BÀI 1: GIỚI THIỆU VỀ PLC

1.1 Giới thiệu.

Kỹ thuật điều khiển đã được phát triển trong thời gian rất lâu. Trước kia việc điều khiển hệ thống chủ yếu do con người thực hiện. Gần đây, việc điều khiển được thực hiện nhờ vào các ứng dụng của ngành điện, thực hiện bằng việc đóng ngắt tiếp điểm relay. Các relay sẽ cho phép đóng ngắt công suất không cần dùng công tắc cơ khí. Ta thường sử dụng relay để tạo nên các thao tác điều khiển đóng ngắt logic đơn giản. Sự xuất hiện của máy tính điện tử đã tạo một bước tiến mới trong điều khiển – Kỹ thuật điều khiển lập trình PLC. PLC xuất hiện vào những năm 1970 và nhanh chóng trở thành sự lựa chọn cho việc điều khiển sản xuất.

1.2 Những ưu điểm của các nhà máy khi sử dụng PLC.

- Giảm giá thành đối với các hệ thống phức tạp.
- Mềm dẽo và dễ thay thế khi cần thay đổi hệ thống điều khiển.
- Khả năng kết hợp với máy tính cho phép điều khiển các hệ thống tinh vi.
- Khả năng hỗ trợ xử lý sự cố làm cho việc lập trình dễ dàng và nhanh chóng.
- Kết cấu chắc chắn và chính xác làm cho hệ thống hoạt động ổn định và tin cậy.

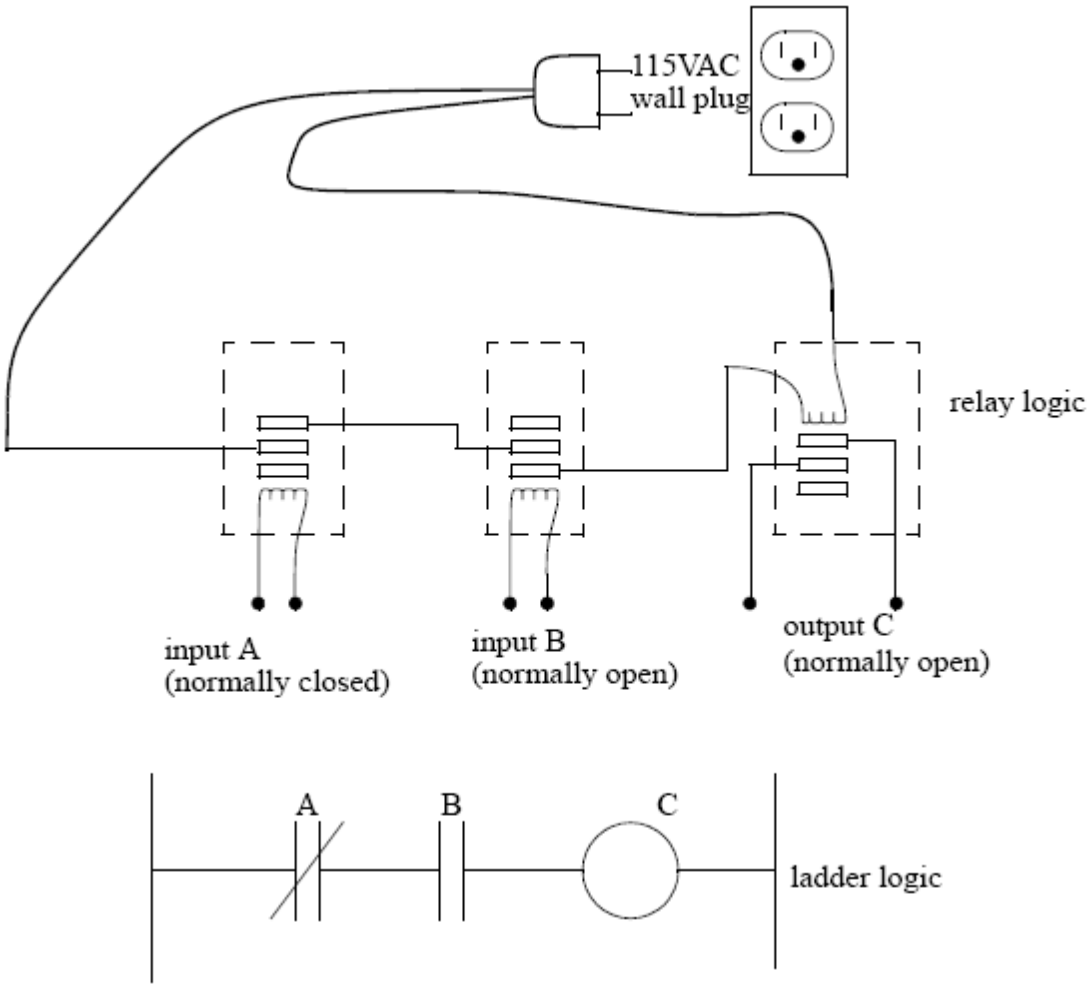
1.3 Logic bậc thang.

Logic bậc thang là phương pháp lập trình chính cho PLC. Logic bậc thang được phát triển để thay thế cho việc điều khiển bằng logic relay. Do đã có sơ đồ điều khiển bằng relay nên khi chọn Logic bậc thang làm phương pháp lập trình chính cho PLC thì việc huấn luyện cho các kỹ sư và người sử dụng sẽ giảm đi rất nhiều.

Các hệ thống điều khiển hiện đại ngày nay vẫn còn sử dụng relay, nhưng chúng không được dùng để tạo ra mức logic mà hoạt động như một thiết bị điện từ dùng để đóng mở tiếp điểm.

Các relay được dùng để đóng mở các nguồn điện công suất lớn dựa vào nguồn năng lượng nhỏ, vẫn giữ cách ly các nguồn này.

Hệ thống điều khiển đơn giản có sử dụng relay được minh họa trên hình 1.1. Relay bên trái sử dụng tiếp điểm thường đóng cho dòng điện qua đến khi có điện áp cấp vào đầu dây A. Relay ở giữa sử dụng tiếp điểm thường hở nên không cho dòng điện qua đến khi đầu dây B có điện. Nếu dòng điện qua 2 tiếp điểm của relay A và B rồi vào cuộn dây của relay C thì sẽ đóng tiếp điểm đầu ra C.

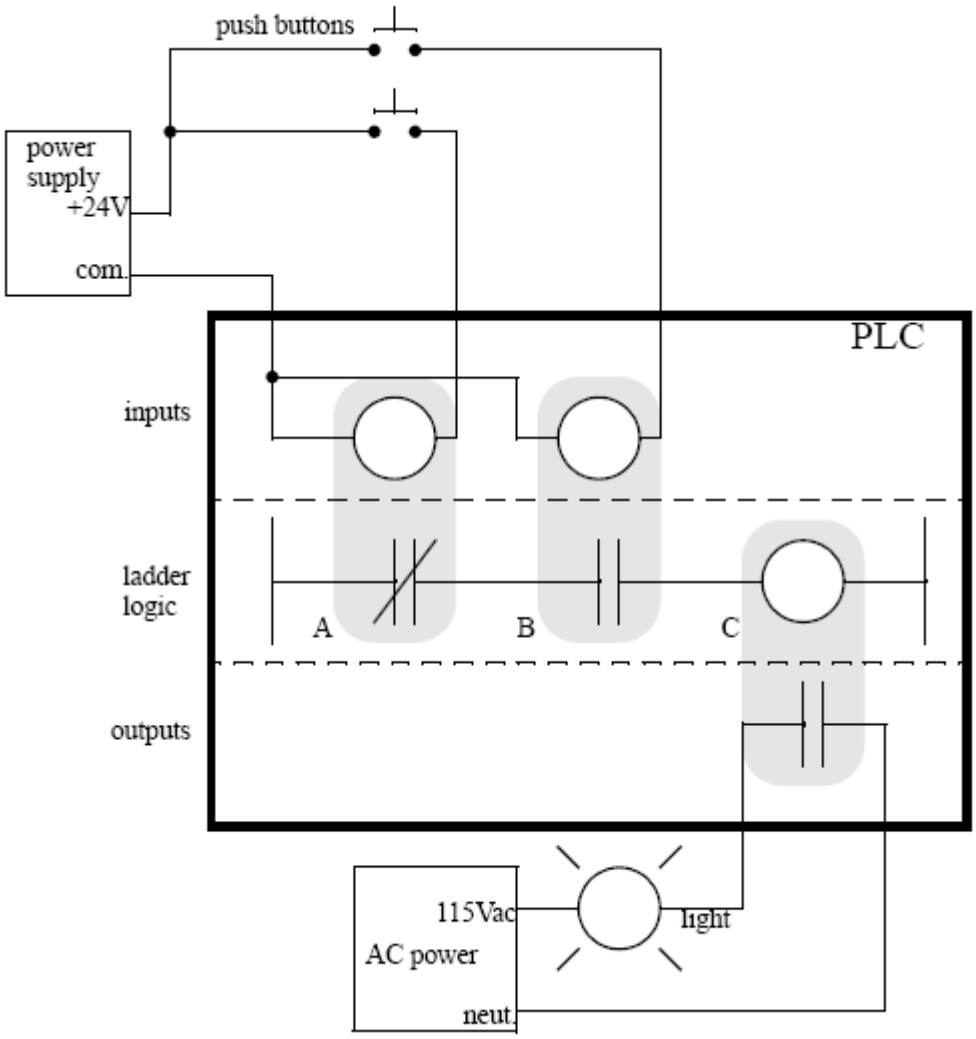


Hình 1.1: Hệ thống điều khiển dùng relay.

Mạch điện được vẽ lại ở dạng sơ đồ logic bậc thang bên dưới trong hình 1.1.
Trạng thái logic được đọc là: C đóng nếu A mở và B đóng.
Hình 1.1 không phải là toàn bộ hệ thống điều khiển, chỉ là sơ đồ logic. Khi xem xét một PLC, ngoài sơ đồ logic còn có các ngõ vào/ra. Hình 1.2 là một minh họa về sơ đồ logic cùng với ngõ vào,ngõ ra của một PLC. PLC này có 2 ngõ vào nút nhấn, giả sử sẽ tác động các cuộn dây relay bên trong PLC, làm ngõ ra relay đóng cấp nguồn 115VAC cho đèn sáng.

Lưu ý:

Các PLC thực tế, ngõ vào không sử dụng relay, nhưng ngõ ra có thể sử dụng relay. Logic bậc thang trong PLC thường là các chương trình do người dùng viết và hiệu chỉnh trên máy tính. Cả 2 ngõ vào PLC là nút nhấn thường hở, nhưng logic bậc thang bên trong PLC có thể sử dụng một thường đóng và một thường hở, không nhất thiết logic bậc thang này phải phù hợp với trạng thái các ngõ vào/ ra.



Hình 1.2: PLC có sử dụng relay

Một số relay có nhiều ngõ ra nên có thể sử dụng một ngõ ra relay như một ngõ vào tức thời, tạo thành mạch duy trì như trong hình 1.3.

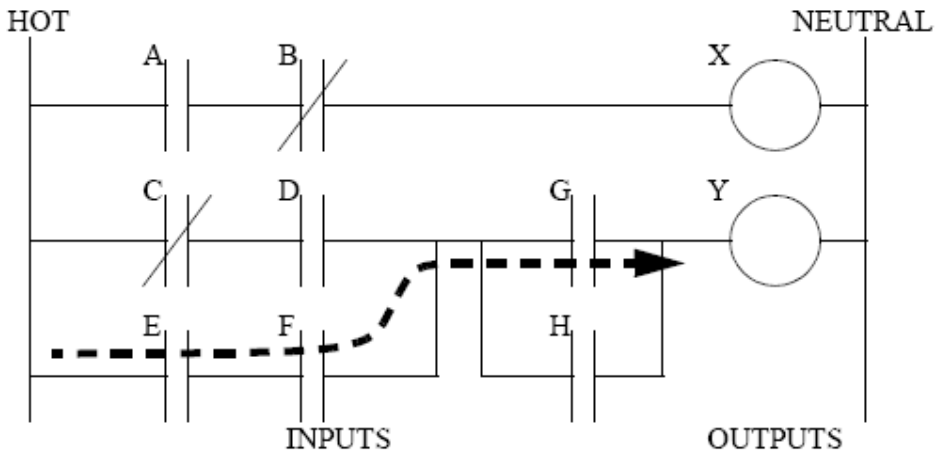
Hình 1.3: Mạch duy trì ngõ ra

Trong mạch này, dòng điện có thể chạy qua cả 2 nhánh là các công tắc A và B. Ngõ vào B chỉ đóng khi ngõ ra B có điện. Nếu B mất điện, đóng ngõ vào A thì B sẽ có điện, làm ngõ vào B đóng. Khi đó cho dù ngõ vào A mở ra nhưng B vẫn có điện nhờ ngõ vào B duy trì. Sau khi ngõ vào B đã đóng thì không thể tắt điện B được.

1.4 Lập trình.

Lập trình là thuật ngữ dùng để nói đến việc con người sử dụng những ngôn ngữ mà PLC hiểu được để giao tiếp với nó, điều khiển nó hoạt động theo ý đồ mà người lập trình đề ra nhằm đáp ứng những yêu cầu trong thực tiễn.

Các PLC trước kia được lập trình bằng kỹ thuật sử dụng các sơ đồ nối dây relay. Do đó không cần phải hướng dẫn nhiều cho các thợ điện, kỹ thuật viên, kỹ sư cách lập trình trên máy tính, nên đây cũng là kỹ thuật lập trình thông dụng cho PLC ngày nay. Xét ví dụ trên hình 1.4.



Hình 1.4: Sơ đồ logic bậc thang đơn giản

Giả sử nguồn nối với đường dây bên trái HOT, gọi là dây nóng, bên phải là dây trung tính. Sơ đồ có 2 nhánh, mỗi nhánh là một tổ hợp các ngõ vào và ngõ ra.

Nếu các ngõ vào đóng hoặc mở thì công suất sẽ chạy từ dây nóng qua các ngõ vào, kết hợp với dây trung tính cấp điện cho ngõ ra.

Ngõ vào PLC có thể được kết nối với các cảm biến hoặc công tắc. Ngõ ra PLC sẽ nối với các thiết bị trung gian đóng ngắt các tải bên ngoài như đèn, động cơ.

Trong nhánh trên, công tắc A thường hở và B thường đóng, nghĩa là nếu A đóng và B mở thì dòng điện sẽ chạy qua công tắc A và B tác động đến ngõ ra X, các trạng thái khác của A và B sẽ làm X mất điện. Tương tự như vậy người đọc có thể giải thích tương tự cho hoạt động của nhánh bên dưới.

Có nhiều phương pháp lập trình khác nhau cho PLC. Một trong những kỹ thuật đó là sử dụng lệnh gợi nhớ. Các lệnh này xuất phát trực tiếp từ sơ đồ logic bậc thang và được nhập vào PLC bằng một thiết bị lập trình.

Trong ví dụ hình 1.5, các lệnh được đọc lần lượt từ trên xuống dưới.

Dòng 00000 có lệnh LDN (input load not) cho ngõ vào 00001. Lệnh này xác định một ngõ vào nối với PLC, nếu nó mở thì sẽ tạo một giá trị 1, và ngược lại sẽ tạo giá trị 0.

Dòng tiếp theo 00001 sử dụng lệnh LD (input load) để xác định giá trị ngõ vào, nếu ngõ vào này mở thì tạo giá trị 0 và ngược lại sẽ tạo giá trị 1.

Lênh AND sử dụng lại 2 số được tạo ra bên trên, nếu chúng cùng bằng 1 thì sẽ tạo ra giá trị 1, còn có một ngõ vào bằng 0 thì tạo giá trị 0. Giá trị này sẽ thay thế cho 2 kết quả trên và lúc này chỉ còn một kết quả của lệnh AND được giữ lại.

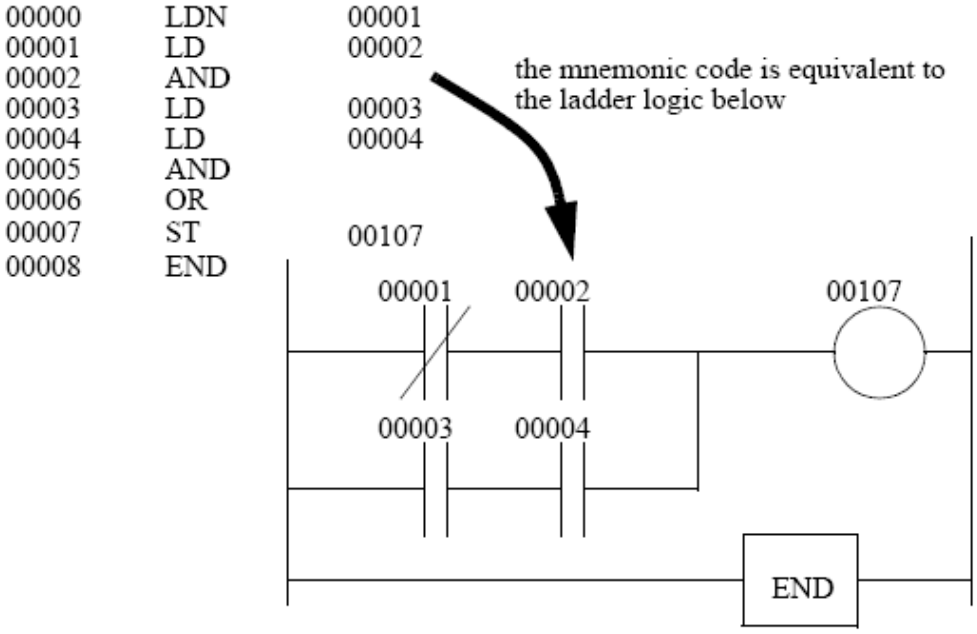
Quá trình này sẽ lặp lại với các hàng 00003 và 00004, sau khi thực hiện xong sẽ có 3 số được lưu lại.

Lệnh AND trong hàng 00005 sẽ AND kết quả của hàng 00003 và 00004, tạo ra 1 kết quả mới.

Lệnh OR trong hàng 00006 sẽ OR kết quả của 2 lệnh AND ở các hàng trên. Lúc này chỉ còn 1 kết quả lưu lại.

Lệnh ST (store ouput) trong hàng 00007 sẽ lưu lại kết quả sau cùng. Nếu kết quả này bằng 1 thì ngõ ra 00107 sẽ tác động, ngược lại ngõ ra này không tác động.

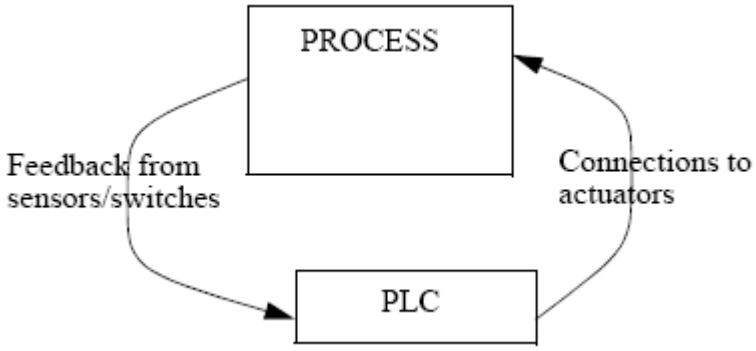
Chương trình logic bậc thang trong hình 1.5 tương đương với chương trình gợi nhớ vừa phân tích trên. Thậm chí nếu ta đã lập trình cho PLC bằng logic bậc thang thì nó có thể sẽ được chuyển về dạng gợi nhớ trước khi được PLC sử dụng.



Hình 1.5: Chương trình gợi nhớ và Sơ đồ logic bậc thang tương đương

1.5 Kết nối PLC.

Khi sử dụng PLC để điều khiển một quá trình nào đó, ta sử dụng các cảm biến nối với ngõ vào PLC, ngõ ra PLC sẽ điều khiển các thiết bị chấp hành, như hình 1.6.



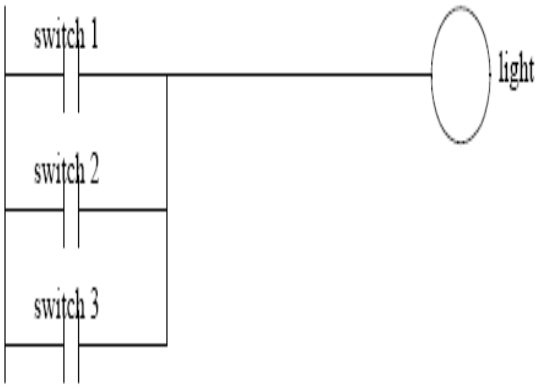
Hình 1.6: Kết nối PLC

Đây là quá trình xử lý thực, thay đổi liên tục theo thời gian. Các thiết bị chấp hành sẽ làm hệ thống thay đổi sang các trạng thái mới, có nghĩa là hệ thống sẽ được giới hạn điều khiển bởi các cảm biến đầu vào. Nếu 1 ngõ vào không tác động thì bộ điều khiển không thể nhận biết được trạng thái hệ thống.

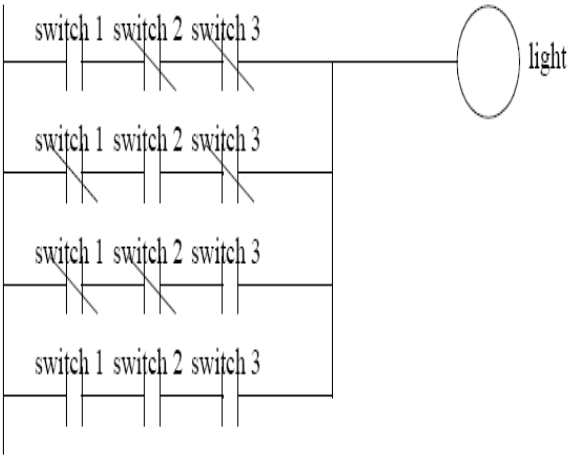
Vòng điều khiển này là 1 chu kỳ liên tục của PLC(Chu kỳ quét của PLC), gồm việc đọc các dữ liệu đầu vào, thực hiện logic bậc thang và làm thay đổi ngõ ra theo ngõ vào.

Ví dụ: Vẽ sơ đồ điều khiển relay sử dụng 3 công tắc điều khiển 1 bóng đèn.

Cách 1:



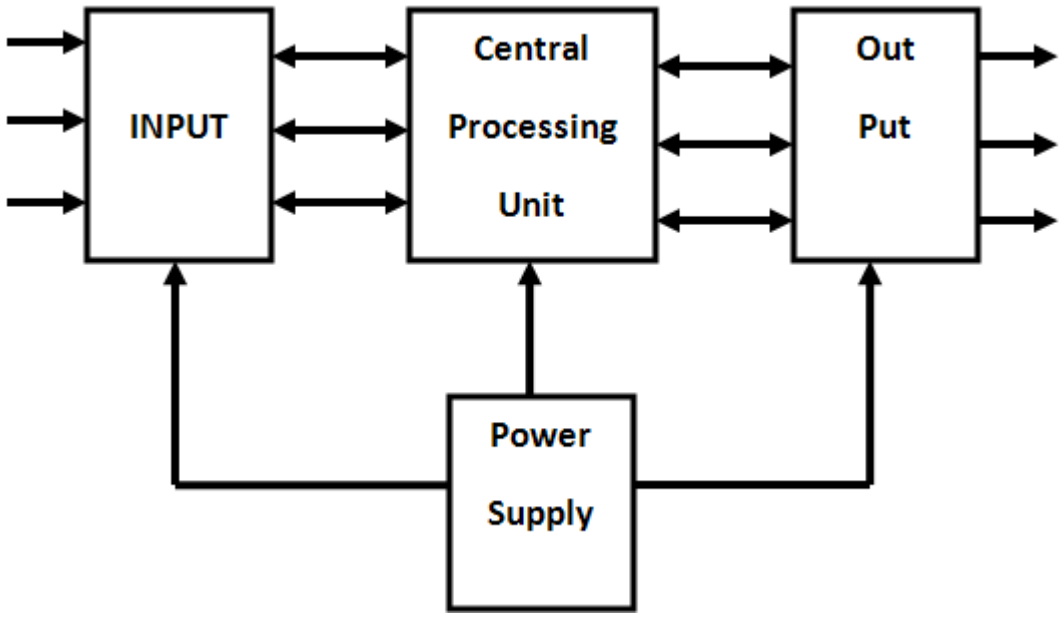
Cách 2:



BÀI 2: CẤU TRÚC VÀ HOẠT ĐỘNG CỦA PLC

2.1 Cấu trúc phần cứng của PLC.

PLC có nhiều hãng sản xuất, nhiều loại và nhiều cấu hình khác nhau. Tuy nhiên, dù là của hãng nào, loại nào, cấu hình nào thì chúng đều có chung các thành phần sau:



Hình 2.1: Cấu trúc tổng quát của một PLC

2.1.1 Nguồn cung cấp.

Nguồn có thể tích hợp sẵn bên trong PLC hoặc làm riêng bên ngoài. Có nhiều cấp điện áp khác nhau tùy loại PLC, gồm 110VAC hoặc 220VAC hoặc 24VDC(Hiện nay có hai cấp điện áp thường được sử dụng là 24VDCvà 220V-AC.

2.1.2 CPU (Central Proceesoing Unit).

Đây là bộ xử lý trung tâm làm việc như 1 máy tính, dùng để lưu trữ và xử lý chương trình theo yêu cầu của người lập trình.

2.1.3 I(Input).

Các loại cảm biến, công tắc, nút nhấn… đưa tín hiệu vào PLC thông qua module Input. Tùy vào loại tín hiệu của cảm biến là số hay tương tự mà moudle ngõ vào của PLC cũng có hai loại là Module số (Digital Module) và Module tương tự (Analog Module).

2.1.4 O(Output).

Các loại cơ cấu chấp hành như: Bóng đèn, cuộn dây, vale, biến tần….. được điều khiển bởi PLC thông qua module Output. Tùy vào đối tượng điều khiển cần tín hiệu số hay tương tự mà moudle ngõ ra của PLC cũng có hai loại là module số ngõ ra (Digital Output Module) và module ngõ ra tương tự (Analog Output Module).

2.1.5 Đèn báo.

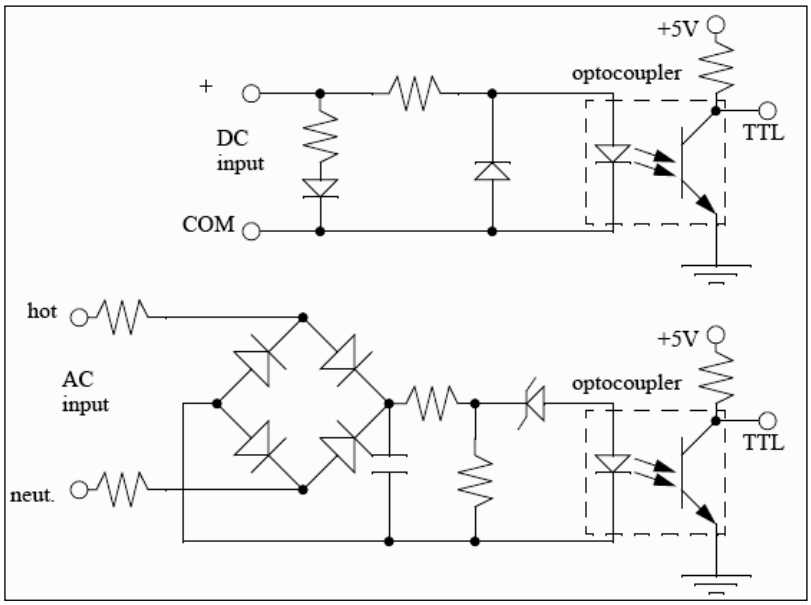
Dùng để chỉ báo trạng thái PLC, gồm nguồn, chạy chương trình, lỗi hệ thống. Các cảnh báo này rất cần thiết trong chẩn đoán sự cố.

2.2 Module ngõ vào.

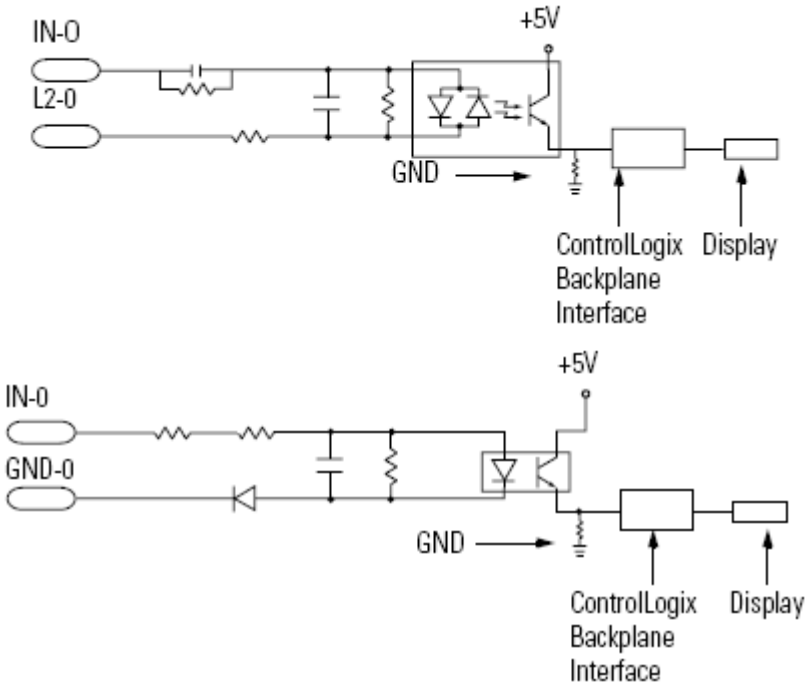
Tùy vào loại tín hiệu của cảm biến là số hay tương tự mà moudle ngõ vào của PLC cũng có hai loại là Module số (Digital Module) và Module tương tự (Analog Module).

2.2.1 Module ngõ vào số.

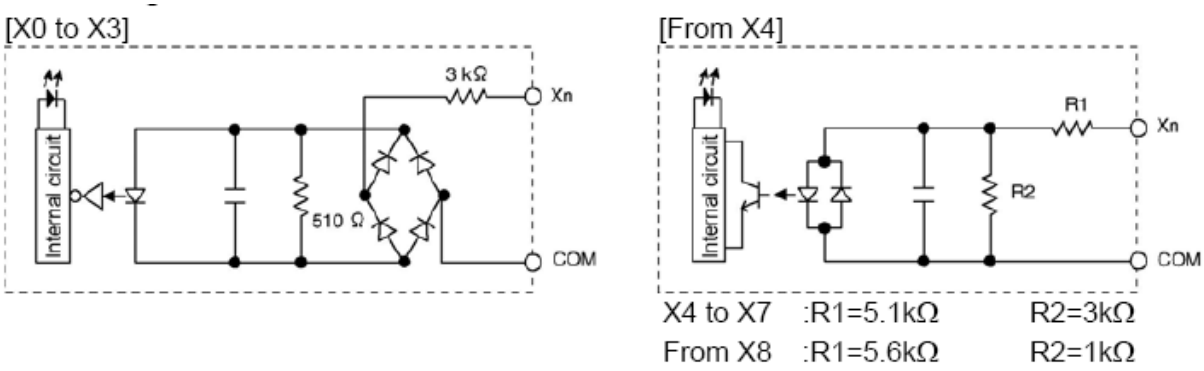
Các loại cảm biến, công tắc, nút nhấn, encoder…. Có tín hiệu ngõ ra dạng số thì được đưa vào PLC thông qua module số. Dưới đây trình bày một số dạng ngõ vào số của một vài PLC.



Hình 2.2: Mạch điện ngõ vào số của PLC Siemens



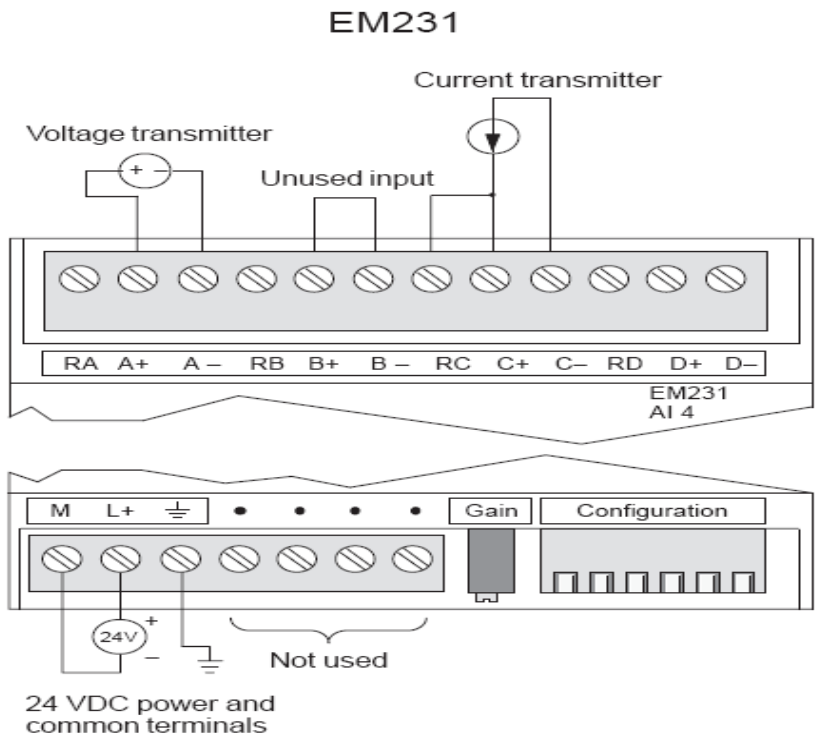
Hình 2.3: Mạch điện ngõ vào số của PLC Rockwell



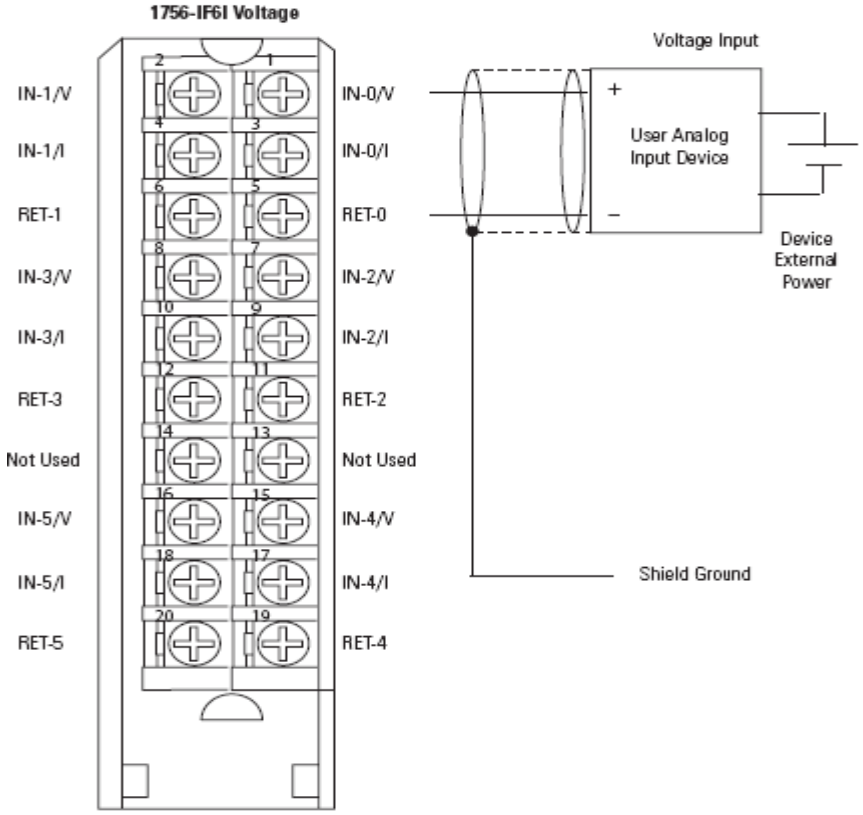
Hình 2.3: Mạch điện ngõ vào số của PLC Panasonic

2.2.2 Module ngõ vào Analog.

Các loại cảm biến, vale, cặp nhiệt, máy phát tốc…. Có tín hiệu ngõ ra dạng analog thì được đưa vào PLC thông qua module analog. Dưới đây trình bày một số dạng module analog ngõ vào của một số PLC.



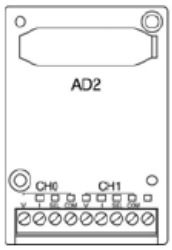
Hình 2.4: Module ngõ vào analog của PLC Siemens

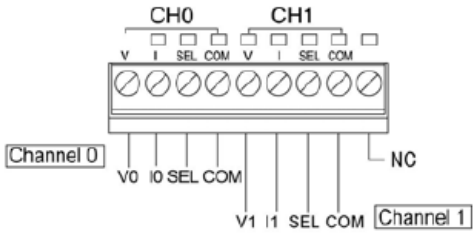


- Do not connect more than 2 wires to any single terminal.

Hình 2.5: Module ngõ vào analog của PLC rockwell

Terminal layout





Note) There is no LED indication.

CH0	**V**	Voltage input
	I	Current input
	SEL	Voltage/current select
	COM	Common
CH1	**V**	Voltage input
	I	Current input
	SEL	Voltage/current select
	COM	Common
NC		Not used

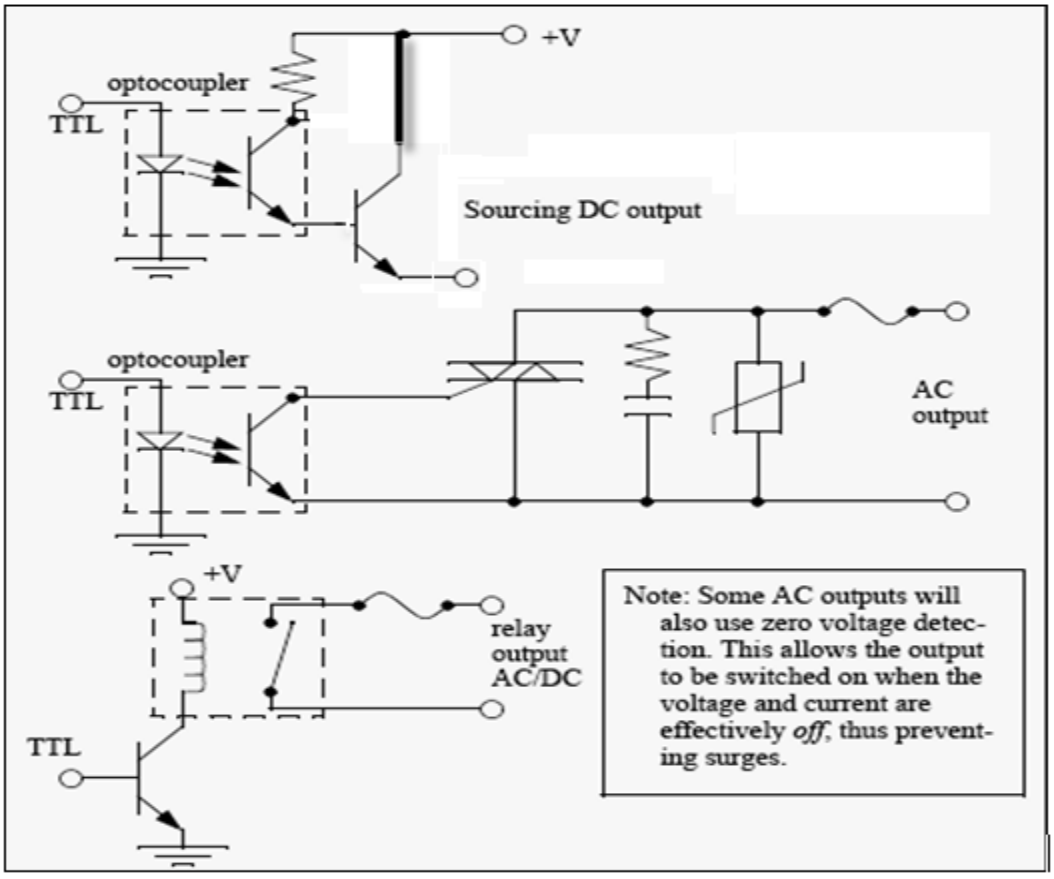
Hình 2.6: Module ngõ vào analog của PLC Panasonic

2.3 Module ngõ ra.

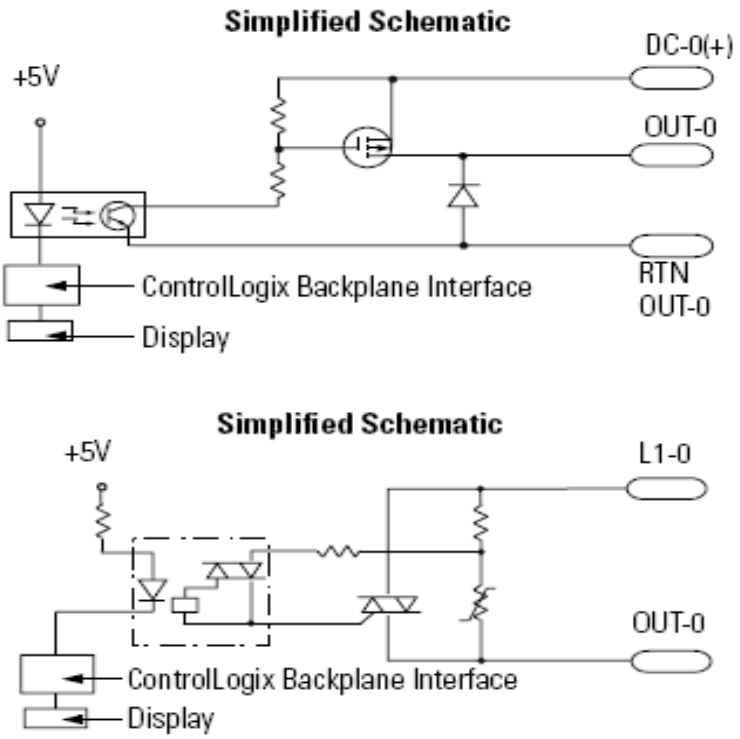
Tùy thuộc vào từng loại cơ cấu chấp mà tín hiệu điều khiển nó có thể là số hay tương tự mà PLC cũng có loại module ngõ ra số và module ngõ ra tương tự tương ứng để điều khiển.

2.3.1 Module ngõ ra số.

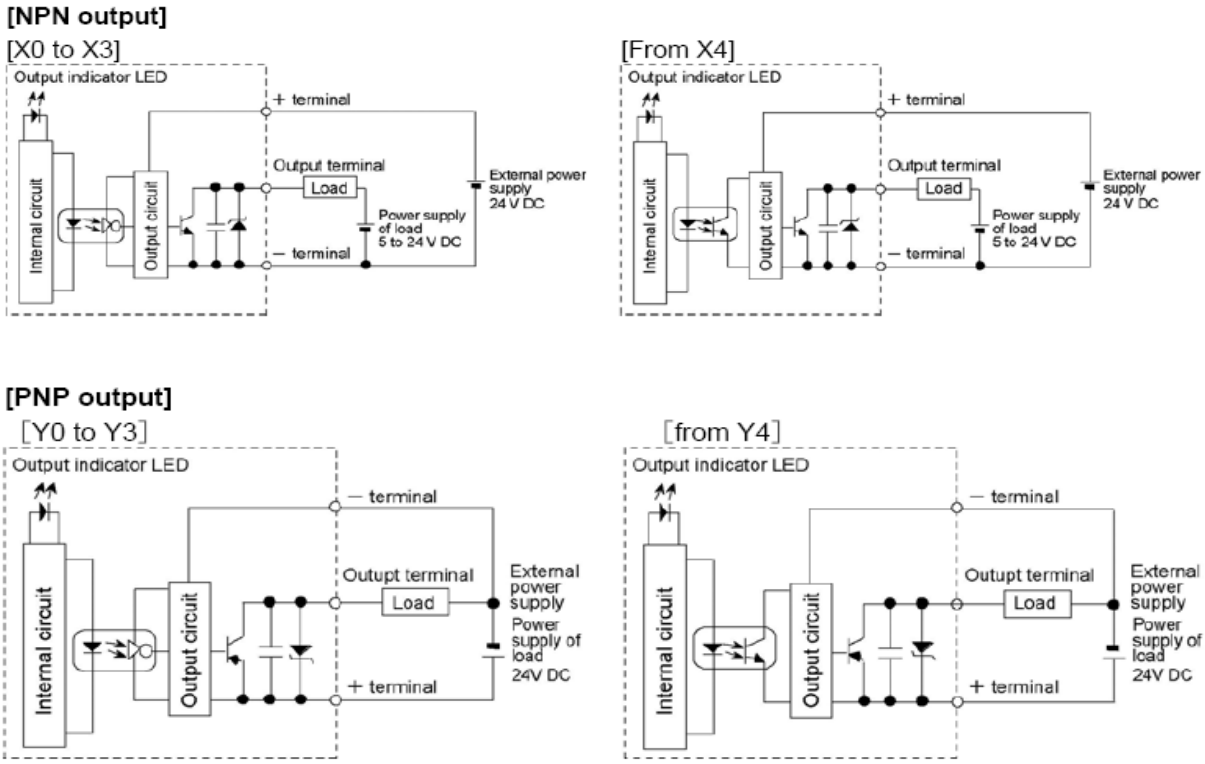
Module ngõ ra số có những kiểu tiêu biểu như: Ngõ ra Transistor, ngõ ra Triac và ngõ ra Relay. Khi cần điều khiển các đối tượng hoạt động theo kiểu ON-OFF thì ta sử dụng module ngõ ra dạng số. Tuy nhiên tùy thuộc vào cấp điện áp, tần số đóng cắt, dòng làm việc mà ta chọn kiểu ngõ ra nào cho phù hợp. Các thông số về dòng, áp, tần số người sử dụng nên tham khảo manual của nhà sản xuất. Dưới đây sẽ trình bày một số kiểu ngõ ra của một số PLC tiêu biểu.



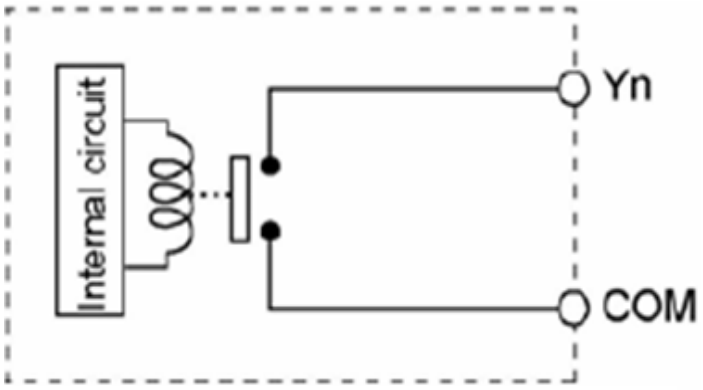
Hình 2.6: Ngõ ra số của PLC siemens.



Hình 2.7: Ngõ ra số của PLC rockwell



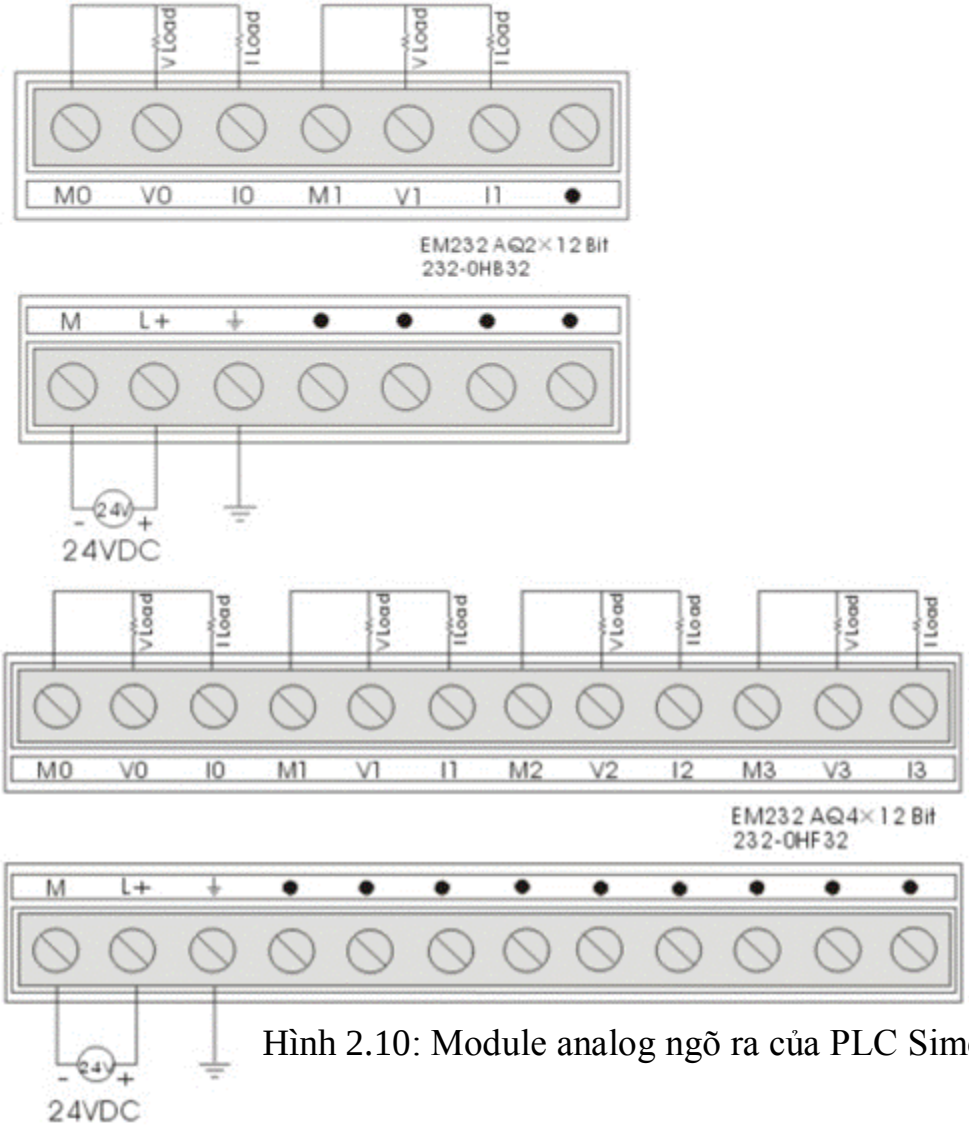
Hình 2.8: Ngõ ra số của PLC Panasonic



Hình 2.9: Ngõ ra Relay của PLC

2.4 Module ngõ ra tương tự.

Module ngõ ra tương tự có hai kiểu tiêu biểu: Ngõ ra điện áp hoặc ngõ ra dòng điện. Đối với những đối tượng yêu cầu tín hiệu điều khiển phải ở dạng analog như: Ngõ vào biến tần, vale tuyến tính, vale thủy lực….. thì cách thường sử dụng nhất là dùng module analog của PLC. Đặc điểm của module analog trong PLC là kết nối đơn giản, dễ sử dụng, không cần phải kết nối thêm các thiết bị bên ngoài. Hình 2.10 là một module analog ngõ ra điển hình của PLC siemens.



Hình 2.10: Module analog ngõ ra của PLC Simens.

2.6 Hoạt động PLC.

Tất cả PLC đều hoạt động theo chu trình lặp, mỗi chu trình hoạt động gồm 4 giai đoạn: Đọc ngõ vào, Ethực thi chương trình, chẩn đoán lỗi và kiểm tra truyền thông, xuất kết quả ra để điều khiển thiết bị. 4 giai đoạn này thường được gọi là 1 chu kỳ quét của PLC.

Hình 2.11: Một chu kỳ quét của PLC

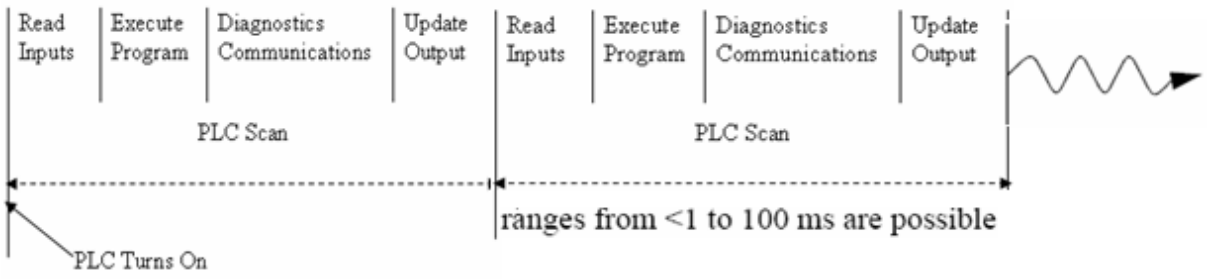
Read Input (Đọc ngõ vào): PLC đọc trạng thái của toàn bộ các ngõ vào và chứa vào bộ đệm ngõ vào.

Execute Program (Thực thi chương trình): PLC dựa vào các trạng thái ngõ vào để thực thi theo chương trình đã được lưu trong bộ nhớ đệm ngõ ra.

Diagnostics Communications (Chẩn đoán và truyền thông): PLC tiến hành chẩn đoán lỗi và kiểm tra quá trình truyền thông.

Update Outputs (Xuất kết quả): PLC xuất kết quả trong vùng nhớ đệm ngõ ra để điều khiển thiết bị ngoại vi.

Quá trình này cứ lặp đi lặp lại từ 10 đến 100 lần mỗi s, như hình 2.12.



Hình 2.12: Thời gian chu kỳ quét của PLC.

2.7 Trạng Thái PLC.

Điều dễ nhận thấy ở PLC là nó thiếu bàn phím và các thiết bị vào ra khác. Tuy nhiên để giúp người lập trình nhanh chóng xác định trạng thái, PLC thường có các đèn chỉ trạng thái, bao gồm:

Đèn báo nguồn.

Đèn chạy chương trình.

Đèn báo sự cố.

Các đèn này thường dùng cho việc sửa lỗi.

Ngoài ra phần cứng PLC còn có các nút nhấn, phổ biến nhất là nút chạy chương trình.

2.8 Bộ Nhớ.

Các loại bộ nhớ được dùng phổ biến hiện nay bao gồm: RAM, ROM, EPROM, EEPROM. (Tương tự như phần bộ nhớ trong Kỹ thuật số).

Tất cả PLC đều sử dụng RAM cho CPU và dùng ROM để lưu hệ điều hành cho PLC.

Khi cấp nguồn, nội dung của RAM sẽ được giữ lại. Nhưng vấn đề cần quan tâm là chuyện gì xảy ra khi bộ nhớ này mất nguồn. Các PLC trước kia sử dụng RAM có nguồn pin nên dữ liệu RAM không bị mất khi mất điện. Phương pháp này vẫn còn sử dụng nhưng không nhiều. Ngày nay người ta sử dụng EPROM làm bộ nhớ cho PLC. Bộ nhớ này được lập trình bên ngoài PLC sau đó đặt vào

PLC. Khi PLC hoạt động chương trình này sẽ được nạp vào PLC và thực hiện. Phương pháp này chính xác nhưng việc lập trình và xóa bộ nhớ sẽ mất nhiều thời gian. Bộ nhớ EEPROM trở thành một phần cố định của PLC, các chương trình lưu trong EEPROM tương tự như lưu trong EPROM. Hiên nay giá thành các bộ nhớ đã giảm đáng kể, và người ta còn phát triển thêm các bộ nhớ khác như Flash ROM.

BÀI 3: CẢM BIẾN

3.1 Giới thiệu.

Cảm biến là một thiết bị dùng để nhận biết sự tồn tại hay thay đổi đặc tính của một đối tượng nào đó. Tùy thuộc vào thuộc tính của đối tượng, cách nhận biết đối tượng đó mà người ta chế tạo ra các loại cảm biến khác nhau. Thông thường thì các cảm biến có thể nhận biết được một số hiện tượng vật lý như sau:

- Có một vật kim loại ở gần hay không?
- Công tắc, nút nhấn có tác động hay không?
- Có sự thay đổi môi trường hay không?
- Có vật che hoặc phản xạ ánh sáng hay không?
- Điện áp hay dòng điện ngõ vào có thay đổi không?

Dựa vào đặc điểm tác động của cảm biến mà chúng được phân làm 2 dạng chính:

Cảm biến có ngõ ra dạng số: Công tắc, nút nhấn, cảm biến điện dung,cảm biến điện cảm, cảm biến quang…

Cảm biến có ngõ ra dạng tương tự: Cảm biến dòng, cảm biến áp, cảm biến nhiệt độ…..…

3.2 Cảm biến có ngõ ra dạng số.

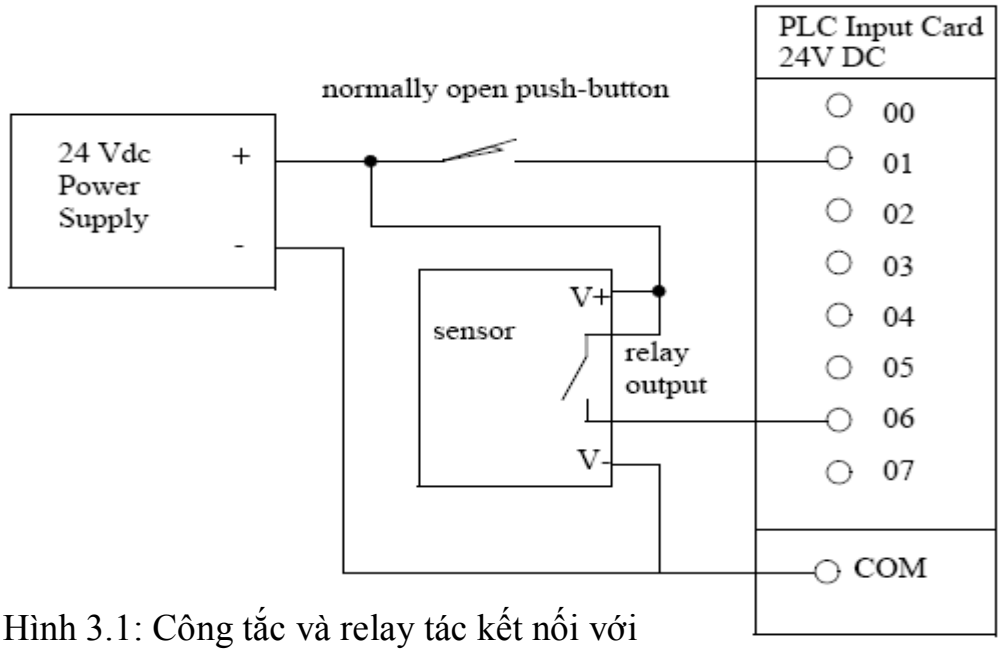
Cảm biến có ngõ ra dạng số được sử dụng để nhận biết có sự thay đổi logic để báo cho bộ xử lý(PLC) biết sự thay đổi này bằng cách đóng/ ngắt một điện áp hoặc dòng điện đến PLC. Trong một số trường hợp ngõ ra của cảm biến sẽ đóng ngắt trực tiếp tải.

Ngõ ra của cảm biến (ngõ vào PLC) bao gồm:

- Các công tắc để đóng ngắt điện áp, dòng điện.
- Ngõ ra cấp dòng hoặc rút dòng.
- Cảm biến điện cảm.
- Cảm biến điện dung.
- Các tiếp điểm relay để đóng ngắt ngõ ra AC.
- Ngõ ra TTL chỉ mức logic 0 hoặc 5V.

3.2.1 Công tắc hoặc tiếp điểm relay.

Công tắc, tiếp điểm relay được xem là một cảm biến đơn giản nhất, nó được sử dụng để đóng ngắt điện áp hoặc dòng điện đến ngõ vào của PLC.



Hình 3.1: Công tắc và relay tác kết nối với

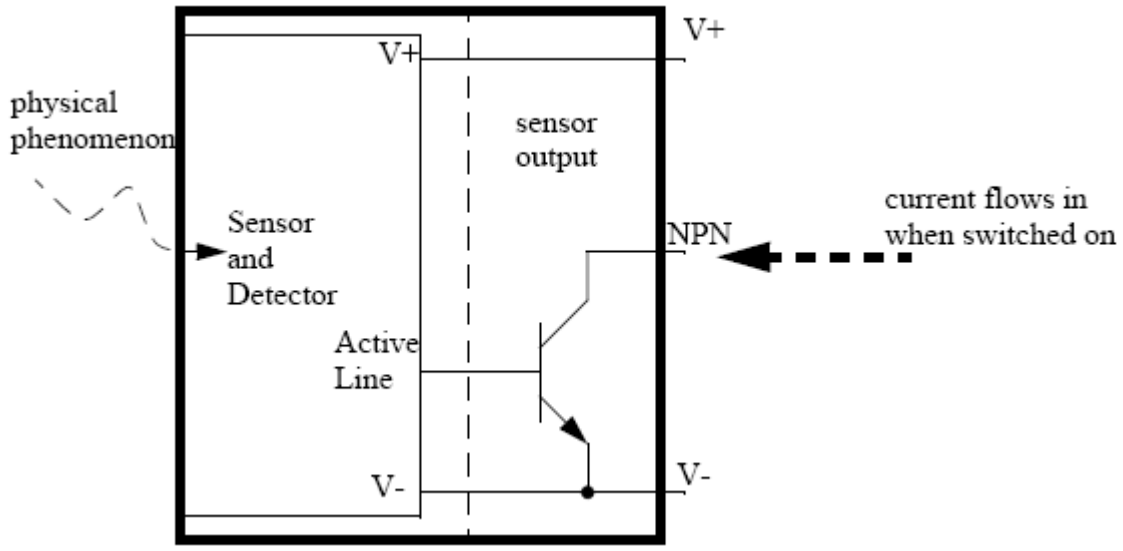
Hình vẽ 3.1 gồm có công tắc thường hở NO (Normal Open) được nối đến ngõ vào I0.1, cảm biến có ngõ ra relay được cấp nguồn +/-V.

Ngõ ra cảm biến sẽ tác động khi xảy ra một hiện tượng nào đó định trước. công tắc bên trong cảm biến sẽ đóng lại cấp điện áp đến ngõ vào I0.6 của PLC.

3.2.2 Ngõ ra rút dòng và cấp dòng (Sinking/Sourcing).

Cảm biến rút dòng cho phép dòng chạy vào cảm biến về mass, còn cảm biến cấp dòng cho phép nguồn từ Vcc trong cảm biến chạy ra ngoài cảm biến.

Ngõ ra của cảm biến sử dụng transistor đóng ngắt (có tổn hao điện áp). Loại NPN dùng cho ngõ ra rút dòng, loại PNP ngõ ra cấp dòng. Minh họa trên hình vẽ 3.2.

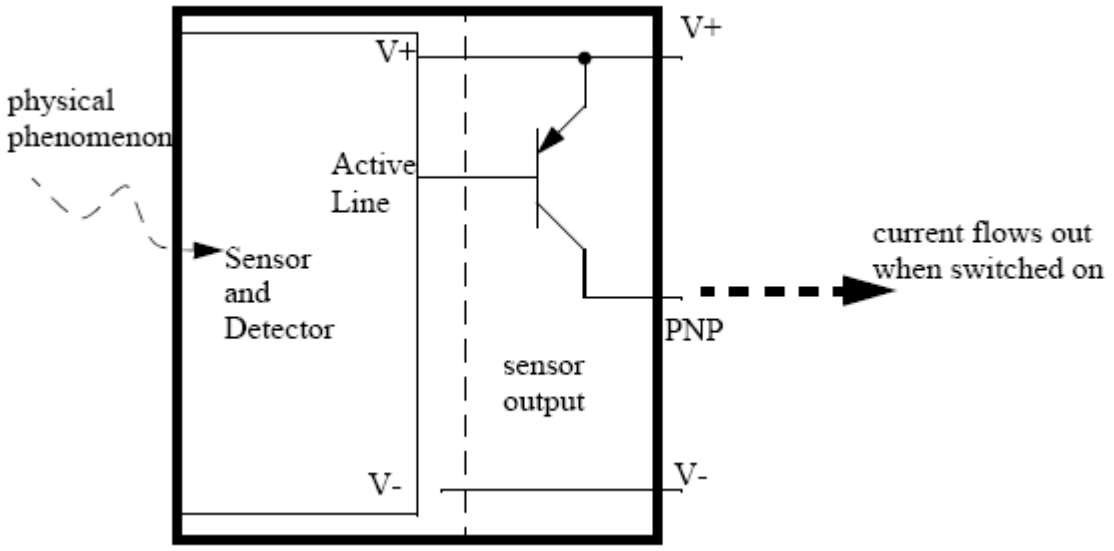


Hình 3.2: Ngõ ra Rút dòng

Cảm biến có bộ phận đầu dò để nhận biết các hiện tượng vật lý xảy ra. Với nguồn cung cấp +/-V, cảm biến sẽ nhận biết các hiện tượng xảy ra và tác động vào chân B của transisotr NPN.

Nếu chân B có 0V thì transistor ngưng dẫn, nếu chân B có điện áp phân cực thì transitor dẫn bảo hòa rút dòng bên ngoài vào.

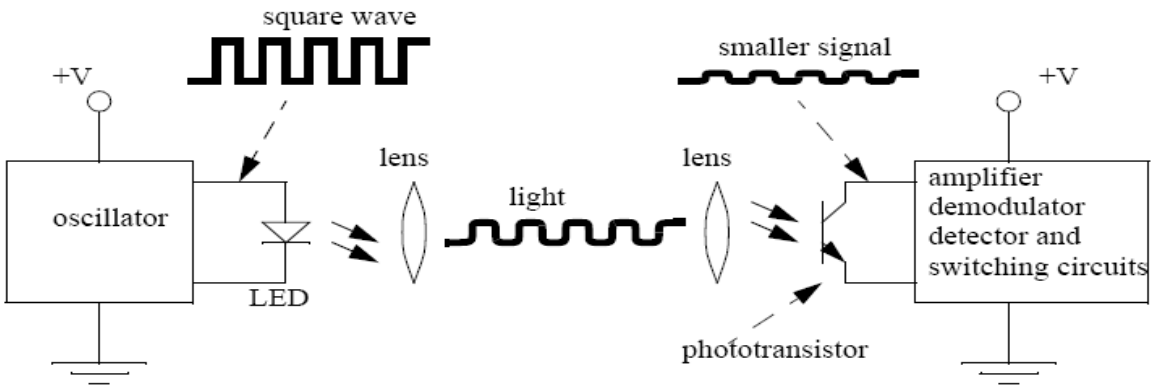
Hoạt động tương tự cho cảm biếm cấp dòng PNP ở hình 3.3.



Hình 3.3: Ngõ ra Cấp dòng

3.2.3 Cảm Biến Quang.

Cảm biến quang được dùng rất lâu, bao gồm một nguồn phát quang và một bộ thu quang. Nguồn quang sử dụng LED hoặc LASER phát ra ánh sáng thấy hoặc không thấy tùy theo bước sóng. Bộ thu quang sử dụng diode hoặc transistor quang. Ta đặt bộ thu và phát sao cho vật cần nhận biết có thể che chắn hoặc phản xạ ánh sáng khi vật xuất hiện. Cảm biến quang thường có 2 dạng: Cảm biến quang thu phát chung và cảm biến quang thu phát độc lập. Sơ đồ sử dụng cảm biến quang cho trên hình 3.4.



Hình 3.4: Cảm biến quang

Ánh sáng do LED phát ra được hội tụ qua thấu kính. Ở phần thu ánh sáng từ thấu kính tác động đến transistor thu quang.

Nếu có vật che chắn thì chùm tia sẽ không tác động đến bộ thu được. Sóng dao động dùng để bộ thu loại bỏ ảnh hưởng của ánh sáng trong phòng. Ánh sáng của mạch phát sẽ tắt và sáng theo tần số mạch dao động.

Phương pháp sử dụng mạch dao động làm cho cảm biến thu phát xa hơn và tiêu thụ ít công suất hơn.

3.2.4 Cảm Biến Điện Dung.

Cảm biến điện dung có thể nhận biết các vật ở khoảng cách lên đến vài cm.

Công thức tính điện dung:

C = A.K/D

C: điện dung (F)

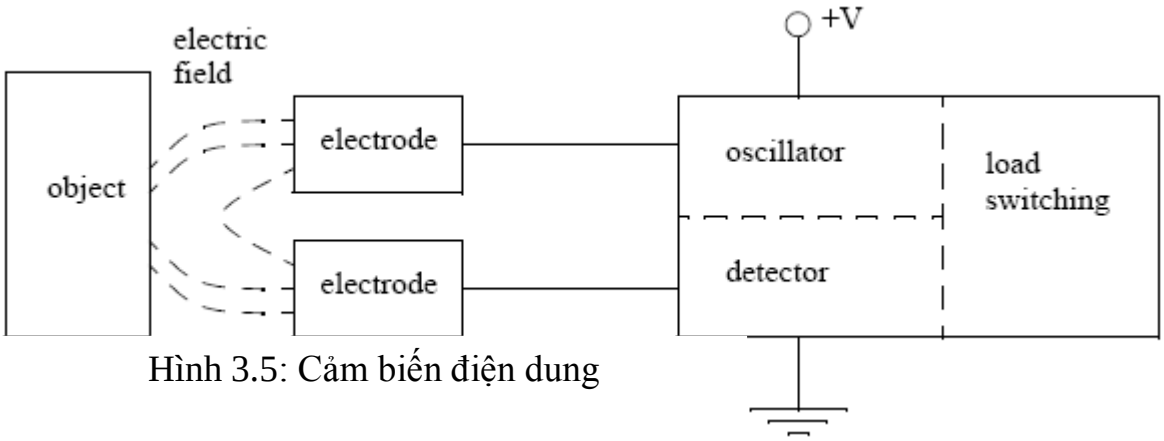
D: hằng số điện môi.

A: diện tích bản cực.

K: khoảng cách 2 bản cực.

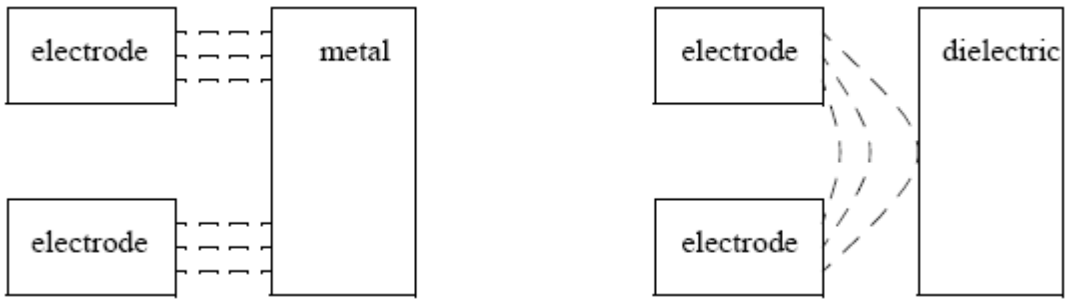
Đối với cảm biến điện dung thì diện tích và khoảng cách 2 bản cực là cố định, nhưng hằng số điện môi của môi trường xung quanh 2 bản cực sẽ thay đổi khi có các vật khác nhau đến gần. Hình 3.5 mô tả một cảm biến điện dung.

Điện dung của 2 bản cực sẽ được xác định bởi một trường thay đổi. Khi có vật đến gần làm thay đổi điện môi giữa 2 bản cực sẽ làm thay đổi điện dung đến giá trị đặt trước nên cảm biến sẽ tác động đóng cắt tải.



Hình 3.5: Cảm biến điện dung

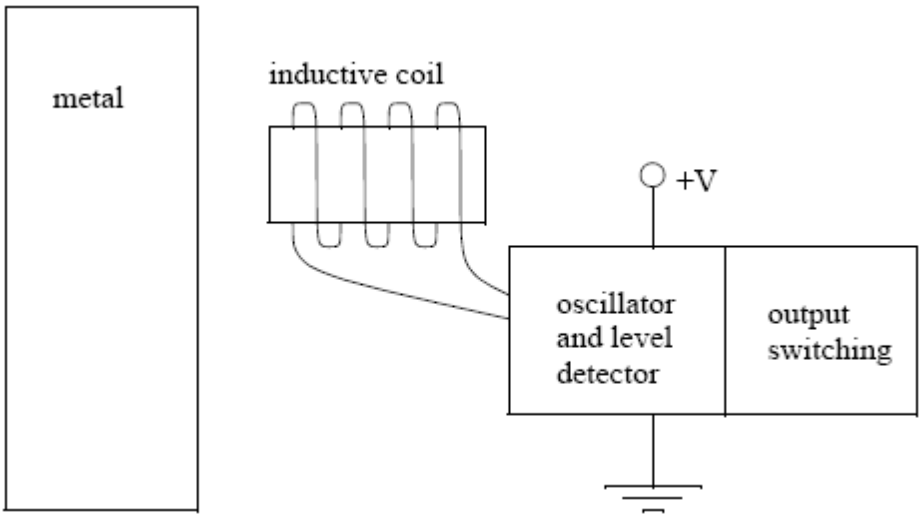
Các cảm biến này làm việc tốt đối với vật cách điện vì chúng có hệ số điện môi lớn nên điện dung lớn. Chúng cũng làm việc tốt đối với kim loại vì các vật dẫn điện tốt xuất hiện sẽ giống như các bản cực lớn hơn nên cũng làm tăng điện dung. Hình 3.7 minh họa ảnh hưởng của vật cách điện và vật dẫn điện đối với cảm biến.



Hình 3.6: Vật cách điện và dẫn điện làm tăng điện dung

3.2.5 Cảm Biến Điện Cảm.

Cảm biến điện cảm sử dụng các từ trường cảm ứng để nhận biết các vật kim loại ở gần. Nó sử dụng cuộn cảm để tạo ra từ trường tần số cao như hình vẽ 3.7.



Hình 3.7: Cảm biến điện cảm.

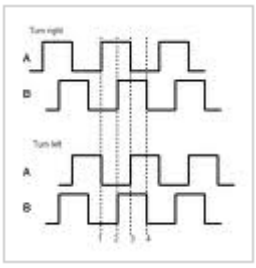
Nếu có một vật kim loại đến gần thì từ trường sẽ thay đổi tạo ra dòng điện qua vật. Dòng điện này tạo ra từ trường mới ngược với từ trường ban đầu nên làm thay đổi cảm kháng cuộn dây bên trong cảm biến. Bằng cách đo điện cảm cảm biến sẽ nhận biết khi có vật kim loại đến gần nó.

Cảm biến loại này có thể dùng để nhận biết bất kỳ vật kim loại nào. Khi cần nhận biết nhiều vật thì ta sử dụng nhiều cảm biến.

3.2.6 Bài tập.

Sinh viên hãy tìm tài liệu, khảo sát đặc tính kỹ thuật, vẽ sơ đồ kết nối các loại cảm biến sau đây với PLC.

3.2.6.1 Bài tập kết nối Encoder với PLC.



3.2.6.2 Bài tập kết nối cảm biến nhiệt độ LM35, LM335 với PLC.

- The output voltage is converted to temperature by a simple conversion factor.
- The sensor has a sensitivity of 10mV / °C.
- Use a conversion factor that is the reciprocal, that is 100 °C/V.
- The general equation used to convert output voltage to temperature is:
  - Temperature (°C) = Vout * (100 °C/V)
  - So if Vout is 1V , then, Temperature = 100 °C
  - The output voltage varies linearly with temperature.

3.2.6.3 Bài tập kết nối cảm biến nhiệt độ PT100 với PLC (Tầm đo từ -200 đến 850 độ c

Rt = R0 * (1 + A* t + B*t2 +C*(t-100)* t3)

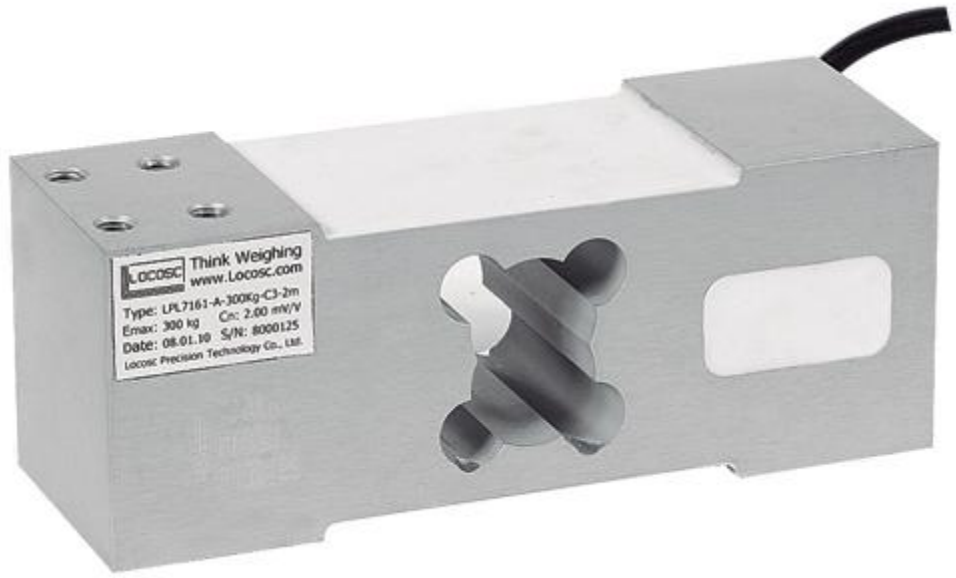
Where:

A = 3.9083 E-3
B = -5.775 E-7
C = -4.183 E -12 (below 0 °C), or
C = 0 (above 0 °C)

3.2.6.4 Bài tập kết nối cảm biến mức (Ultrasonic level sensor) với PLC.

water level sensor

3.2.6.5 Bài tập kết nối Loadcell với PLC.



Technical data:

Item	Specification
Rated Load	50g100g
Dimension	5.5*14*77
Rated Output	0.9mv/V±20%
Rated Output	±0.1000mv/V
Non-linearity	±0.03%R.O.
Non-repeatability,Hysteresis	±0.03%R.O.
Creep in 2 Min.	±0.03%R.O./5Min
Temp. effect on Output	±0.05%R.O./10
Temp. effect on Zero	±2%R.O./10
Input Resistance	1130±20Ω
Ouput Resistance	1000±10Ω
Safe Overload	150%.R.C
Insulation Resistance	≥2000MΩ
Compensated Temp.Range	-10to+40
Operating Temp. Range	-20to+60
Excitation,Recommende	≤6VDC
Cable,Lenth	95mm(φ0.65)

Bảng 3.1: Thông số kỹ thuật của Loadcell

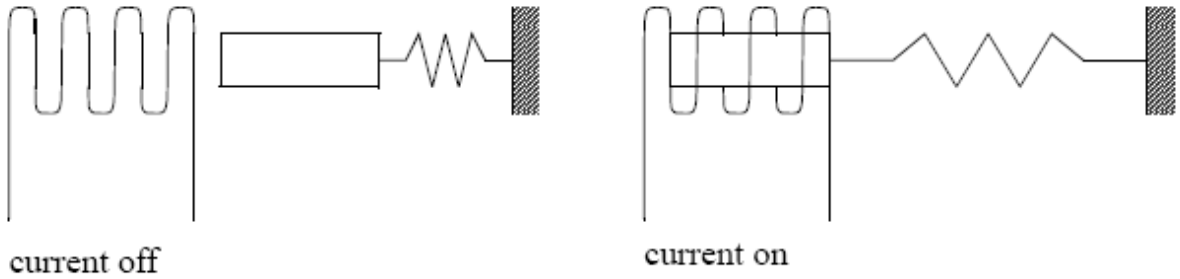
BÀI 4: CƠ CẤU CHẤP HÀNH

4.1 Giới thiệu.

Thiết bị chấp hành là các thiết bị nhận tín hiệu điều khiển từ ngõ ra của PLC và tác động đến chuyển động của các thiết bị cơ khí – quá trình chuyển hóa cơ/điện.

4.1.1 Solenoid.

Solenoid là một trong những thiết bị chấp hành thông dụng nhất. Nguyên lý hoạt động dựa vào chuyển động của lõi sắt (gọi là piston) trong lòng cuộn dây. Thông thường piston được giữa bên ngoài cuộn dây bằng lò xo. Khi có dòng điện chạy qua cuộn dây sẽ tạo ra từ trường hút piston và kéo nó vào trong cuộn dây. Piston có thể tạo ra lực tuyến tính. Hình vẽ 4.1 minh họa một solenoid.



Hình 4.1: Solenoid

Solenoid công nghiệp sử dụng điện áp 24VDC tạo ra dòng điện vài trăm mA.

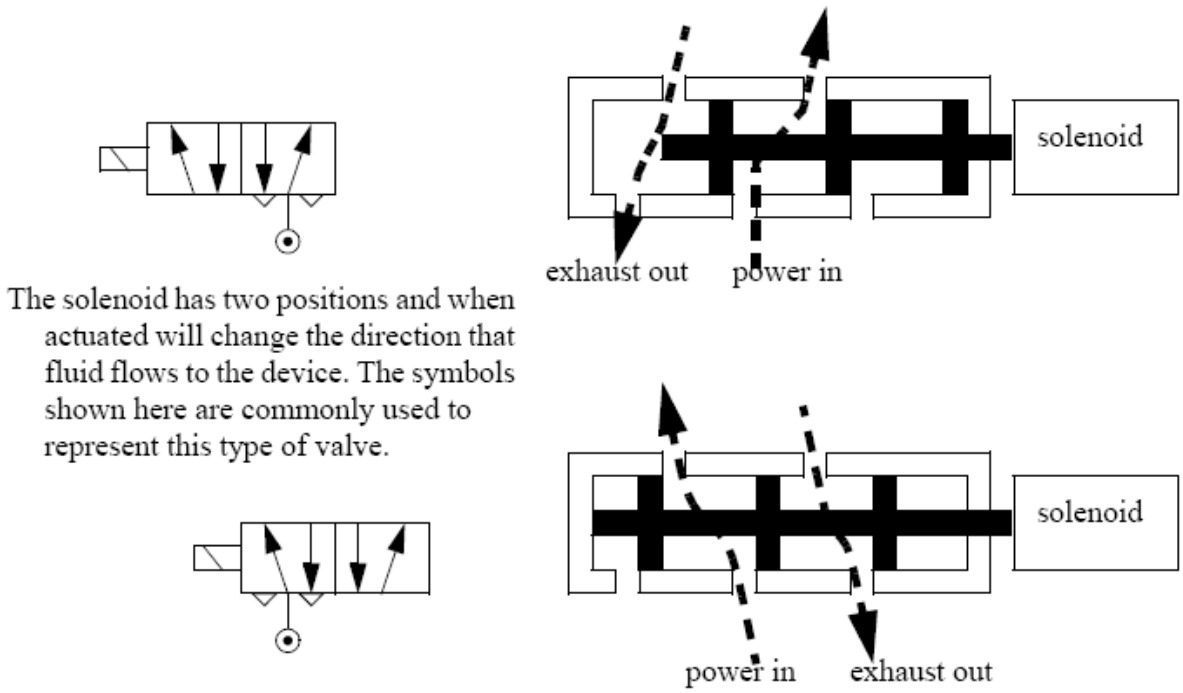
4.1.2 Valve.

Dòng chảy của chất lỏng và chất khí sẽ được điều khiển bằng các Valve, được đóng ngắt bằng Solenoid. Xem hình vẽ 4.1.

Solenoid được gắn vào một mặt phẳng. Khi được kích thích nó sẽ tác động đến ống bên trái. Phía trên của valve có 2 port nối với các thiết bị, như xy lanh chất lỏng. Phía dưới của valve có 1 đường áp suất ở tâm và 2 ống xả 2 bên.

Trong hình vẽ 4.1, công suất đi qua tâm đến port phải của xy lanh. Port trái của xy lanh cho phép thoát qua ống xả.

Trong hình vẽ dưới, Solenoid ở vị trí mới nên áp suất được cấp đến port trái, còn port phải dùng để xả.



Hình 4.2: Solenoid điều khiển Valve 5 port, 4 đường , 2 vị trí

4.1.2.1 Các loại Valve thường dùng bao gồm:

Valve có 2 đường thường đóng: 1 đường ra, 1 đường vào, bình thường valve đóng, khi có điện sẽ mở. Ứng dụng trong cho phép dòng chảy.

Valve có 2 đường thường mở: Dùng trong ngắt dòng chảy.

Valve có 3 đường thường đóng: 1 vào, 1 ra, 1 ống xả, bình thường port ra nối với ống xả, khi có điện port vào sẽ nối với port ra.

…

4.1.2.2 Khi chọn Valve cần quan tâm các vấn đề sau:

Kích cỡ ống dẫn.

Tốc độ dòng chảy.

Áp suất vận hành.

Điện áp Solenoid.

Thời gian đáp ứng. 5-150ms

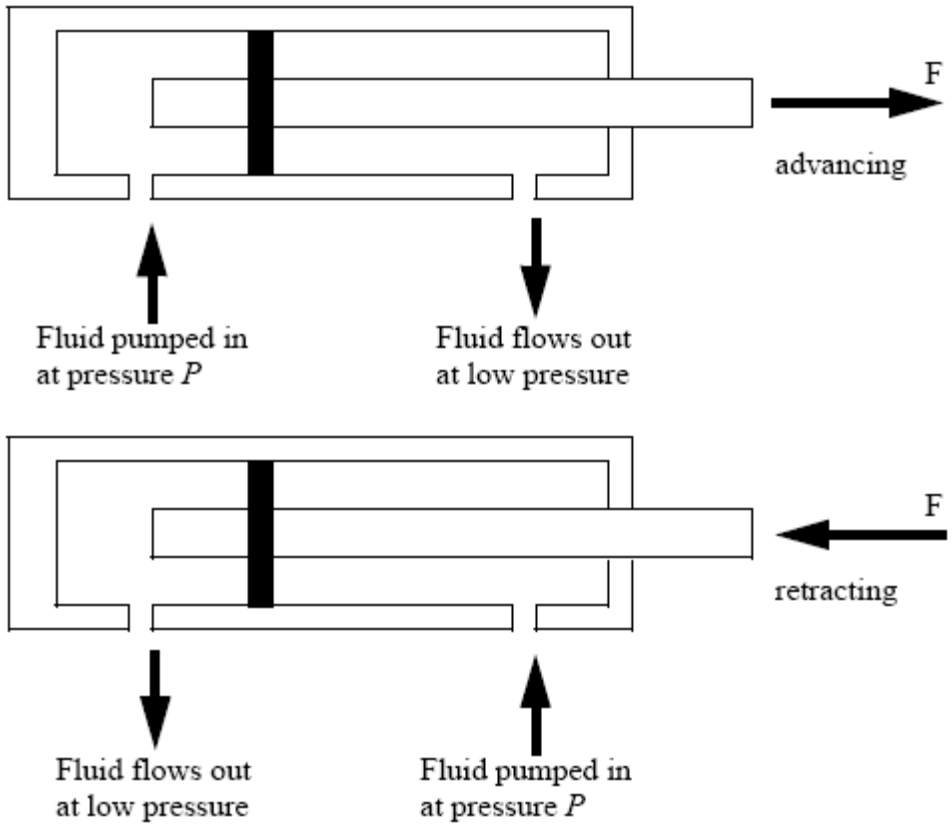
4.1.3 Xy Lanh.

Xy lanh dùng khí hoặc chất lỏng đã nén áp suất để tạo ra lực tuyến tính, như hình 4.3.

Chất lỏng được bơm vào 1 bên của xy lanh làm nó nở ra đẩy piston. Chất lỏng sẽ thoát tự do ở mặt bên kia. Lực do xy lanh tạo ra tỉ lệ với tiết diện mặt cắt ngang của xy lanh.

Xy lanh đơn tạo ra lực khi nở ra và có lò xo kéo piston lại.

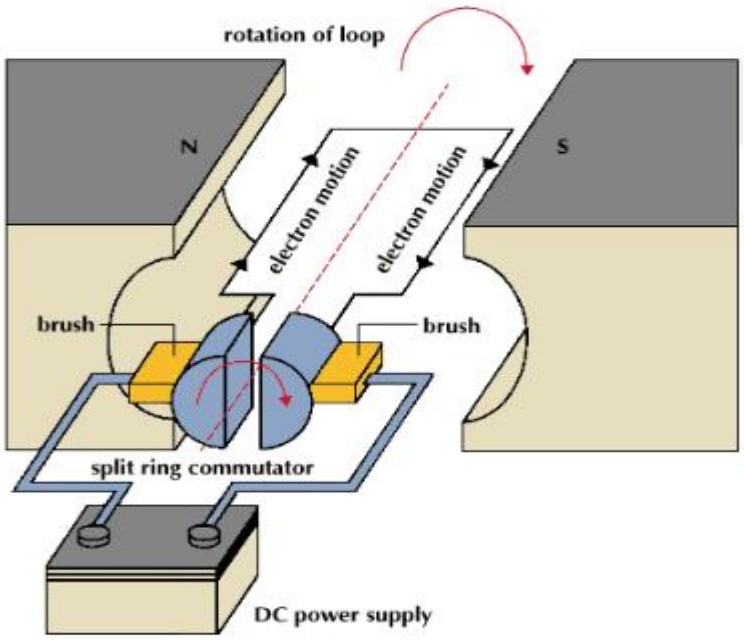
Xy lanh đôi tạo lực ở cả 2 hướng.



Hình 4.3: Mắt cắt của xylanh chất lỏng

4.1.4 Động cơ DC chổi Quét.

Động cơ DC có bộ dây quấn trên rotor quay bên trong startor. Khi có dòng điện chạy qua vòng dây sẽ tạo ra một từ trường, năng lượng cấp cho rotor qua bộ chuyển mạch và chổi quét., như hình 4.4.

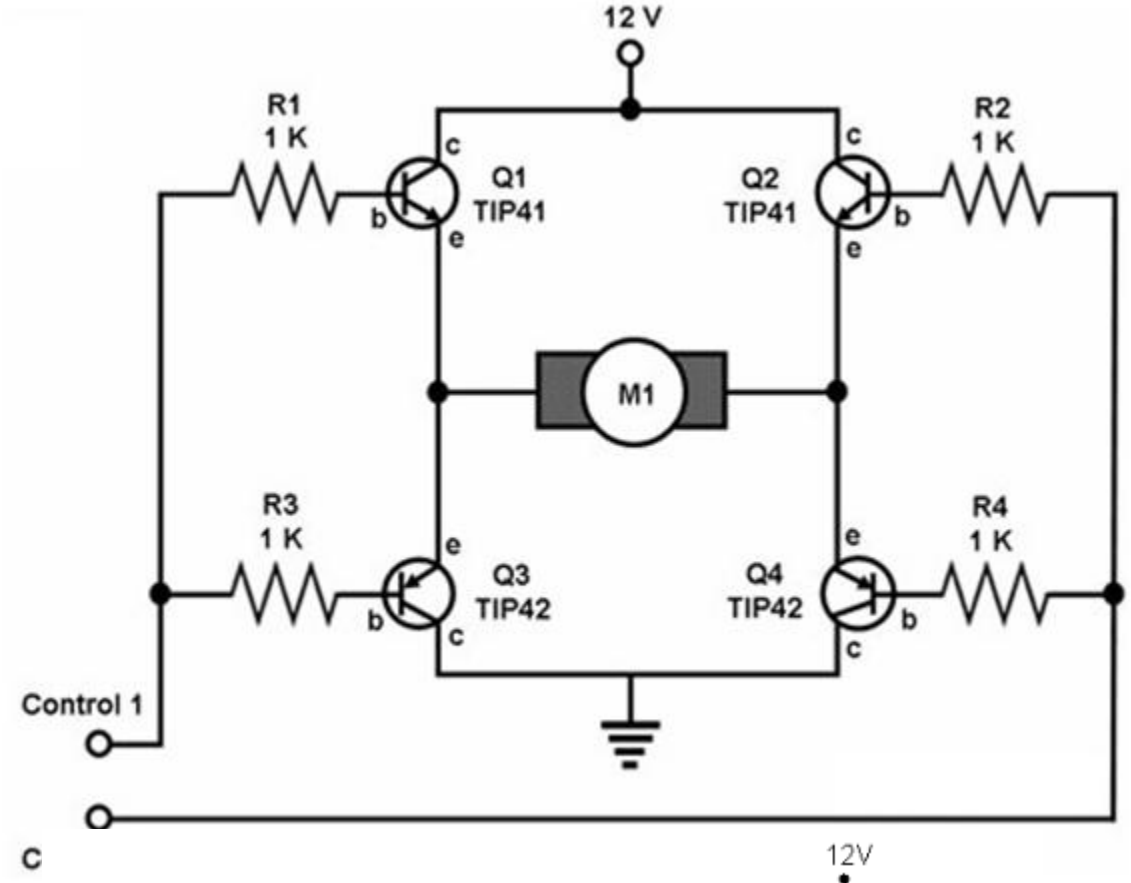


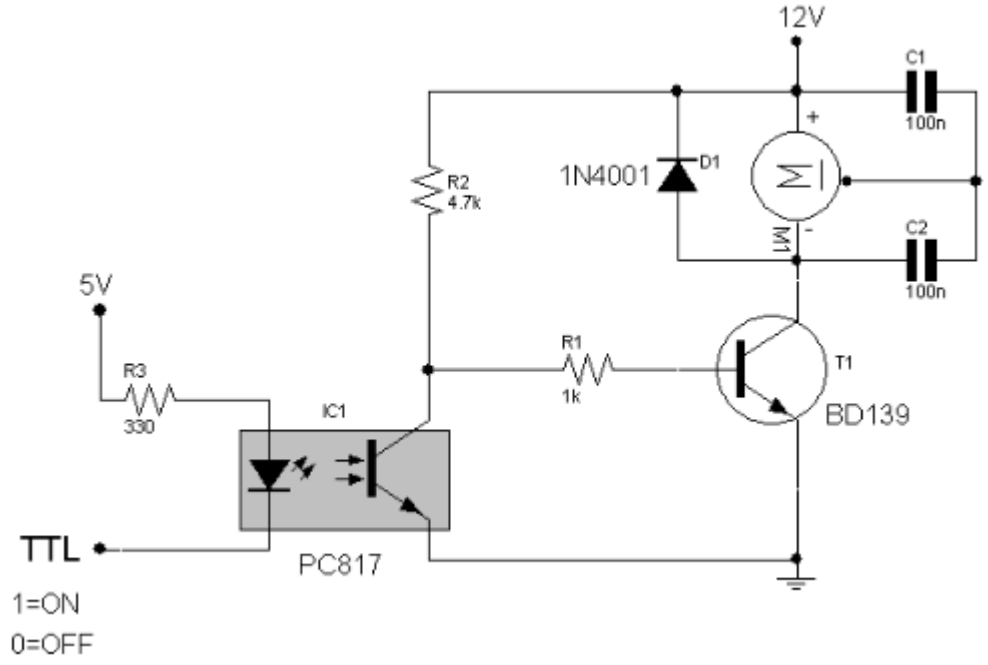
Hình 4.4: Mô hình động cơ DC

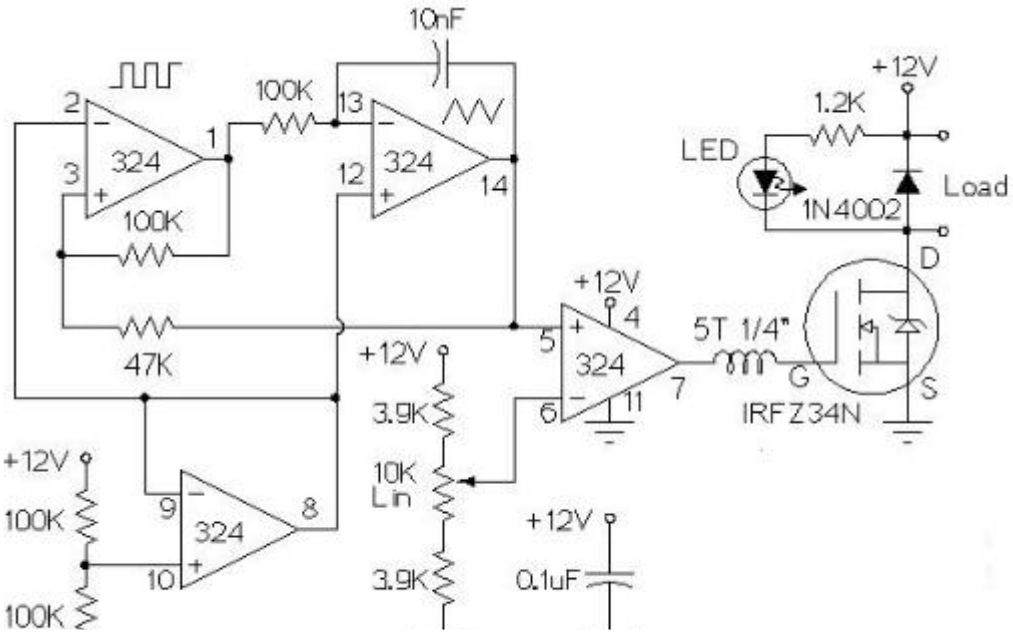
Để điều khiển tốc độ quay của động cơ DC thường có 2 phương pháp: Thay đổi điện áp cuộn kích từ (Từ trường của startor), Thay đổi dòng điện chạy qua cuộn dây phần ứng (Thay đổi điện áp đặt lên hai đầu cuộn dây phần ứng). Phương pháp sử dụng phổ biến nhất là PWM. Tuy nhiện tùy thuộc vào các yếu tố như:

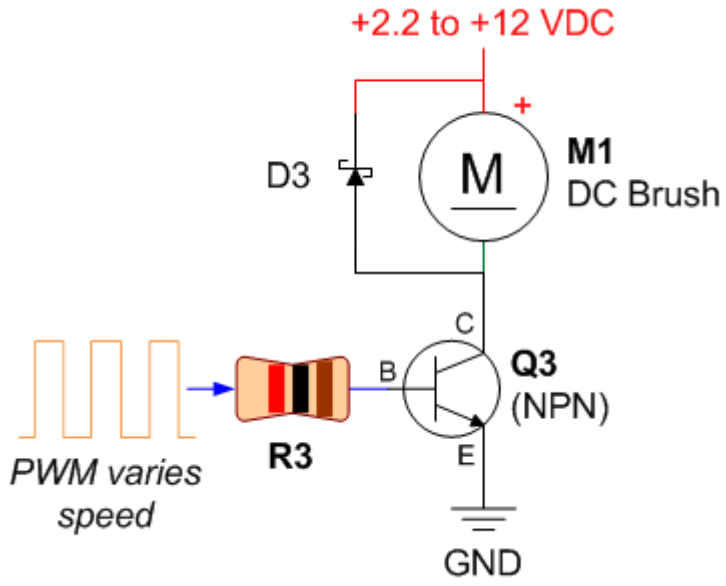
Bộ điều khiển, giá thành sản phẩm, yêu cầu công nghệ mà người sử dụng thiết kế bộ điều khiển nào cho phù hợp.

Một số mạch thường được sử dụng để điều khiển động cơ DC.



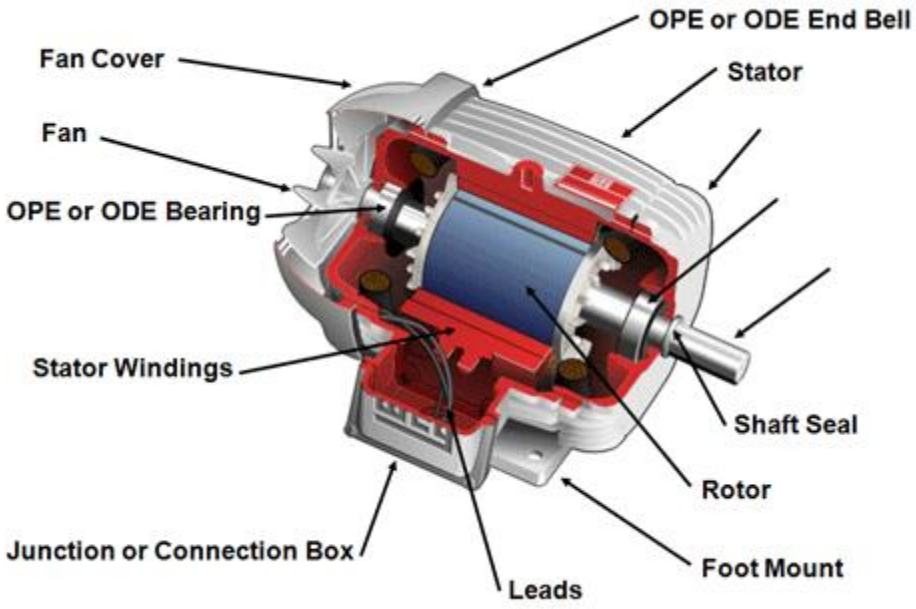



10nF
100K
324
100K
47K
+12V
1.2K
LED
1N4002
Load
D
5T 1/4"
G
S
IRFZ34N
3.9K
10K
Lin
3.9K
0.1uF
100K
100K


+2.2 to +12 VDC
+
D3
M
M1
DC Brush
C
B
Q3
(NPN)
E
R3
PWM varies
speed
GND

4.1.5 Động Cơ AC Đồng Bộ.

Động cơ đồng bộ có dây quấn trên Startor, Rotor lồng sóc. Lồng sóc là 1 lõi nhôm khi đặt vào từ trường thay đổi sẽ tạo ra trường ngược lại. khi cấp nguồn AC đến cuộn dây Startor sẽ tạo ra một từ trường AC. Lồng sóc sẽ tạo ra từ trường ngược lại và tạo ra moment xoắn làm động cơ quay.



Hình 4.5: Các bộ phận của động cơ không đồng bộ.

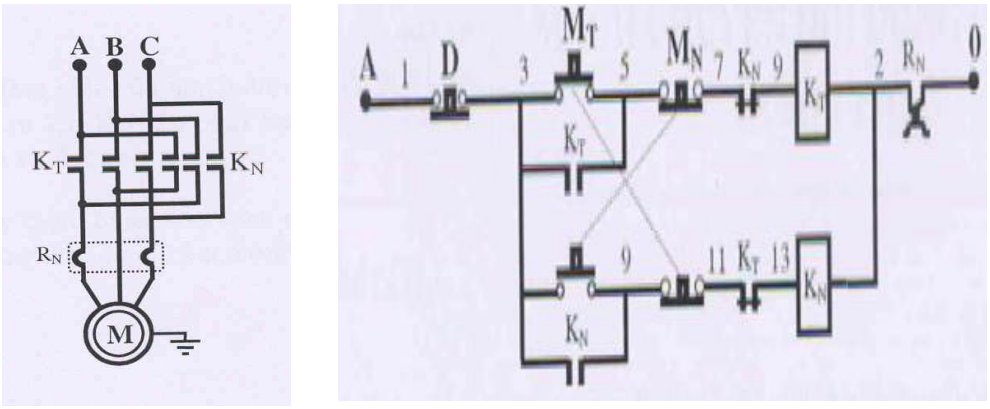
Động cơ gọi là đồng bộ vì có nó quay với tần số gần bằng tần số nguồn.

Thực tế, động cơ đạt moment quay max nhỏ hơn tốc độ đồng bộ.

Ví dụ: Động cơ có 2 cực có tốc độ đồng bộ 2x60x60/2 = 3600 vòng/phút nhưng chỉ đạt vận tốc 3520 vòng/phút.

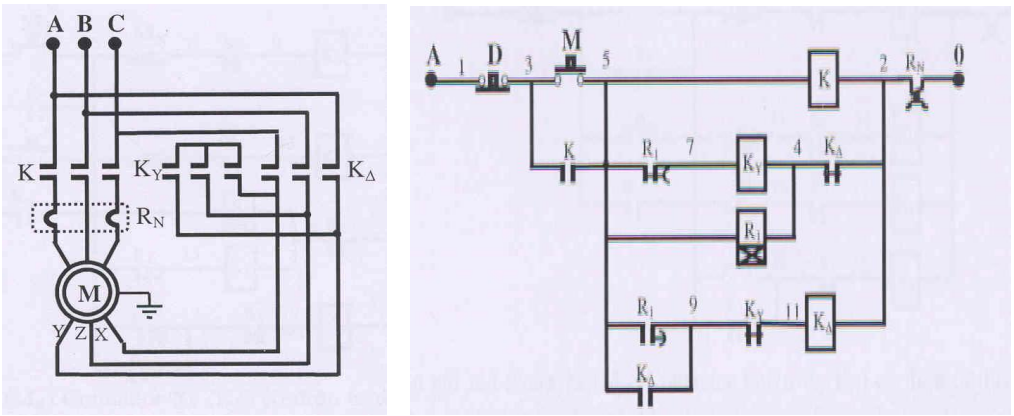
Ngày nay động cơ AC thường được ứng dụng rộng rãi trong công nghiệp. Trong lĩnh vực điều khiển, đối với những ứng dụng đơn giản thì động cơ AC thường được điều khiển hoạt động theo các kiểu sau:

4.1.5.1 Điều khiển ON-OFF, đảo chiều.



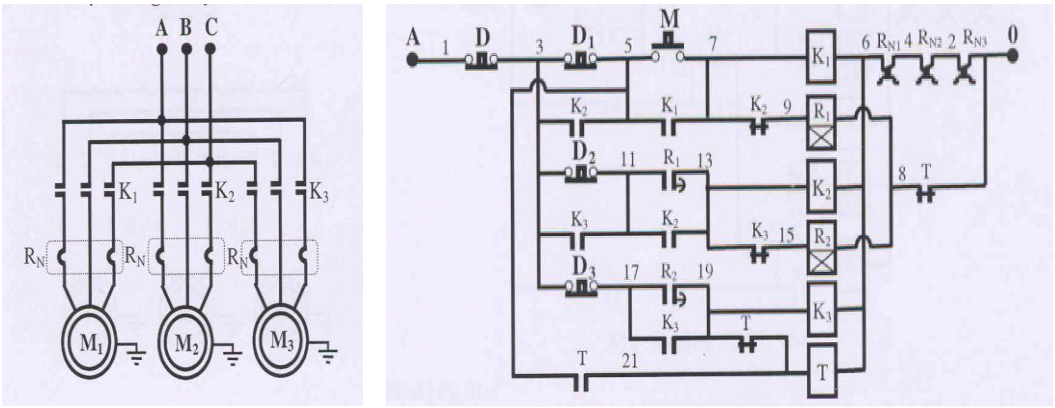
Hình 4.6: Mạch điều khiển ON-OFF, đảo chiều động cơ

4.1.5.2 Điều khiển khởi động sao – tam giác.



Hình 4.7: Mạch điều khiển đảo chiều sao – tam giác

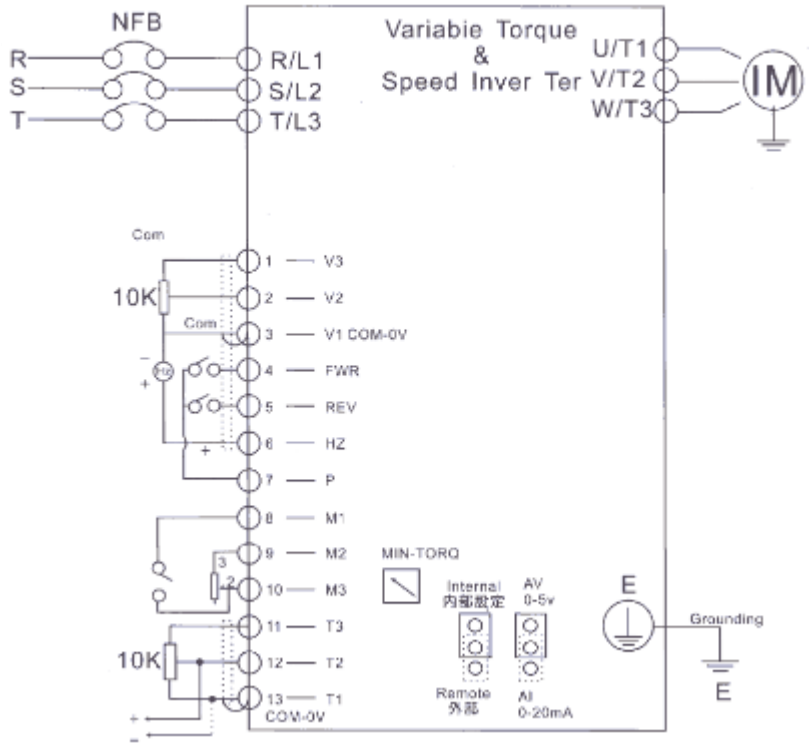
4.1.5.3 Điều khiển khởi động tuần tự.



Hình 4.8: Mạch điều khiển khởi động tuần tự

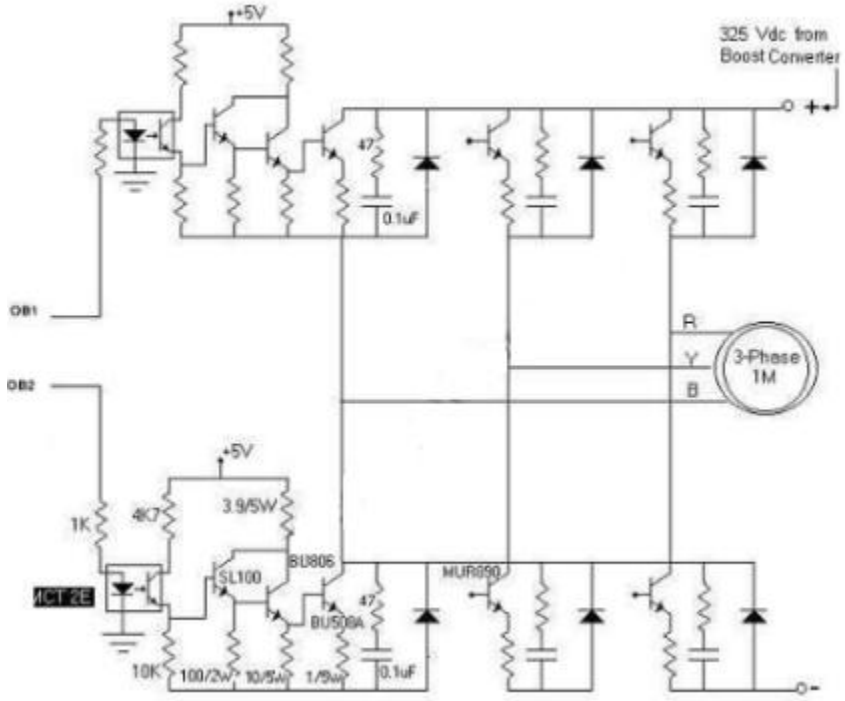
Đối với những ứng dụng đòi hỏi tính kỹ thuật cao thì phương pháp điều khiển được sử dụng phổ biến nhất là thay đổi tần số nguồn điện cấp cho động cơ.

4.1.5.4 Điều khiển dùng biến tần.



Hình 4.9: Mạch điều khiển tốc độ dùng biến tần

4.1.5.6 Điều khiển SCR hoặc Triac.

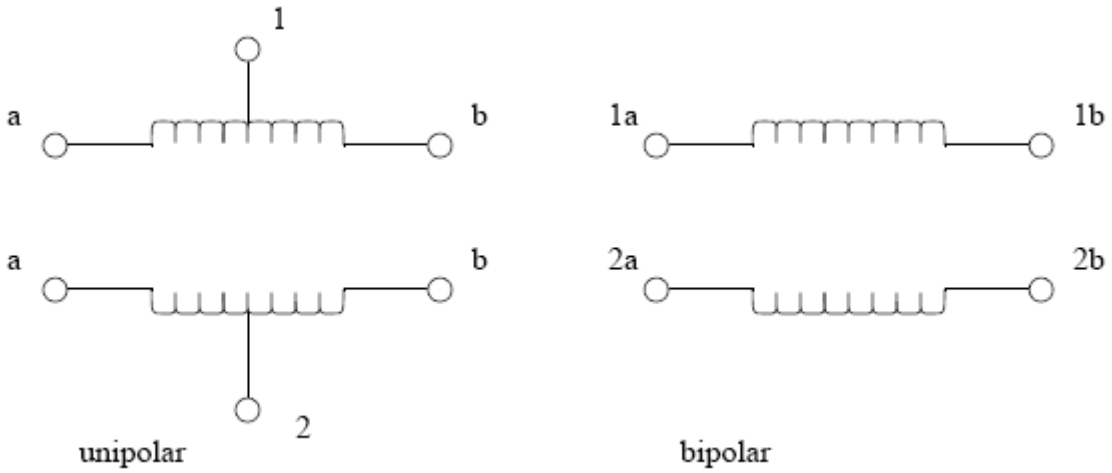


Hình 4.10: Mạch điều khiển tốc độ dùng Transistor

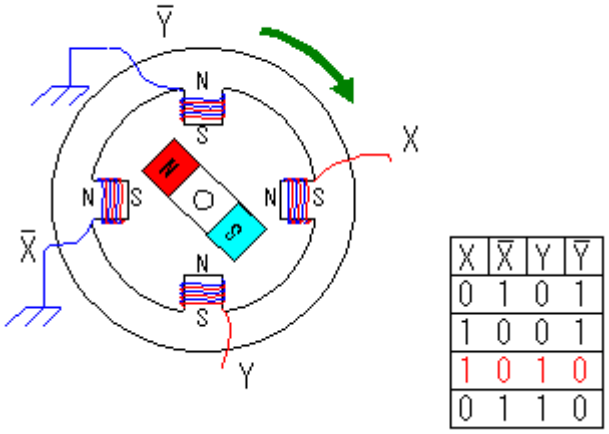
4.1.6 Động Cơ Bước.

Động cơ bước được thiết kế trong điều khiển định ví trí. Chúng di chuyển mỗi lần một bước, thường có góc quay là 1.8^0 – tương ứng 200 bước/vòng. Những loại khác có góc quay 2^0, 2.5^0, 5^0, 15^0, 30^0.

Có 2 loại động cơ bước: cực đơn và cực kép như hình vẽ 4.8.



Hình 4.11 : Sơ đồ dây quấn động cơ bước.



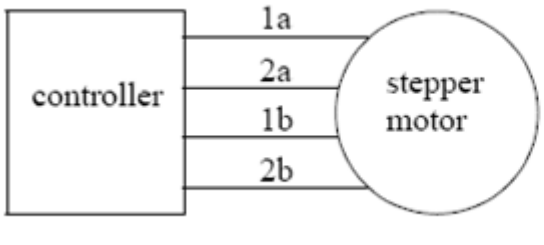
X	X	Y	Y
0	1	0	1
1	0	0	1
1	0	1	0
0	1	1	0

Hình 4.12 : Cấu trúc và cách điều khiển động cơ

Loại đơn cực sử dụng cuộn dây có nhánh rẽ ở tâm và dùng nguồn đơn.

Loại cực kép có cấu tạo đơn giản hơn nhưng phải dùng nguồn đôi, mạch điều khiển phức tạp hơn. Động cơ bước quay được nhờ điện áp lệch ở các đầu dây.

Các giá trị điện áp khác nhau cấp cho động cơ đơn cực minh họa ở hình 4.13



Step	1a	2a	1b	2b
1	1	0	1	0
2	0	1	1	0
3	0	1	0	1
4	1	0	0	1

Hình 4.13: Bảng mã điều khiển động cơ bước

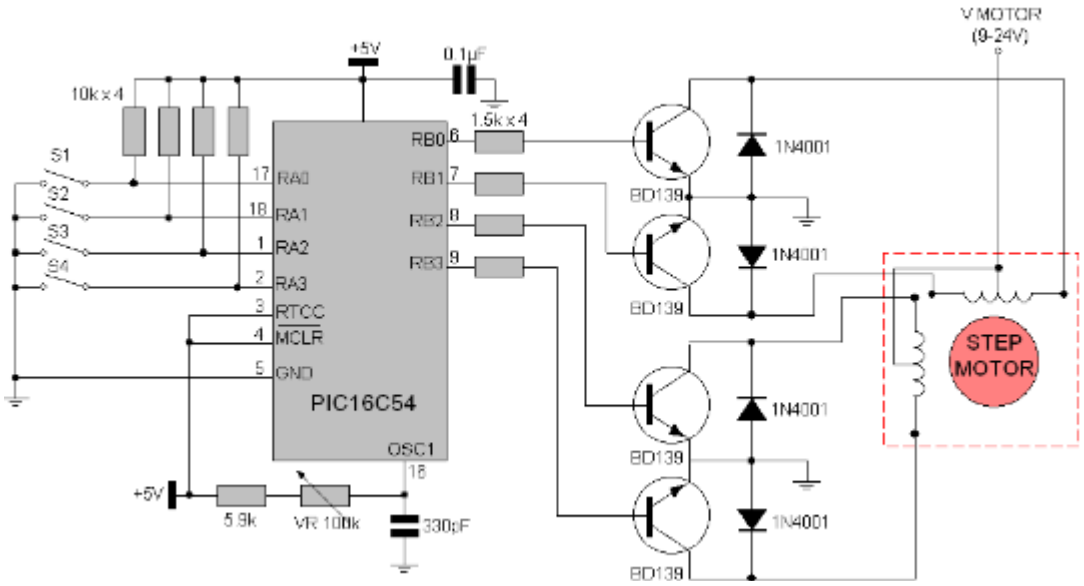
Giả sử ban đầu các cuộn dây có trạng thái như hàng 1. Để động cơ quay, ta phải thay đổi áp theo hàng 2, rồi hàng 3, hàng 4 và lặp lại. Nếu đảo thứ tự này động cơ sẽ quay theo chiều ngược lại. Động năng của động cơ và tải sẽ giới hạn tốc độ cực đại của các chuyển mạch đóng ngắt điện áp các đầu dây. Thông thường khoảng vài ngàn bước mỗi giây.

Khi không thay đổi, điện áp các đầu dây sẽ giữ động cơ ở 1 vị trí nào đó.

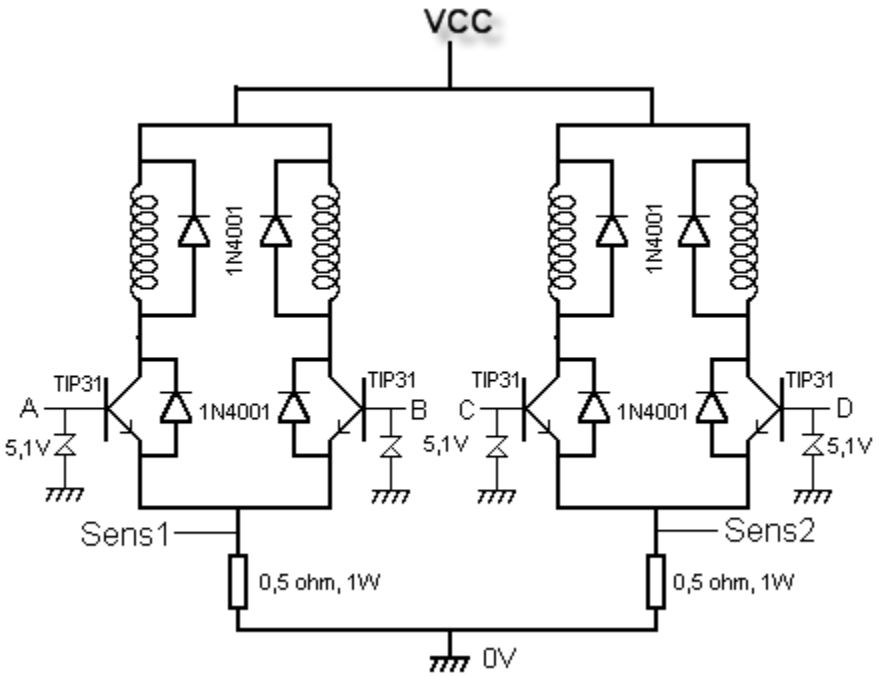
Động cơ bước không cần bộ phận hồi tiếp, trừ khi sử dụng trong các ứng dụng có độ tin cậy rất cao.

Động cơ bước sẽ bị lệch khi moment giữ bị vượt hoặc khi nó tăng tốc quá nhanh. Khi động cơ bị lệch nó sẽ quay một góc nào đó không xác định từ vị trí hiện tại, muốn xác định giá trị này cần có hệ thống hồi tiếp vị trí.

Do cấu trúc của động cơ bước cơ bản là giống nhau nên các mạch công suất điều khiển cũng có những nét giống nhau. Dưới đây trình bày một số mạch điều khiển động cơ bước thường được sử dụng.



Hình 4.14 : Điều khiển động cơ bước dùng PIC



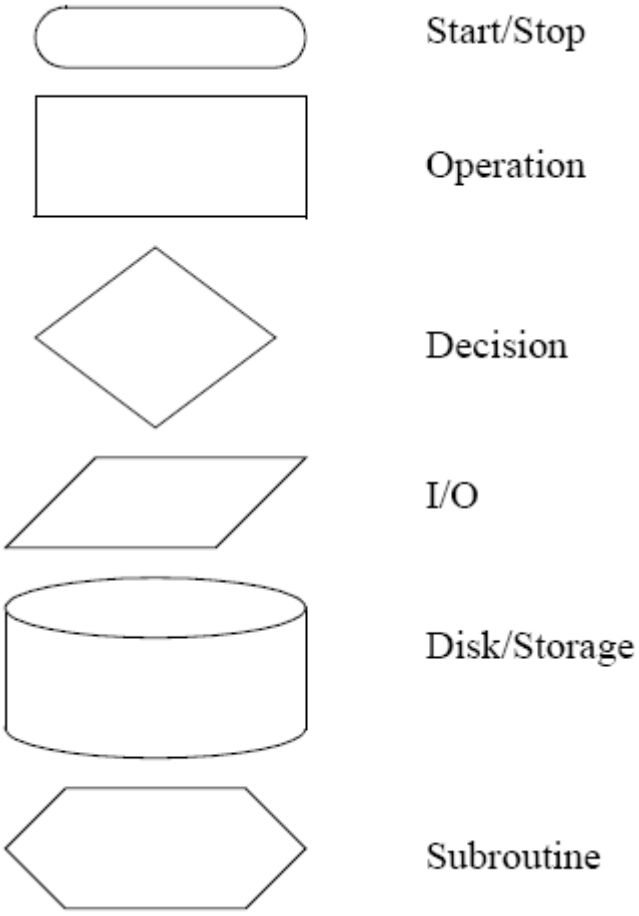
Hình 4.15: Mạch điều khiển dùng Transistor

BÀI 5: THIẾT KẾ CHƯƠNG TRÌNH THEO LƯU ĐỒ

5.1 Giới thiệu.

Lưu đồ là một sơ đồ mô tả toàn bộ quá trình xử lý của một hệ thống điều khiển. Nó giúp người lập trình kiểm tra tính khả thi của việc lập trình, nhanh chóng đưa ra những giải thuật để viết chương trình một cách nhanh chóng và hiệu quả. Một quá trình có các bước xử lý tuần tự sẽ thích hợp khi sử dụng lưu đồ để thiết kế chương trình. Các bước trong lưu đồ được thực hiện theo một trình tự đơn giản.

Các ký hiệu dùng trong lưu đồ bao gồm:



Hình 5.1: Ký hiệu dùng trong lưu đồ

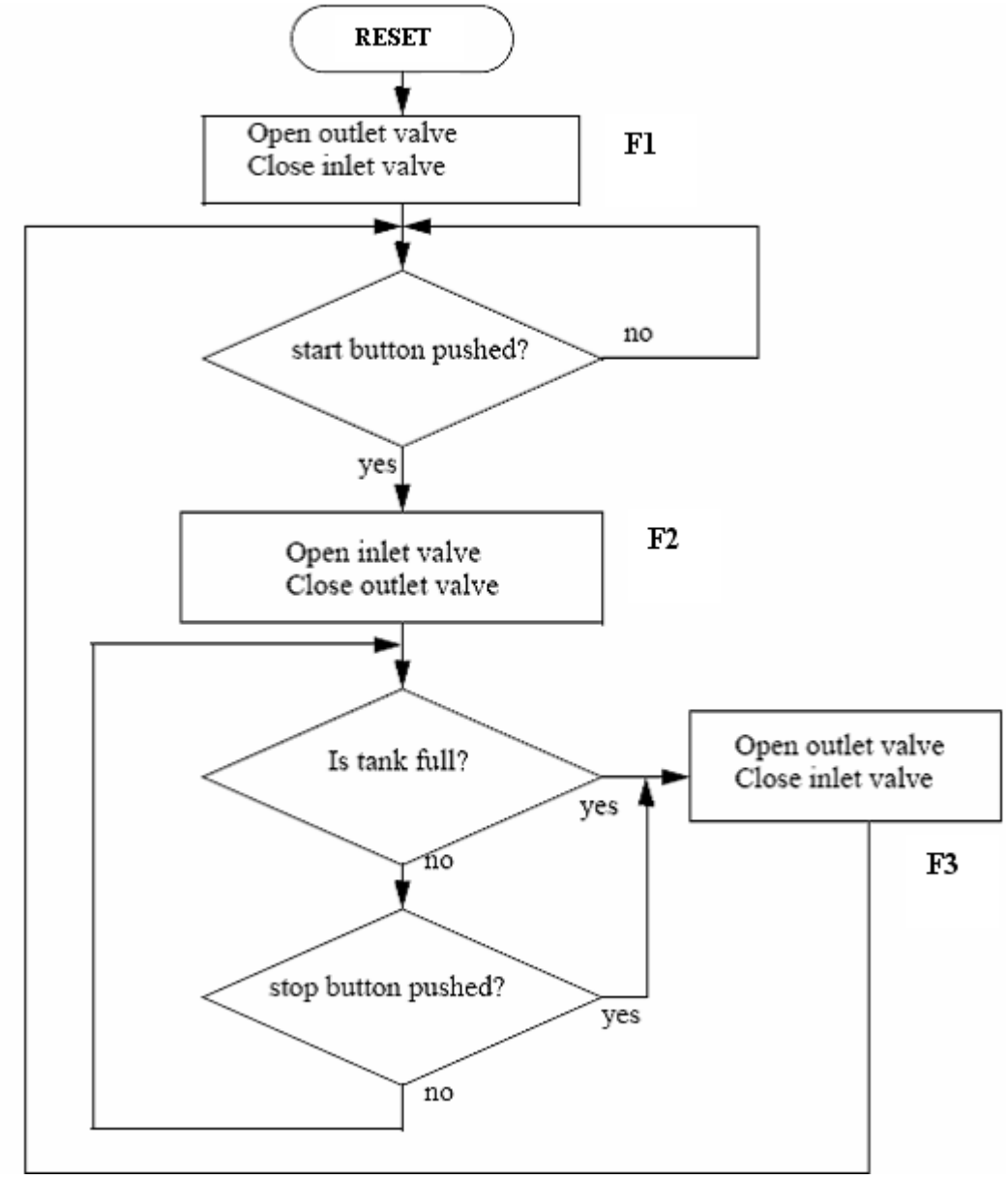
Trong các khối ở trên thì khối thực thi chương trình (Operation) và khối quyết định(Decision) thường được sử dụng nhiều nhất trong các ứng dụng lập trình của PLC, các khối khác chỉ được sử dụng trong một số trường hợp nhất định.

Các khối được nối với nhau bằng các mũi tên nhằm chỉ ra các bước thực hiện tuần tự.

Chương trình PLC luôn bắt đầu bằng khối Start, kế đến là các khối, kiểm tra, xử lý và đưa ra kết quả. Thường thì chương trình được thực hiện một cách tuần tự, liên tục. Tuy nhiên trong quá trình thực hiện, chương trình cũng kiểm tra khối Stop để dừng chương trình khi cần thiết.

5.2 Lưu đồ điều khiển.

Để thuận tiện cho việc tìm hiểu ta hãy xét một lưu đồ điều khiển bồn nước như hình 5.2.



Hình 5.2: Mô tả lưu đồ của hệ thống điều khiển một bồn nước.

Ở trạng thái RESET hệ thống thì nước được xã ra bởi Outlet Valve và đóng ngõ vào bởi Inlet Valve.

Khi nhấn nút Start, bồn bắt đầu cho nước vào và tắt đường chảy ra.

Khi bồn đầy nước, hoặc nhấn nút Stop sẽ mở đường chảy ra và đóng đường chảy vào. Quá trình bắt đầu từ sau khi Reset hệ thống. Đầu tiên là mở van ngõ

ra và đóng van ngõ vào. Tiếp theo, khối Decision sẽ chờ xem có nút nào được nhấn không. Nếu có nút được nhấn, theo nhánh Yes sẽ mở van vào và đóng van ra. Tiếp theo đến một vòng gồm hai khối Decision để chờ đến khi bồn đầy hoặc nhấn nút Stop. Nếu một trong hai trường hợp xảy ra thì đóng van vào và mở van ra. Hệ thống sẽ quay lại để kiểm tra trạng thái nút Start lần nữa.

5.2.1 Những đặc điểm cần chú ý khi viết lưu đồ:

- Mô tả được quá trình hoạt động của hệ thống.
- Xác định các hoạt động chính, vẽ thành các khối.
- Xác định tuần tự vận hành, vẽ bằng các mũi tên.
- Mỗi khối thực thi trong lưu đồ phải được đặt bằng một tên nhất định.
- Khi tuần tự này thay đổi thì sử dụng các khối Decision để rẽ nhánh.
- Giải thích hoạt động theo lưu đồ để sửa chữa, bổ sung và hoàn thiện lưu đồ trước khi bắt đâu viết chương trình.

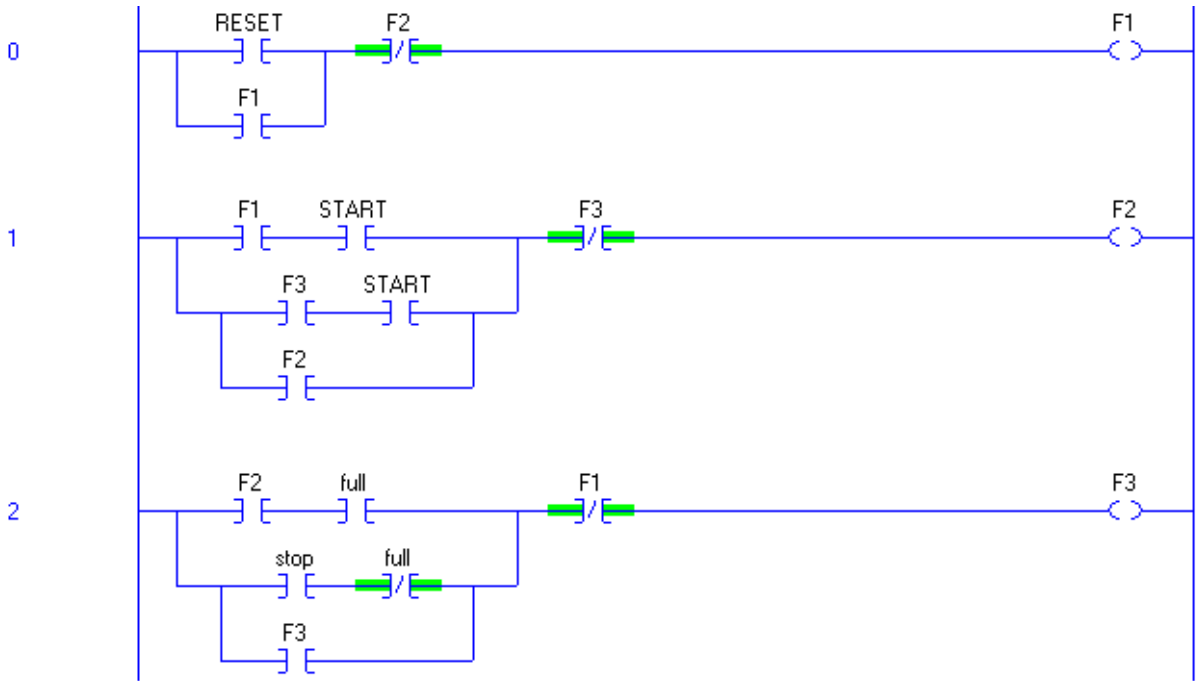
Một lưu đồ có thể được viết bằng nhiều ngôn ngữ khác nhau. Việc lựa chọn ngôn ngữ nào để viết là tùy thuộc vào sự thành thạo của người lập trình. Trong phạm vi tài liệu này, ngôn ngữ LAD được sử dụng để viết chương trình cho lưu đồ.

5.2.2 Một số đặc điểm cần lưu ý khi viết chương trình theo lưu đồ.

- Quá trình chỉ được chuyển từ khối này đến khối khác chỉ được thực hiện khi thỏa mãn một số yêu cầu nhất định.
- Khi chuyển đến khối hiện tại thì phải xóa trạng thái ở khối trước đó.
- Chương trình thường được viết thành 2 đoạn: Đoạn chương trình chuyển trạng thái giữa các khối và đoạn chương trình thực thi trong từng khối để tác động ngõ ra.

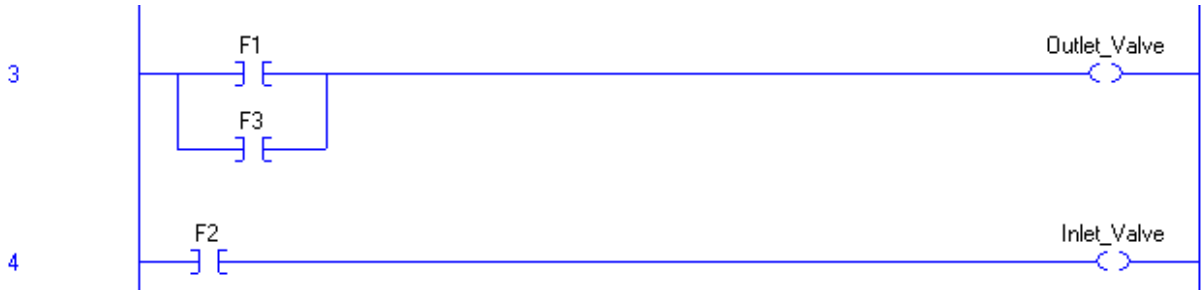
Dưới đây sẽ trình bày cách viết chương trình cho lưu đồ điều khiển bồn nước.

5.2.3 Đoạn chương trình chuyển trạng thái giữa các khối cho lưu đồ H 5.2.



Hình 5.3: Chương trình chuyển trạng thái giữa các khối.

5.2.4 Đoạn chương trình thực thi của các khối để tác động ngõ ra.



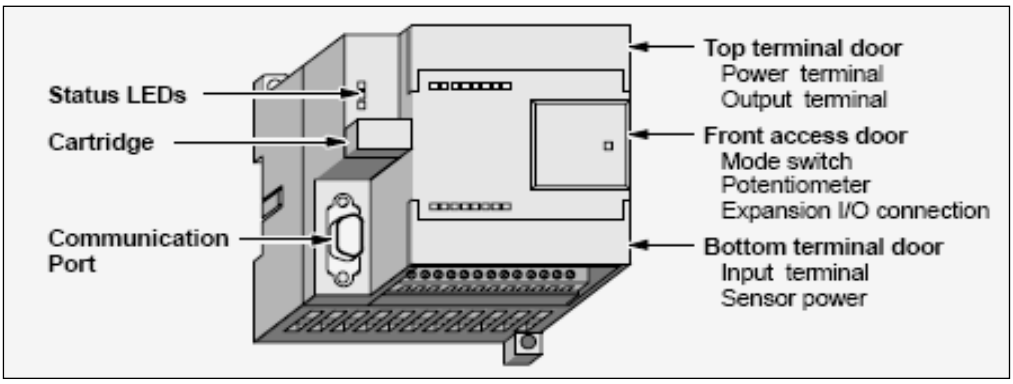
Hình 5.4: Chương trình tác động tải ngõ ra.

BÀI 6: PLC S7 – 200

6.1 Giới thiệu về PLC S7 200.

S7-200 là thiết bị điều khiển logic lập trình loại nhỏ của hãng Siemens (Đức), có cấu trúc theo kiểu module và có các module mở rộng. Các module này đươc sử dụng cho nhiều ứng dụng lập trình khác nhau.

Hình dạng bên ngoài của PLC S7-200 được mô tả như hình 6.1.



Hình 6.1: Bộ điều khiển lập trình S7-200

6.1.1 Thông số kỹ thuật của PLC S7-200 (CPU 22X).

Các loại CPU khác nhau thường có vùng nhớ, module mở rộng và một số chức năng khác nhau. Bảng 6.1 trình bày sự khác biệt này.

Đặc trưng	CPU 221	CPU 222	CPU 224	CPU 226
Kích thước(mm)	90x80x62	90x80x62	120.5x80x62	190x80X62
Bộ nhớ chương trình	2048 words	2048words	4096words	4096words
Bộ nhớ dữ liệu	1024 words	1024words	2560words	2560words
Cổng logic vào	6	8	14	24
Cổng logic ra	4	6	10	16
Modul mở rộng	None	2	7	7
Digital I/O cực đại	128/128	128/128	128/128	128/128
Analog I/O cực đại	None	16In/16Out	32In/32Out	32In/32Out
Bộ đếm (Counter)	256	256	256	256
Bộ định thì (Timer)	256	256	256	256
Tốc độ thực thi lệnh	0.37µs	0.37µs	0.37µs	0.37µs
Khả năng lưu trữ khi mất điện	50 giờ	50 giờ	190 giờ	190 giờ

Bảng 6.1: Thông số kỹ thuật của PLC S7200

6.1.2 Các đèn báo.

SF (System Failure): Đèn đỏ SF báo hiệu hệ thống bị hỏng.

RUN: Đèn xanh RUN chỉ định PLC đang ở chế độ làm việc và thực hiện chương trình được nạp vào trong máy.

STOP: Đèn vàng STOP chỉ định rằng PLC đang ở chế độ dừng chương trình.

6.1.3 Tín hiệu ngõ vào.

Điện áp mức logic 1: 15-30VDC, dòng nhỏ nhất 4mA; 35VDC ở thời gian tức thời 500ms.

Trạng thái mức logic 1 chuẩn: 24 VDC, 7mA.

Trạng thái mức logic 0: Tối đa 5 VDC, 1mA.

Đáp ứng thời gian lớn nhất ở các chân I0.0 đến I1.5: có thể chỉnh từ 0,2 đến 8,7ms. Thời gian mặc định 0,2ms.

Sự cách ly về quang 500VAC.

6.1.4 Các ngõ ra.

Kiểu đầu ra: Relay hoặc Transistor cấp dòng điện.

Điện áp mức 1: 24.4 đến 28.8VDC.

Dòng tải tối đa: 2A/ điểm; 8A/common.

Quá dòng: 7A với contact đóng.

Điện trở cách ly: nhỏ nhất 100 MΩ.

Thời gian chuyển mạch: tối đa 10ms.

Thời gian sử dụng: 10.000.000 lần với công tắc cơ khí; 100.000 lần với tốc độ tải.

Điện trở công tắc: tối đa 200 mΩ.

Chế độ bảo vệ ngắn mạch: không có.

6.1.5 Nguồn cung cấp.

Điện áp nguồn cung cấp: 20.4 đến 24.8VDC

Dòng vào max load: 900mA tại 24VDC

Cách ly điện ngõ vào: không có

Thời gian duy trì khi mất nguồn: 10ms ở 24 VDC

Cầu chì bên trong: 2A, 250V

6.1.6 Chế độ làm việc.

PLC có 3 chế độ làm việc:

RUN: Cho phép PLC thực hiện chương trình từ bộ nhớ, PLC sẽ chuyển từ RUN sang STOP nếu trong máy có sự cố hoặc trong chương trình gặp lệnh STOP.

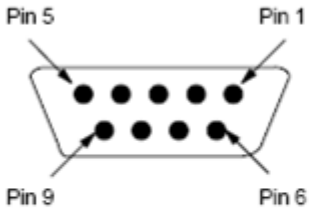
STOP: Cưỡng bức PLC dừng chương trình đang chạy và chuyển sang chế độ STOP.

TERM: Cho phép máy lập trình tự quyết định chế độ hoạt động cho PLC ở chế độ RUN hoặc STOP.

6.1.7 Cổng truyền thông.

S7-200 sử dụng cổng truyền thông nối tiếp RS485 với đầu 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 (Point to Point Interface) là 9600 bauds. Tốc độ truyền của PLC theo kiểu tự do là 300 ÷38.400 bauds.

Sơ đồ chân cổng truyền thông vẽ trên hình 6.2.

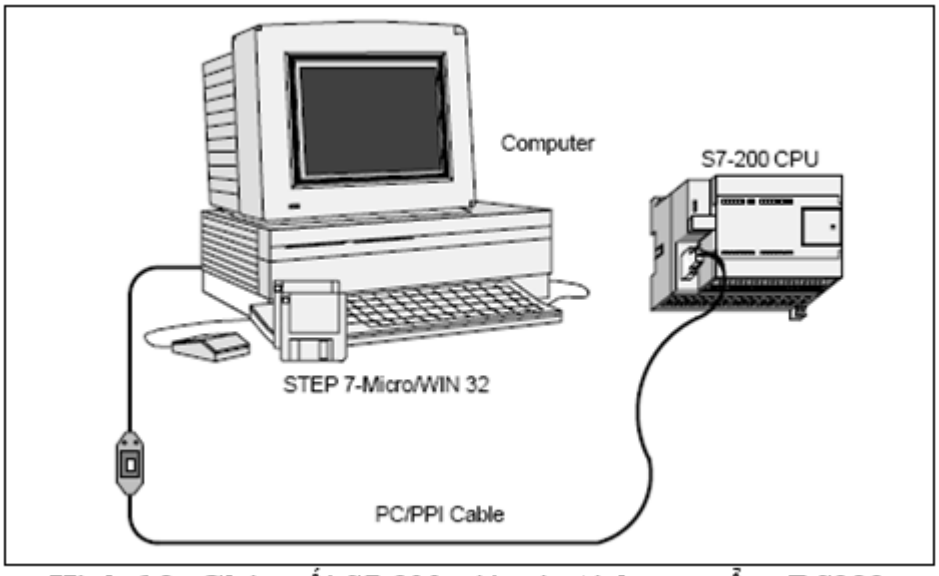


Chân	Chức năng
1	GND
2	24 VDC
3	Tín hiệu A của RS485 (RxD/TxD+)
4	RTS (theo mức TTL)
5	GND
6	+5 VDC
7	Nguồn cấp 24 VDC 120mA max
8	Tín hiện B RS485 (RxD/TxD-)
9	Chọn lựa cách giao tiếp

Hình 6.2: Sơ đồ chân của cổng truyền thông.

6.1.8 Kết nối S7 200 với máy tính PC.

Để ghép S7-200 với các máy tính PC qua cổng RS-232 cần có cáp nối PC/PPI với bộ chuyển đổi từ RS232 sang RS485, theo hình vẽ 6.3.



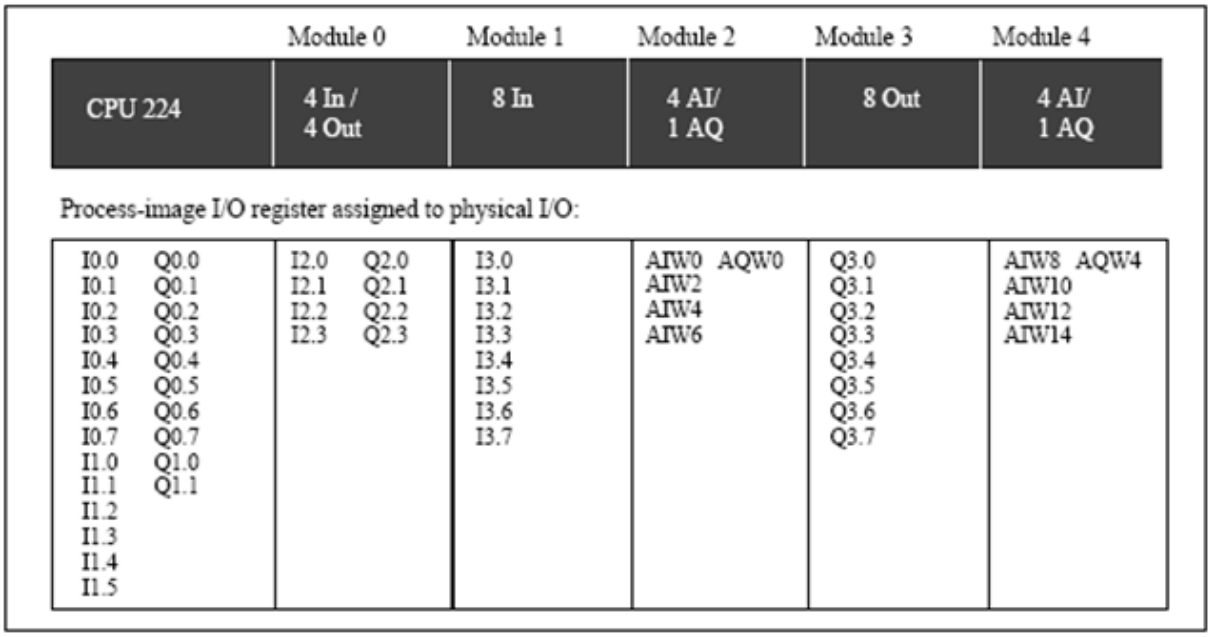
Hình 6.3: Ghép nối S7-200 với máy tính qua cổng RS232

6.1.9 Các module mở rộng.

Có thể mở rộng ngõ vào/ra của PLC bằng cách ghép nối thêm vào PLC các module mở rộng về phía bên phải của CPU. CPU 224 có thể ghép nhiều nhất 7 module theo hình 6.4.

Các module mở rộng Digital hay Analog đều chiếm chỗ trong bộ đệm, tương ứng với số đầu vào/ ra của các module.

Ngoài ra còn có thể thêm các module khác dùng để kết nối mạng Profibus và AS-Interface.



Hình 6.4: Cách xác định địa chỉ các modue mở rộng

6.2 Nguyên lý hoạt động của PLC.

6.3.1 Đơn vị xử lý trung tâm CPU.

CPU điều khiển các hoạt động bên trong PLC. Bộ xử lý sẽ đọc và kiểm tra chương trình được chứa trong bộ nhớ, sau đó sẽ thực hiện tuần tự từng lệnh trong chương trình, kết quả làm đóng hay ngắt các ngõ ra. Các trạng thái ngõ ra ấy được đưa đến tác động các thiết bị bên ngoài. Toàn bộ các hoạt động này đều phụ thuộc vào chương trình điều khiển được lưu trữ trong bộ nhớ.

6.3.2 Hệ thống BUS.

Hệ thống Bus là tuyến truyền tín hiệu, hệ thống gồm nhiều đường tín hiệu song song:

Address Bus: Bus địa chỉ dùng để truyền địa chỉ đến các Module khác nhau.

Data Bus: Bus dùng để truyền dữ liệu.

Control Bus: Bus điều khiển dùng để truyền các tín hiệu định thời và điều khiển đồng bộ các hoạt động trong PLC

Trong PLC dữ liệu được trao đổi giữa bộ vi xử lý và các module vào ra thông qua Data Bus. Address Bus và Data Bus gồm 8 đường, ở cùng thời điểm cho phép truyền đồng thời 8 bit của 1 byte, còn gọi là truyền song song 8 bit.

Nếu một module vào nhận được địa chỉ của nó trên Address Bus, nó sẽ chuyển tất cả trạng thái ngõ vào của nó vào Data Bus. Nếu một địa chỉ byte của 8 ngõ ra xuất hiện trên Address Bus, modul ra tương ứng sẽ nhận được dữ liệu từ Data bus. Control Bus sẽ chuyển các tín hiệu điều khiển vào chu trình hoạt động của PLC.

Hệ thống Bus sẽ làm nhiệm vụ trao đổi thông tin giữa CPU, bộ nhớ và I/O. Bên cạnh đó, CPU được cung cấp một xung Clock có tần số từ 1÷8 MHZ. Xung này quyết định tốc độ hoạt động của PLC và cung cấp các yếu tố về định thời, đồng hồ của hệ thống.

6.3.3 Bộ nhớ.

PLC thường sử dụng bộ nhớ trong các trường hợp:

- Làm bộ lưu trữ tạm thời các bảng trạng thái I/O.
- Làm bộ đệm trạng thái các chức năng trong PLC như định thời, đếm, thanh ghi.

Mỗi lệnh của chương trình có một vị trí riêng trong bộ nhớ, tất cả mọi vị trí trong bộ nhớ đều được đánh số, những số này chính là địa chỉ trong bộ nhớ .

Địa chỉ của từng ô nhớ sẽ được trỏ đến bởi một bộ đếm địa chỉ ở bên trong bộ vi xử lý. Bộ vi xử lý sẽ tăng giá trị bộ đếm này lên một trước khi xử lý lệnh tiếp theo. Với một địa chỉ mới, nội dung của ô nhớ tương ứng sẽ xuất hiện ở đầu ra, quá trình này được gọi là quá trình đọc.

Bộ nhớ bên trong PLC được tạo bởi các vi mạch bán dẫn, mỗi vi mạch này có khả năng chứa 2000 ÷ 16000 dòng lệnh, tùy loại vi mạch. Trong PLC các bộ nhớ như RAM, EEPROM đều được sử dụng.

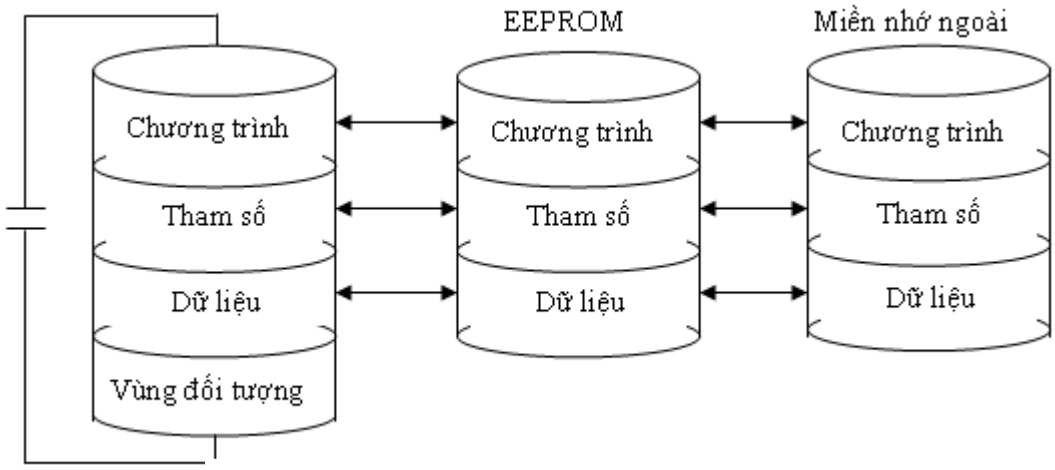
RAM (Random Access Memory) có thể nạp chương trình, thay đổi hay xoá bỏ nội dung bất kỳ lúc nào. Nội dung của RAM sẽ bị mất nếu nguồn điện nuôi bị mất . Để tránh tình trạng này các PLC đều được trang bị một pin khô, có khả năng cung cấp năng lượng dự trữ cho RAM từ vài tháng đến vài năm. Trong thực tế RAM được dùng để khởi tạo và kiểm tra chương trình. Khuynh hướng hiện nay dùng CMOSRAM nhờ khả năng tiêu thụ năng lượng thấp và tuổi thọ lớn.

EEPROM (Electrically Eraseable Programmable Read Only Memory) là bộ nhớ mà người sử dụng bình thường chỉ có thể đọc chứ không ghi nội dung vào được. Nội dung của EEPROM không bị mất khi mất nguồn, nó được gắn sẵn trong máy, đã được nhà sản xuất nạp và chứa hệ điều hành sẵn. Nếu người sử dụng không muốn mở rộng bộ nhớ thì chỉ dùng EEPROM gắn bên trong PLC.

Ngoài ra PLC còn cho phép gắn thêm bộ nhớ mở rộng như RAM, EPROM ở khe Cartridge.

6.3 Cấu trúc bộ nhớ.

Bộ nhớ của S7-200 được chia thành 4 vùng có một tụ điện làm nhiệm vụ duy trì dữ liệu trong một khoảng thời gian nhất định khi mất nguồn. Bộ nhớ S7-200 có tính năng động cao, có thể đọc ghi được trong toàn vùng, ngoại trừ các bit nhớ đặc biệt SM (Special memory) chỉ có thể truy nhập để đọc. Hình vẽ 6.4 mô tả bộ nhớ trong và ngoài của PLC.



Hình 6.4: Các vùng nhớ của PLC S7 200

Vùng chương trình: Miền bộ nhớ được sử dụng để lưu trữ các lệnh chương trình.

Vùng tham số: Miền lưu trữ các tham số như từ khóa, địa chỉ trạm,… cũng giống như vùng chương trình.

Vùng dữ liệu: Được sử dụng để cất các dữ liệu của chương trình bao gồm các kết quả các phép tính, bộ đệm truyền thông…

Vùng dữ liệu là một miền nhớ động. Nó có thể được truy cập theo từng bit, từng byte, từng từ đơn, hoặc theo từng từ kép và được sử dụng làm miền lưu trữ dữ liệu cho các thuật toán, hàm truyền thông, lập bảng, hàm dịch chuyển, xoay vòng thanh ghi, con trỏ địa chỉ,…

Ghi các dữ liệu kiểu bảng bị hạn chế rất nhiều vì các dữ liệu này thường chỉ được sử dụng theo những mục đích nhất định.

Vùng dữ liệu lại được chia thành những miền nhớ nhỏ với các công dụng khác nhau. Chúng được ký hiệu bằng các chữ cái đầu của tên tiếng Anh, đặc trưng cho công dụng riêng của chúng như sau:

V: Variable memory

I: Input image register

O: Output image register

M: Internal memory bits

SM: Special memory bits

Vùng đối tượng: Timer, counter, bộ đếm tốc độ cao và các cổng vào/ra tương tự được đặt trong vùng nhớ cuối cùng.

Vùng đối tượng được sử dụng để lưu trữ dữ liệu cho các đối tượng lập trình như các giá trị tức thời, giá trị đặt trước của bộ đếm Counter, bộ định thời Timer. Dữ liệu kiểu đối tượng bao gồm các Timer, Counter, các bộ đếm tốc độ cao, bộ đệm vào/ra tương tự và các thanh ghi Accumulator(AC).

Kiểu dữ liệu đối tượng bị hạn chế rất nhiều vì các dữ liệu kiểu đối tượng chỉ được ghi theo mục đích cần sử dụng đối tượng đó.

Phân chia vùng nhớ và các toán hạng tương ứng cho từng loại PLC cho ở bảng 6.2:

Vùng nhớ		CPU 221	CPU 222	CPU 224	CPU 226
Vùng dữ liệu	V	V0.0÷V2047.7	V0.0÷V2047.7	V0.0÷V5119.7	V0.0÷V5119.7
	I	I0.0÷I15.7	I0.0÷I15.7	I0.0÷I15.7	I0.0÷I15.7
	Q	Q0.0÷Q15.7	Q0.0÷15.7	Q0.0÷Q15.7	Q0.0÷Q15.7
	M	M0.0÷M31.7	M0.0÷M31.7	M0.0÷M31.7	M0.0÷M31.7
	SM	SM0.0÷SM179.7	SM0.0÷SM179.7	SM0.0÷SM179.7	SM0.0÷SM179.7
	S	S0.0÷S31.7	S0.0÷S31.7	S0.0÷S31.7	S0.0÷S31.7
	L	L0.0÷L63.7	L0.0÷L63.7	L0.0÷63.7	L0.0÷L63.7
Vùng đối tượng	Timer	T0÷T255	T0÷T255	T0÷T255	T0÷T255
	Counter	C0÷C255	C0÷C255	C0÷C255	C0÷C255
	Analog inputs	None	AIW0÷AIW30	AIW0÷AIW62	AIW0÷AIW62
	Analog outputs	None	AQW0÷AQW30	AQW0÷AQW62	AQW0÷AQW62
	Thanh nghi ACC	AC0÷AC3	AC0÷AC3	AC0÷AC3	AC0÷AC3
	Bộ đếm tốc độ cao	HC0,HC3,HC4,HC5	HC0,HC3,HC4,HC5	HC0÷HC5	HC0÷HC5

Bảng 6.2: Phân chia vùng nhớ và toán hạng trong PLC

6.3.4 Phương thức truy cập bộ nhớ.

Truy cập theo bit: Tên miền(+) địa chỉ byte (+) • (+) chỉ số bit.

Truy cập theo byte: Tên miền (+) B (+) địa chỉ của byte trong miền.

Truy cập theo từ: Tên miền (+) W (+) địa chỉ byte cao của từ trong miền.

Truy cập theo từ kép: Tên miền (+) D (+) địa chỉ byte cao của từ trong miền.

Access Method	CPU 221	CPU 222	CPU 224	CPU 226
Bit access	V0.0÷V2047.7 I0.0÷15.7 Q0.0÷15.7 M0.0÷M31.7 SM0.0÷SM179.7 S0.0÷S31.7 T0÷T255 C0÷C255 L0.0÷L63.7	V0.0÷V2047.7 I0.0÷I15.7 Q0.0÷Q15.7 M0.0÷M31.7 SM0.0÷SM179.7 S0.0÷S31.7 T0÷T255 C0÷C255 L0.0÷L63.7	V0.0÷V5119.7 I0.0÷I15.7 Q0.0÷Q15.7 M0.0÷M31.7 SM0.0÷SM179.7 S0.0÷S31.7 T0÷T255 C0÷C255 L0.0÷L63.7	V0.0÷V5119.7 I0.0÷I15.7 Q0.0÷Q15.7 M0.0÷M31.7 SM0.0÷SM179.7 S0.0÷S31.7 T0÷T255 C0÷C255 L0.0÷L63.7
Byte access	VB0÷VB2047 IB0÷IB15 QB0÷QB15 MB0÷MB31 SMB0÷SMB179 AC0÷AC3 SB0÷SB31 LB0÷LB63	VB0÷VB2047 IB0÷IB15 QB0÷QB15 MB0÷MB31 SMB0÷SMB179 AC0÷AC3 SB0÷SB31 LB0÷LB63	VB0÷VB5119 IB0÷IB15 QB0÷QB15 MB0÷MB31 SMB0÷SMB179 AC0÷AC3 SB0÷SB31 LB0÷LB63	VB0÷VB5119 IB0÷IB15 QB0÷QB15 MB0÷MB31 SMB0÷SMB179 AC0÷AC3 SB0÷SB31 LB0÷LB63

Word access	VW0÷VW2046 T0÷T255 C0÷C255 IW0÷IW14 QW0÷QW14 MW0÷MW30 SMW0÷SMW178 AC0÷AC3 LW0÷LW62 SW0÷SW30 Constant	VW0÷VW2046 T0÷T255 C0÷C255 IW0÷IW14 QW0÷QW14 MW0÷MW30 SMW0÷SMW178 AC0÷AC3 AIW0÷AIW30 AQW0÷AQW30 LW0÷LW62 SW0÷SW30 Constant	VW0÷VW5118 T0÷T255 C0÷C255 IW0÷IW14 QW0÷QW14 MW0÷MW30 SMW0÷SMW178 AC0÷AC3 AIW0÷AIW62 AQW0÷AQW62 LW0÷LW62 SW0÷SW30 Constant	VW0÷VW5118 T0÷T255 C0÷C255 IW0÷IW14 QW0÷QW14 MW0÷MW30 SMW0÷SMW178 AC0÷AC3 AIW0÷AIW62 AQW0÷AQW62 LW0÷LW62 SW0÷SW30 Constant
Double word access	VD0÷VD2044 ID0÷ID12 QD0÷QD12 MD0÷MD28 SMD0÷SMD176 AC0÷AC3 HC0,3,4,5 SD0÷SD28 LD0÷LD60 Constant	VD0÷VD2044 ID0÷ID12 QD0÷QD12 MD0÷MD28 SMD0÷SMD176 AC0÷AC3 HC0,3,4,5 SD0÷SD28 LD0÷LD60 Constant	VD0÷VD5116 ID0÷ID12 QD0÷QD12 MD0÷MD28 SMD0÷SMD176 AC0÷AC3 HC0÷HC5 SD0÷SD28 LD0÷LD60 Constant	VD0÷VD5116 ID0÷ID12 QD0÷QD12 MD0÷MD28 SMD0÷SMD176 AC0÷AC3 HC0÷HC5 SD0÷SD28 LD0÷LD60 Constant

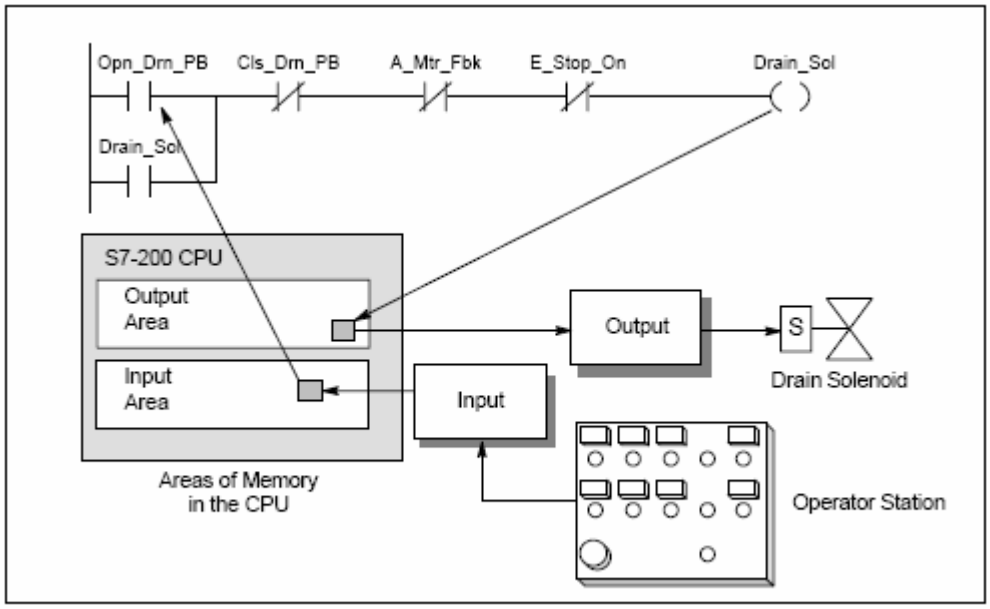
Bảng 6.3: Phương pháp truy cập vùng nhớ PLC S7-200.

6.4 Lập trình cho PLC S7 200.

6.4.1 Hoạt động cơ bản của PLC bao gồm các bước:

- CPU đọc trạng thái các ngõ vào.
- Thực hiện chương trình logic chứa trong bộ nhớ.
- CPU xuất dữ liệu đến ngõ ra.
- Chương trình của PLC bao gồm một dãy các tập lệnh. PLC S7-200 thực hiện chương trình bắt đầu từ lệnh lập trình đầu tiên và kết thúc ở lệnh lập trình cuối trong một vòng quét.

Quan hệ giữa chương trình PLC và các ngõ vào, ngõ ra như hình 6.6.



Hình 6.6: Quan hệ giữa chương trình và các ngõ vào/ra PLC.

6.4.2 Các kiểu ngôn ngữ lập trình.

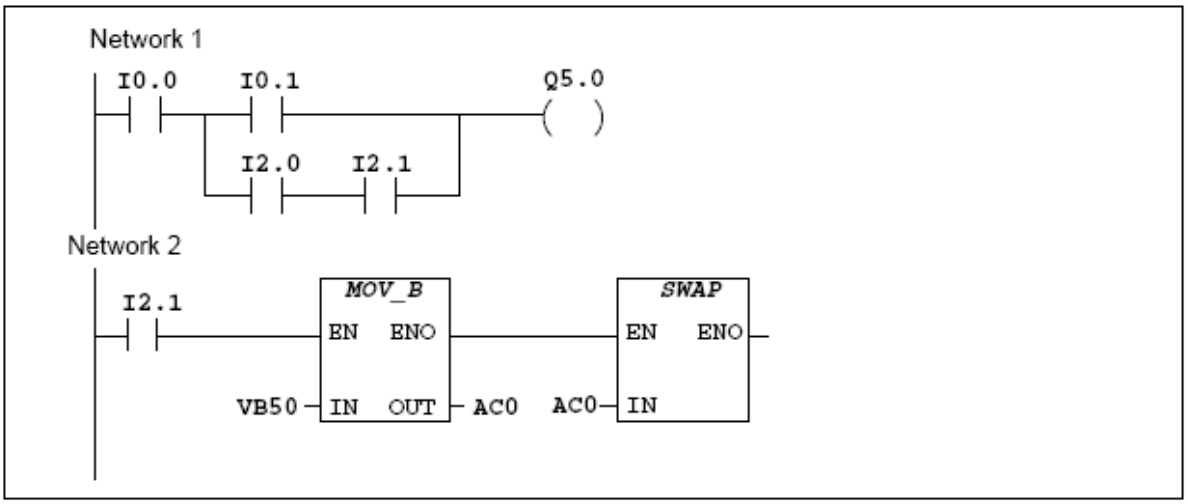
Các CPU S7-200 hỗ trợ nhiều loại lệnh khác nhau cho phép ta người lập trình sử dụng để giải quyết nhiều công việc tự động hóa. Việc lựa chọn ngôn ngữ nào để lập trình là tùy thuộc vào kinh nghiệm, khả năng và sở thích của từng người. Các kiểu lệnh mà S7 200 hỗ trợ bao gồm:

- Ladder Logic (LAD).
- Function Block Diagram (FBD).
- Statement List (STL).

6.4.3 Ngôn ngữ lập trình LAD (Ladder Loic)

Ngôn ngữ LAD cho phép ta viết chương trình tương tự như mạch tương đương của sơ đồ nối dây mạch điện. Rất nhiều người lập trình và các nhân viên kỹ thuật chọn lựa sử dụng phương pháp này.

Chương trình LAD cho phép CPU mô phỏng di chuyển của dòng điện từ nguồn, qua một loạt các điều kiện ngõ vào để tác động đến ngõ ra.



Hình 6.7:Chương trình LAD của PLC S7-200

Các lệnh khác nhau được biểu diễn bằng các ký hiệu đồ họa, gồm có các dạng cơ bản:

Tiếp điểm: Biểu diễn các điều kiện logic ngõ vào, như các công tắc, nút nhấn, trạng thái của cảm biến, …

Tiếp điểm thường mở —| |—

Tiếp điểm thương đóng —|/|—

Cuộn dây (coil): —( ) biểu diễn cho kết quả logic ngõ ra, như đèn, động cơ, cuộn dây của relay, …

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

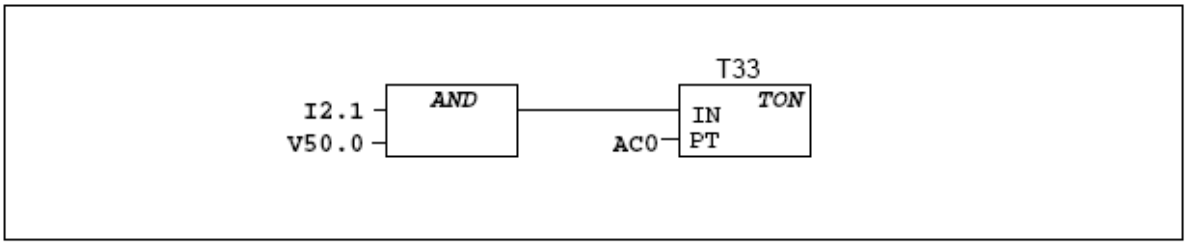
Các vấn đề chính cần quan tâm khi sử dụng ngôn ngữ LAD:

- LAD thích hợp cho người mới bắt đầu lập trình.

- Biểu diễn đồ họa dễ hiểu và thông dụng hơn.
- Luôn chuyển từ dạng LAD sang STL.

6.4.4 Ngôn ngữ FBD (Function Block Diagram)

Ngôn ngữ FBD cho phép ta xem các lệnh như là các hộp logic, tương tự như sơ đồ cổng logic. Không có các tiếp điểm và cuộn dây, nhưng sẽ có các hộp. Chương trình logic sẽ được tạo ra bằng việc kết nối các hộp, ngõ ra lệnh này sẽ tác động đến ngõ vào lệnh kia tạo thành chương trình điều khiển logic. Phương pháp kết nối này cho phép ta giải quyết được nhiều bài toán logic khác nhau.



Hình 6.8:Chương trình FBD của PLC S7-200.

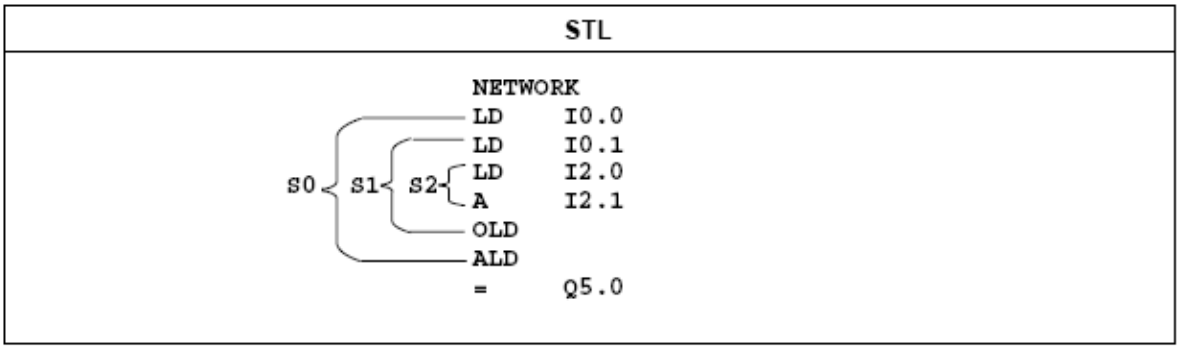
Luôn chuyển đổi từ chương trình FBD sang STL.

6.4.5 Ngôn ngữ lập trình STL (Statement List)

Soạn thảo chương trình theo phương pháp STL cho phép ta viết chương trình điều khiển bằng các lệnh gợi nhớ. Nói chung soạn thảo bằng STL phù hợp cho người có kinh nghiệm lập trình và đã quen với PLC cũng như cách lập trình logic.

Soạn thảo bằng ngôn ngữ STL cũng cho phép ta tạo ra các chương trình mà các ngôn ngữ LAD và FBD không thực hiện được. Vì STL là cách lập trình theo ngôn ngữ tự nhiên của CPU, trong khi các phương pháp khác là lập trình đồ họa.

Ví dụ viết chương trình theo ngôn ngữ STL như sau:

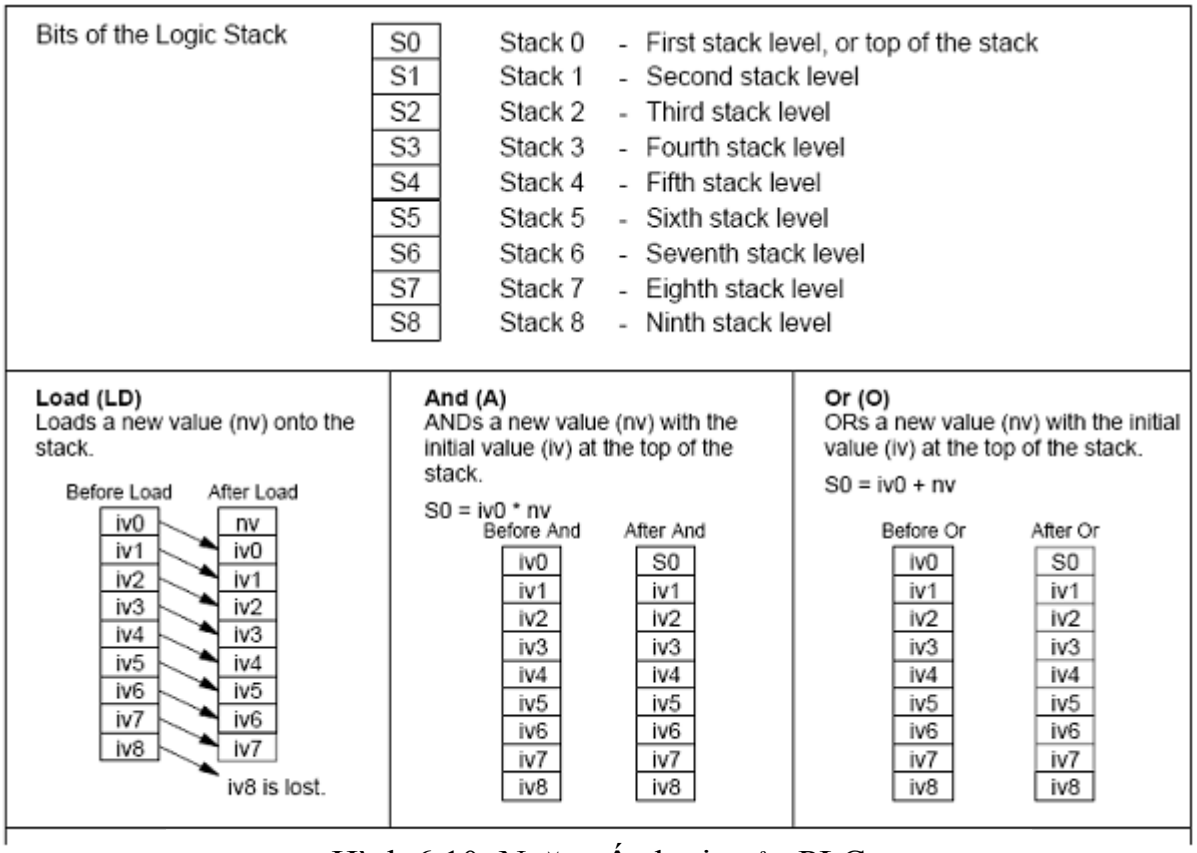


Hình 6.9: Lập trình sử dụng ngôn ngữ STL

Chương trình này tương tự như lập trình băng ngôn ngữ Assembler. CPU thực hiện chương trình bằng cách chạy các lệnh từ trên xuống dưới, rồi lặp lại.

Các điểm chính cần quan tâm khi chọn ngôn ngữ lập trình STL:

- STL thích hợp cho những người lập trình kinh nghiệm.
- STL cho phép ta giải quyết các điều khiển phức tạp mà LAD và FBD không thực hiện được.
- STL chỉ thực hiện với tập lệnh SIMATIC.
- Có thể chuyển từ chương trình STL sang LAD và FBD nhưng ngược lại thì sẽ bị giới hạn.



Hình 6.10: Ngăn xếp logic của PLC

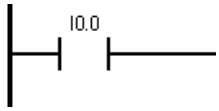
BÀI 7: TẬP LỆNH S7 – 200

7.1 Nhóm lệnh về tiếp điểm.

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

Toán hạng gồm I, Q, M, SM, V, C, T.

Dạng LAD : Tiếp điểm thường mở I0.0 sẽ đóng nếu ngõ vào PLC có địa chỉ I0.0 tác động.

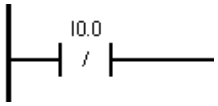


Dạng STL : LD I0.0

Load Not (LDN) : Lệnh LDN nạp giá trị logic của một tiếp điểm vào trong bit đầu tiên của ngăn xếp, các giá trị còn lại trong ngăn xếp bị đẩy lùi xuống một bit.

Toán hạng gồm : I, Q, M, SM, V, C, T.

Dạng LAD : Tiếp điểm thường đóng sẽ mở khi ngõ vào PLC có địa chỉ I0.0 tác động.



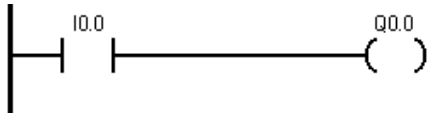
Dạng STL : LDN I0.0

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

Toán hạng bao gồm : I,Q,M,SM,T,C (bit)

Mô tả lệnh OUTPUT bằng LAD :

Nếu tiếp điểm I0.0 đóng thì cuộn dây Q0.0 sẽ được cấp điện (làm cho ngõ ra của PLC có địa chỉ Q0.0=1)



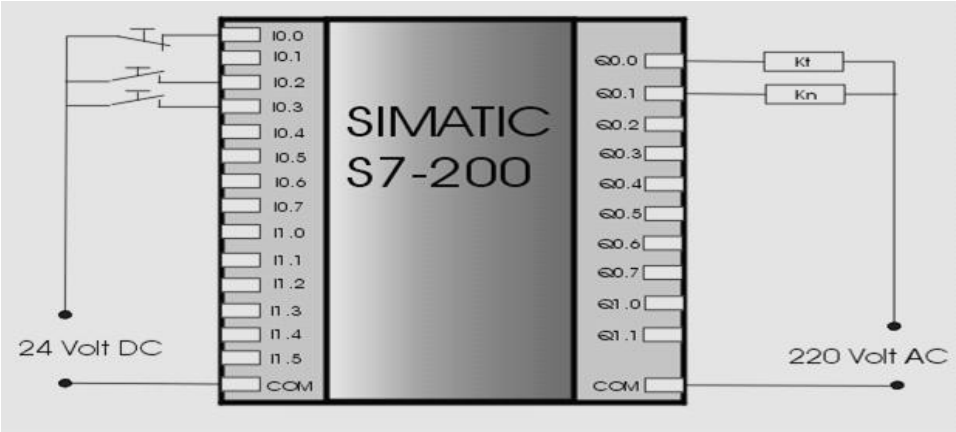
Dạng STL : Giá trị logic I0.0 được đưa vào bit đầu tiên của ngăn xếp, và bit này được sao chép vào bit ngõ ra Q0.0 .

LD I0.0

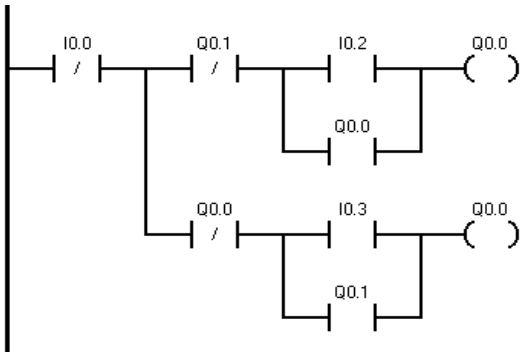
= Q0.0

Ví dụ: Lập trình PLC điều khiển đảo chiều động cơ.

Sơ đồ kết nối PLC với các thiết bị :



Lập trình dạng LAD :



Chú thích :

I0.0 : nút nhấn dừng

I0.2 : nút nhấn mở máy thuận

I0.3 : nút nhấn mở máy nghịch

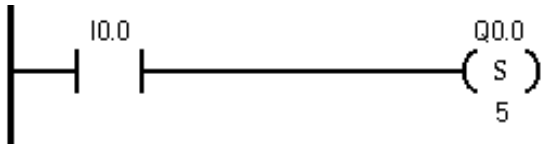
Q0.0 : ngõ ra nối với cuộn dây Kt.

Q0.1 : ngõ ra nối với cuộn dây Kn.

SET (S) : Lệnh dùng để đóng các điểm liên tục đã được chọn trước. Trong LAD, logic ngõ ra sẽ điều khiển đóng dòng điều khiển đến các cuộn dây đầu ra. Khi dòng điều khiển đến các cuộn dây thì các cuộn dây đóng các tiếp điểm. Trong STL, lệnh truyền trạng thái bit đầu tiên của ngăn xếp đến các điểm thiết kế. Nếu bit này có giá trị bằng 1, các lệnh S sẽ đóng 1 tiếp điểm hoặc một dãy các tiếp điểm (giới hạn từ 1 đến 255). Nội dung của ngăn xếp không bị thay đổi bởi các lệnh này.

***Dạng LAD* :** Khi tiếp điểm I0.0 đóng lệnh SET sẽ đóng một mảng gồm n các tiếp điểm kể từ địa chỉ Q0.0.

Toán hạng bao gồm I, Q, M, SM,T, C,V (bit)

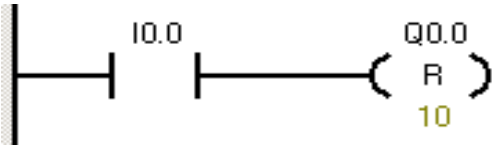


***Dạng STL*:** Ghi giá trị logic vào một mảng gồm n bit từ địa chỉ Q0.0

```
LD      I0.0
S       Q0.0,  5
```

RESET (R): Lệnh dùng để ngắt các điểm liên tục được chọn trước. Trong LAD, logic ngõ ra sẽ điều khiển ngắt dòng điện các cuộn dây đầu ra. Khi dòng điều khiển đến các cuộn dây thì các cuôn dây mở các tiếp điểm. Trong STL, lệnh truyền trạng thái bit đầu tiên của ngăn xếp đến các điểm thiết kế. Nếu bit này có giá trị bằng 1, các lệnh R sẽ ngắt 1 tiếp điểm hoặc một dãy các tiếp điểm (giới hạn từ 1 đến 255). Nội dung của ngăn xếp không bị thay đổi bởi các lệnh này.

Dạng LAD : Ngắt một mảng gồm 10 các tiếp điểm kể từ tiếp điểm có địa chỉ Q0.0. Toán hạng bao gồm I, Q, M, SM,T, C,V (bit)



Dạng STL : Xóa một mảng gồm 10 bit kể từ địa chỉ Q0.0.

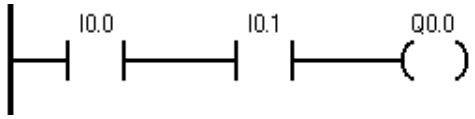
```
LD    I0.0
R     Q0.0,  10
```

Lệnh Đại số Bool.

Các lệnh tiếp điểm đại số Bool cho phép tạo lập các mạch logic (không có nhớ). Trong LAD các lệnh này được biểu diễn thông qua cấu trúc mạch, mắc nối tiếp hay song song các tiếp điểm thường đóng hay các tiếp điểm thường mở.

AND (A) :

Dạng LAD :

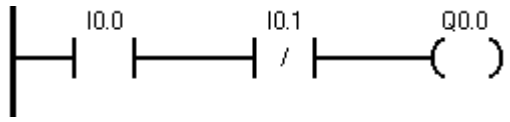


Dạng STL :

```
LD    I0.0
A     I0.1
=     Q0.0
```

AND NOT(AN) :

Dạng LAD :

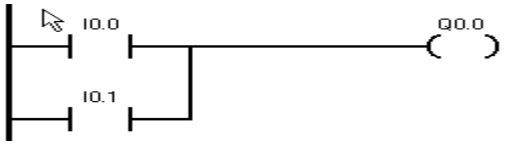


Dạng STL :

```
LD    I0.0
AN    I0.1
=     Q0.0
```

OR (O):

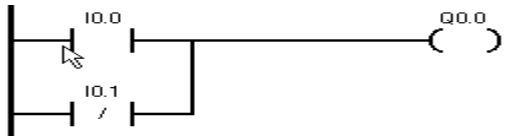
Dạng LAD : Dạng STL :




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

OR NOT (ON):

Dạng LAD : Dạng STL :

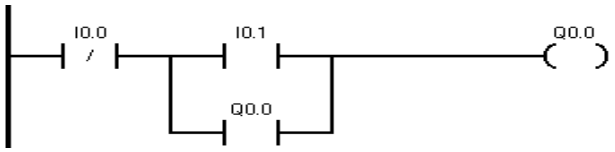



```
LD   I0.0
OR   I0.1
=    Q0.0
```

Ví dụ :

Viết chương trình điều khiển mở máy động cơ bằng PLC.

Lập trình LAD Ghi chú :



I0.0 : Nút nhấn dừng

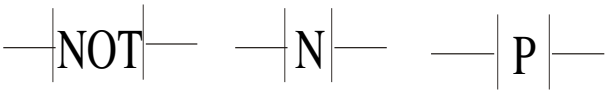
I0.1 : Nút nhấn mở

Q0.0 : Cuộn dây KĐT

Q0.0 : Tiếp điểm duy trì

Lệnh Tiếp điểm đặc biệt.

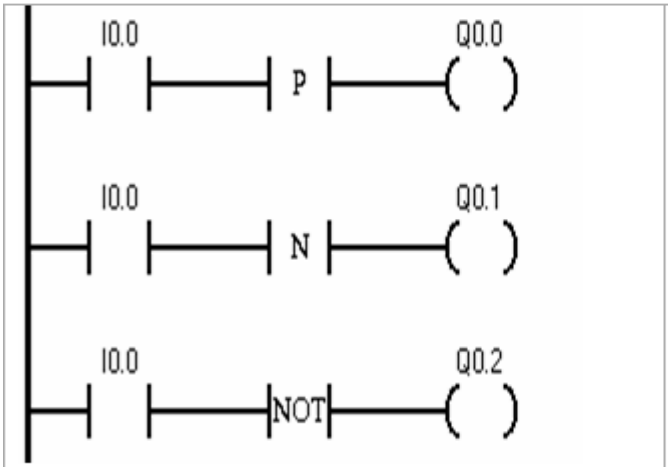
Tiếp điểm đảo, tác động cạnh xuống, tác động cạnh lên :



Có thể dùng các lệnh tiếp điểm đặc biệt để phát hiện sự chuyển tiếp trạng thái của xung (sườn xung) và đảo lại trạng thái của dòng cung cấp (giá trị đỉnh của ngăn xếp). LAD sử dụng các tiếp điểm đặc biệt này để tác động vào dòng cung cấp. Các tiếp điểm đặc biệt không có toán hạng riêng của nên ta phải đặt chúng phía trước cuộn dây hoặc hộp đầu ra. Tiếp điểm chuyển tiếp dương/âm (các lệnh sườn trước và sườn sau) có yêu cầu về bộ nhớ, bởi vậy đối với CPU 214 có thể sử dụng nhiều nhất là 256 lệnh.

Ví dụ minh họa:

Dạng LAD : Dạng STL :

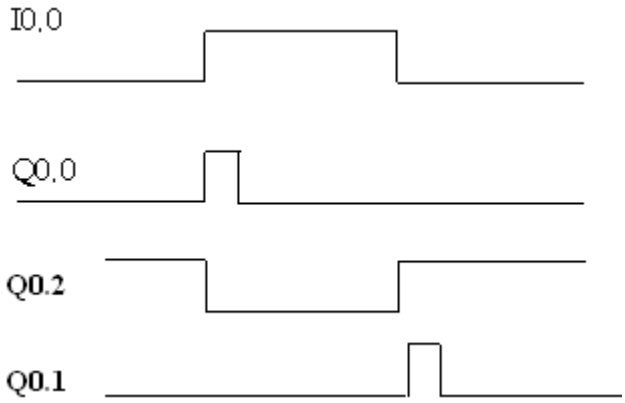



```
LD    I0.0
EU
=     Q0.0

LD    I0.0
ED
=     Q0.1

LD    I0.0
NOT
=     Q0.2
```

Biểu đồ thời gian :



Tiếp điểm trong vùng nhớ đặc biệt :

SM0.0 : Ngược lại với SM0.1, vòng quét đầu tiên thì mở nhưng từ vòng quét thứ hai trở đi thì đóng.

SM0.1 : Vòng quét đầu tiên tiếp điểm này đóng, kể từ vòng quét thứ hai thì mở ra và giữ nguyên trong suốt quá trình hoạt động.

SM0.4 : Tiếp điểm tạo xung với nhịp xung với chu kỳ là 1 phút.

SM0.5 : Tiếp điểm tạo xung với nhịp xung với chu kỳ là 1s

7.2 Nhóm lệnh về Timer.

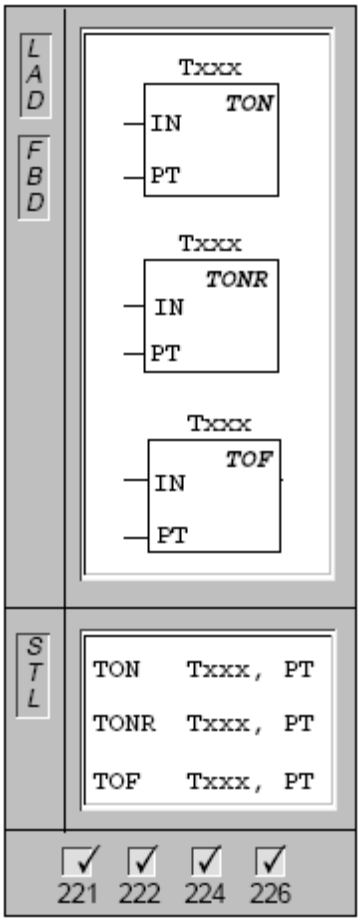
Timer là bộ tạo thời gian trễ giữa tín hiệu vào và tín hiệu ra nên trong điều khiển thường được gọi là khâu trễ. Nếu ký hiệu tín hiệu (logic) vào là x(t) và thời gian trễ tạo ra bằng Timer là τ thì tín hiệu đầu ra của Timer đó sẽ là $x(t - \tau)$

S7-200 có 64 bộ Timer (với CPU 212) hoặc 128 Timer (với CPU 214) được chia làm 3 loại khác nhau:

Timer đóng chậm không có nhớ (On-Delay Timer), ký hiệu là TON.

Timer mở chậm không nhớ (Off –Delay Timer), ký hiệu TOF.

Timer đóng chậm có nhớ (Retentive On-Delay Timer), ký hiệu TONR.



Ba loại Timer của S7-200 phân biệt với nhau ở phản ứng của nó đối với trạng thái logic đầu vào.

Cả hai Timer kiểu TON và TONR cùng bắt đầu đếm thời gian kể từ thời điểm có sườn lên ở tín hiệu đầu vào, đầu vào chuyển trạng thái logic từ 0 lên 1, được gọi là thời gian Timer được kích. Khi giá trị thời gian của Timer lớn hơn hoặc bằng giá trị đặt trước thì tiếp điểm của Timer sẽ đóng lại. Lúc này Timer vẫn tiếp tục đếm thời gian cho đến giá trị max.

Khi đầu vào có giá trị logic bằng 0, TON tự động reset còn TONR thì không. Timer TON được dùng để tạo thời gian trễ trong một khoảng thời gian (miền liên thông), còn với TONR thời gian trễ sẽ được tạo ra trong nhiều khoảng thời gian khác nhau.

Timer TOF dùng để mở chậm một ngõ ra sau thời gian đặt trước nào đó, kể từ khi logic ngõ vào chuyển từ 1 xuống 0. Khi ngõ vào Timer lên 1, tiếp điểm của nó đóng ngay lập tức và đặt giá trị thời gian bằng 0. Khi ngõ vào xuống 0, Timer sẽ đếm thời gian cho đến giá trị đặt trước thì tiếp điểm Timer sẽ mở ra.

Timer TON, TOF và TONR đều có 3 loại với 3 độ phân giải khác nhau, độ phân giải 1ms, 10ms và 100ms. Thời gian trễ τ được tạo ra chính là tích của độ phân giải của bộ Timer được chọn và giá trị đặt trước cho Timer. Ví dụ Timer có độ phân giải 10ms và giá trị đặt trước 50 thì thời gian trễ là 500ms.

Độ phân giải các loại Timer của S7-200, được trình bày trong bảng 7.1.

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

Bảng 7.1: Độ phân giải của các Timer

7.2.1 Khai báo sử dụng Timer.

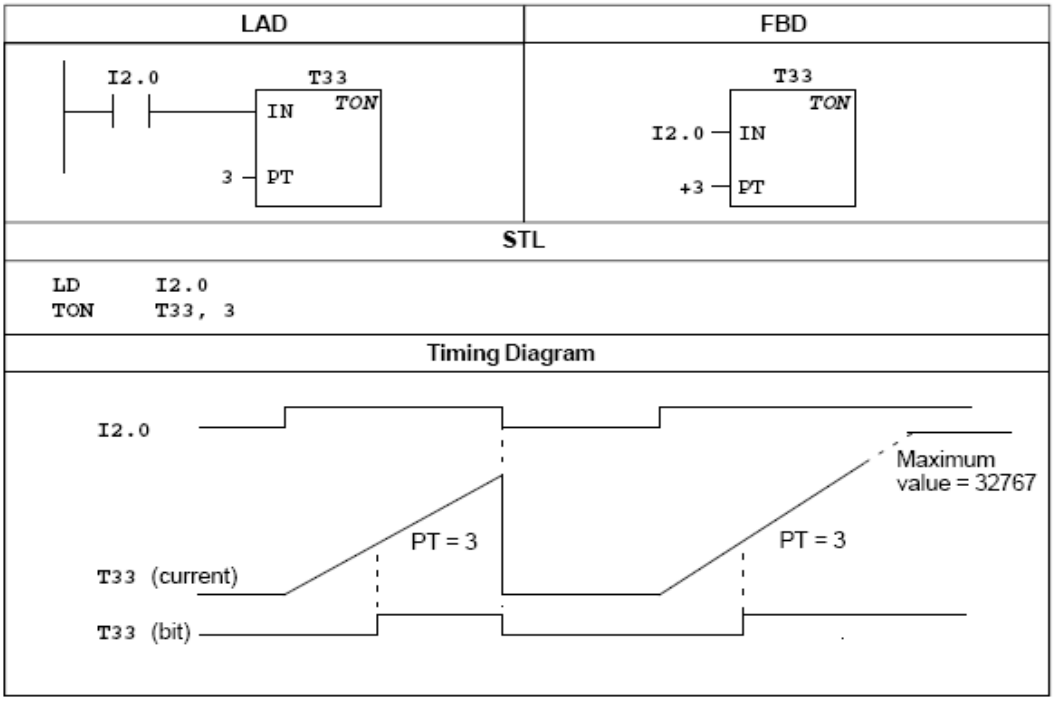
LAD	Mô tả	Toán hạng
TON-Txx TOF-Txx IN PT	Khai báo Timer số hiệu xx kiểu TON để tạo thời gian trễ tính từ khi đầu vào IN được kích. Nếu như giá trị đếm tức thời lớn hơn hoặc bằng giá trị đặt trước PT thì T-bit có giá trị logic bằng 1. có thể reset Timer kiểu TON bằng lệnh R hoặc bằng giá trị logic 0 tại đầu vào IN.	Txx: Số hiệu Timer PT: VW, T, (*word*) C, IW, QW, MW, SMW, C, IW, hằng số
TONR-Txx IN PT	Khai báo Timer số hiệu xx kiểu TONR để tạo thời gian trễ tính từ khi đầu vào IN được kích. Nếu như giá trị đếm tức thời lớn hơn hoặc bằng giá trị đặt trước PT thì T-bit có giá trị logic bằng 1. Chỉ có thể reset Timer kiểu TON bằng lệnh R cho T-bit.	Txx: Số hiệu Timer PT: VW, T, (*word*) C,IW,QW, MW, SMW, AC, AIW, hằng số

Khi sử dụng Timer TONR, giá trị đếm tức thời được lưu lại và không bị thay đổi trong khoảng thời gian khi tín hiệu đầu vào có logic 0. Giá trị của T-bit không được nhớ mà hoàn toàn phụ thuộc vào kết quả so sánh giữa giá trị đếm tức thời và giá trị đặt trước.

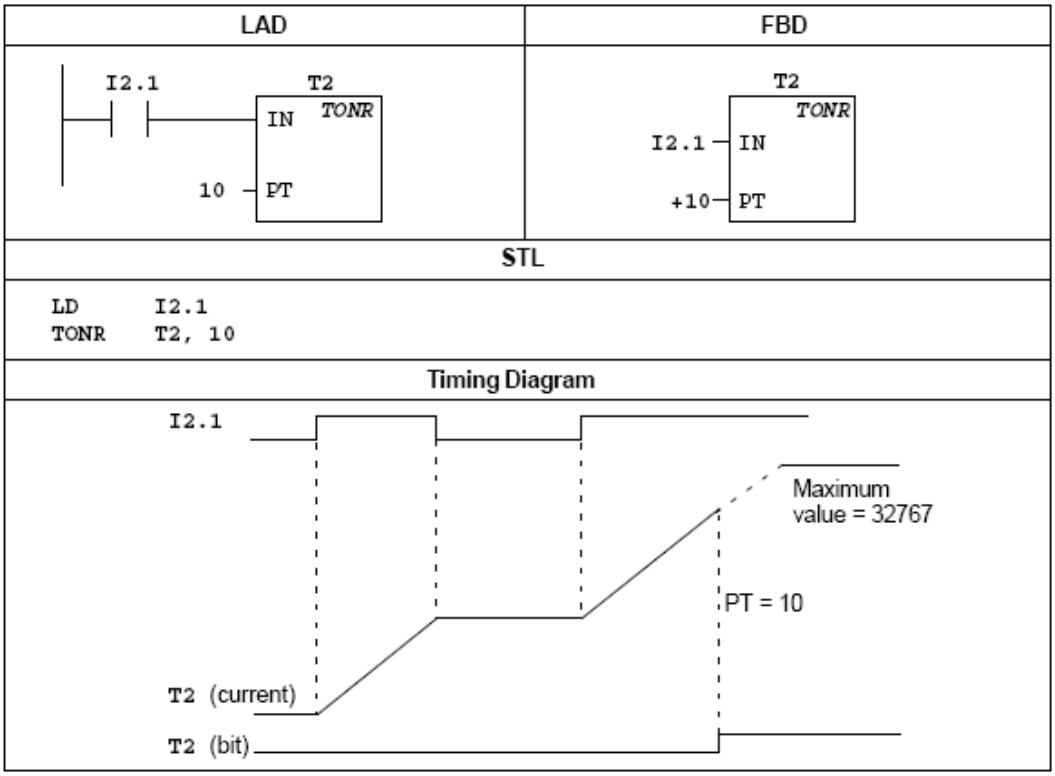
Khi Reset một bộ Timer, T-word và T-bit của nó đồng thời được xóa và có giá trị bằng 0, như vậy giá trị đếm tức thời được đặt về 0 và tín hiệu đầu ra cũng có trạng thái logic 0.

7.2.2 Một số ví dụ về Timer.

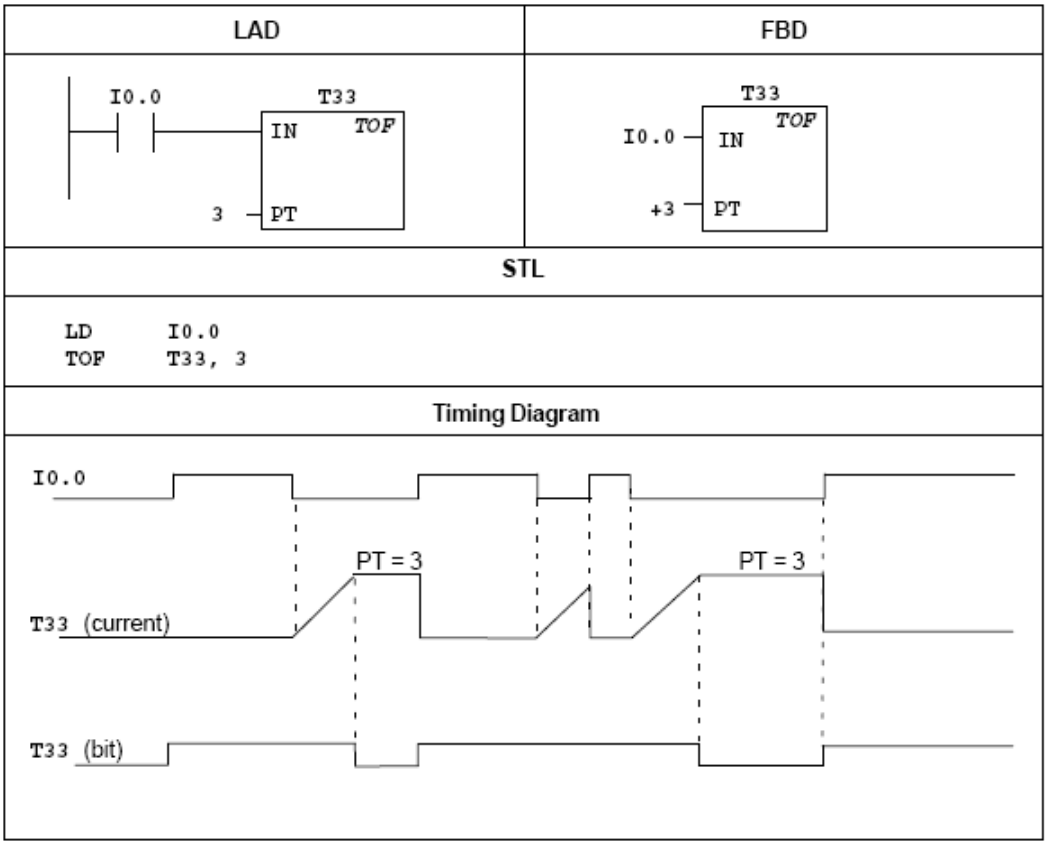
Ví dụ 1: Sử dụng Timer kiểu TON



Ví dụ 2: Sử dụng timer kiểu TONR



Ví dụ 3: Sử dụng timer kiểu TOF



Ví dụ 4:

Lập trình logic cho PLC điều khiển dây chuyền sản xuất gồm 3 động cơ hoạt động theo yêu cầu sau :

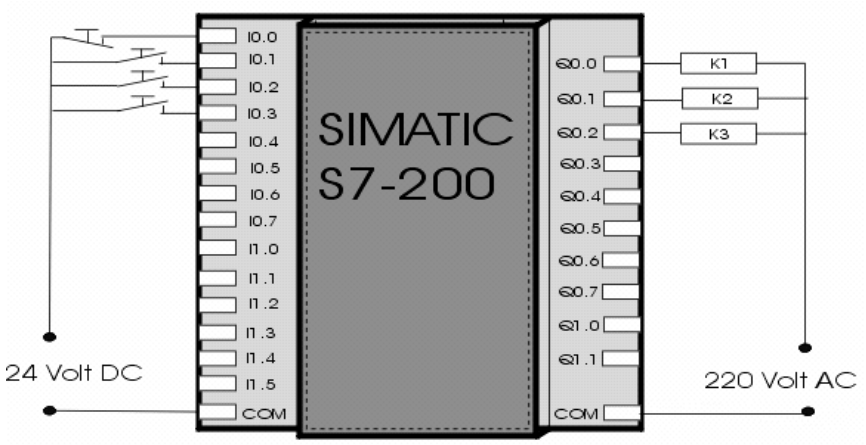
Nhấn nút khởi động cho động cơ Đ1 chạy, sau 5s cho phép vận hành Đ2.

Nhấn nút khởi động cho động cơ Đ2 chạy đồng thời lúc này động cơ Đ1 ngừng, sau 10s thì cho phép vận hành động cơ Đ3.

Nhấn nút khởi động cho động cơ Đ3 chạy đồng thời động cơ Đ2 ngừng.

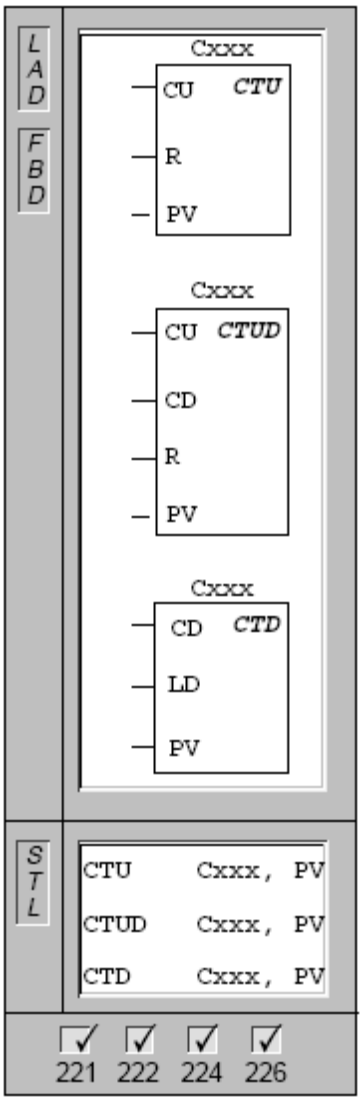
Nhấn nút dừng thì bất kỳ động cơ nào chạy cũng phải ngừng.

Sơ đồ kết nối PLC :



7.3 Lệnh Counter.

Counter là bộ đếm thực hiện chức năng đếm sườn xung, trong S7-200 các bộ đếm được chia làm 3 loại : Bộ đếm lên (CTU), bộ đếm xuống (CU) và bộ đếm lên/xuống (CTUD).



7.3.1 Bộ đếm lên CTU:

Đếm số sườn lên của tín hiệu logic đầu vào, tức là đếm số lần thay đổi trạng thái logic từ 0 lên 1 của tín hiệu. Số xung đếm được ghi vào thanh ghi 2 byte của bộ đếm, gọi là thanh ghi C-word. Nội dung của thanh ghi C-word, gọi là giá trị đếm tức thời của bộ đếm, luôn được so sánh với giá trị đặt trước của bộ đếm, được ký hiệu là PV. Khi giá trị đếm tức thời bằng hoặc lớn hơn giá trị đặt trước này thì bộ đếm báo ra ngoài bằng cách đặt giá trị logic 1 vào một bit đặc biệt của nó, gọi là C-bit. Trường hợp giá trị đếm tức thời nhỏ hơn giá trị đặt trước thì C-bit có giá trị logic là 0.

7.3.2 Bộ đếm lên/xuống CTUD**:**

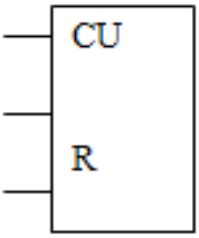
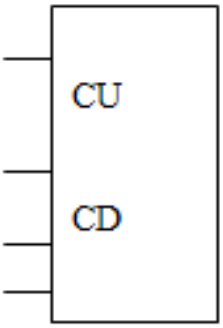
Đếm lên khi gặp sườn lên của xung vào cổng đếm lên, ký hiệu là CU hoặc bit thứ 3 của ngăn xếp trong STL, và đếm xuống khi gặp sườn lên của xung vào cổng đếm xuống, ký hiệu là CD trong LAD hoặc bit thứ 2 của ngăn xếp trong STL.

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

Bộ đếm lên CTU có miền giá trị đếm tức thời từ 0 đến 32.767. Bộ đếm lên/xuống CTUD có miền giá trị đếm tức thời từ –32.768 đến 32.767.

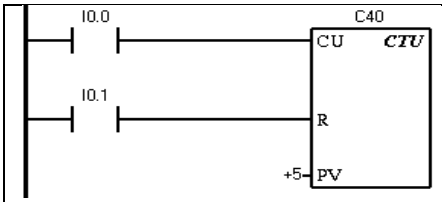
7.3.3 Bộ đếm xuống CTD**:**

Đếm xuống từ giá trị đặt trước khi có sườn lên tác động vào cổng đếm xuống. Khi giá trị đếm bằng 0 thì dừng đếm và tiếp điểm của nó sẽ đóng. Khi chân Load LD tác động thì bộ đếm xuống sẽ mở tiếp điểm và nạp giá trị đặt trước vào.

LAD	Mô tả	Toán hạng
CTU – Cxx  	Khai báo bộ đếm tiến theo sườn lên của CU. Khi giá trị đếm tức thời C-word Cxx lớn hơn hoặc bằng giá trị đặt trước PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm được reset khi đầu vào R có giá trị logic bằng 1. Bộ đếm ngừng đếm khi C-word Cxx đạt được giá trị cực đại.	Cxx : Số hiệu bộ đếm. PV(*word*) : VW, T, C, IW, QW, MW, SMW, AC, AIW, hằng số, *VD, *AC
CTD-Cxx  	Khai báo bộ đếm tiến/lùi, đếm tiến theo sườn lên của CU, đếm lùi theo sườn lên của CD. Khi giá trị đếm tức thời C-word Cxx lớn hơn hoặc bằng giá trị đặt trước PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm ngừng đếm tiến khi C-word Cxx đạt được giá trị cực đại 32.767 và ngừng đếm lùi khi C-word Cxx đạt được giá trị cực đại –32.768. CTUD reset khi đầu vào R có giá trị logic bằng 1.	Cxx : Số hiệu bộ đếm. PV(*word*) : VW, T, C, IW, QW, MW, SMW, AC, AIW, hằng số, *VD, *AC

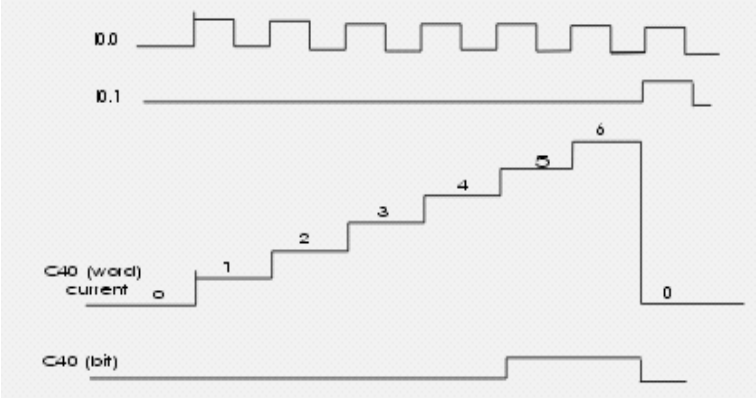
7.3.4 Một số ví dụ về Counter

Ví dụ 5: Sử dụng bộ đếm CTU :

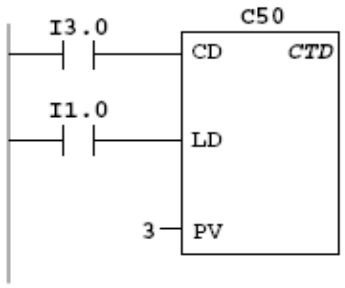
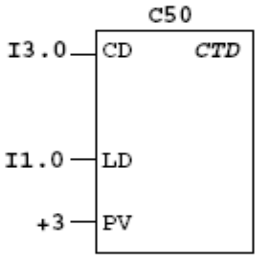



```
LD    I0.0
LD    I0.1
CTU   C40, +5
```

Giản đồ thời gian :



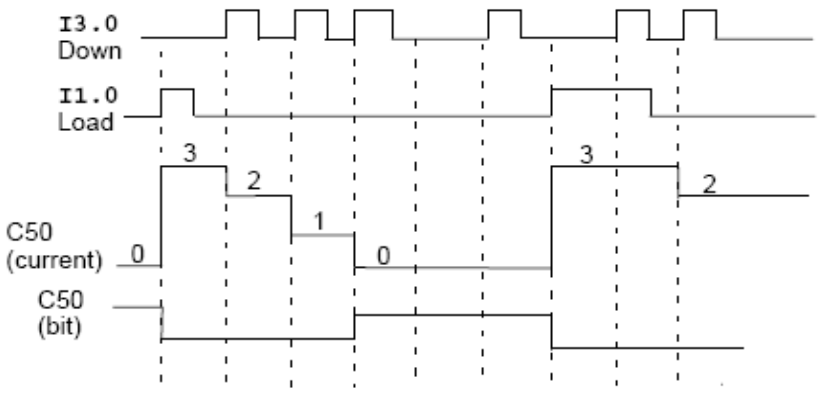
Ví dụ 6: Sử dụng bộ đếm CTD:

LAD	FBD
	

STL

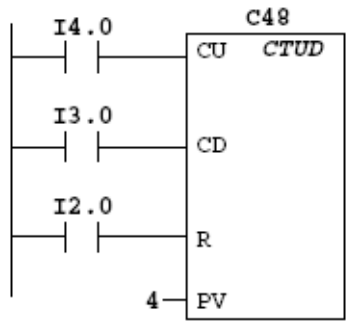
```
LD      I3.0      //Count Down Input
LD      I1.0      //Load Input
CTD     C50, 3
```

Timing Diagram

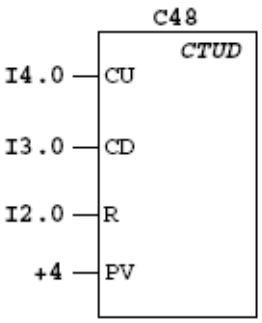


Ví dụ 7: Sử dụng bộ đếm CTUD :

LAD



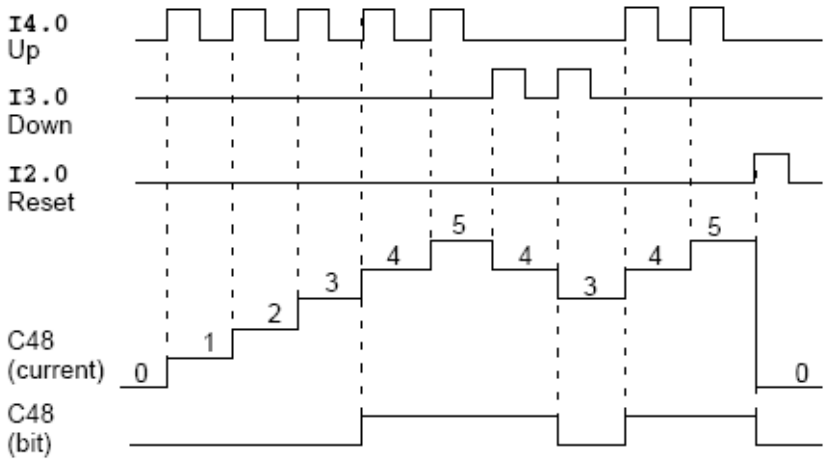
FBD



STL

```
LD      I4.0        //Count Up Input
LD      I3.0        //Count Down Input
LD      I2.0        //Reset Input
CTUD    C48, 4
```

Timing Diagram

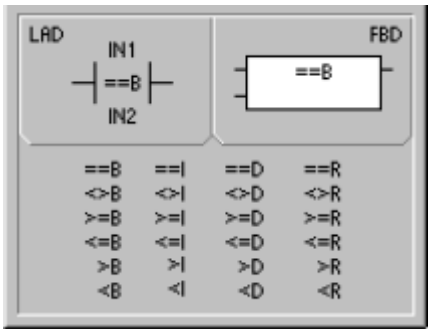


7.4 Nhóm lệnh so sánh.

Khi lập trình, nếu các quyết định về điều khiển được thực hiện dựa trên kết quả của việc so sánh thì có thể sử dụng lệnh so sánh theo Byte, Word, DWord và Real của S7-200. Những lệnh so sánh mà PLC hỗ trợ: So sánh bằng, lớn hơn hoặc bằng, nhỏ hơn hoặc bằng, lớn hơn, nhỏ hơn.

IN1 = IN2	IN1 >= IN2	IN1 <= IN2
IN1 > IN2	IN1 < IN2	IN1 <> IN2

Mỗi phép toán đều hỗ trợ đối với kiểu dữ liệu Byte, Word, Double word và Real.



Cách thức thực hiện lệnh so sánh: Lấy giá trị ngõ vào IN1 so sánh với ngõ vào IN2. Nếu thỏa mãn phép toán trong câu lệnh thì tiếp điểm so sánh sẽ được đóng lại.

Ví dụ 8 : Sử dụng lệnh về Timer và So sánh, Viết chương trình điều khiển đèn giao thông tại ngả 4 theo yêu cầu: Xanh 4s, Vàng 1s, Đỏ 5s. Có 2 nút nhấn ở chế độ bình thường và ưu tiên.

Thực hiện tương tự với lệnh Counter.

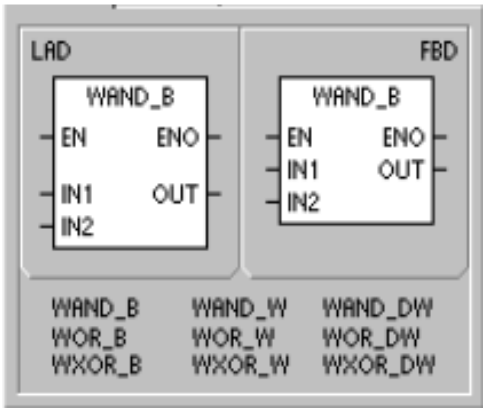
7.5 Nhóm lệnh về cổng Logic.

Ngoài những lệnh ghép nối tiếp, song song và tổng hợp các tiếp điểm thì tập lệnh của S7-200 còn cung cấp các cổng logic AND, OR, EXOR thực hiện đối

với byte (8 bit hay 8 tiếp điểm), word (16 bit hay 16 tiếp điểm) và double word (32 bit hay 32 tiếp điểm). Sau đây là chi tiết của từng cổng:

Lệnh AND:

PLC hỗ trợ lệnh AND 2 Byte, AND 2 Word, AND 2 Double Word. Cú pháp lệnh AND, OR và WXOR viết trong LAD và FBD va toán hạng hỗ trợ trong câu lệnh được trình bày như hình 7.1.



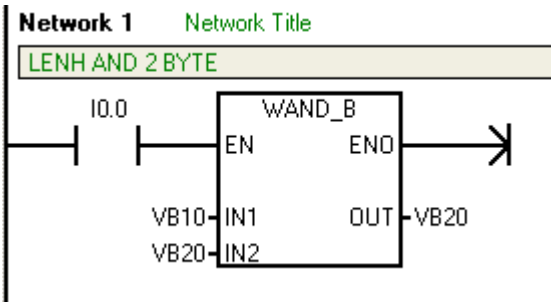
Inputs/Outputs	Data Types	Operands
IN1, IN2	BYTE	IB, QB, VB, MB, SMB, SB, LB, AC, *VD, *LD, *AC, Constant
	WORD	IW, QW, VW, MW, SMW, SW, T, C, LW, AC, AIW, *VD, *LD, *AC, Constant
	DWORD	ID, QD, VD, MD, SMD, SD, LD, AC, HC, *VD, *LD, *AC, Constant
OUT	BYTE	IB, QB, VB, MB, SMB, SB, LB, AC, *VD, *AC, *LD
	WORD	IW, QW, VW, MW, SMW, SW, T, C, LW, AC, *VD, *AC, *LD
	DWORD	ID, QD, VD, MD, SMD, SD, LD, AC, *VD, *AC, *LD

Hình 7.1: Lện AND và các toán hạng hỗ trợ ngõ vào, ngõ ra

Ý nghĩa:

Lệnh thực hiện phép AND từng bit của hai Byte, hai Word hoặc hai Double Word ngõ vào IN1 và IN2, kết quả được ghi vào 1 byte, 1 Word hay 1 Double Word ở ngõ ra OUT, địa chỉ ngõ ra có thể giống ngõ vào.

Ví dụ: Lệnh AND 2 Byte:



VB10	1	0	0	0	1	1	1	0
AND								
VB20	0	0	1	1	0	1	1	1
Kết quả VB20	0	0	0	0	0	1	1	0

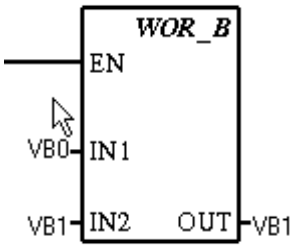
7.6 Lệnh OR:

PLC hỗ trợ lệnh OR 2 Byte, OR 2 Word, OR 2 Double Word. Cú pháp lệnh AND, OR và WXOR viết trong LAD và FBD và các phép toán hỗ trợ trong câu lệnh được trình bày hình 7.1.

Ý nghĩa:

Lệnh thực hiện phép OR từng bit của hai Byte, hai Word hoặc hai Double Word ngõ vào IN1 và IN2, kết quả được ghi vào 1 byte, 1 Word hay 1 Double Word ở ngõ ra OUT, địa chỉ ngõ ra có thể giống ngõ vào.

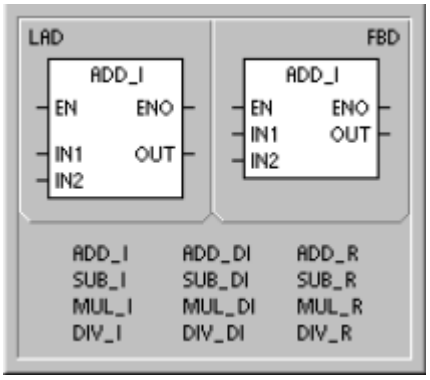
Ví dụ: Lệnh OR hai Byte:



VB0	1	0	0	0	1	1	1	0
OR								
VB1	0	0	1	1	0	1	1	1
Kết quả VB1	1	0	1	1	1	1	1	1

7.7 Nhóm lệnh về các phép toán logic.

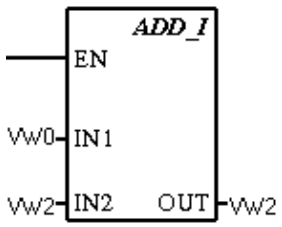
Các lệnh toán học trong PLC gồm có: Lệnh cộng , lệnh trừ, lệnh nhân, lệnh chia. Các lệnh toán học này đều hỗ trợ với kiểu dữ liệu Word, Double Word và Real.



Hình 7.3: Các lệnh toán học

7.7.1 Lệnh Cộng số nguyên 16 bit.

Dạng LAD : Dạng STL :



+I VW0, VW2

Ý nghĩa :

Lệnh thực hiện cộng các số nguyên 16 bit IN1 và IN2 , kết quả là số nguyên 16 bit được ghi vào OUT, IN1 + IN2 = OUT, IN2 và OUT có thể cùng địa chỉ, thuộc các vùng nhớ sau

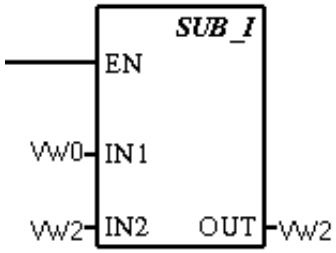
IN1, IN2 : VW,T ,C, IW, QW, MW, SMW, AC, AIW, const.

OUT : VW,T ,C, IW, QW, MW, SMW, AC, AIW

Các lệnh cộng với số nguyên 32 bit và số thực cũng được thực hiện tương tự.

7.7.2 Lệnh trừ số nguyên 16 bit.

Dạng LAD : Dạng STL :



+I VW0, VW2

Ý nghĩa :

Lệnh được thực hiện phép trừ các số nguyên 16 bit IN1và IN2 , kết quả là số nguyên 16 bit và được ghi vào OUT, tức là : IN1 – IN2 = OUT, địa chỉ thuộc các vùng nhớ sau

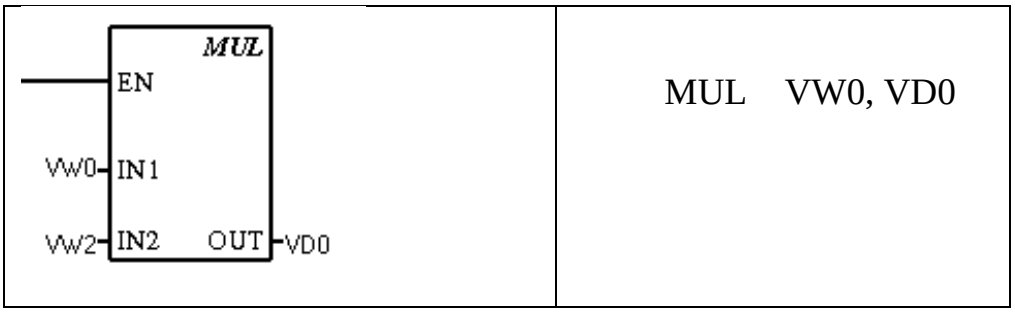
IN1, IN2 : VW,T ,C, IW, QW, MW, SMW, AC, AIW, const.

OUT : VW,T ,C, IW, QW, MW, SMW, AC, AIW.

Các lệnh trừ với số nguyên 32 bit và số thực cũng được thực hiện tương tự.

7.7.3 Lệnh nhân số nguyên 16 bit.

Dạng LAD : Dạng STL.

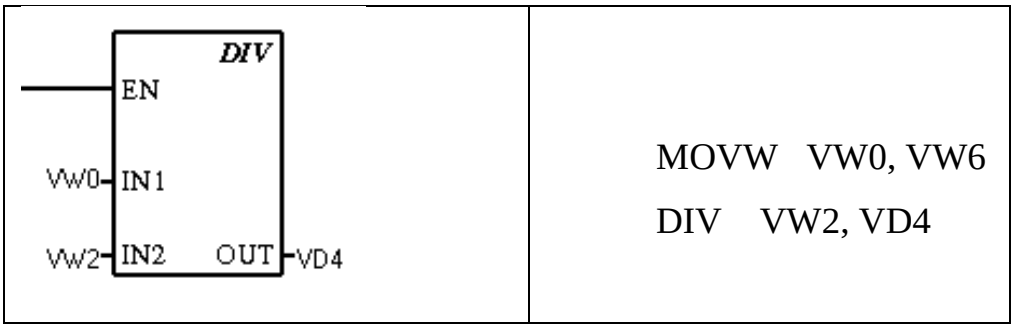


Ý nghĩa :

Lệnh thực hiện phép nhân 2 số nguyên 16bit IN1 và IN2. Kết quả 32 bit chứa trong từ kép OUT (4 byte).

7.7.4 Lệnh chia số nguyên 16 bit .

Dạng LAD : Dạng STL :



Ý nghĩa :

Lệnh thực hiện phép chia số nguyên 16 bit IN1 cho số nguyên 16 bit IN2. Kết quả 32 bit chứa trong từ kép OUT (4 byte) gồm thương số ghi trong mảng 16 bit từ bit 0 đến bit 15 (từ thấp) và phần dư cũng 16 bit ghi trong mảng từ bit 16 đến bit 31 (từ cao).

Trong lệnh này có sử dụng các bit nhớ đặc biệt sau để báo trạng thái.

Kết quả tính	SM1.0	SM1.1	SM1.2	SM1.3
= 0	1			
Báo tràn		1		
Số âm			1	
Mẫu = 0				1

7.8 Nhóm lệnh di chuyển và trao đổi dữ liệu.

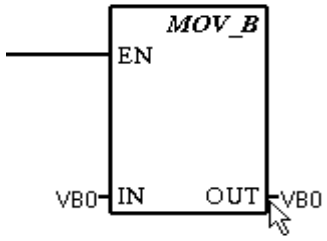
Các lệnh di chuyển thực hiện việc di chuyển hoặc sao chép dữ liệu từ vùng này sang vùng khác trong bộ nhớ. Lệnh dịch chuyển thực hiện việc di chuyển hay sao chép nội dung một byte, một từ đơn, hoặc một từ kép từ vùng này sang vùng khác trong bộ nhớ.

Lệnh trao đổi nội dung của hai byte trong một từ đơn thực hiện việc chuyển nội dung của byte thấp sang byte cao và ngược lại chuyển nội dung của byte cao sang byte thấp của từ đó.

7.8.1 Lệnh di chuyển dữ liệu.(MOV).

Lệnh di chuyển dữ liệu từ vùng nhớ này sang vùng nhớ khác. Lệnh MOV hỗ trợ cho các kiểu dữ liệu Byte, Word, Double Word và Real.

MOV_B :

Dạng LAD	Dạng STL
	MOVB VB0, VB0

Ý nghĩa:

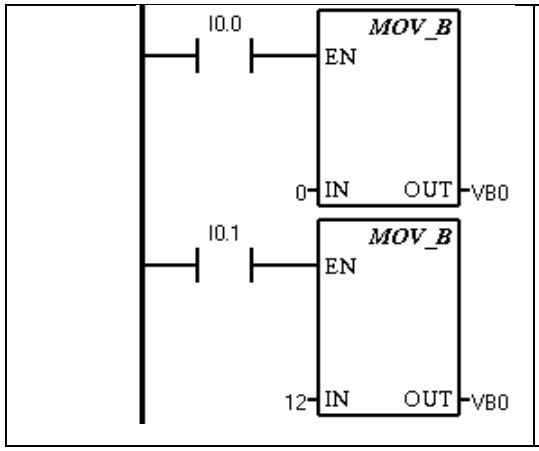
Lệnh sao chép nội dung của byte ở địa chỉ ngõ vào IN sang byte có địa chỉ ở ngõ ra OUT. Địa chỉ của byte ngõ vào IN và địa chỉ byte ngõ ra OUT có thể giống nhau, thuộc các vùng sau:

IN : VB, IB, QB, MB, SMB, AC, const

OUT: VB, IB, QB, MB, SMB, AC,

Ví dụ :

Dạng LAD	Dạng STL

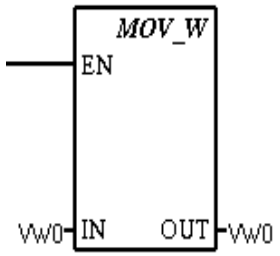
	LD I0.0
	MOVB 0, VB0
	LD I0.1
	MOVB 12, VB0

Giải thích:

Nếu tiếp điểm I0.0 đóng thì lấy giá trị 0 ghi vào byte VB0 (xóa VB0)

Tiếp theo đóng tiếp điểm I0.1 thì lấy số 12 ghi vào VB0. Kết quả địa chỉ byte VB0 có giá trị bằng 12 (nhị phân).

MOV_W:

Dạng LAD	Dạng STL
	MOVW VW0, VW0

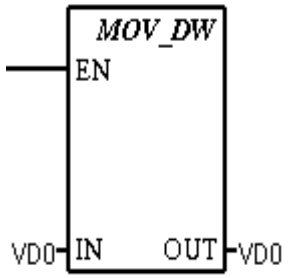
Ý nghĩa :

Lệnh sao chép nội dung của Word ở địa chỉ ngõ vào IN sang Word có địa chỉ ở ngõ ra OUT, địa chỉ ngõ ra có thể giống ngõ vào, nằm trong các vùng sau:

IN: VW, IW, QW, MW, SMW, AC, const

OUT: VW, IW, QW, MW, SMW, AC

MOV_DW :

Dạng LAD	Dạng STL
	MOVDW VD0, VD0

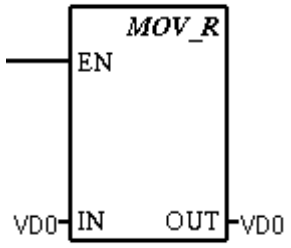
Ý nghĩa :

Lệnh sao chép nội dung của DWord ở địa chỉ ngõ vào IN sang DWord có địa chỉ ở ngõ ra OUT, địa chỉ ngõ ra có thể giống ngõ vào, nằm trong các vùng sau:

IN: VD, ID, QD, MDW, SMD, AC, const

OUT: VD, ID, QD, MD, SMD, AC

MOV_R :

Dạng LAD	Dạng STL
	MOVR VD0, VD0

Ý nghĩa :

Lệnh sao chép nội dung của số thực chứa trong double word có địa chỉ ở ngõ vào IN sang double word có địa chỉ ở ngõ vào OUT, địa chỉ ngõ ra có thể giống ngõ vào, thường nằm trong các vùng sau:

IN: VD, ID, QD, MD, SMD, AC, const

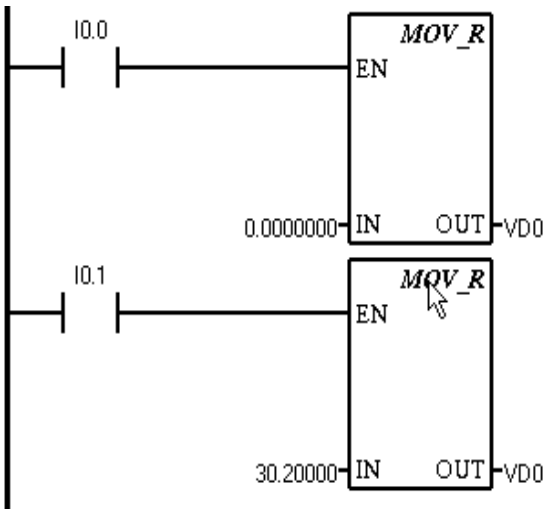
OUT: VD, ID, QD, MD, SMD, AC

Dữ liệu được ghi vào trong các địa chỉ này theo nguyên tắc sau :

Phần nguyên ghi vào word thấp

Phần thập phân ghi vào word cao

Ví dụ :



Giải Thích :

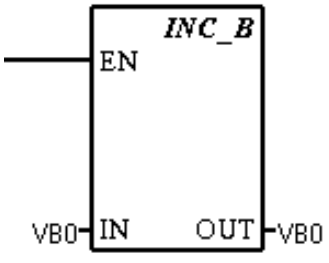
Tiếp điểm I0.0 đóng thì xóa double word 0 (VD0), tiếp điểm I0.1 đóng thì ghi số thực 30,2 vào double word VD0, kết quả như sau :

VW2 (word cao)	VW0 (word thấp)
20 (nhị phân)	30 (nhị phân)

7.8.2 Lệnh tăng giảm dữ liệu.

Lệnh Tăng Byte.

Tăng giá trị ngõ vào(IN) lên 1 và đưa kết quả sang ngõ ra(OUT).

Dạng LAD :	Dạng STL :
	INCB VB0

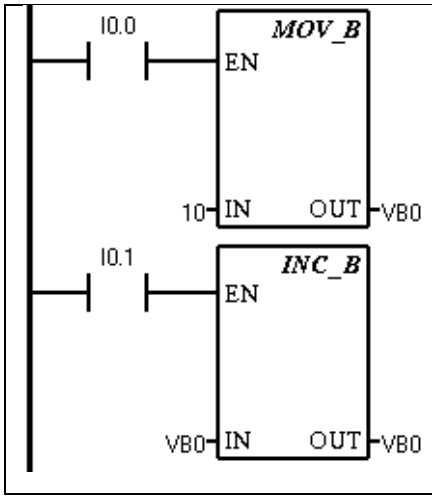
Ý nghĩa :

Lệnh này có tác dụng cộng số nguyên 1 đơn vị với nội dung của byte có địa chỉ ở ngõ vào IN, kết quả được ghi vào byte có địa chỉ ở ngõ ra OUT. Byte IN và byte OUT có thể cùng địa chỉ. Lệnh này có sử dụng các bit nhớ đặc biệt SM1.0, SM1.1, SM1.2 để báo trạng thái kết quả phép tính theo nguyên tắc như sau :

Kết quả tính	SM1.0	SM1.1	SM1.2
= 0	1		
Số âm			1
> byte		1	

Ví dụ :

Dạng LAD :



Dạng STL :

```
MOVB   10,  VB0

INCB     VB0
```

Giải thích :

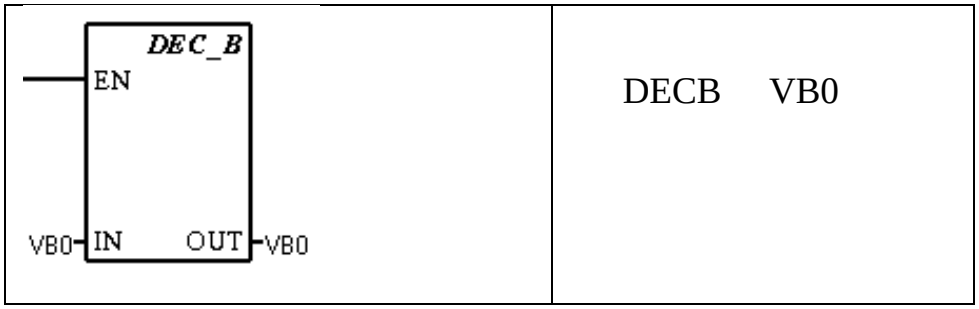
Tiếp điểm I0.0 đóng thì số 10 được ghi vào VB0, tiếp điểm I0.1 đóng thì nội dung của VB0 tăng lên 1 đơn vị và kết quả được lưu lại VB0. Lúc này VB0 = 11.

Tương tự như vậy người học có thể thử cho lệnh tăng Word, tăng Double Word.

Lệnh Giảm Byte.

Lệnh trừ nội dung của byte ngõ vào(IN) bớt 1 đơn vị sau đó lưu kết quả đạt được vào ngõ ra OUT.

Dạng LAD : Dạng STL :



Ý nghĩa :

Lệnh này có tác dụng lấy nội dung của byte có địa chỉ ở ngõ vào IN trừ đi 1 đơn vị , kết quả được ghi vào byte có địa chỉ ở ngõ ra OUT , byte IN và byte OUT có thể cùng địa chỉ và ở lệnh này cũng sử dụng các bit nhớ đặc biệt SM1.0, SM1.1, SM1.2 để báo trạng thái kết quả phép tính theo nguyên tắc như sau :

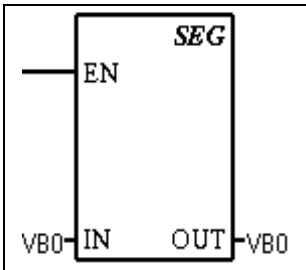
Kết quả tính	SM1.0	SM1.1	SM1.2
= 0	1		
Số âm			1
> byte		1	

Tương tự như lệnh giảm byte, người học tự kiểm tra lệnh giảm Word và Double Word.

7.9 Lệnh Chuyển đổi.

7.9.1 Chuyển đổi số nguyên sang mã led 7 đoạn:

Dạng LAD : Dạng STL :



1.1.1 SEG VB0, VB0

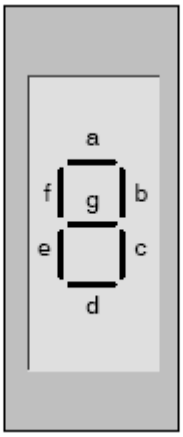
Ý nghĩa:

Lệnh này có tác dụng chuyển đổi các số trong hệ thập lục phân từ 0 đến F (dạng nhị phân) chứa trong 4 bit thấp của byte có địa chỉ ở ngõ vào IN thành giá trị BIT chứa trong 8 bit của byte có địa chỉ ở ngõ ra OUT tương ứng với thanh led 7 đoạn CK, địa chỉ ngõ ra có thể giống ngõ vào, nằm trong những vùng sau:

IN: VB, IB, QB, MB, SMB,AC, const

OUT: VB, IB,AB,MB,SMB,AC

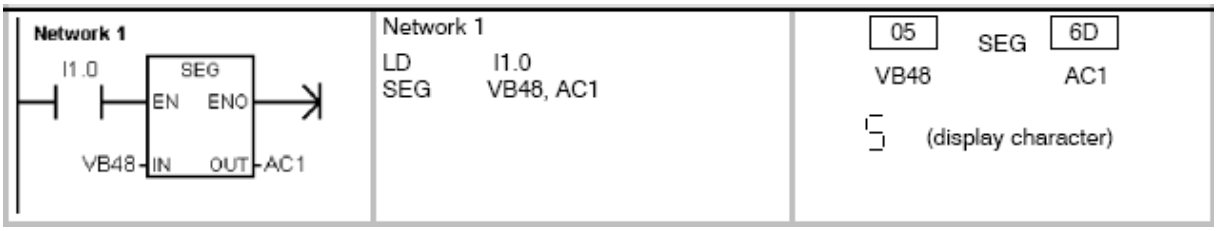
(IN) LSD	Segment Display	(OUT) - g f e d c b a
0	0	0011 1111
1	1	0000 0110
2	2	0101 1011
3	3	0100 1111
4	4	0110 0110
5	5	0110 1101
6	6	0111 1101
7	7	0000 0111



(IN) LSD	Segment Display	(OUT) - g f e d c b a
8	8	0111 1111
9	9	0110 0111
A	A	0111 0111
B	B	0111 1100
C	C	0011 1001
D	D	0101 1110
E	E	0111 1001
F	F	0111 0001

Hình 7.4: Bảng mã 7 đoạn cho led loại Catot Chung

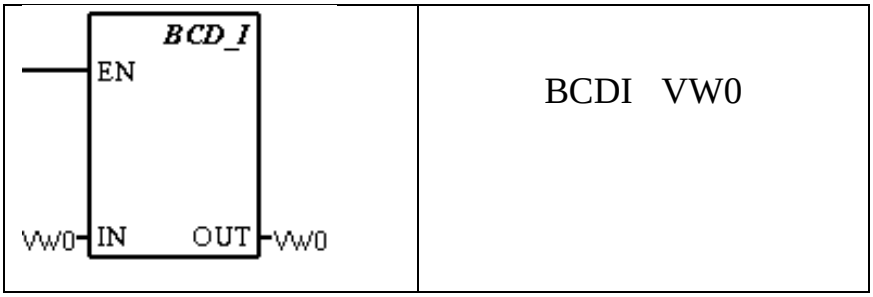
Ví dụ :



Giải thích: Giã sử trước khi chuyển đổi VB48 có giá trị bằng 05H, sau khi chuyển đổi sẽ cho kết quả là 6DH.

7.9.2 Lệnh chuyển đổi mã BCD sang số nguyên:

Dạng LAD : Dạng STL :



Ý nghĩa.

Lệnh này thực hiện phép biến đổi một số dạng mã BCD 16 bit chứa trong word có địa chỉ ở ngõ vào IN sang số nguyên dạng nhị phân 16 bit chứa trong word có địa chỉ ở ngõ ra OUT, địa chỉ ngõ ra có thể giống ngõ vào, thường nằm trong các vùng sau :

IN: VW, T, C, IW, QW, MW, SMW, AC, AIW, const

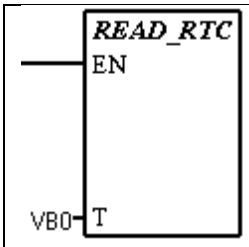
OUT: VW, T, C, IW, QW, MW, SMW, AC.

7.10 Lệnh về đồng hồ thời gian thực.

Trong thiết bị điều khiển lập trình PLC S7-200 kể từ CPU 214 trở đi thì trong CPU có một đồng hồ ghi giá trị thời gian thực gồm các thông số về Năm, tháng, ngày, giờ, phút, giây và ngày trong tuần. Đồng hồ này được cấp điện liên tục bởi nguồn pin 3V.

Khi thực hiện lập trình cho các hệ thống tự động điều khiển cần cập nhật giá trị đồng hồ thời gian thực này ta sử dụng 2 lệnh sau :

Lệnh đọc thời gian thực:

Dạng LAD :	Dạng STL :
	TODR VB0

Ý nghĩa :

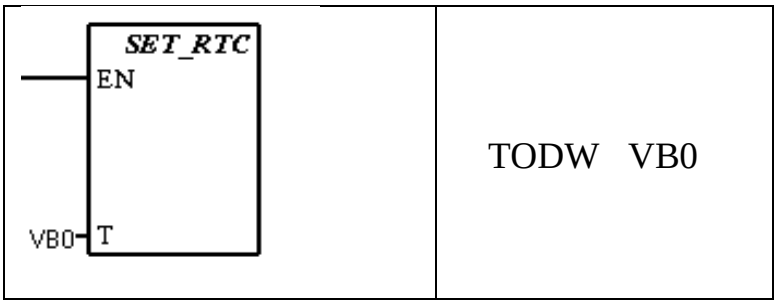
Lệnh này đọc nội dung của đồng hồ thời gian thực rồi chuyển sang mã BCD và lưu vào bộ đệm 8 byte liên tiếp nhau theo thứ tự như sau:

Byte 0	Năm (0 → 99)
Byte 1	Tháng (1 → 12)
Byte 2	Ngày (1 → 31)
Byte 3	Giờ (0 → 23)
Byte 4	Phút (0 → 59)
Byte 5	Giây (0 → 59)
Byte 6	0
Byte 7	Ngày trong tuần (1→7)

Trong đó byte đầu tiên được chỉ định bởi toán hạng T trong câu lệnh, byte 7 chỉ sử dụng 4 bit thấp để lưu giá trị các ngày trong tuần.

Lệnh đặt thời gian thực:

Dạng LAD : Dạng STL :



Ý nghĩa :

Lệnh này có tác dụng ghi nội dung của bộ đệm 8 byte với byte đầu tiên được chỉ định trong toán hạng T vào đồng hồ thời gian thực. Trong đó T thuộc 1 trong những vùng nhớ sau : VB, IB, QB, MB, SMB.

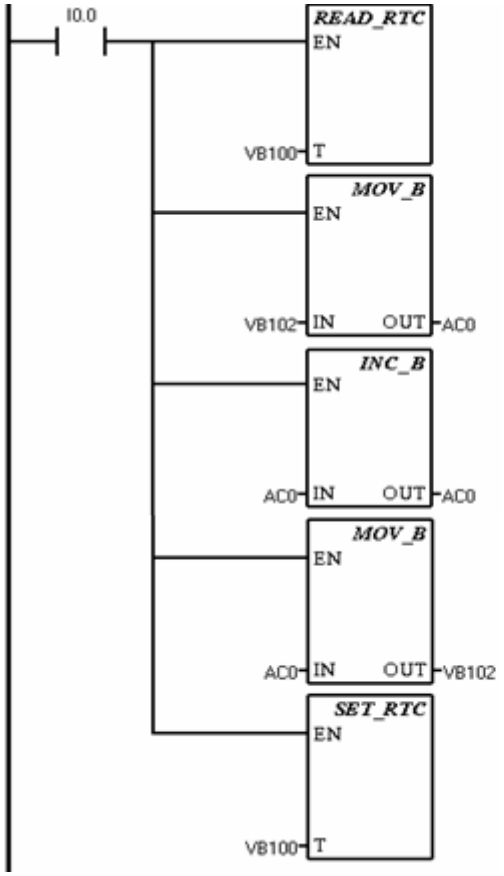
Nếu cần điều chỉnh các thônng số về năm, tháng, ngày, giờ, phút, giây, ngày trong tuần thì điều chỉnh các byte như sau :

Byte 0	Năm (0 → 99)
Byte 1	Tháng (1 → 12)
Byte 2	Ngày (1 → 31)
Byte 3	Giờ (0 → 23)
Byte 4	Phút (0 → 59)
Byte 5	Giây (0 → 59)
Byte 6	0
Byte 7	Ngày trong tuần (1→7)

Ví dụ :

Để điều chỉnh giá trị ngày ta làm như sau :

Dạng LAD



Dạng STL

```
LD    I0.0
TODR  VB100
MOVB  VB102, AC0
INCB  AC0
MOVB  AC0, VB102
TODW  VB100
```

Giá trị trước xử lý :

VB100	2003
B101	02
VB102	30
B103	09
VB104	20
VB105	35
VB106	0
VB107	4

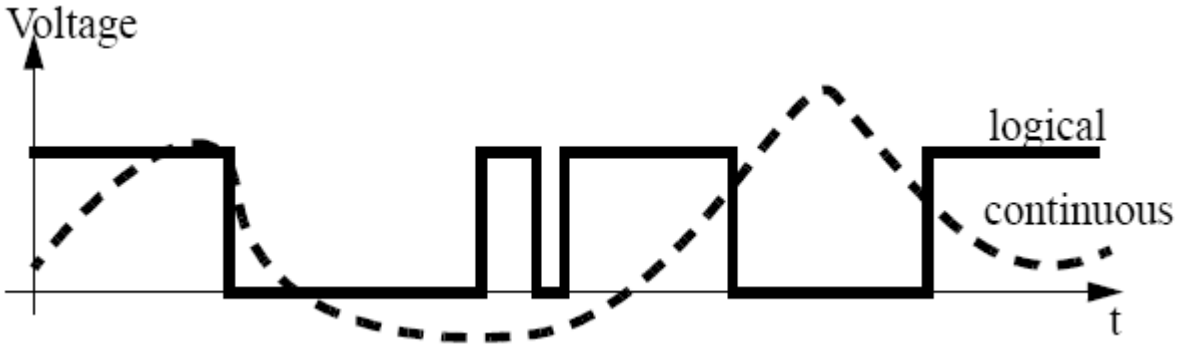
Giá trị sau khi xử lý :

VB100	2003
VB101	02
VB102	31
VB103	09
VB104	20
VB105	35
VB106	0
VB107	4

BÀI 8: TÍN HIỆU ANALOG

8.1 Tín hiệu analog và bộ A/D

Khác với tín hiệu số, ngõ vào và ngõ ra chỉ có hai trạng thái là ON hoặc OFF (mức 1 hoặc 0), tín hiệu analog có biên độ liên tục theo thời gian.

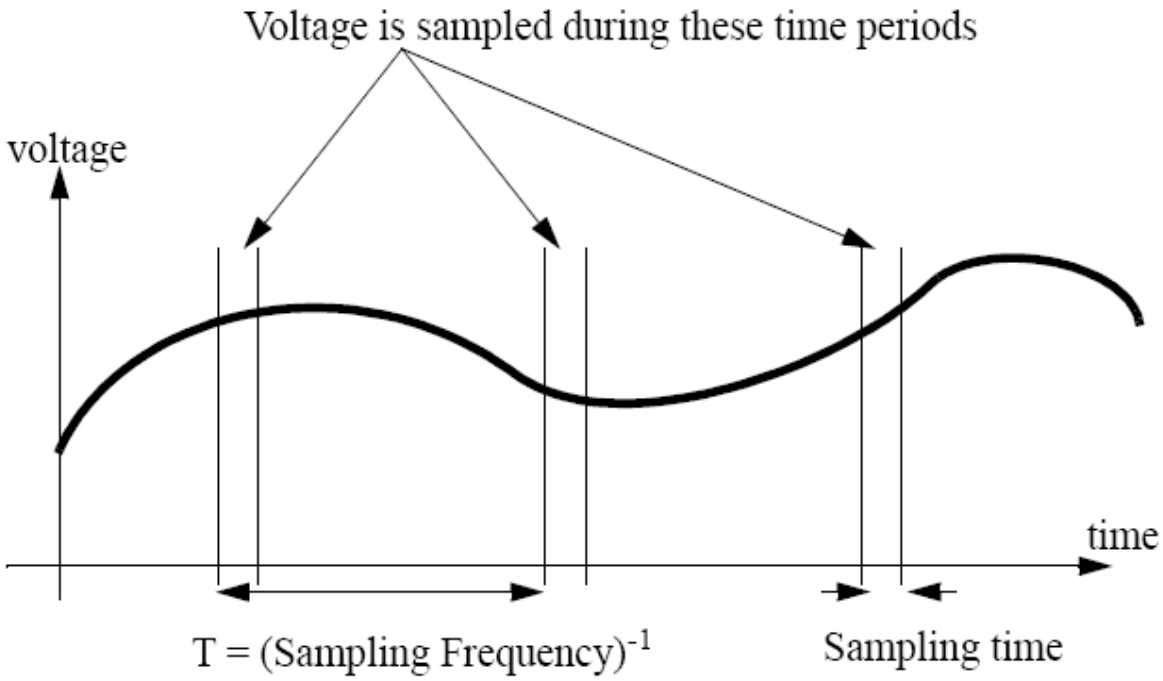


Hình 8.1: Sự khác biệt giữa tín hiệu số và tín hiệu tương tự

Phần lớn những hiện tượng xảy ra trong thực tế đều ở dạng analog. Các cảm biến ngõ có tín hiệu ra dạng analog như: Cảm biến nhiệt độ, cảm biến áp suất, cảm biến dòng chảy, cảm biến mức….. Những cơ cấu chấp hành có tín hiệu điều khiển dạng analog: Vale tuyến tính, biến tần.....

8.2 Thời gian lấy mẫu, tần số lấy mẫu.

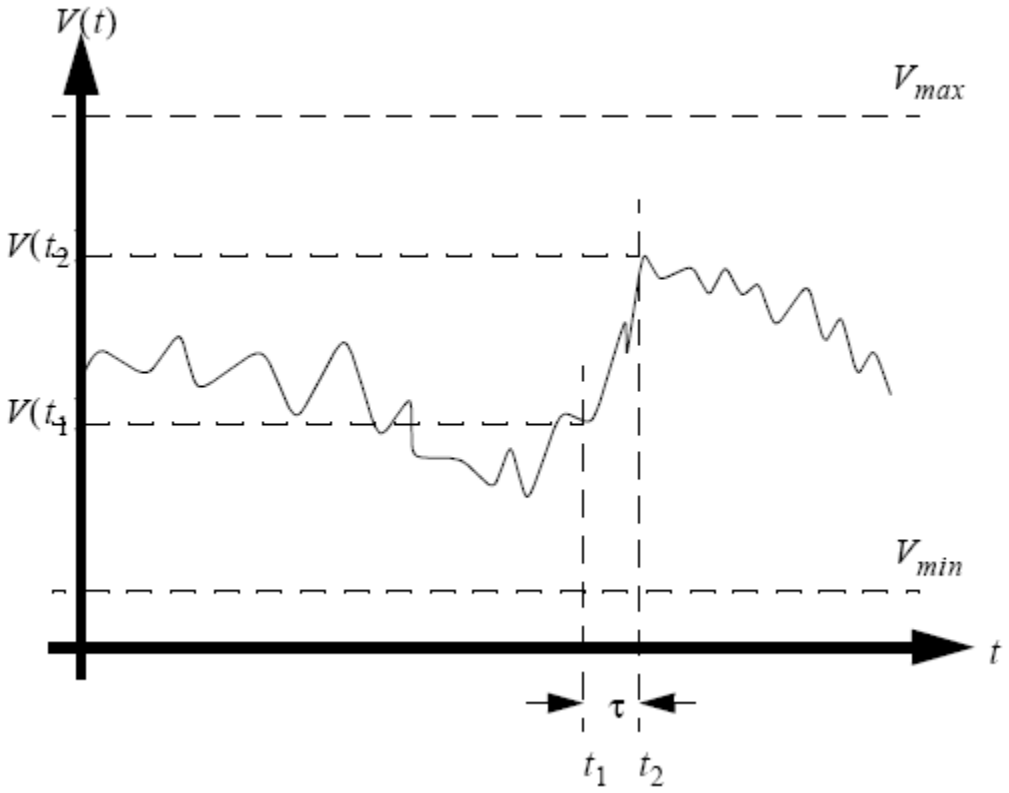
Đối với ngõ vào của PLC hay máy tính, tín hiệu analog không được đọc liên tục mà sẽ được lấy mẫu vào những khoảng thời gian nhất định. Sau đó tín hiệu analog được chuyển đổi sang tín hiệu số nhờ bộ A/D. Trong một khoảng thời gian nhất định, nếu số mẫu lấy càng nhiều thì độ chính xác càng tăng. Tuy nhiên mỗi bộ A/D chỉ có thể thu thập được một số mẫu nhất định trong một giây. Đối với PLC thì tần số lấy mẫu có thể đạt 20hZ.



Hình 8.2: Cách lấy mẫu tín hiệu tương tự

8.3 Các thông số của bộ A/D.

Một bộ A/D được đánh giá dựa vào các thông số như: Số bit chuyển đổi, thời gian lấy mẫu, tốc độ chuyển đổi, sai số chuyển đổi, tầm điện áp hoặc dòng điện mà bộ A/D có thể chuyển đổi. Các thông số này thường được cho bởi nhà sản xuất. Hình 8.3thể hiện các thông số của bộ A/D.



$V(t)$ = Tín hiệu analog.

τ = Khoảng thời gian lấy mẫu của bộ A/D.

t = Thời gian.

t_1, t_2 = Thời gian bắt đầu và kết thúc lấy mẫu.

$V(t_1), V(t_2)$ = Điện áp bắt đầu, kết thúc của mẫu.

V_{min}, V_{max} = Dãy điện áp ngõ vào của bộ A/D.

N = Số lượng bit của bộ A/D.

Hình 8.3: Các thông số cần quan tâm của bộ A/D

$$R = 2^N \quad (1)$$

$$V_{ERROR} = \left(\frac{V_{max} - V_{min}}{2R}\right) \quad (2)$$

$$\mathbf{N} = INT\left[\left(\frac{V_{in} - V_{min}}{V_{max} - V_{min}}\right)(R-1)\right] \quad (3)$$

$$V_C = \left(\frac{\mathbf{N}}{(R-1)}\right)(V_{max} - V_{min}) + V_{min} \quad (4)$$

Hình 8.4: Quan hệ giữa các thông số trong bộ A/D

Ghi chú:

Công thức 1: Quan hện giữa số lượng bit và độ phân giải của timer

Công thức 2: Sai số lượng tử.

Công thức 3: Quan hệ giữa giá trị số nguyên của bộ A/D với $V_{in,}$ V_{min}, V_{max}, $V_{min,}$ R.

Công thức 4: Quan hệ giữa điện áp đưa vào với N , V_{max}, $V_{min,}$ R.

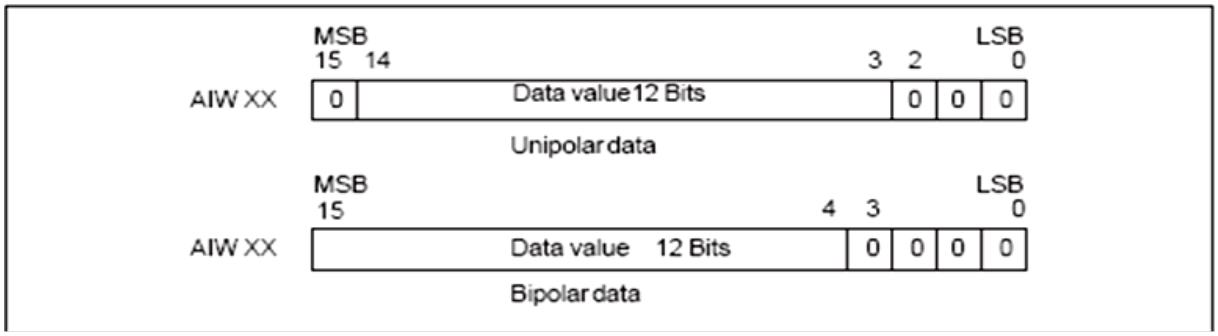
8.4 A/D trong PLC S7 200.

Đối với PLC thì bộ chuyển đổi A/D thường sử dụng 8 bit, 12 bit, 16 bit. Tùy theo yêu cầu kỹ thuật, độ chính xác, tính kinh tế mà người lập trình chọn bộ A/D nào cho phù hợp.

8.4.1 Cấu trúc dữ liệu của bộ A/D trong PLC S7 200.

Module analog trong S7 200 thường sử dụng loại 12 bit. Tín hiệu vào của module analog ở dạng điện áp hoặc dòng điện, điện áp có thể dương hoặc âm, dữ liệu chuyển đổi có thể ở dạng đơn cực hoặc lưỡng cực. Tùy thuộc vào dạng

chuyển đổi mà cách sắp xếp các bit dữ liệu cũng có sự khác nhau. Hình …. Trình bày cách sắp xếp các bit dữ liệu dạng đơn cực và lưỡng cực.



Hình 8.5: Cách định dạng các bit dữ liệu trong module analog của S7 200

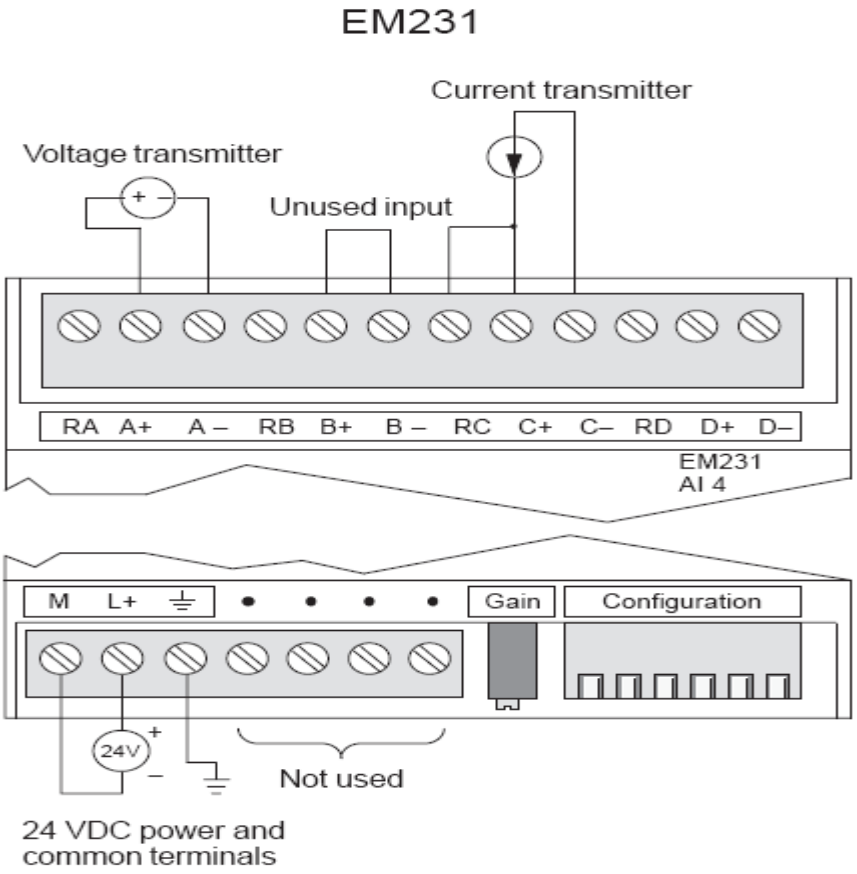
8.4.2 Một số module analog của S7 200 thường được sử dụng.

8.4.2.1 Module analog EM231

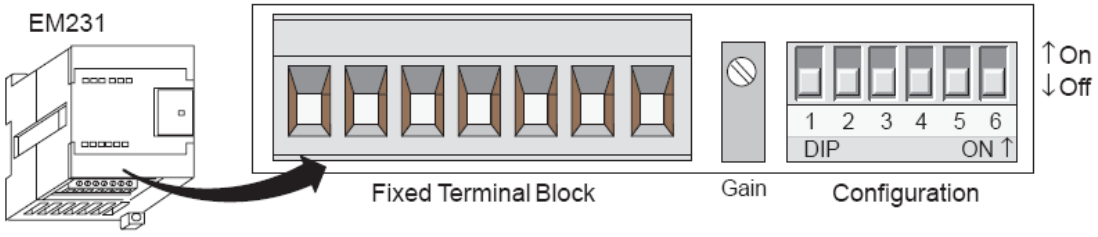
Các thông số kỹ thuật.

Module	EM 231	EM 231
Number of inputs	4	8
Input type	0-10V / 0-20mA	0-10V / 0-20mA
Voltage ranges	0-10V, 0-5V +/- 5V, +/- 2,5V	0-10V, 0-5V, +/- 5V, +/- 2,5V (Ch 0-5) 0-10V, 0-5V, +/- 5V +/- 2,5V,0-20mA (Ch 6-7)
Resolution	12 Bit	12 Bit
Isolation	no	no
Dimensions (W x H x D)	71,2 x 80 x 62 mm	71,2 x 80 x 62 mm

Cách kết nối ngõ vào.



Switch chọn giá trị ngõ vào và độ phân giải.



Unipolar			Full-Scale Input	Resolution
SW1	SW2	SW3		
ON	OFF	ON	0 to 10 V	2.5 mV
	ON	OFF	0 to 5 V	1.25 mV
			0 to 20 mA	5 µA
Bipolar			Full-Scale Input	Resolution
SW1	SW2	SW3		
OFF	OFF	ON	± 5 V	2.5 mV
	ON	OFF	± 2.5 V	1.25 mV

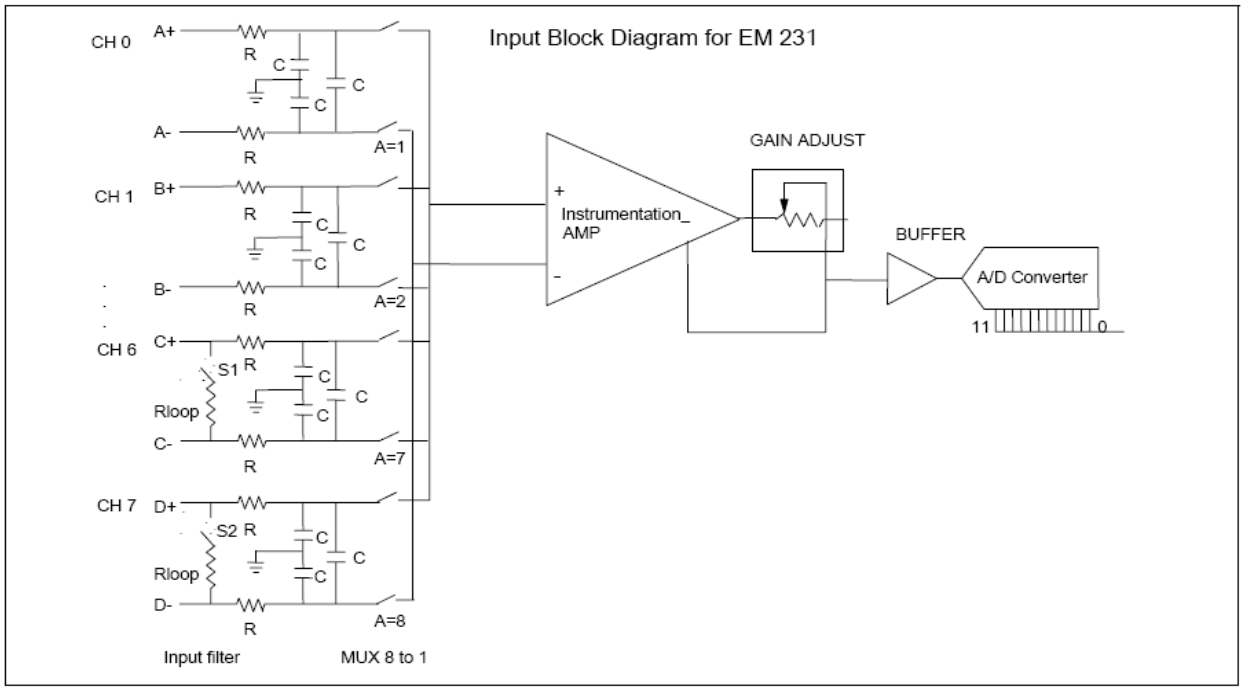
Lưu ý:

Dòng điện ngõ vào: 0 đến 20mA.

Độ phân giải: 5uA hay từ 1,25mV đến 2,5mV.

Giá trị số ngõ vào: -32000 đến 32000(lưỡng cực) hay từ 0 đến 32000(đơn cực).

Mạch ngõ vào của Module EM231.

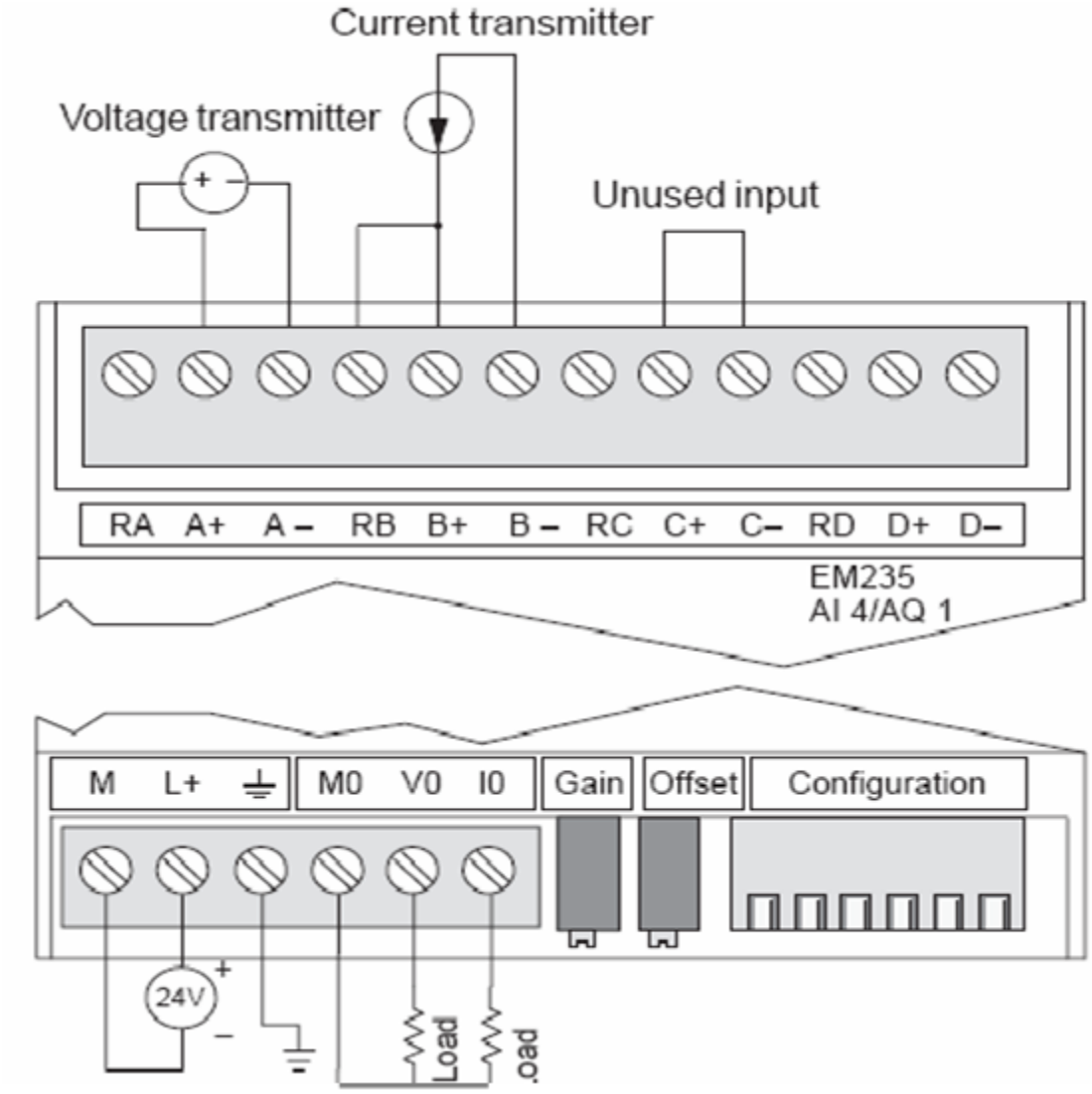


8.4.2.2 Module analog EM235

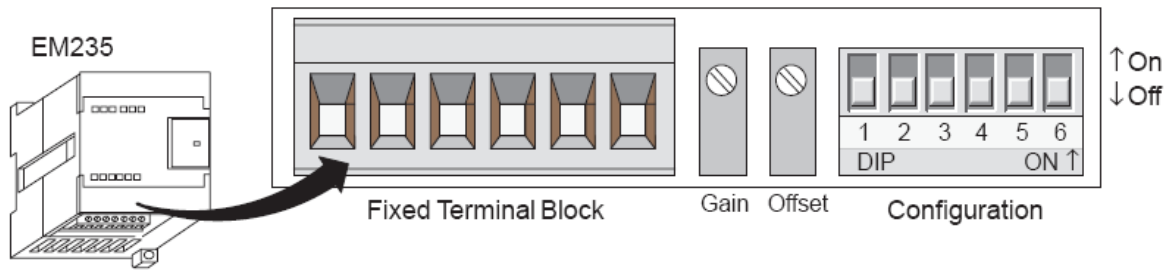
Các thông số kỹ thuật.

Module	EM 235
Number of inputs	4
Input type	0-10V / 0-20mA
Voltage ranges	0-10V / 0-5 V
Resolution	12 Bit
Isolation	no
Number of outputs	1
Output type	0-10V / 0-20mA
Resolution	12 Bit voltage 11 Bit current
Isolation	no
Dimensions (W x H x D)	71,2 x 80 x 62 mm

Cách kết nối ngõ vào, ngõ ra.



Switch chọn giá trị ngõ vào và độ phân giải.



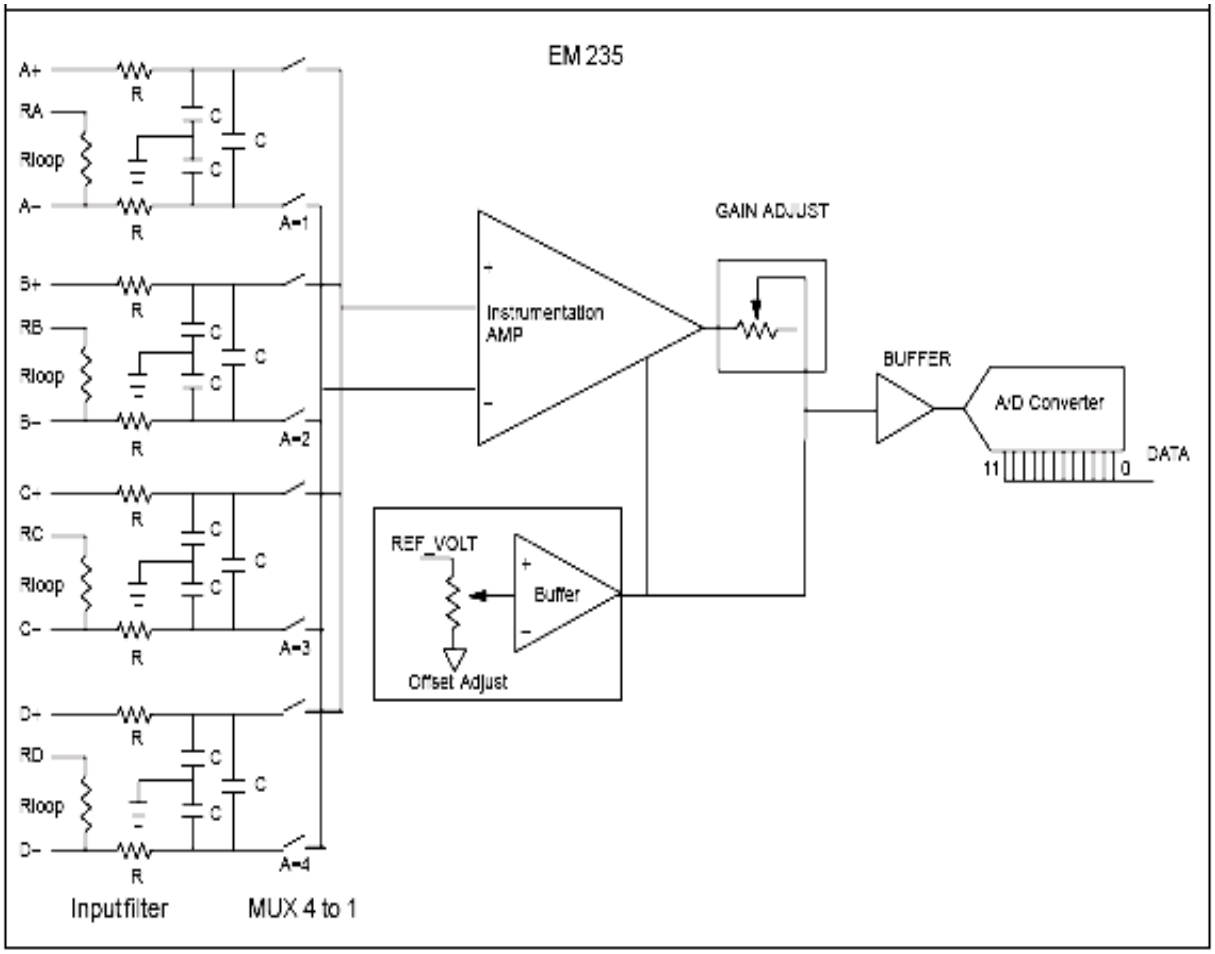
Configuration Switch						Full-Scale Input	Resolution
1[1]	3	5	7	9	11		
ON	ON	OFF	ON	OFF	OFF	0 to 50 mV	12.5 µV
ON	ON	OFF	OFF	ON	OFF	0 to 100 mV	25 µV
ON	OFF	ON	ON	OFF	OFF	0 to 500 mV	125 µV
ON	OFF	ON	OFF	ON	OFF	0 to 1 V	250 µV
ON	OFF	OFF	ON	OFF	OFF	0 to 5 V	1.25 mV
ON	OFF	OFF	ON	OFF	OFF	0 to 20 mA[2]	5 µA
ON	OFF	OFF	OFF	ON	OFF	0 to 10 V	2.5 mV
OFF	ON	OFF	ON	OFF	OFF	±25 mV	12.5 µV
OFF	ON	OFF	OFF	ON	OFF	±50 mV	25 µV
OFF	ON	OFF	OFF	OFF	ON	±100 mV	50 µV
OFF	OFF	ON	ON	OFF	OFF	±250 mV	125 µV
OFF	OFF	ON	OFF	ON	OFF	±500 mV	250 µV
OFF	OFF	ON	OFF	OFF	ON	±1 V	500 µV
OFF	OFF	OFF	ON	OFF	OFF	±2.5 V	1.25 mV
OFF	OFF	OFF	OFF	ON	OFF	±5 V	2.5 mV
OFF	OFF	OFF	OFF	OFF	ON	±10 V	5 mV

Lưu ý:

Độ phân giải: 5uA hay từ 12,5uV đến 5mV.

Giá trị số ngõ vào: -32000 đến 32000 hay từ 0 đến 32000.

Mạch ngõ vào của Module EM235.



8.4.2.3 Module analog EM232

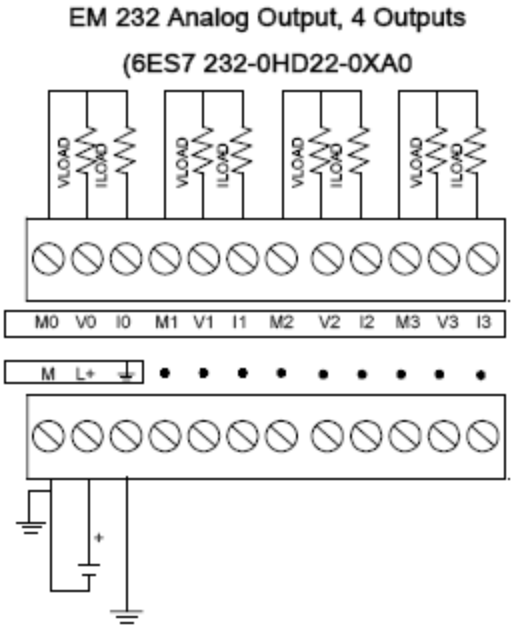
Các thông số kỹ thuật.

Module	EM 232	EM 232
Number of outputs	2	4
Output type	0-10V / 0-20mA	0-10V / 0-20mA
Resolution	12 Bit voltage 11 Bit current	12 Bit voltage 11 Bit current
Isolation	no	no
Dimensions (W x H x D)	46 x 80 x 62 mm	71,2 x 80 x 62 mm

Một vài thông số chi tiết.

VDC requirements +5 VDC + 24 V	 20 mA 60 mA (with outputs at 20 mA)
Number of outputs	4
Removable connector	no
Type	Voltage and output
Isolation (field to logic)	None
Signal range Voltage output Current output	 ± 10 V 0 to 20 mA
Resolution, full-scale Voltage Current	 11 bits plus 1 sign bit 11 bits
Data word format Voltage Current	 -32000 to +32000 0 to +32000
Accuracy	± 0.5% at 25° C ± 2.0% 0° C to 55° C
Setting time Voltage output Current output	 100 µS 2 mS

Cách kết nối ngõ ra.



8.4.3 Hiệu chỉnh giá trị analog.

Module analog thường có nhiều tầm đo khác nhau, tín hiệu ngõ vào có thể là dòng điện hoặc điện áp. Việc chuyển đổi từ tầm đo này sang tầm đo khác thì kết quả chuyển đổi thường có những sai số nhất định do cấu trúc của mạch chuyển đổi. Do vậy thông thường khi sử dụng module analog, người lập trình cần phải hiệu chỉnh trước khi sử dụng để kết quả chuyển đổi được chính xác hơn. Dưới đây trình bày việc hiệu chỉnh cho ngõ vào là điện áp, tầm đo 10V, ngõ vào chuyển đổi là AIW0.

- Cấp điện cho module analog hoạt động khoảng 10 phút.
- Chọn điện áp vào là 10V (độ phân giải 2,5mV)
- Chỉnh biến trở tại ngõ vào AIW0 để ngõ vào đạt giá trị 0V.
- Dùng chương trình đọc giá trị analog vào và quan sát giá trị. Nếu chưa bằng không thì hiệu chỉnh độ lợi (Gain) để đạt giá trị = 0.
- Chỉnh biến trở tại ngõ vào AIW0 để ngõ vào đạt giá trị 10V.
- Dùng chương trình đọc giá trị analog vào và quan sát giá trị. Nếu chưa bằng 32000 thì hiệu chỉnh độ lợi (Gain) để đạt giá trị = 32000.

8.5 Bài tập ứng dụng:

8.5.1 Bài tập về cảm biến nhiệt độ LM35.

Một cảm biến nhiệt độ LM 35 có thông số kỹ thuật như sau:

Tầm đo từ -55 độ C đến 150 độ C.

Tỉ lệ chuyển đổi từ nhiệt độ sang điện áp: 10mV/độ C.

Cảm biến này được sử dụng để đo nhiệt độ của một lò sấy.

Dùng PLC S7 200 và module analog EM231 để đọc đọc tín hiệu của LM35 và điều khiển hoạt động của lò. Biết rằng lò được điều khiển tại ngõ ra Q0.0

Yêu cầu:

Vẽ sơ đồ kết nối phần cứng.

Viết chương trình điều khiển: Khi nhiệt độ nhỏ hơn 50 độ C thì mở lò. Khi nhiệt độ lớn hơn 60 độ C thì tắt lò.

Viết chương trình hiển thị nhiệt độ của lò theo độ C và lưu kết quả vào vùng nhớ MW0.

8.5.2 Bài tập về Loadcell.

Một Load cell có thông số kỹ thuật như sau:

Tầm đo từ 0 đến 1000 kg.

Điện áp kích thích: 12-24V.

Độ nhạy: 1,96mV/V.

Cảm biến này được sử dụng để đo khối lượng của một bể nước.

Dùng PLC S7 200 và module analog EM231 để đọc đọc tín hiệu của Loadcell này và điều khiển hoạt động của 1 bơm. Biết rằng bơm được điều khiển tại ngõ ra Q0.0

Yêu cầu:

Vẽ sơ đồ kết nối phần cứng.

Viết chương trình điều khiển: Khi khối lượng nước trong bể nhỏ hơn 600kg thì bơm, khi khối lượng nước trong bể lớn hơn 800kg thì ngừng bơm .

Viết chương trình hiển thị khối lượng nước của bể theo kg và lưu kết quả vào vùng nhớ MW0.

8.5.3 Bài tập điều khiển công suất lò nhiệt.

Một bộ điều khiển công suất của lò nhiệt có ngõ vào từ 0 đến 10V tương ứng với công suất của lò từ 0 đến 100%. Hãy vẽ sơ đồ khối kết nối, viết chương trình để thực hiện yêu cầu trên.

BÀI 9: HOẠT ĐỘNG NGẮT TRONG PLC

9.1 Giới thiệu về ngắt trong PLC.

Ngắt là quá trình mà PLC dừng chương trình đang thực thi để thực hiện chương trình ngắt khi có những sự kiện ngắt xảy ra. Những sự kiện gây ra ngắt có thể là từ bên ngoài(ngắt ngoài) hay từ bên trong PLC(Ngắt xảy ra bên trong PLC). Khi một sự kiện ngắt xuất hiện thì chương trình chính ngay lập tức bị dừng lại để thực thi chương trình ngắt(trình phục vụ ngắt). Sau khi thực hiện xong chương trình ngắt thì PLC sẽ quay về chương trình chính và thực hiện tiếp tại vị trí trước khi xảy ra ngắt.

Khi có nhiều yêu cầu ngắt xảy ra đồng thời thì các ngắt sẽ thực hiện theo thứ tự ưu tiên. Ngắt có mức ưu cao được thực hiện trước, ngắt có mức ưu tiên thấp hơn thực hiện sau. Tuy nhiên khi một ngắt đang thực thi thì nó vẫn tiếp tục thực hiện cho đến khi hoàn thành bất chấp các ngắt khác có mức ưu tiên cao hơn đang yêu cầu.

Các ngắt trong PLC thường tập trung vào 3 nhóm chính: Ngắt truyền thông, Ngắt I/O, Ngắt định thời.

9.2 Hoạt động ngắt trong PLC S7 200.

Cũng giống như các PLC khác, S7 200 cũng hỗ trợ các ngắt về truyềnt thông, ngắt I/O và ngắt định thời. Tùy thuộc vào loại CPU mà số lượng ngắt cũng như sự kiện ngắt có khác nhau.

Inputs/Outputs	Data Types	Operands		
INT	BYTE	Constant (0 to 127)		
EVNT	BYTE	Constant	*CPU 221 and CPU 222:*	0 to 12, 19 to 23, and 27 to 33
			CPU 224:	0 to 23 and 27 to 33
			CPU 226 and CPU 226XM:	0 to 33

Hình 9.1: Sô lượng và sự kiện ngắt trong S7 200

9.3 Các sự kiện gây ra ngắt trong S7 – 200.

Event	Description		CPU 221 CPU 222	CPU 224	CPU 226 CPU 226XM
0	I0.0	Rising edge	Y	Y	Y
1	I0.0	Falling edge	Y	Y	Y
2	I0.1	Rising edge	Y	Y	Y
3	I0.1	Falling edge	Y	Y	Y
4	I0.2	Rising edge	Y	Y	Y
5	I0.2	Falling edge	Y	Y	Y
6	I0.3	Rising edge	Y	Y	Y
7	I0.3	Falling edge	Y	Y	Y
8	Port 0	Receive character	Y	Y	Y
9	Port 0	Transmit complete	Y	Y	Y
10	Timed interrupt 0	SMB34	Y	Y	Y
11	Timed interrupt 1	SMB35	Y	Y	Y
12	HSC0	CV=PV (current value = preset value)	Y	Y	Y
13	HSC1	CV=PV (current value = preset value)		Y	Y
14	HSC1	Direction changed		Y	Y
15	HSC1	External reset		Y	Y
16	HSC2	CV=PV (current value = preset value)		Y	Y
17	HSC2	Direction changed		Y	Y
18	HSC2	External reset		Y	Y
19	PLS0	PTO pulse count complete interrupt	Y	Y	Y
20	PLS1	PTO pulse count complete interrupt	Y	Y	Y
21	Timer T32	CT=PT interrupt	Y	Y	Y
22	Timer T96	CT=PT interrupt	Y	Y	Y
23	Port 0	Receive message complete	Y	Y	Y
24	Port 1	Receive message complete			Y
25	Port 1	Receive character			Y
26	Port 1	Transmit complete			Y
27	HSC0	Direction changed	Y	Y	Y
28	HSC0	External reset	Y	Y	Y
29	HSC4	CV=PV (current value = preset value)	Y	Y	Y
30	HSC4	Direction changed	Y	Y	Y
31	HSC4	External reset	Y	Y	Y
32	HSC3	CV=PV (current value = preset value)	Y	Y	Y
33	HSC5	CV=PV (current value = preset value)	Y	Y	Y

Hình 9.2: Các sự kiện ngắt trong S7 200

9.4 Thứ tự ưu tiên ngắt trong PLC S7 200

Khi có nhiều ngắt xảy ra đồng thời thì chương trình sẽ xảy ra theo thứ tự ưu tiên. Dưới đây là thứ tự ưu tiên của các ngắt trong PLC S7 200.

Event	Description		Priority Group	Priority in Group
8	Port 0	Receive character	Communications *Highest Priority*	0
9	Port 0	Transmit complete		0
23	Port 0	Receive message complete		0
24	Port 1	Receive message complete		1
25	Port 1	Receive character		1
26	Port 1	Transmit complete		1
19	PLS0	PTO pulse count complete interrupt	Discrete *Medium Priority*	0
20	PLS1	PTO pulse count complete interrupt		1
0	I0.0	Rising edge		2
2	I0.1	Rising edge		3
4	I0.2	Rising edge		4
6	I0.3	Rising edge		5
1	I0.0	Falling edge		6
3	I0.1	Falling edge		7
5	I0.2	Falling edge		8
7	I0.3	Falling edge		9
12	HSC0	CV=PV (current value = preset value)		10
27	HSC0	Direction changed		11
28	HSC0	External reset		12
13	HSC1	CV=PV (current value = preset value)		13
14	HSC1	Direction changed		14
15	HSC1	External reset		15
16	HSC2	CV=PV (current value = preset value)		16
17	HSC2	Direction changed		17
18	HSC2	External reset		18
32	HSC3	CV=PV (current value = preset value)		19
29	HSC4	CV=PV (current value = preset value)		20
30	HSC4	Direction changed		21
31	HSC4	External reset		22
33	HSC5	CV=PV (current value = preset value)		23
10	Timed interrupt 0	SMB34	Timed *Lowest Priority*	0
11	Timed interrupt 1	SMB35		1
21	Timer T32	CT=PT interrupt		2
22	Timer T96	CT=PT interrupt		3

Hình 9.3: Thứ tự ưu tiên ngắt trong S7 200

9.5 Các lệnh sử dụng khi lập trình điều khiển ngắt.

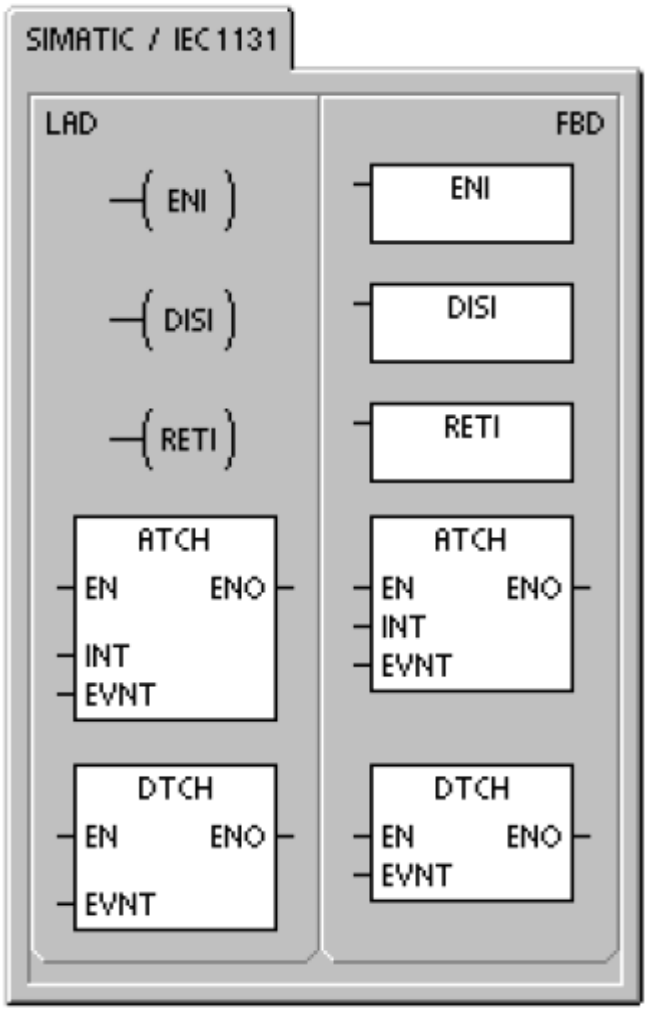
Lệnh cho phép ngắt toàn cục

Lệnh cấm ngắt toàn cục

Lệnh thoát khỏi chương trình ngắt.

Lệnh gán chương trình ngắt với sự kiện ngắt tương ứng.

Lệnh loại bỏ chương trình ngắt với sự kiện ngắt tương ứng.



Hình 9.4: Các lệnh sử dụng khi lập trình với ngắt

Lưu ý:

Một chương trình ngắt có thể được gọi bởi nhiêu sự kiện ngắt, Tuy nhiên một sự kiện ngắt thì không thể gán cho nhiều chương trình ngắt.

9.6 Lập trình khi sử dụng ngắt.

Tất cả các ngắt phải được khởi tạo trước khi làm việc. Việc khởi tạo các ngắt chỉ thực hiện một lần trong chương trình, chỉ khi nào cần thay đổi các thông số(chương trình ngắt, sự kiện ngắt) thì mới khởi tạo lại. Vì vậy thông thường chương trình khởi tạo được viết trong một chương trình con và được gọi một lần (SM0.1) khi chạy chương trình.

Khi sử dụng ngắt, người lập trình cần xác định các đặc điểm sau đây.

Loại ngắt: Ngắt truyền thông, ngắt I/O hay ngắt định thời.

Điều kiện ngắt: Khi nào sự kiện ngắt xảy ra, chuẩn bị các điều kiện để ngắt xảy ra.

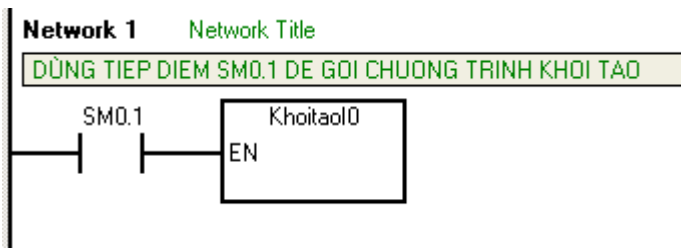
Khởi tạo ngắt: Gán chương trình ngắt và sự kiện ngắt tương ứng, cho phép ngắt, Lập trình cho chương trình ngắt.

Lưu ý: Chương trình viết trong chương trình ngắt phải theo nguyên tắc **"Càng ngắn gọn càng hiệu quả"**

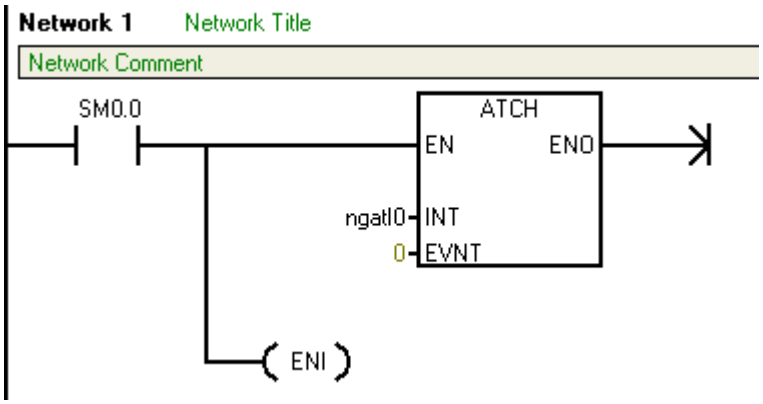
Mỗi ngắt đều có cách khởi tạo riêng, dưới đây sẽ trình bày việc khởi tạo một số ngắt cơ bản.

9.6.1 Khởi tạo ngắt cạnh lên tại I0.0 (Sự kiện 0).

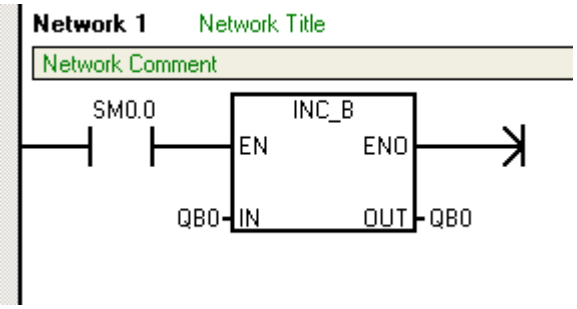
Chương trình chính.(Gọi chương trình con khởi tạo ngắt).



Chương trình con KhoitaoI0: (Gán chương trình ngắt với sự kiện ngắt tương) ứng, cho phép ngắt).



Chương trình ngắt ngatI0:



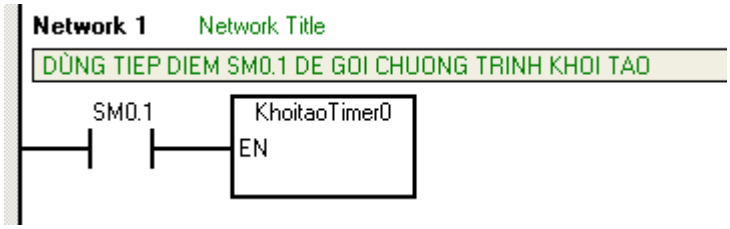
Chương trình ngắt:

Mỗi lần có cạnh lên tại I0.0 thì chương trình ngắt được gọi. Trong ví dụ này, chương trình ngắt có nhiệm vụ tăng giá trị QB0 thêm 1.

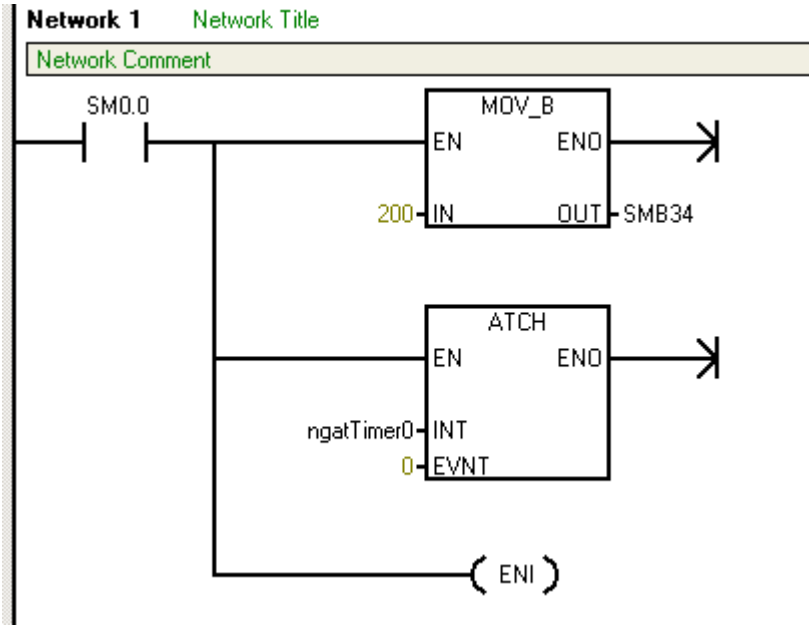
Tương tự như trên, người lập trình có thể khởi tạo ngắt cho các ngõ vào còn lại.

9.6.2 Khởi tạo ngắt định thời 0: Timed interrupt0

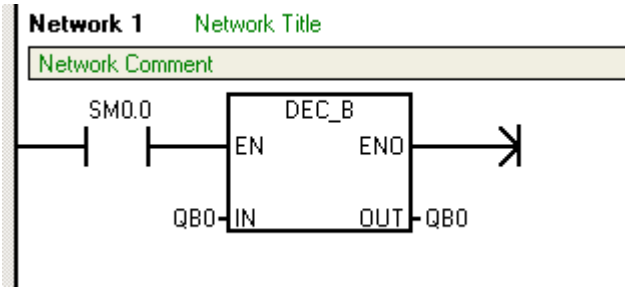
Chương trình chính.(Gọi chương trình con khởi tạo ngắt).



Chương trình con KhoitaoTimer0: (Nạp thời gian gây ra ngắt, gán chương trình ngắt với sự kiện ngắt tương ứng, cho phép ngắt).



Chương trình ngắt ngatTimer0: Sau một khoảng thời gian 200mS thì chương trình ngắt được gọi. Trong ví dụ này, chương trình ngắt có nhiệm vụ giảm giá trị QB0 thêm 1.



Tương tự như trên, người lập trình có thể khởi tạo ngắt cho Timer1

BÀI 10: BỘ ĐẾM TỐC ĐỘ CAO
(HSC: HIGH SPEED COUNTER).

10.1 Giới thiệu về HSC.

Bộ đếm thường: Bộ đếm thường trong PLC như đếm lên (CTU), đếm xuống(CTD), đếm lên xuông(CTUD), chỉ đếm được các sự kiện xảy ra với tần số thấp (Chu kỳ xuất hiện của sự kiện nhỏ hơn chu kỳ quét của PLC).

HSC là bộ đếm tốc độ cao có khả năng đếm được những sự kiện xảy ra với tần số lớn mà các bộ đếm thông thường trong PLC không đếm được.

VD: Tín hiệu xung từ encoder, tín hiệu xung từ động cơ servo…

10.2 Số lượng bộ đếm HSC có trong PLC và tần số tối đa cho phép.

Tùy thuộc vào loại CPU mà số lượng bộ đếm HSC và tốc độ tối đa cho phép khác nhau.

Bộ đếm	Ngõ vào	Tần số cho phép	Loại CPUs
HC0	I0.0	30 kHz	221,222,224,224XP,226
HC1	I0.6	30 kHz	221,222,224,224XP,226
HC2	I1.2	30 kHz	221,222,224,224XP,226
HC3	I0.1	30 kHz	221,222,224,224XP,226
HC4	I0.3	200 kHz	224XP
HC5	I0.4	200 kHz	224XP

Khi sử dụng HSC, người lập trình cần lưu ý một số đặc điểm sau đây.

- Tần số đếm của HSC phù thuộc vào loại HSC và loại CPU.
- Chỉ có một số ngõ vào đặc biệt mới nhận được xung tần số cao.
- Thường thì PLC của hãng nào cũng có HSC. Tuy nhiên có hãng thì tích hợp HSC sẵn trong CPU, có hãng thì thiết kế theo kiểu từng module riêng.

10.3 Vùng nhớ đặc biệt sử dụng để lập trình cho HSC.

Mỗi HSC có một vùng nhớ đặc biệt riêng, vùng nhớ này được sử dụng để khai báo chọn mode đếm, đặt giá trị, lưu giá trị đếm cho HSC tương ứng.

STT	Bộ đếm	Vùng nhớ khai báo	Chú thích
1	HSC0	SMB36 đến SMB45	Mỗi HSC sử dụng 10 byte
2	HSC1	SMB46 đến SMB55	
3	HSC2	SMB56 đến SMB65	
4	HSC3	SMB136 đến SMB145	
5	HSC4	SMB146 đến SMB155	
6	HSC5	SMB156 đến SMB165	

Mỗi HSC được hỗ trợ 10 byte để khai báo thông số. Tuy nhiên các byte tương ứng của các HSC đều có chức năng cơ bản là giông nhau. Vì vậy người lập trình chỉ cần tìm hiểu kỹ cách sử dụng một HSC là có thể sử dụng được các bộ đếm khác.

10.4 Các Mode đếm của bộ đếm:

Mỗi bộ đếm đều có những Mode đếm khác nhau. Tùy vào từng ứng dụng cụ thể mà người lập trình lựa chọn Mode đếm cho phù hợp. Bảng 10.1 mô tả các mode đếm của HSC.

Mode	Description	Inputs			
	HSC0	I0.0	I0.1	I0.2	
	HSC1	I0.6	I0.7	I1.0	I1.1
	HSC2	I1.2	I1.3	I1.4	I1.5
	HSC3	I0.1			
	HSC4	I0.3	I0.4	I0.5	
	HSC5	I0.4			
0	Single-phase counter with internal direction control	Clock			
1		Clock		Reset	
2		Clock		Reset	Start
3	Single-phase counter with external direction control	Clock	Direction		
4		Clock	Direction	Reset	
5		Clock	Direction	Reset	Start
6	Two-phase counter with 2 clock inputs	Clock Up	Clock Down		
7		Clock Up	Clock Down	Reset	
8		Clock Up	Clock Down	Reset	Start
9	A/B phase quadrature counter	Clock A	Clock B		
10		Clock A	Clock B	Reset	
11		Clock A	Clock B	Reset	Start

Bảng 10.1: Các mode đếm của HSC

Lưu ý:

Mỗi mode đếm khác nhau thường có khác nhau ở một vài đặc điểm như: Ngõ vào nhận xung, điều khiển hướng đếm, tác động reset, tác động start, đếm lệch pha… Người lập trình phải chú ý đặc điểm này để tránh nhầm lẫn khi sử dụng.

10.5 Chức năng các bit trong byte trạng thái của các bộ HSC.

10.5.1 Byte trạng thái của HSC0.

SM36.0	Không sử dụng
SM36.1	Không sử dụng
SM36.2	Không sử dụng
SM36.3	Không sử dụng
SM36.4	Không sử dụng

SM36.5	Chiều đang đếm, 1:Đếm lên, 0:Đếm xuống.
SM36.6	Kết quả so sánh tức thời, 0:Nếu CV ≠ PV, 1:Nếu CV = PV
SM36.7	Kết quả so sánh tức thời, 0:Nếu CV ≤ PV, 1:Nếu CV > PV

10.5.2 Byte trạng thái của HSC1.

SM46.0	Không sử dụng
SM46.1	Không sử dụng
SM46.2	Không sử dụng
SM46.3	Không sử dụng
SM46.4	Không sử dụng
SM46.5	Chiều đang đếm, 1:Đếm lên, 0:Đếm xuống.
SM46.6	Kết quả so sánh tức thời, 0:Nếu CV ≠ PV, 1:Nếu CV = PV
SM46.7	Kết quả so sánh tức thời, 0:Nếu CV ≤ PV, 1:Nếu CV > PV

10.5.3 Byte trạng thái của HSC2.

SM56.0	Không sử dụng
SM56.1	Không sử dụng
SM56.2	Không sử dụng
SM56.3	Không sử dụng
SM56.4	Không sử dụng
SM56.5	Chiều đang đếm, 1:Đếm lên, 0:Đếm xuống.

SM56.6	Kết quả so sánh tức thời, 0:Nếu CV ≠ PV, 1:Nếu CV = PV
SM56.7	Kết quả so sánh tức thời, 0:Nếu CV ≤ PV, 1:Nếu CV > PV

10.6 Ý nghĩa các bit của byte điều khiển của các bộ HSC.

10.6.1 Byte điều khiển của HSC0

SM37.0	Không sử dụng
SM37.1	Không sử dụng
SM37.2	Không sử dụng
SM37.3	Chiều đếm: 0 đếm lùi, 1 : đếm lên
SM37.4	Cho phép đổi chiều đếm, 0: không cho phép, 1: cho phép.
SM37.5	Cho phép sửa đổi giá trị đặt trước.0: không cho phép. 1: cho phép
SM37.6	Cho phép sửa đổi giá trị đếm tức thời, 0: không cho phép, 1: cho phép
SM37.7	1- Cho phép kích HSC0, 0:Không cho phép HSC0

10.6.2 Byte điều khiển của HSC1.

SM47.0	Kiểu reset cho tín hiệu xóa tại cổng I1.0
SM47.1	Kiểu start cho tín hiệu kích tại cổng I1.1
SM47.2	Tần số đếm của HSC1

SM47.3	Chiều đếm: 0 đếm lùi, 1 : đếm lên
SM47.4	Cho phép đổi chiều đếm, 0: không cho phép, 1: cho phép
SM47.5	Cho phép sửa đổi giá trị đặt trước, 0: không cho phép, 1: cho phép
SM47.6	Cho phép sửa đổi giá trị đếm tức thời, 0: không cho phép, 1: cho phép
SM47.7	1- Cho phép kích HSC1, 0:Không cho phépHSC1

10.6.3Byte điều khiển của HSC2.

SM57.0	Kiểu reset cho tín hiệu xóa tại cổng I1.0
SM57.1	Kiểu start cho tín hiệu kích tại cổng I1.1
SM57.2	Tần số đếm của HSC2
SM57.3	Chiều đếm: 0 đếm lùi, 1 : đếm lên
SM57.4	Cho phép đổi chiều đếm, 0: không cho phép, 1: cho phép
SM57.5	Cho phép sửa đổi giá trị đặt trước, 0: không cho phép, 1: cho phép
SM57.6	Cho phép sửa đổi giá trị đếm tức thời, 0: không cho phép, 1: cho phép
SM57.7	1- cho phép kích HSC2, 0 – cho phép hủy HSC2

10.6.4Chọn kiểu Reset, Start và tần số đếm cho HSC

HSC1	HSC2	Ghi chú	
SM47.0	**SM57.0**	**0: Reset mức cao**	**1: Reset mức thấp**
SM47.1	**SM57.1**	**0: Start mức cao**	**1: Start mức thấp**
SM47.2	**SM57.2**	**0: 4X giá trị đếm**	**1: 1X giá trị đếm**

Lưu ý: Kiểu đếm 4X chỉ có giá trị đối với mode đếm 9,10,11.

9.6.3 Byte trạng thái và byte điều khiển của HSC3,HSC4,HSC5

Bộ đếm	Byte trạng thái	Byte điều khiển	Ghi chú
HSC3	SMD136	SMD137	
HSC4	SMD146	SMD147	
HSC5	SMD156	SMD157	

Chức năng các bit của byte trạng thái và byte điều khiển của HSC3, HSC4, HSC5 về cơ bản cũng tương tự như HSC0, HSC1, HSC2 đã trình bày ở trên.

Giá trị đếm tức thời, giá trị đặt cho các bộ HSC.

Bộ đếm	Giá trị đếm tức thời	Giá trị đặt	Ghi chú
HSC0	SMD38	SMD42	
HSC1	SMD48	SMD52	
HSC2	SMD58	SMD62	
HSC3	SMD148	SMD142	
HSC4	SMD158	SMD152	
HSC5	SMD168	SMD162	

10.7 Các bước khởi tạo bộ đếm HSC.

Khác với các lệnh thông thường. HSC chỉ hoạt động sau khi nó được khởi tạo và cho phép bằng phần mềm. Dưới đây sẽ trình bày một gợi ý cho việc khởi tạo bộ đếm HSC.

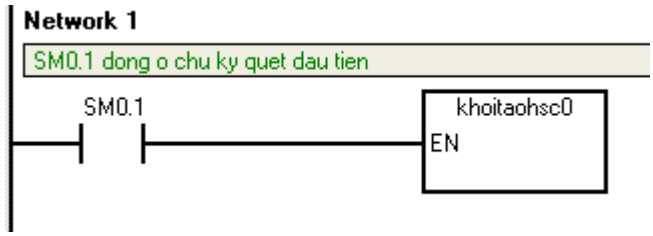
- Dùng chu kỳ quét đầu tiên(SM0.1) để gọi chương trình con khởi tạo. Trong chương trình con khởi tạo thực hiện các công việc sau đây.
- Nạp giá trị cho byte điều khiển.
- Gán bộ đếm với Mode đếm tương ứng dùng lệnh HDEF
- Nạp giá trị đếm tức thời.
- Nạp giá trị đặt trước.
- Gán chương trình ngắt với sự kiện ngắt dùng lệnh ATCH nếu sử dụng ngắt.
- Cho phép ngắt dùng lệnh ENI.
- Chọn bộ đếm để thực thi dùng lệnh HSC.

Lưu ý: Toàn bộ các bước trên đều được thực hiện trong một chương trình con khởi tạo HSC0. Việc khởi tạo này chỉ thực hiện một lần, khi nào cần thay đổi giá trị, chế độ làm việc thì mới khởi tạo lại.

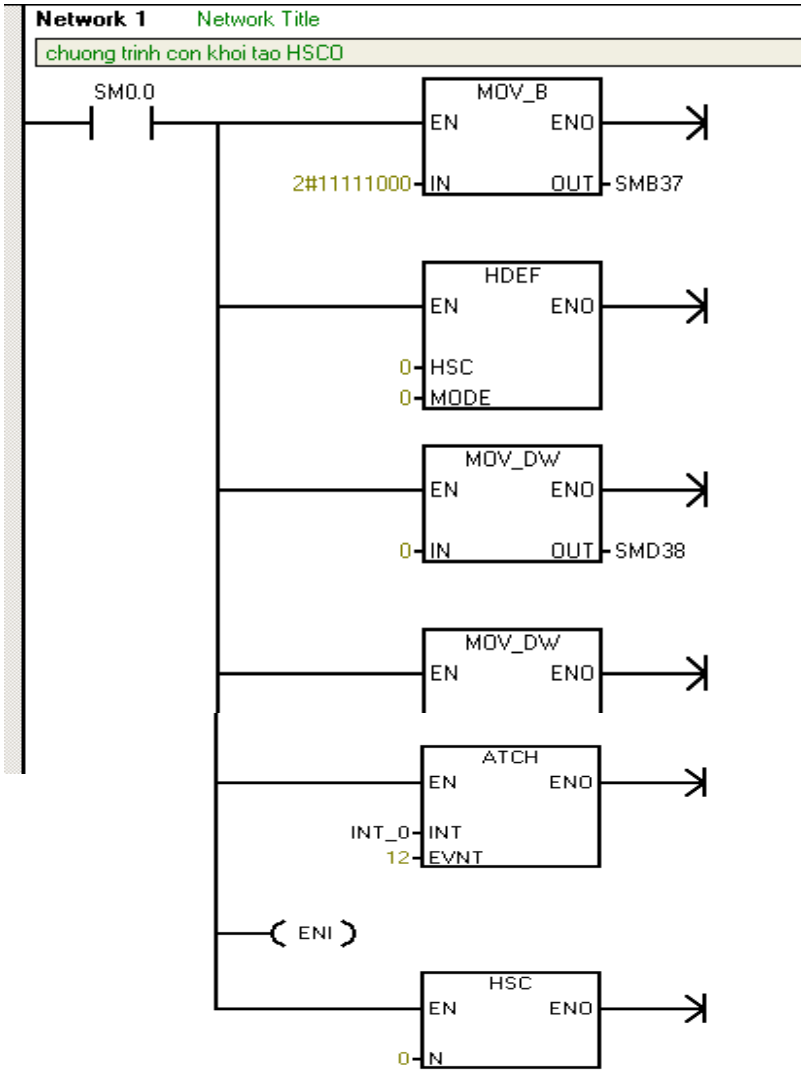
10.8 Một vài ví dụ khởi tạo HSC.

10.8.1 Khởi tạo bộ đếm HSC0 hoạt động ở Mode 0.

Chương trình chính.



Chương trình con khởi tạo.



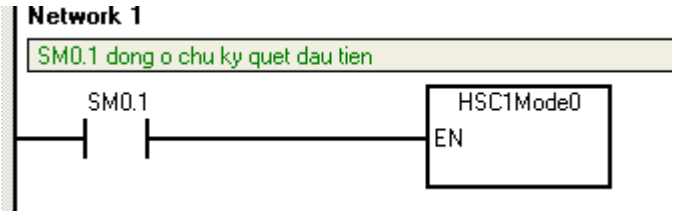
Lưu ý:

Một bộ đếm có thể có một hoặc nhiều mode đếm. Sử dụng mode đếm nào là tùy thuộc vào từng yêu cầu cụ thể của từng ứng dụng. Khi người lập trình muốn bộ HSC hoạt động theo mode nào thì chỉ cần dùng lệnh HDEF để gán bộ đếm đó với mode đếm tương ứng là được.

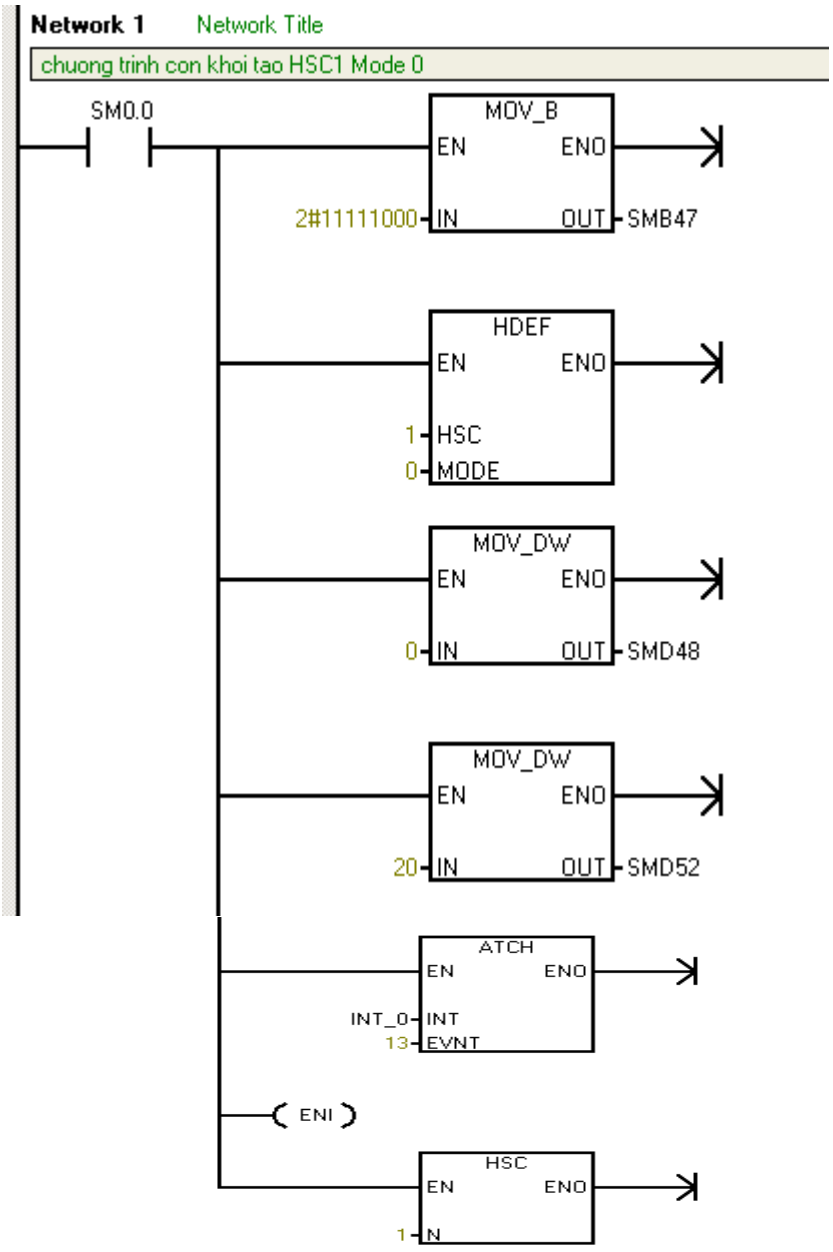
Người học có thể đối chiếu với các vùng nhớ khai báo cho HSC để hiểu về ý nghĩa của từng câu lệnh.

10.8.2 Khởi tạo bộ đếm HSC1 hoạt động ở Mode 0.

Chương trình chính.



Chương trình con khởi tạo.



10.9 Bài tập ứng dụng.

10.9.1 Lập trình với HSC0.

Viết chương trình khởi tạo bộ đếm HSC0 hoạt động ở Mode 0. Xuất kết quả đếm ra vùng nhớ QW2.

Tạo xung CK có tần số 1hZ tại Q0.0 cấp vào ngõ vào xung Clock, quan sát trạng thái ngõ ra tại QW2 khi HSC0 hoạt động ở Mode 0.

Vẽ sơ đồ kết nối phần cứng và viết lại đoạn chương trình khi HSC0 hoạt động ở Mode 0

10.9.2 Lập trình với HSC1.

Viết chương trình khởi tạo bộ đếm HSC1 hoạt động ở Mode 1, chế độ 1X, xuất kết quả đếm ra vùng nhớ QW2.

Tạo xung CK có tần số 1hZ tại Q0.0 cấp vào ngõ vào xung Clock, quan sát trạng thái ngõ ra tại QW2 khi HSC1 hoạt động ở Mode 1. Trong quá trình hoạt động, tác động chân Reset để kiểm tra việc xóa bộ đếm.

Vẽ sơ đồ kết nối ngõ vào xung CK.

10.9.3 Lập trình với HSC2.

Viết chương trình khởi tạo bộ đếm HSC2 hoạt động ở Mode,2,3,4,5. Xuất kết quả đếm ra vùng nhớ QW2. Trong mỗi trường hợp, tác động xung Clock,Reset,Start dùng các Switch để kiểm tra kết quả tại ngõ ra.

Tạo xung CK có tần số 0.5hZ tại Q0.0 cấp vào ngõ vào xung Clock, quan sát trạng thái ngõ ra tại QW2 khi HSC2 hoạt động ở Mode 4.

Vẽ sơ đồ kết nối phần cứng khi HSC2 hoạt động ở Mode 4.

10.9.4 Lập trình với HSC3.

Viết chương trình khởi tạo bộ đếm HSC3 hoạt động ở Mode 4,5,6. Xuất kết quả đếm ra vùng nhớ QW2. Trong mỗi trường hợp, tác động xung Clock,Reset,Start dùng các Switch để kiểm tra kết quả tại ngõ ra.

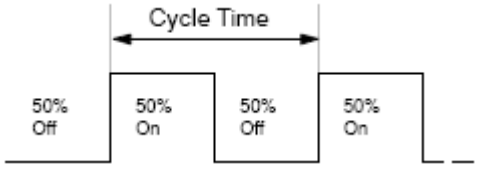
Tạo xung CK có tần số 0.5hZ tại Q0.0 cấp vào ngõ vào xung Clock, quan sát trạng thái ngõ ra tại QW2 khi HSC3 hoạt động ở Mode 5.

Vẽ sơ đồ kết nối phần cứng khi HSC3 hoạt động ở Mode 5

BÀI 11: BỘ PHÁT XUNG VÀ ĐIỀU CHẾ ĐỘ RỘNG XUNG

11.1 Giới thiệu về PLS và PWM.

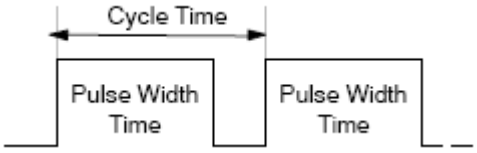
PLS(Pulse): Lệnh phát xung ngõ ra được sử dụng để phát xung ngõ ra PTO (Pulse Train Output). PTO phát xung vuông, có độ rộng luôn bằng nữa chu kỳ.



PTO được ứng dụng trong những trường hợp cần tín hiệu điều khiển dạng xung tần số cao. Những đối tượng thường được điều khiển bởi PTO như: Động cơ bước, động cơ servo…..

Điểm đặc nỗi bật của PTO đó là khả năng phát nhiều đoạn xung liên tiếp có tần số phát khác nhau. Vì vậy khi người lập trình cần tạo ra những chuỗi xung có tần số thay đổi để điều khiển đối tượng điều khiển thì sử dụng PTO là một trong những lựa chọn hiệu quả nhất.

PWM (Pulse Wide Modulation): Điều chế độ rộng xung được sử dụng khi người lập trình cần phát chuỗi xung với chu kỳ và độ rộng có thể thay đổi được.



PWM thường được ứng dụng để điều khiển sự thay đổi tốc độ của động cơ, điều khiển thời gian đóng mở để cấp điện cho lò nhiệt, điều khiển tốc độ quay cho biến tần(Biến tần của panasonic)…..

Các loại PLC nói chung thường có hỗ trợ chức năng PLS và PWM. Tuy nhiên có PLC thì tích hợp sẵn chức năng này trong CPU, có loại thì thiết kế theo kiểu module riêng.

11.2 PTO và PWM trong S7 200.

S7 200 hỗ trợ 2 ngõ ra Q0.0 và Q0.1 để phát xung PLS/PWM với tần số lớn (có thể lên đến 100khZ). Với tần số này thì các ngõ ra bình thường, hoạt động phụ thuộc vào chu kỳ quét của PLC không thể phát được.

PLS: Điều khiển phát xung vuông(PTO) với chu kỳ thay đổi từ 50uS đến 65535uS hay từ 2mS đến 65535mS. Số lượng xung phát nằm trong khoảng từ 1 đến 4,294,967,295.

PLS có thể phát đoạn xung đơn (single segment hoặc nhiều đoạn xung kế tiếp nhau (Multiple segment).

PWM: Phát xung với chu kỳ và độ rộng xung có thể thay đổi được.

Chu kỳ: Từ 50uS đến 65535uS hay từ 2mS đến 65535mS.

Độ rộng xung: Từ 0 đến 65535uS hay từ 0 đến 65535mS.

Cũng giống như các chức năng đặc biệt khác trong PLC. PLS và PWM chỉ sử dụng được sau khi được khai báo và khởi tạo. Việc khai báo và khởi tạo được thực hiện bằng việc nạp giá trị cho vùng nhớ đặc biệt trong PLC.

11.3 Vùng nhớ đặc biệt được sử dụng khi lập trình điều khiển PTO và PWM.

11.3.1 Byte điều khiển.

Q0.0	Q0.1	Control Bits		
SM67.0	SM77.0	PTO/PWM update the cycle time:	0 = no update	1 = update cycle time
SM67.1	SM77.1	PWM update the pulse width time: width	0 = no update	1 = update pulse
SM67.2	SM77.2	PTO update the pulse count value: count	0 = no update	1 = update pulse
SM67.3	SM77.3	PTO/PWM time base:	0 = 1 µs/tick	1 = 1 ms/tick
SM67.4	SM77.4	PWM update method:	0 = asynchronous	1 = synchronous
SM67.5	SM77.5	PTO single/multiple segment operation:	0 = single	1 = multiple
SM67.6	SM77.6	PTO/PWM mode select:	0 = PTO	1 = PWM
SM67.7	SM77.7	PTO/PWM enable:	0 = disable	1 = enable

Tùy thuộc vào việc sử dụng ngõ ra Q0.0 hay Q0.1 ở chế độ PTO hay PWM và tần số mong muốn mà người lập trình nạp các giá trị thích hợp vào byte điều khiển SMB67 hay SMB77.

11.3.2 Các vùng nhớ đặc biệt khác.

Q0.0	Q0.1	Other PTO/PWM Registers	
SMW68	SMW78	PTO/PWM cycle time value	range: 2 to 65,535
SMW70	SMW80	PWM pulse width value	range: 0 to 65,535
SMD72	SMD82	PTO pulse count value	range: 1 to 4,294,967,295
SMB166	SMB176	Number of the segment in progress	Multiple-segment PTO operation only
SMW168	SMW178	Starting location of the profile table (byte offset from V0)	Multiple-segment PTO operation only

11.3.3 Một số giá trị nạp cho byte điều khiển và kết quả thực thi của lệnh PLS.

Control Register (Hex Value)	Enable	Result of Executing the PLS Instruction						
		Select Mode	PTO Segment Operation	PWM Update Method	Time Base	Pulse Count	Pulse Width	Cycle Time
16#81	Yes	PTO	Single		1 µs/cycle			Load
16#84	Yes	PTO	Single		1 µs/cycle	Load		
16#85	Yes	PTO	Single		1 µs/cycle	Load		Load
16#89	Yes	PTO	Single		1 ms/cycle			Load
16#8C	Yes	PTO	Single		1 ms/cycle	Load		
16#8D	Yes	PTO	Single		1 ms/cycle	Load		Load
16#A0	Yes	PTO	Multiple		1 µs/cycle			
16#A8	Yes	PTO	Multiple		1 ms/cycle			
16#D1	Yes	PWM		Synchronous	1 µs/cycle			Load
16#D2	Yes	PWM		Synchronous	1 µs/cycle		Load	
16#D3	Yes	PWM		Synchronous	1 µs/cycle		Load	Load
16#D9	Yes	PWM		Synchronous	1 ms/cycle			Load
16#DA	Yes	PWM		Synchronous	1 ms/cycle		Load	
16#DB	Yes	PWM		Synchronous	1 ms/cycle		Load	Load

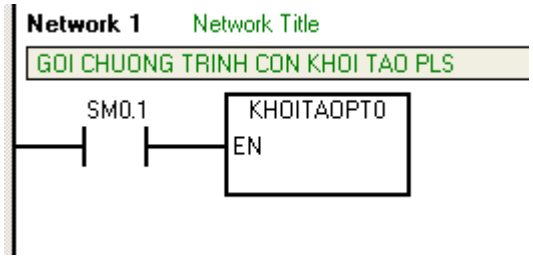
11.4 Khởi tạo phát một đoạn xung đơn (single segment).

Sử dụng chu kỳ quét đầu tiên để gọi chương trình con khởi tạo PLS. Trong chương trình con khởi tạo thực hiện các công việc sau.

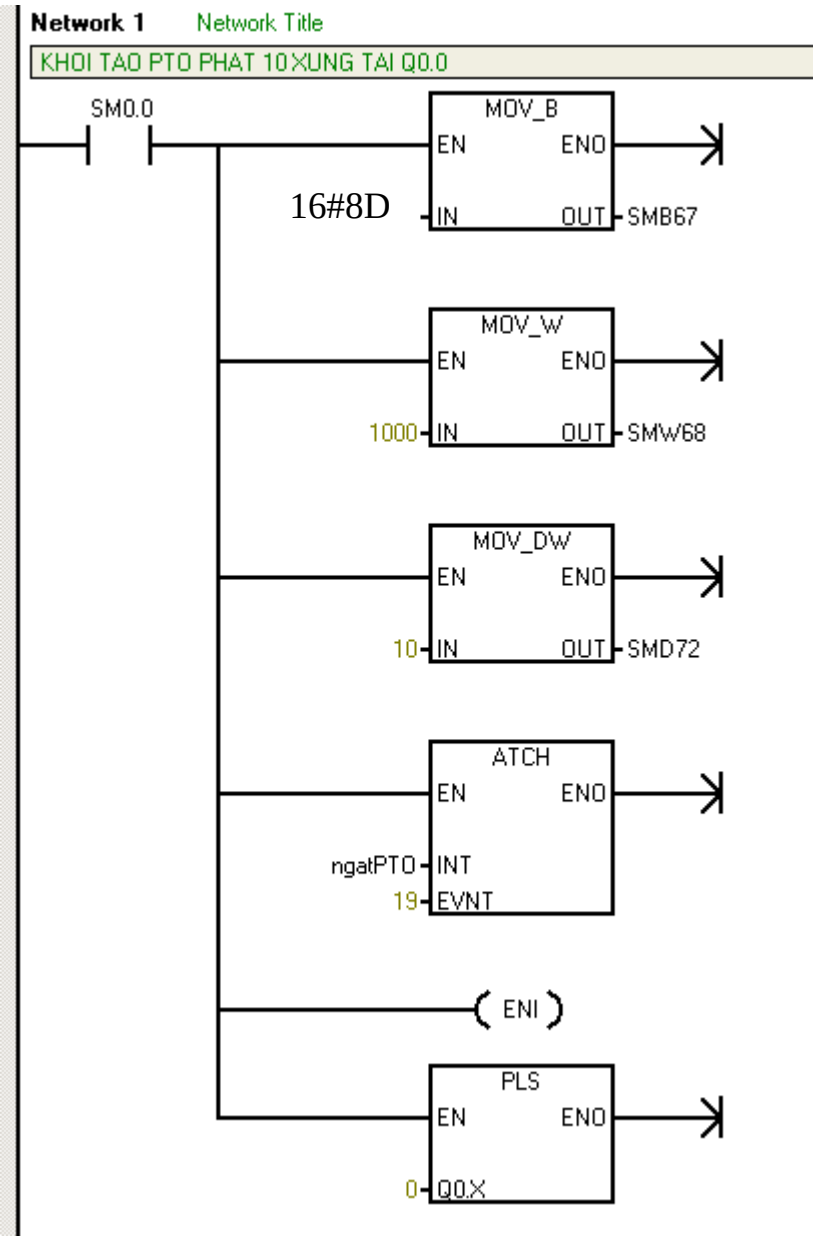
- Nạp giá trị cho byte điều khiển để chọn chế độ phát theo yêu cầu.
- Nạp giá trị thời gian để chọn chu kỳ.
- Nạp giá trị để chọn số lượng xung phát.
- Gán chương trình ngắt với sự kiện ngắt tương ứng.
- Cho phép ngắt.
- Thực hiện lệnh PLS để phát xung.

11.4.1 Ví dụ: Khởi tạo PTO phát 10 xung vuông, tần số 1hZ.

Chương trình chính.



Chương trình con khởi tạo.



11.5 Khởi tạo PTO phát nhiều đoạn xung (Multiple segment).

Bên cạnh việc phát một đoạn xung như đã trình bày ở trên thì PTO có thể phát nhiều đoạn xung liên tiếp nhau (Tối thiểu là 1 và tối đa là 255 đoạn). PTO phát nhiều đoạn xung liên tiếp với tần số khác nhau có ý nghĩa rất quan trong trong

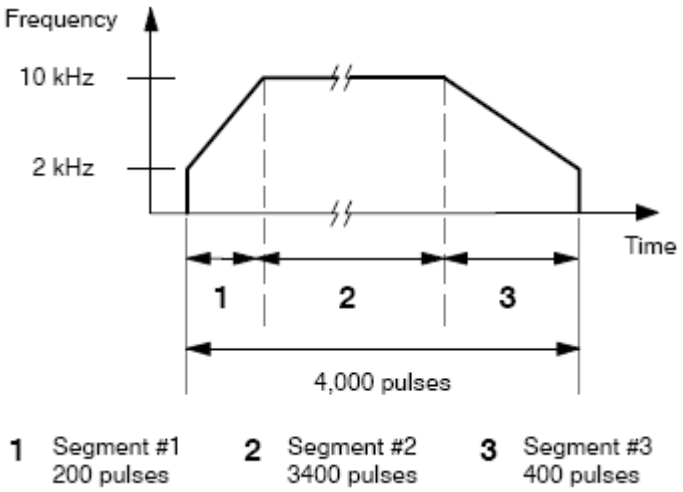
một số ứng dụng thực tiễn. Chẳng hạn như việc điều khiển động cơ bước hay servo motor quay nhanh dần khi khởi và giảm dần tốc đọ khi chuẩn bị dừng…

11.5.1 Một số đặc điểm cần chú ý khi lập trình phát nhiều chuỗi xung.

Việc khởi tạo phát nhiều đoạn xung liên có một số khác biệt so với phát một đoạn xung. Vì vậy, người học trình cần chú ý một số đặc điểm sau đây khi lập trình phát nhiều đoạn xung.

- Vùng nhớ sử dùng để khai báo thông số cho các chuỗi xung là vùng nhớ V.
- Mỗi chuỗi xung sử dụng 8 byte để khai báo thông số.
- Địa chỉ bắt đầu của vùng nhớ V tùy thuộc vào địa chỉ offset. Địa chỉ offset được nạp trong vùng nhớ đặc biệt(SMW168: Q0.0 và SMW178: Q0.1).
- Byte đầu tiên trong vùng nhớ V được sử dụng để nạp số chuỗi cần phát.

Dưới đây là ví dụ phát 3 đoạn xung liên tiếp tại ngõ ra Q0.0, bảng khai báo thông số và chương trình viết cho PLC s7 200.

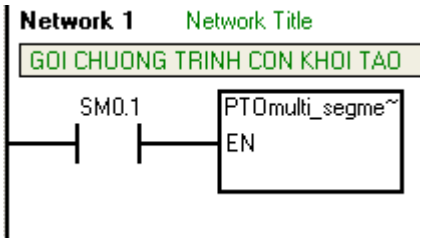


Khai báo thông số.

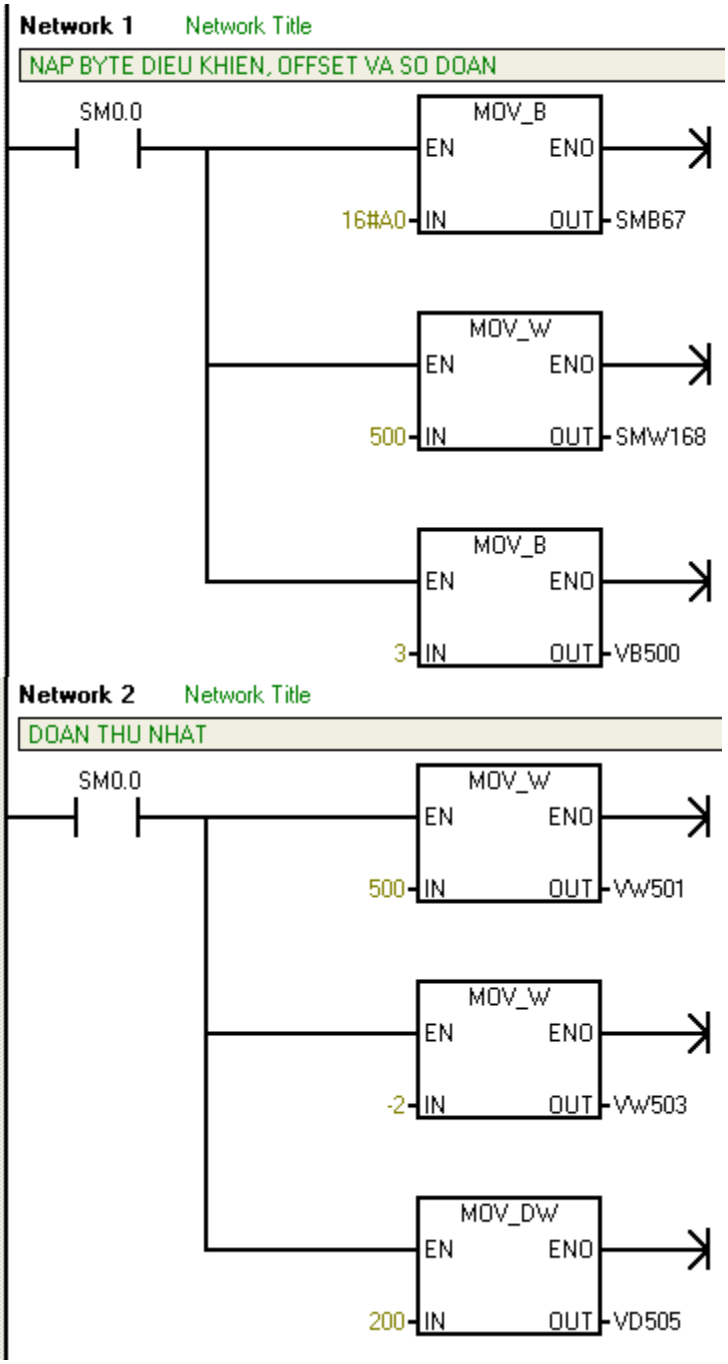
Address	Value	Description	
VB500	3	Total number of segments	
VW501	500	Initial cycle time	Segment 1
VW503	-2	Initial delta cycle time	
VD505	200	Number of pulses	
VW509	100	Initial cycle time	Segment 2
VW511	0	Delta cycle time	
VD513	3400	Number of pulses	
VW517	100	Initial cycle time	Segment 3
VW519	1	Delta cycle time	
VD521	400	Number of pulses	

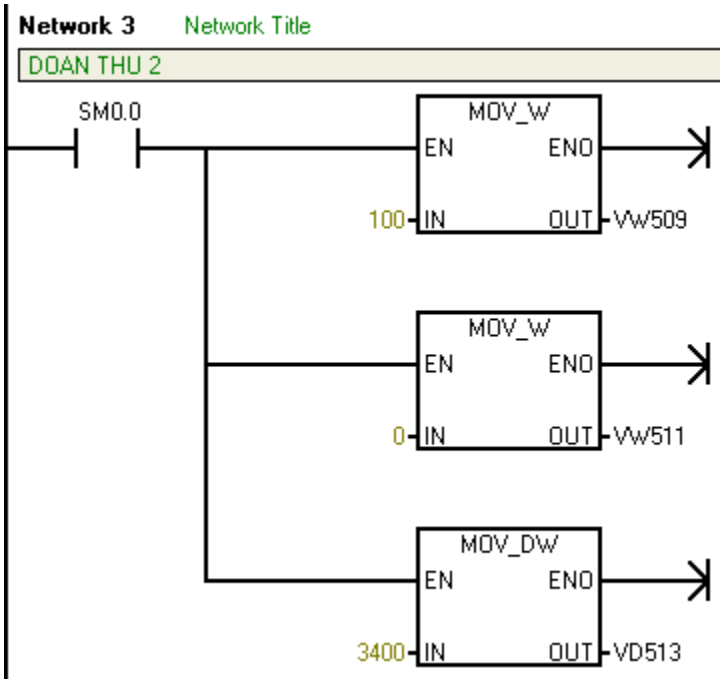
Lập trình cho PLC.

Chương trình chính.



Chương trình con khởi tạo.


Network 1
Network Title
NAP BYTE DIEU KHIEN, OFFSET VA SO DOAN
SM0.0
MOV_B
EN
ENO
16#A0
IN
OUT
SMB67
MOV_W
EN
ENO
500
IN
OUT
SMW168
MOV_B
EN
ENO
3
IN
OUT
VB500
Network 2
Network Title
DOAN THU NHAT
SM0.0
MOV_W
EN
ENO
500
IN
OUT
VW501
MOV_W
EN
ENO
-2
IN
OUT
VW503
MOV_DW
EN
ENO
200
IN
OUT
VD505



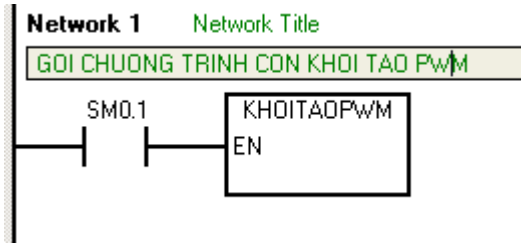
11.6 Khởi tạo bộ phát xung PWM.

Sử dụng chu kỳ quét đầu tiên để gọi chương trình con khởi tạo PWM. Trong chương trình con khởi tạo thực hiện các công việc sau.

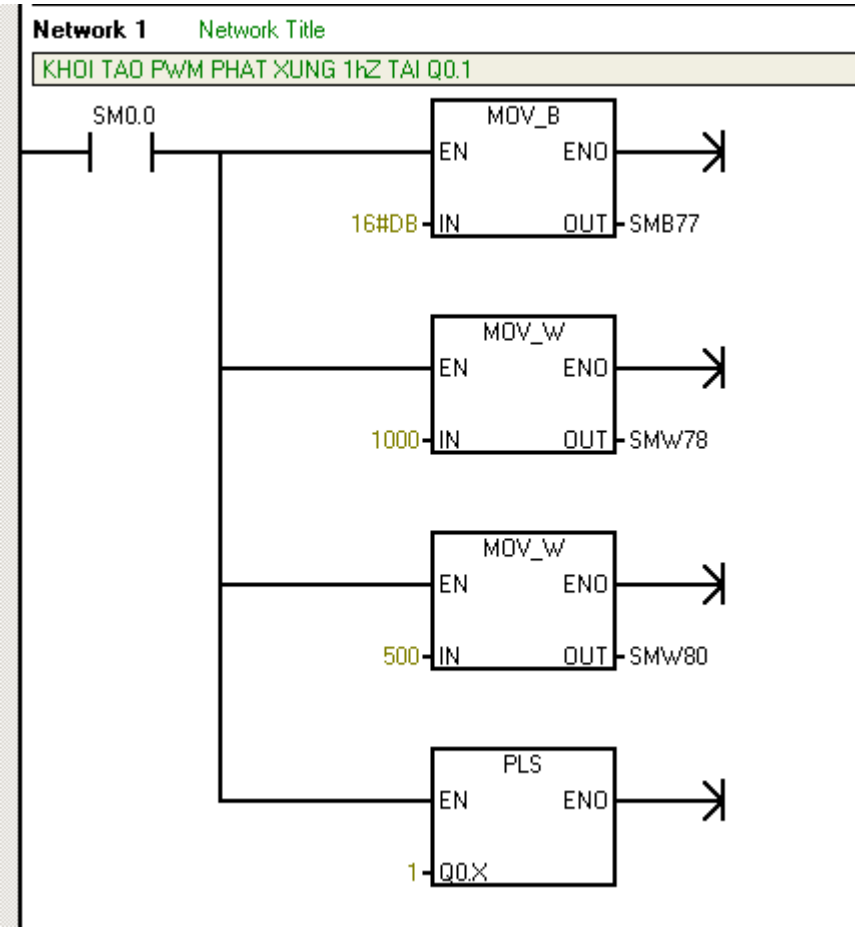
- Nạp giá trị cho byte điều khiển để chọn chế độ phát theo yêu cầu.
- Nạp giá trị thời gian để chọn chu kỳ.
- Nạp giá trị thời gian để chọn độ rộng xung.
- Thực hiện lệnh PLS để phát xung.

Ví dụ: Khởi tạo phát xung vuông tần số 1hZ tại ngõ ra Q0.1 dùng PWM.

Chương trình chính



Chương trình con khởi tạo



11.7 Bài tập ứng dụng.

11.7.1 Viết chương trình điều khiển PTO phát xung theo yêu cầu:

Mỗi lần nhấn START, phát 20 xung tần số 1hZ tại ngõ ra Q0.0.

11.7.2 Viết chương trình điều khiển PTO phát xung theo yêu cầu:

Mỗi lần nhấn START, phát 20 xung tần số 1hZ tại ngõ ra Q0.1.

11.7.3 Viết chương trình điều khiển PTO phát xung tại Q0.0 theo yêu cầu.

Nhấn START: Phát xung PTO có chu kỳ 1000ms tại Q0.0

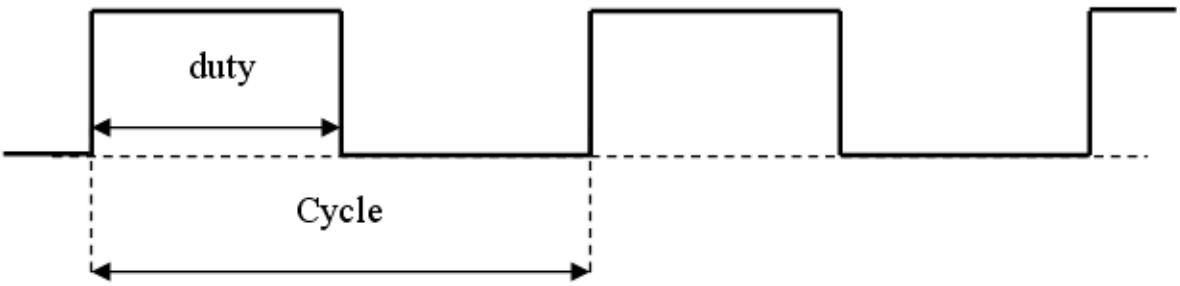
Mỗi lần nhấn Up: Chu kỳ tăng thêm 100ms, tối đa không quá 10000ms.

Mỗi lần nhấn Down: Chu kỳ giảm bớt 100ms, tối thiểu không nhỏ hơn 100ms.

Symbol	Address	comment
START	I0.0	
Up	I0.1	
Down	I0.2	
PULSE	Q0.0	

11.7.4 Viết chương trình điều khiển Q0.0 hoạt động theo 2 Mode.

Mode 1: Phát xung vuông tần với Cycle = 1 giây, duty = 0,5 giây.



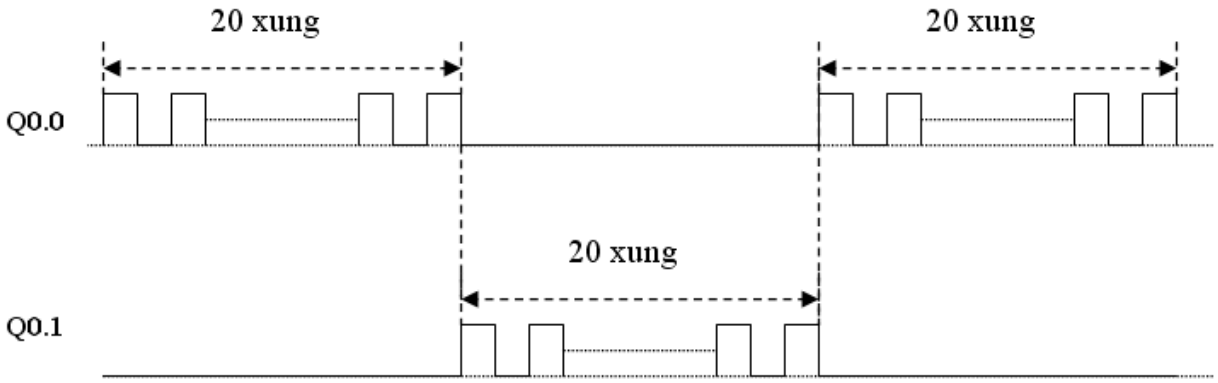
Mode 2: Cycle = 1 giây.

Sử dụng Module analog, chọn tầm điện áp vào từ 5V. Dùng biến trở để chỉnh duty: Khi điện áp vào bằng 0V thì duty = 0, khi điện áp vào bằng 5V thì duty = 1giây.

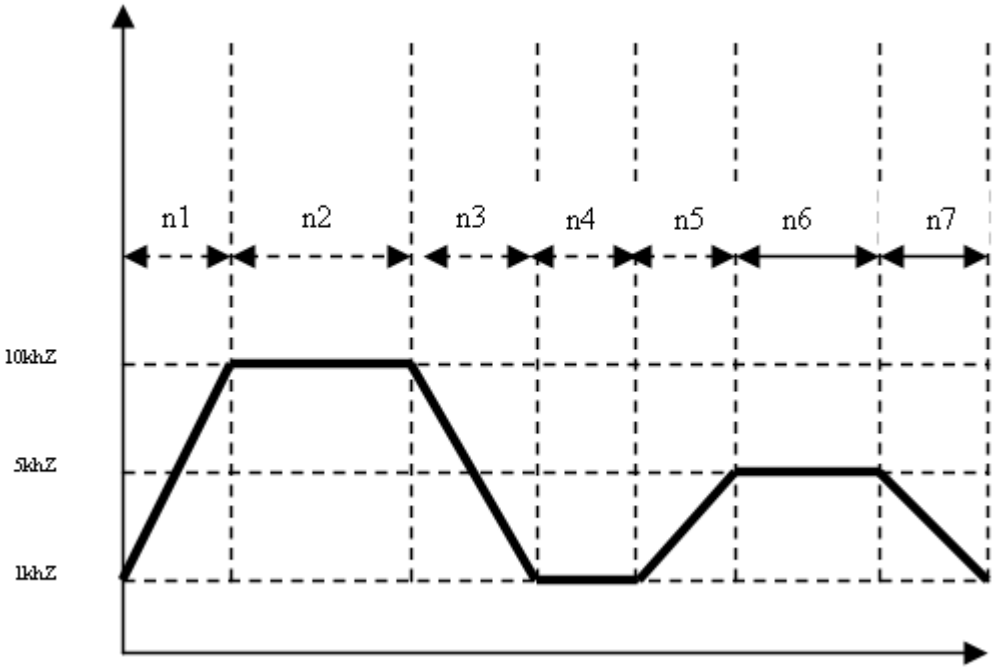
Bảng mô tả địa chỉ

Symbol	Address	comment
Mode1	I0.0	
Mode2	I0.1	
PULSE	Q0.0	

11.7.5 Viết chương trình điều khiển Q0.0 và Q0.1 hoạt động như hình vẽ sau. Biết rằng tần số phát ra tai Q0.0 và Q0.1 là 2hZ.



11.7.6 Viết chương trình phát xung tại ngõ ra Q0.0 theo sơ đồ hình sau:



11.7.7 Viết chương trình thực hiện yêu cầu sau:

Phát chuỗi xung tại Q0.1 tần số 10khZ giã lập xung của một encoder được gắn trên trục động cơ.

Khởi tạo HSC1 hoạt động ở Mode 2 để đếm số xung tại Q0.0.

Khởi tạo ngắt của Timer để đọc số xung đếm được trong khoảng thời gian lấy mẫu Ts.

Giã sử encoder có thông số 1000 xung/vòng. Hãy viết chương trình tính tốc độ động cơ ra vòng/phút.