Đồ án Tốt Nghiệp

Thết Kế Hệ Thống Điều Khiển Đèn Giao Thông Trên Micro PLC SIMATIC S7-200

--------🕮 --------

Nhiệm Vụ Thiết Kế Tốt Nghiệp

Họ và Tên **:**

MSSV:

Niên Khố:

Khoa: Điện.

Ngành: Điều khiển tự động.

1. Đầu đề thiết kế:

Thết Kế Hệ Thống Điều Khiển Đèn Giao Thông

Trên Micro PLC SIMATIC S7- 200

2. Các số liệu ban đầu:

.....

.....

.....

.....

.....

3. Nội dung các phần thuyết minh và tính tốn:

.....

.....

.....

.....

.....

4. Các bản vẽ và đồ thị:

.....

.....

.....

.....

.....

5. Cán bộ hướng dẫn:

Phần	Tên Cán Bộ
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....

6. Ngày giao nhiệm vụ thiết kế:.

7. Ngày hồn thành nhiệm vụ:

Ngày……Tháng……Năm

Chủ nhiệm bộ môn Cán bộ hướng dẫn

Học sinh đã hồn thành

Ngày ……Tháng ……Năm……

LỜI CẢM ƠN

Sau quá trình học tập và rèn luyện nghiệm túc tại Khoa Điện trường ĐHBKHN cùng với sự hướng dẫn và đôn đốc tận tình của Thầy giáo Nguyễn Dỗn Phước , tôi đã hồn thành Đồ án tốt nghiệp Cao đẳng.

Tôi xin chân thành gửi lời cảm ơn sâu sắc đến Thầy Nguyễn Dỗn Phước, người thầy đã động viên và giúp đỡ tôi nhiều về mặt tinh thần cũng như kiến thức để tôi vượt qua những ngày tháng khó khăn trong sự tìm tòi hiểu biết về lĩnh vực mới để rồi cuối cùng hồn thành được Đồ án tốt nghiệp ngày hôm nay.Một lần nữa xin được gửi lời cảm ơn đến Thầy,chúc Thầy luôn khoẻ mạnh và có được những tháng năm công tác tốt như thầy mong đợi.

Tôi xin chân thành gửi lời cảm ơn đến các thầy cô trong bộ môn Điều Khiển Tự Động cũng như các thầy cô trong Khoa Điện và những người đã dìu dắt tôi ,cho tôi kiến thức chuyên ngành và những kinh nghiệm quý báu để cùng với sự nỗ lực của bản thân tôi đã hồn thành đồ án tốt nghiệp ngày hôm nay.

Tôi cũng xin gửi lời cảm ơn đến gia đình ,bạn bè và tất cả những người thân của tôi đã tạo điều kiện và giúp đỡ tôi rất nhiều để tôi có được kết quả đồ án ngày hôm nay.

Một lần nữa xin cảm ơn tất cả mọi người .

LỜI NÓI ĐẦU

Trong những năm gần đây cùng với sự phát triển của nền kinh tế là tốc độ ra tăng không ngừng về các loại phương tiện giao thông. Sự phát triển nhanh chóng của các phương tiện giao thông đã dẫn đến tình trạng tắc nghẽn giao thông xảy ra rất thường xuyên .Vấn đề đặt ra ở đây là làm sao để đảm bảo giao thông thông suốt và sử dụng đèn điều khiển giao thông ở những ngã tư ,những nơi giao nhau của các làn đường là một giải pháp .

Để viết chương trình điều khiển đèn giao thông ta có thể viết trên nhiều hệ ngôn ngữ khác nhau . Nhưng với những ưu điểm vượt trội của PLC S7- 200 như : giá thành hạ , dễ thi công , sửa chữa , chất lượng làm việc ổn định linh hoạt ….nên ở đây tôi đã chọn hệ thống điều khiển có thể lập trình được PLC (Programmble Logic Control) với ngôn ngữ lập trình của S7 – 200 để viết chương trình điều khiển đèn giao thông .

Xuất phát từ những nhu cầu thực tế và những ham muốn hiểu biết về về lĩnh vực này , tôi xin chọn đề tài làm đồ án tốt nghiệp về : ‘’ Thiết kế hệ thống điều khiển đèn giao thông trên Micro PLC SIMATIC S7 – 200 ‘’ . Mục đích của đề tài này là hiểu biết về các thiết bị tự động hố , các giải pháp tự động hố tích hợp tồn diện thông qua PLC S7 – 200 và quan trọng nhất là những ứng dụng của PLC trong cuộc sống (Điều khiển đèn giao thông , tự động hố trong mọi lĩnh vực của ngành sản xuất . . .)

Báo cáo về đề tài gồm 3 phần chính:

Chương 1: Nguyên Tắc Hoạt Động Đèn Giao Thông

Trong chương này chủ yếu trình bầy về cấu tạo và nguyên tắc hoạt động của đèn giao thông .

Chương 2 : Công Cụ Thực Hiện Bài Tốn .Nội dung chủ yếu về giới thiệu cấu tạo phần cứng của PLC S7 – 200 , các hệ lệnh cơ bản và Mircowin.

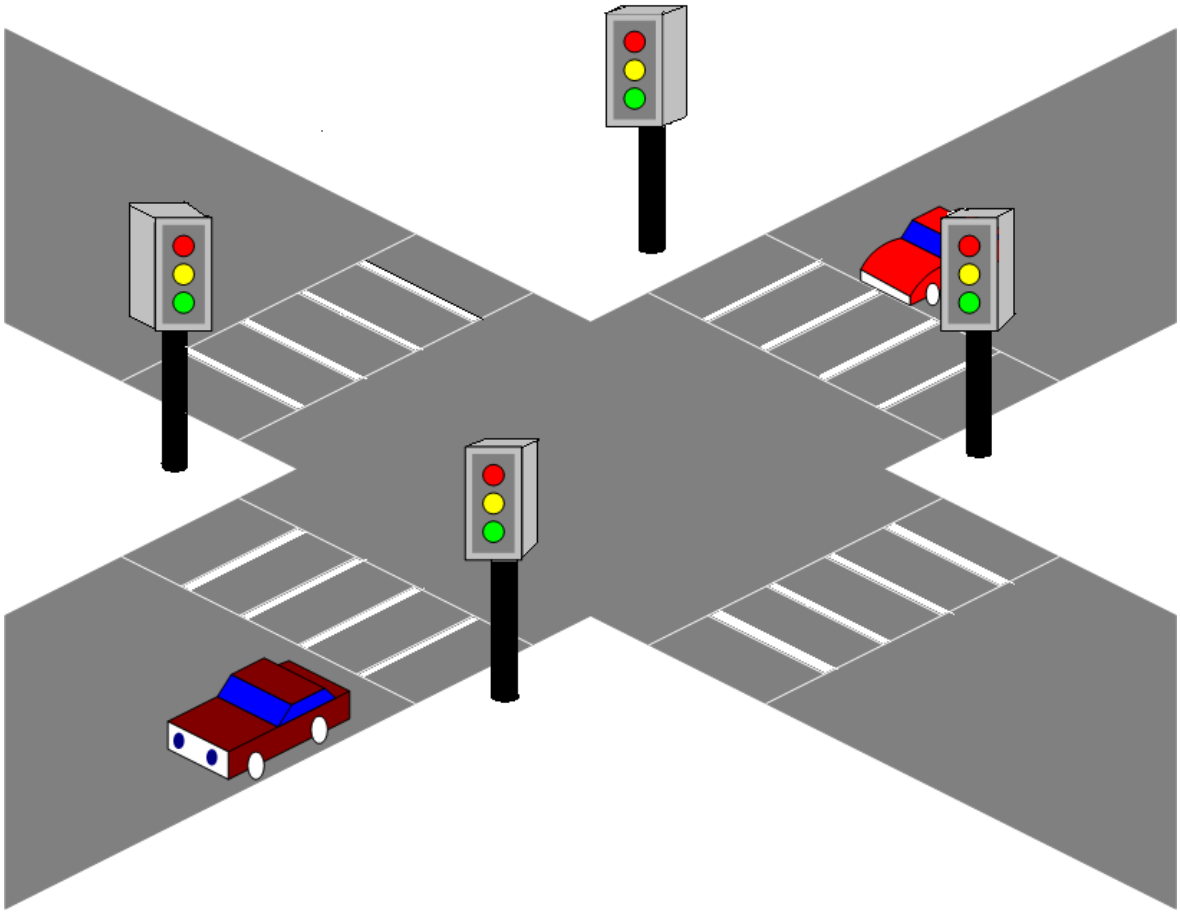
Chương 3 : Chương Trình Điều Khiển Đèn Giao Thông Bằng S7 -200 .

MỤC LỤC

Trang

Chương 1:NGUYÊN TẮC HOẠT ĐỘNG CỦA ĐÈN GIAO THÔNG

1.1 Cấu tạo và nguyên tắc hoạt động của đèn giao thông



Mô hình đèn giao thông ở ngã tư.

Cấu tạo

Hệ thống đèn giao thông hay là đèn điều khiển giao thông gồm hai cột đèn chính được lắp đặt tại hai đầu của hai làn đường khác nhau ở ngã tư. Mỗi một cột đèn gồm 6 đèn đó là 3 đèn chính gồm: đèn xanh, đèn đỏ và đèn đỏ; 2 đèn phụ là 2 đèn

dùng điều khiển làn đường dành cho người đi bộ: đèn xanh người đi bộ và đèn đỏ người đi bộ.

Ngồi ra, mỗi một hệ thống đèn có một hộp điều khiển từ đó sẽ phát ra tín hiệu điều khiển đèn. Tín hiệu điều khiển của đèn từ CPU thông qua các cổng ra rồi đến các rơle, rồi qua hệ thống dây nối đến các đèn.

Nguyên tắc hoạt động

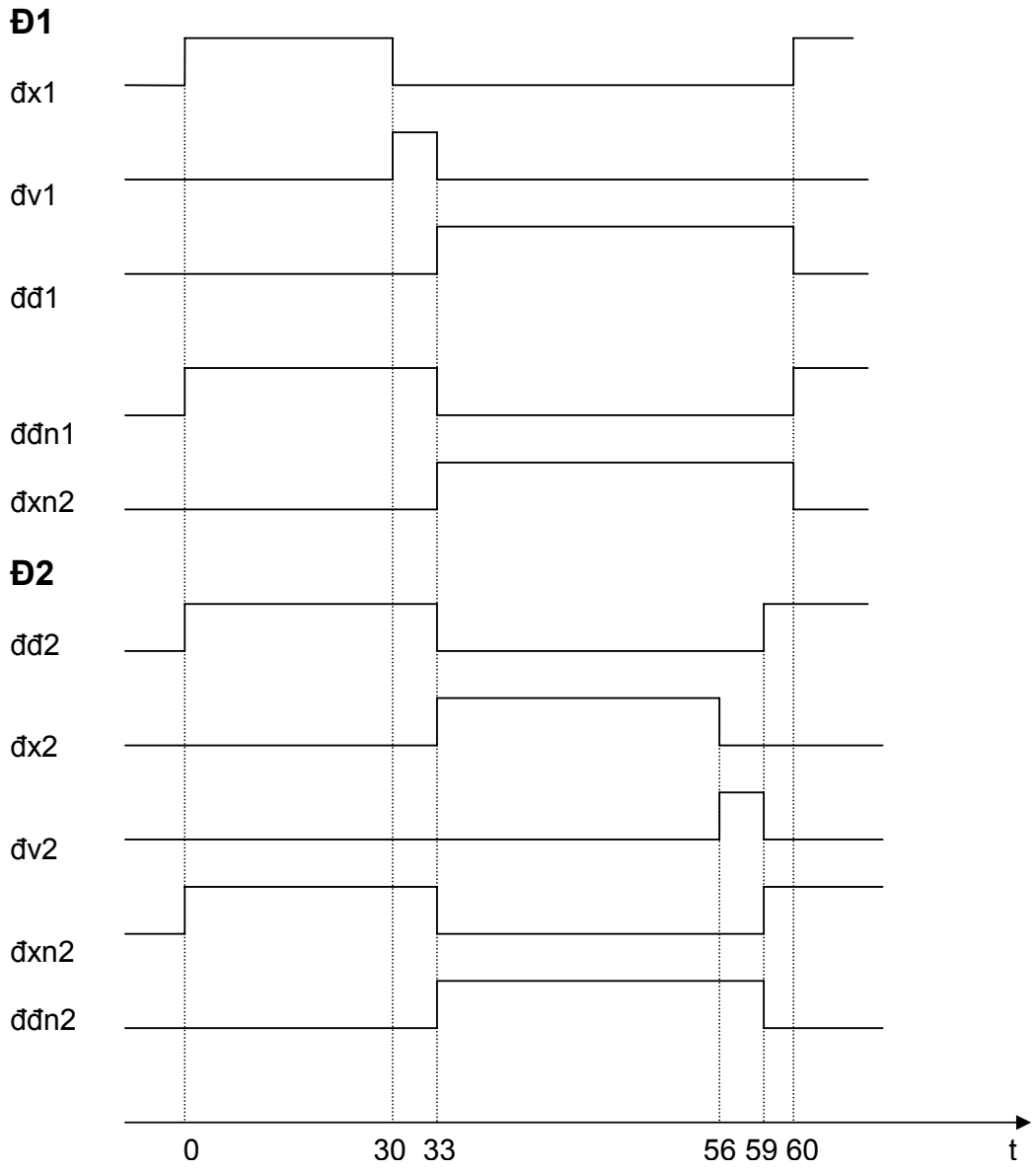
Cơ chế hoạt động của đèn giao thông thật ra rất đơn giản: Khi đèn của làn đường 1(đx1) được bật sáng thì cùng lúc đó đèn đỏ của làn đường 2 (đđ2), đèn đỏ cho người đi bộ ở làn đường 1(đđn1), đèn xanh người đi bộ làn đường 2 (đxn2) cũng được bật sáng.Sau một khoảng thời gian nhất định đx1 tắt,đèn vàng 1(đv1) được bật lên .

Khi đv1 tắt thì đđ2, đđn1,đxn2 mới tắt cùng lúc đó đèn xanh 2(đx2) , đèn đỏ 1(đđ1),đèn đỏ cho người đi bộ 2(đđn2), đèn xanh cho người đi bộ 1(đxn1) được bật sáng.

Lúc đèn vàng 2(đv2) được bật lên cũng là lúc đx2 tắt ,đv2 tắt chu kì được lập lại với đđ2,đx1…

1.2 Giản đồ thời gian cho từng đèn

Với một chu kỳ đèn bất kỳ ta có giản đồ thời gian hoạt động của từng đèn như sau:



1.3 ‘’ Làn Xanh ‘’

Khái niệm đèn xanh được đề cập đến ở đây chính là làm thế nào để phương tiện tham gia giao thông có thể gặp hai đèn xanh liên tiếp ở hai ngã tư liền nhau. Muốn được như vậy chúng ta phải làm sao cho chu kỳ của đèn ở ngã tư tiếp theo phù hợp với tốc độ của phương tiện và khoảng cách giữa hai ngã tư. Và giải pháp tôi đề cập ở đây là ở ngã tư thứ hai ta lắp đặt một Timer có tác dụng tạo thời gian trễ của chu kỳ đèn thứ hai so với đèn thứ nhất phù hợp.

Bài tốn đèn giao thông trong đồ án này chưa đề cập đến ‘’ làn xanh ‘’ mà chỉ là chương trình cho điều khiển cho một ngã tư.

Chương 2 : CÔNG CỤ THỰC HIỆN BÀI TỐN

2.1 Thiết bị điều khiển logic khả trình PLC S7 – 200

Trong công nghiệp sản xuất, để điều khiển một dây chuyền, một thiết bị máy móc công nghiệp … người ta thực hiện kết nối các linh kiện điều khiển rời (rơle, timer, contactor …) lại với nhau tuỳ theo mức độ yêu cầu thành một hệ thống điện điều khiển. Công việc này khá phức tạp trong thi công, sửa chữa bảo trì do đó giá thành cao. Khó khăn nhất là khi cần thay đổi một hoạt động nào đó.

Một hệ thống điều khiển ưu việt mà chúng ta phải chọn được điều khiển cho một máy sản xuất cần phải hội đủ các yêu cầu sau: giá thành hạ, dễ thi công, sửa chữa, chất lượng làm việc ổn định linh hoạt … Từ đô hệ thống điều khiển có thể lập trình được PLC (Programable Logic Control) ra đời đã giải quyết được vấn đề trên.

Thiết bị điều khiển lập trình đầu tiên đó được những nhà thiết kế cho ra đời năm 1968 (Công ty General Moto - Mỹ). Tuy nhiên, hệ thống này còn khá đơn giản và cồng kềnh, người sử dụng gặp nhiều khó khăn trong việc vận hành hệ thống. Vì vậy các nhà thiết kế từng bước cải tiến hệ thống đơn giản, gọn nhẹ, dễ vận hành, nhưng việc lập trình cho hệ thống còn khó khăn, do lúc này không có các thiết bị lập trình ngoại vi hỗ trợ cho công việc lập trình.

Để đơn giản hóa việc lập trình, hệ thống điều khiển lập trình cầm tay (programmable controller handle) đầu tiên được ra đời vào năm 1969. Trong giai đoạn này các hệ thống điều khiển lập trình (PLC) chỉ đơn giản nhằm thay thế hệ thống Relay và dây nối trong hệ thống điều khiển cổ điển. Qua quá trình vận hành, các nhà thiết kế đã từng bước tạo ra được một tiêu chuẩn mới cho hệ thống, tiêu chuẩn đó là: dạng lập trình dùng giản đồ hình thang. Trong những năm đầu thập niên 1970, những hệ thống PLC còn có thêm khả năng vận hành với những thuật tốn hỗ trợ (arithmetic), "vận hành với các dữ liệu cập nhật" (data manipulation). Do sự phát triển của loại màn hình dùng cho máy tính (Cathode Ray Tube: CRT), nên việc giao tiếp giữa người điều khiển để lập trình cho hệ thống càng trở nên thuận tiện hơn. Ngồi ra các nhà thiết kế còn tạo ra kỹ thuật kết nối với các hệ thống PLC riêng lẻ thành một hệ thống PLC chung, tăng khả năng của từng hệ thống riêng lẻ. Tốc độ xử lý của hệ thống được cải thiện, chu kỳ quét (*scan*) nhanh hơn làm cho hệ thống PLC xử lý tốt với những chức năng phức tạp, số lượng cổng ra/vào lớn.

Một PLC có đầy đủ các chức năng như: bộ đếm, bộ định thời, các thanh ghi (*register*) và tập lệnh cho phép thực hiện các yêu cầu điều khiển phức tạp khác nhau. Hoạt động của PLC hồn tồn phụ thuộc vào chương trình nằm trong bộ nhớ, nó luôn cập nhật tín hiệu ngõ vào, xử lý tín hiệu để điều khiển ngõ ra.

Những đặc điểm của PLC:

-Thiết bị chống nhiễu.

-Có thể kết nối thêm các modul để mở rộng ngõ vào/ra.

-Ngôn ngữ lập trình dễ hiểu.

-Dễ dàng thay đổi chương trình điều khiển bằng máy lập trình hoặc máy tính cá nhân.

-Độ tin cậy cao, kích thước nhỏ.

-Bảo trì dễ dàng.

Do các đặc điểm trên, PLC cho phép người điều hành không mất nhiều thời gian nối dây phức tạp khi cần thay đổi chương trình điều khiển, chỉ cần lập chương trình mới thay cho chương trình cũ.

Việc sử dụng PLC vào các hệ thống điều khiển ngày càng thông dụng, để đáp ứng yêu cầu ngày càng đa dạng này, các nhà sản xuất đã đưa ra hàng loạt các dạng PLC với nhiều mức độ thực hiện đủ để đáp ứng các yêu cầu khác nhau của người sử dụng.

Để đánh giá một bộ PLC người ta dựa vào 2 tiêu chuẩn chính: dung lượng bộ nhớ và số tiếp điểm vào/ra của nó. Bên cạnh đó cũng cần chú ý đến các chức năng như: bộ vi xử lý, chu kỳ xung clock, ngôn ngữ lập trình, khả năng mở rộng số ngõ vào/ra.

2.1.1 Cấu hình cứng

PLC viết tắt của Programmable Logic Control, là thiết bị điều khiển logic lập trình được, cho phép thực hiện linh hoạt các thuật tốn điều khiển thông qua một ngôn ngữ lập trình.

S7 – 200 là thiết bị điều khiển khả trình loại nhỏ của hãng Siemens, có cấu trúc theo kiểu modul và có các modul mở rộng. Các modul này sử dụng cho nhiều ứng dụng lập trình khác nhau. Thành phần cơ bản của S7 – 200 là khối vi xử lý CPU 212 hoặc CPU 214. Về hình thức bên ngồi, sự khác nhau của hai loại CPU này nhận biết được nhờ số đầu vào/ra và nguồn cung cấp.

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

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

S7 – 200 có nhiều loại modul mở rộng khác nhau.

CPU 214 bao gồm:

-2048 từ đơn (4K byte) thuộc miền nhớ đọc/ghi non-volatile để lưu chương trình (vựng nhớ có giao diện với EEPROM).

-2048 từ đơn (4K byte) kiểu đọc/ghi để lưu dữ liệu, trong đó 512 từ đầu thuộc miền nhớ non-volatile.

-14 cổng vào và 10 cổng ra logic.

-Cú 7 modul để mở rộng thêm cổng vào/ra bao gồm luôn cả modul analog.

-Tổng số cổng vào/ra cực đại là 64 cổng vào và 64 cổng ra.

-128 Timer chia làm 3 loại theo độ phân giải khác nhau: 4 Timer 1ms, 16 Timer 10ms và 108 Timer 100ms.

-128 bộ đếm chia làm 2 loại: chỉ đếm tiến và vừa đếm tiến vừa đếm lùi.

-688 bit nhớ đặc biệt dùng để thông báo trạng thái và đặt chế độ làm việc.

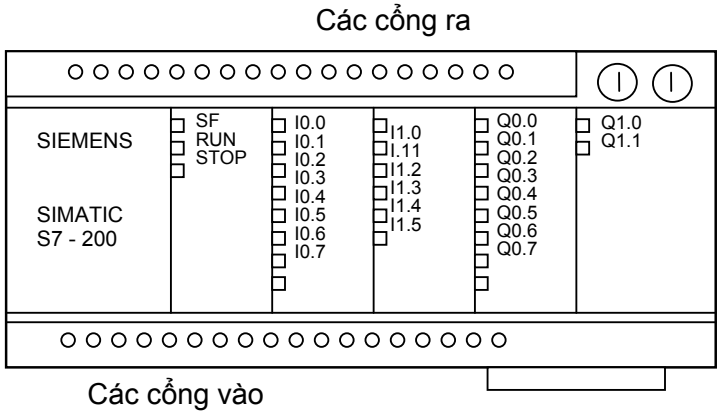
-Các chế độ ngắt và xử lý ngắt bao gồm: ngắt truyền thông, ngắt theo sườn lên hoặc xuống, ngắt thời gian, ngắt của bộ đếm tốc độ cao và ngắt truyền xung.

-3 bộ đếm tốc độ cao với nhịp 2 KHz và 7KHz.

-2 bộ phát xung nhanh cho dãy xung kiểu PTO hoặc kiểu PWM.

-2 bộ điều chỉnh tương tự.

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



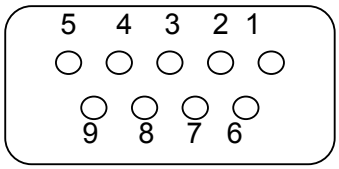
Hình 1 : Bộ điều khiển lập trình được (khả trình) S7 -200 với khối vi xử lý CPU 214

Mô tả các đèn báo trên S7 -200 CPU 214:

SF (đèn đỏ)	Đèn đỏ SF báo hiệu hệ thống bị hỏng .Đèn SF sádng lên khi PLC có hỏng hóc .
RUN (đèn xanh)	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)	Đèn vàng STOP chỉ định rằng PLC đang ở chế độ dừng .Dừng chương trình đang thực hiện lại .
Ix .x (đèn xanh)	Đèn xanh ở cổng vào chỉ định trạng thái tức thời của cổng Ix.x (x.x = 0.0 ÷ 1.5).Đèn này báo hiệu trạng thái của tín hiệu theo giá trị logic của cổng .
Qy.y (đèn xanh)	Đèn xanh ở cổng ra báo hiệu trạng thái tức thời của cổng Qy.y (y.y = 0.0 ÷1.1).Đèn này báo hiệu trạng thái của tín hiệu theo giá trị logic của cổng.

Cổng truyền thông :

S7 – 200 sử dụng cổng truyền thông nối tiếp RS485 với phích nối 9 chân để phục vụ cho việc ghép nối với thiết bị lập trình hoặc với các trạm PLC khác. Tốc độ truyền cho máy lập trình kiểu PPI là 9600 baud. Tốc độ truyền cung cấp của PLC theo kiểu tự do là 300 đến 38.400.



Hình 2 : Sơ đồ chân của cổng truyền thông

Trong đó : Chân	Giải thích
1	Đất
2	24 VDC
3	Truyền và nhận dữ liệu
4	Không sử dụng
5	Đất
6	5 VDC (điện trở trong 100Ω)
7	24 VDC (120 mA tối đa)
8	Truyền và nhận dữ liệu

9	Không sử dụng

Để ghép nối S7 – 200 với máy lập trình PG702 hoặc với các loại máy lập trình thuộc họ PG7xx có thể sử dụng cáp nối thẳng qua MPI .Cáp đó đi kèm theo máy lập trình .

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

Công tắc chọn chế độ làm việc của PLC

Công tắc chọn chế độ làm việc nằm phía trên, bên cạnh các cổng ra của S7 – 200 có ba vị trí cho phép chọn các chế độ làm việc khác nhau cho PLC.

-RUN cho phép PLC thực hiện chương trình trong bộ nhớ. PLC S7 – 200 sẽ rời khỏi chế độ RUN và chuyển sang chế độ STOP nếu trong máy có sự cố hoặc trong chương trình gặp lệnh STOP, thậm chí ngay cả khi công tắc ở chế độ RUN. Nên quan sát trạng thái thực tại của PLC theo đèn báo.

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

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

Chỉnh định tương tự

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

Pin và nguồn nuôi bộ nhớ

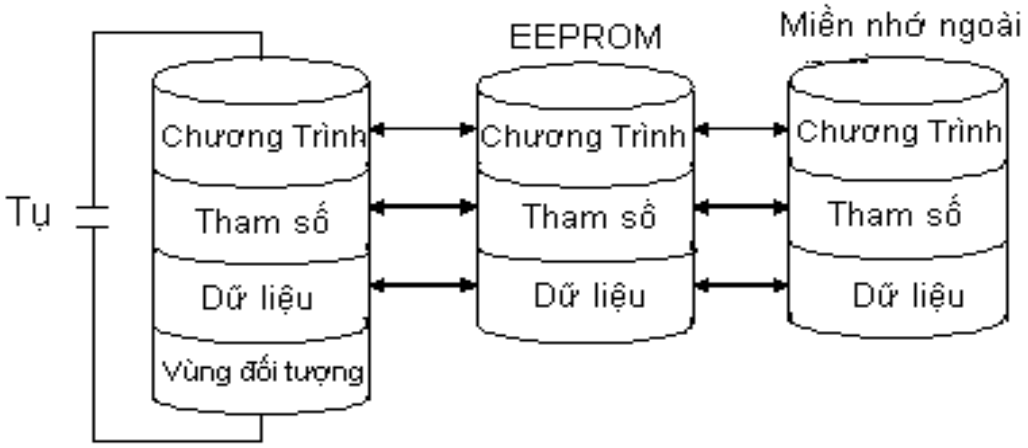
Nguồn nuôi dùng để mở rộng thời gian lưu giữ cho các dữ liệu có trong bộ nhớ. Nguồn pin tự động được chuyển sang trạng thái tích cực nếu như dung lượng tụ nhớ bị cạn kiệt và nó phải thay thế vào vị trí đó để dữ liệu trong bộ nhớ không bị mất đi.

2.1.2 Cấu trúc bộ nhớ

Phân chia bộ nhớ:

Bộ nhớ của S7 – 200 được chia thành 4 vùng với một tụ có nhiệm vụ duy trì dữ liệu trong một khoảng thời gian nhất định khi mất nguồn. Bộ nhớ của S7 – 200 có tính

năng động cao, đọc và ghi được trong tồn vùng, loại trừ phần bit nhớ đặc biệt được kí hiệu SM (Special Memory) chỉ có thể truy nhập để đọc.



Hình 3 : Bộ nhớ trong và ngồi của S7 - 200

Vùng chương trình: là miền nhớ được sử dụng để lưu các lệnh chương trình. Vùng này thuộc kiểu non-volatile đọc/ghi được.

Vùng tham số: là miền lưu giữ các tham số như: từ khóa, địa chỉ trạm … cũng như vùng chương trình, vùng tham số thuộc kiểu non-volatile đọc/ghi được.

Vùng dữ liệu: dùng để cất các dữ liệu của chương trình bao gồm các kết quả các phép tính, hằng số được định nghĩa trong chương trình, bộ đệm truyền thông … một phần của vùng nhớ này thuộc kiểu non-volatile.

Vùng đối tượng: Timer, bộ đếm, bộ đếm tốc độ cao và các cổng vào/ra tương tự được đặt trong vùng nhớ cuối cùng. Vùng này không kiểu non-volatile nhưng đọc/ghi được.

Vùng dữ liệu

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

Vùng dữ liệu lại được chia thành các 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 từng công dụng của chúng như sau:

V - Variable memory.

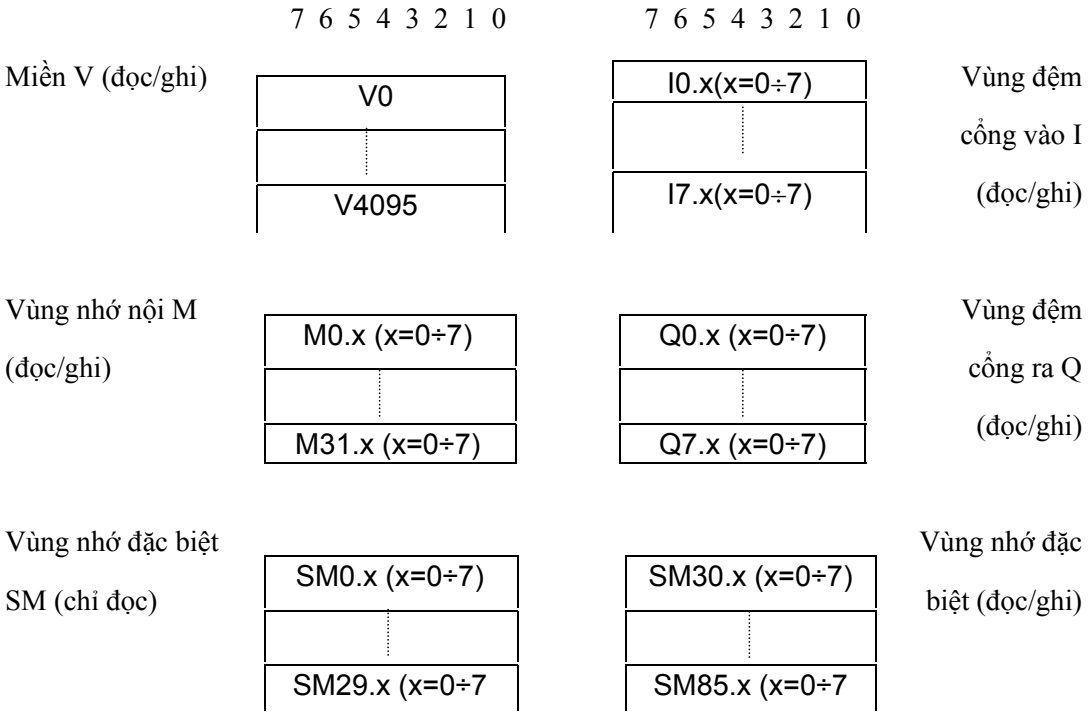
I - Input image regigter.

O - Output image regigter.

M - Internal memory bits.

SM - Speacial memory bits.

Tất cả các miền này đều có thể truy nhập được theo từng bit, từng byte, từng từ đơn (word-2byte) hoặc từ kép (2 word).



Hình 4 : Mô tả vùng dữ liệu của CPU 214

Địa chỉ truy nhập được qui ước theo công thức:

-*Truy nhập theo bit*: Tên miền (+) địa chỉ byte (+)•(+) chỉ số bit. Ví dụ V150.4 chỉ bit 4 của byte 150 thuộc miền V.

-*Truy nhập theo byte*: Tên miền (+) **B** (+) địa chỉ của byte trong miền. Ví dụ VB150 chỉ 150 thuộc miền V.

-*Truy nhập theo từ*: Tên miền (+) **W** (+) địa chỉ byte cao của từ trong miền. Ví dụ VW150 chỉ từ đơn gồm 2 byte150 và 151 thuộc miền V, trong đó byte 150 có vai trò byte cao trong từ.

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

VW150	VB150 (byte cao)	VB151 (byte thấp)

-*Truy nhập theo từ kép*: Tên miền (+) **D** (+) địa chỉ byte cao của từ trong miền.Ví dụ VD150 chỉ từ kép gồm 4 byte150, 151, 152 và 153 thuộc miền V, trong đó byte 150 có vai trò byte cao và byte 153 là thấp trong từ kép.

bit	63	32 31	16 15	8	0

VD150	VB150	VB151	VB152	VB153

Tất cả các byte thuộc vùng dữ liệu đều có thể truy nhập được bằng con trỏ. Con trỏ được định nghĩa trong miền V hoặc các thanh ghi AC1, AC2 và AC3. Mỗi con trỏ địa chỉ chỉ gồm 4 byte (từ kép).

Vùng đối tượng:

Vùng đối tượng được sử dụng để lưu giữ dữ liệu cho các đối tượng lập trình như các giá trị tức thời, giá trị đặt trước của bộ đếm, hay Timer. Dữ liệu kiểu đối tượng bao gồm của thanh ghi của Timer, bộ đếm, 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 của đối tượng đó.

Hình 5. Vùng nhớ đối tượng được phân chia như sau:

CPU214

	15 0	bit
Timer (đọc/ghi)	T0 … T127	T0 … T127
Bộ đếm (đọc/ghi)	C0 … C27	C0 … C127
Bộ đệm cổng vào	AW0 … AW30	

tương tự (chỉ đọc)

Bộ đệm cổng ra tương tự (chỉ ghi)

AQW0
AQW30

Thanh ghi Accumulator (đọc/ghi)

31	23	8	0
AC0 (không có khả năng làm con trỏ)			
AC1			
AC2			
AC3			

Bộ đếm tốc độ cao (đọc/ghi)

HSC0
HSC1 (chỉ có trong CPU 214)
HSC2 (chỉ có trong CPU 214)

2.1.3 Mở rộng ngõ vào/ra:

Có thể mở rộng ngõ vào/ra của PLC bằng cách ghép nối thêm vào nó các modul mở rộng về phía bên phải của CPU (CPU 214 nhiều nhất 7 modul), làm thành một móc xích, bao gồm các modul có cùng kiểu.

Các modul mở rộng số hay rời rạc đều chiếm chỗ trong bộ đệm, tương ứng với số đầu vào/ra của các modul.

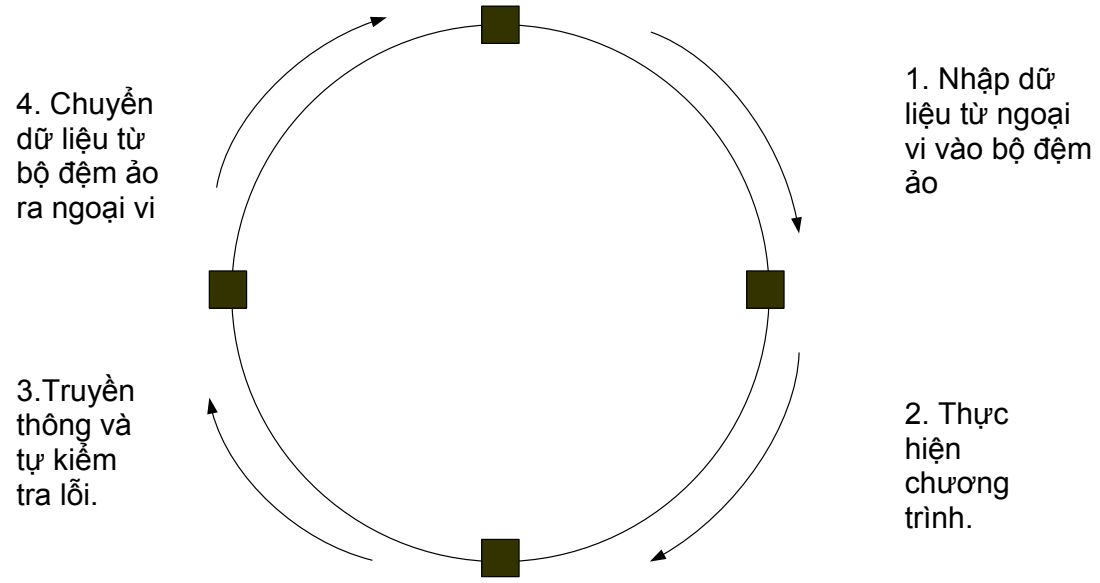
Sau đây là một ví dụ về cách đặt địa chỉ cho các modul mở rộng trên CPU 214:

CPU214	MODUL 0	MODUL 1	MODUL 2	MODUL 3	MODUL 4

		(4vào/4ra)	(8 vào)	(3vào analog /1ra analog)	(8 ra)	(3vào analog /1ra analog)
I0.0	Q0.0	I2.0	I3.0	AIW0	Q3.0	AIW8
I0.1	Q0.1	I2.1	I3.1	AIW2	Q3.1	AIW10
I0.2	Q0.2	I2.2	I3.2	AIW4	Q3.2	AIW12
I0.3	Q0.3	I2.3	I3.3	AQW0	Q3.3	AQW4
I0.4	Q0.4		I3.4		Q3.4	
I0.5	Q0.5	Q2.0	I3.5		Q3.5	
I0.6	Q0.6	Q2.1	I3.6		Q3.6	
I0.7	Q0.7	Q2.2	I3.7		Q3.7	
I1.1	Q1.0	Q2.3				
I1.2	Q1.1					
I1.3						
I1.4						
I1.5						

2.1.4 Thực hiện chương trình:

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



Hình 6: Vòng quét (scan) trong S7- 200.

Như vậy, tại thời điểm thực hiện lệnh vào/ra, thông thường lệnh không làm việc mà chỉ thông qua bộ đệm ảo của cổng trong vùng nhớ tham số. Việc truyền thông giữa bộ đệm ảo với ngoại vi trong các giai đoạn 1 và 4 do CPU quản lý. Khi gặp *lệnh vào/ra ngay lập tức* thì hệ thống sẽ cho dừng mọi công việc khác, ngay cả chương trình xử lý ngắt, để thực hiện lệnh này một cách trực tiếp với cổng vào/ra.

Nếu sử dụng các chế độ xử lý ngắt, chương trình con tương ứng với từng tín hiệu ngắt được soạn thảo và cài đặt như một bộ phận của chương trình. Chương trình xử lý ngắt chỉ được thực hiện trong vòng quét khi xuất hiện tín hiệu báo ngắt và có thể xảy ra ở bất cứ điểm nào trong vòng quét.

Cấu trúc chương trình của S7 – 200

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

-STEP 7 – Micro/DOS

-STEP 7 – Micro/WIN

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

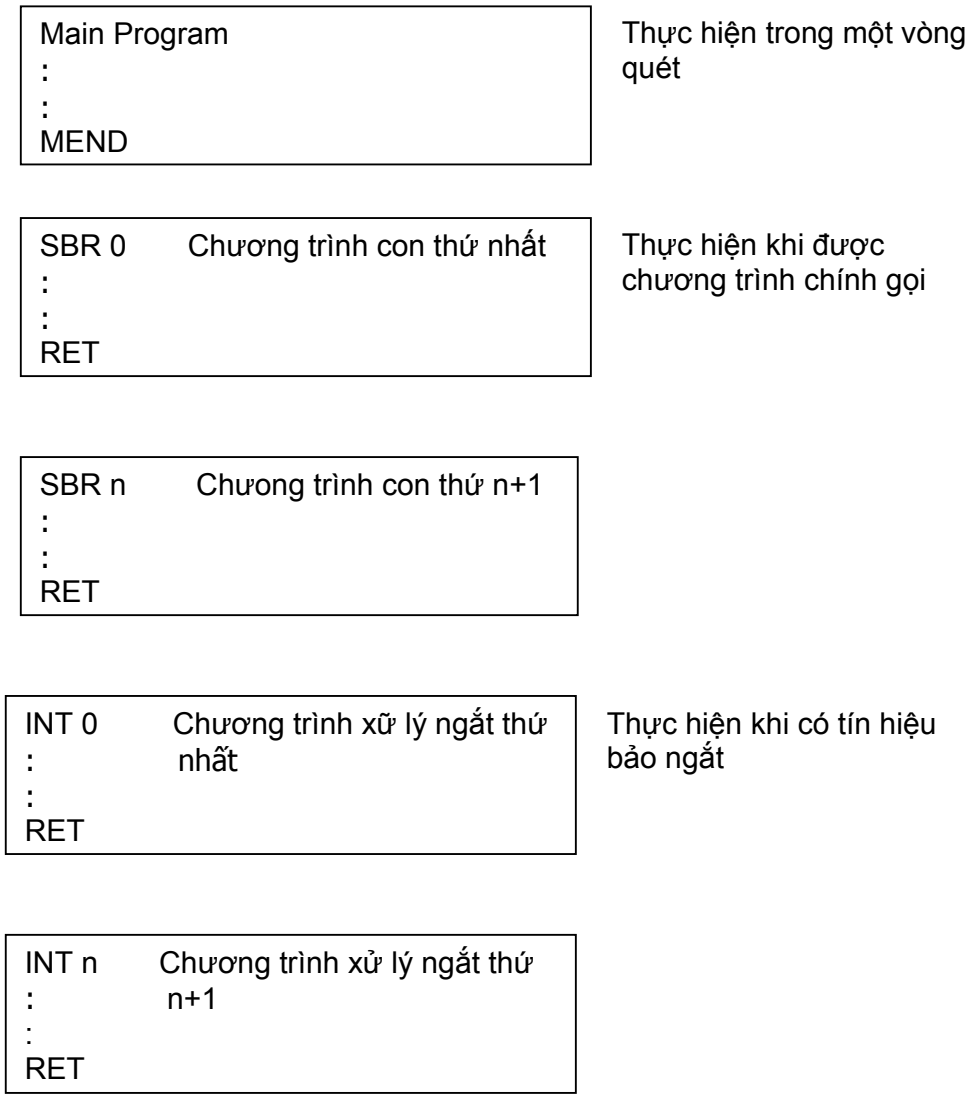
Các chương trình cho S7 – 200 phải có cấu trúc bao gồm chương trình chính (*main program*) và sau đó đến các chương trình con và các chương trình xử lý ngắt được chỉ ra sau đây:

-Chương trình chính được kết thúc bằng lệnh kết thúc chương trình (MEND)

-Chương trình con là một bộ phận của chương trình. Các chương trình con phải được viết sau lệnh kết thúc chương trình chính, đó là lệnh MEND.

-Các chương trình xử lý ngắt là một bộ phận của chương trình. Nếu cần sử dụng chương trình xử lý ngắt phải viết sau lệnh kết thúc chương trình chính MEND.

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



Hình 7: Cáu trúc chương trình S7 – 200

Hình 8: Hình ảnh thực tế của PLC S7 – 200

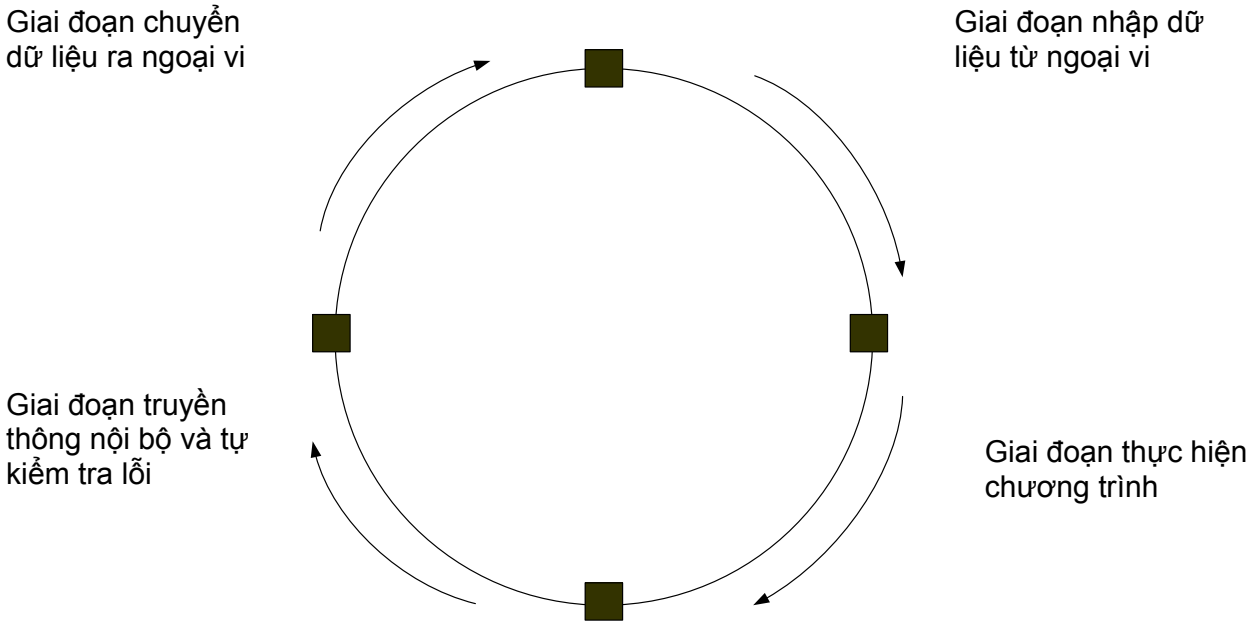
Hình 9: hình ảnh thực tế của một modul analog

2.1.5 Ngôn ngữ lập trình S7 – 200

2.1.5.1 Phương pháp lập trình

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

Một vòng (*scan cycle*) quét được bắt đầu bằng việc đọc trạng thái của đầu vào, và sau đó thực hiện chương trình. Scan cycle kết thúc bằng việc thay đổi trạng thái đầu ra. Trước khi bắt đầu một vòng quét tiếp theo S7 – 200 thực thi các nhiệm vụ bên trong và nhiệm vụ truyền thông. Chu trình thực hiện chương trình là chu trình lặp.



Hình 10: Thực hiện chương trình theo vòng quét trong S7 – 200.

Cách lập trình cho S7 – 200 nói riêng và cho các PLC của Siemens nói chung dựa trên hai phương pháp lập trình cơ bản: Phương pháp hình thang (*Ladder Logic* viết tắt là LAD) và phương pháp liệt kê lệnh (*Statement List* viết tắt là STL).

Nếu chương trình được viết theo kiểu LAD, thiết bị lập trình sẽ tự tạo ra một chương trình theo kiểu STL tương ứng. Nhưng ngược lại không phải mọi chương trình được viết theo kiểu STL cũng có thể chuyển được sang LAD.

Định nghĩa về LAD:

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

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

- *Cuộn dây* (*coil*): là biểu tượng ─( )─ mô tả các rơle được mắc theo chiều dòng điện cung cấp cho rơle.

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

-*Mạng LAD*: là đường nối các phần tử thành một mạch hồn thiện, đi từ đường nguồn bên trái sang đường nguồn bên phải. Đường nguồn bên trái là dây nóng, đường nguồn bên phải là dây trung hòa hay là đường trở về nguồn cung cấp (đường nguồn bên phải thường không được thể hiện khi dùng chương trình tiện dụng STEP7-Micro/DOS hoặc STEP7-Micro/WIN). Dòng điện chạy từ bên trái qua các tiếp điểm đến các cuộn dây hoặc các hộp trở về bên phải nguồn.

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

Định nghĩa về ngăn xếp logic (*logic stack*):

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

Để tạo ra một chương trình dạng STL, người lập trình cần phải hiểu rõ phương thức sử dụng 9 bit của ngăn xếp logic của S7 – 200. Ngăn xếp logic là một khối gồm 9 bit chồng lên nhau. Tất

cả các thuật tốn liên quan đến ngăn xếp đều chỉ làm việc với bit đầu tiên hoặc với bit đầu tiên và bit thứ hai của ngăn xếp. Giá trị logic mới đều có thể được gửi (hoặc được nối thêm) vào ngăn xếp. Khi phối hợp hai bit đầu tiên của ngăn xếp, thì ngăn xếp sẽ được kéo lên một bit.

Ví dụ về Ladder Logic và Statement List:

LAD	**STL**
I0.0 Q1.0 —┤ ├—( )	LD I0.0 = Q1.0

2.1.5.2 Cú pháp lệnh của S7 – 200

Hệ lệnh của S7 – 200: được chia làm ba nhóm:

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

-Các lệnh chỉ thực hiện khi bit đầu tiên của ngăn xếp có giá trị logic bằng 1.

-Các nhãn lệnh đánh dấu trong vị trí tập lệnh.

Các tốn hạng giới hạn cho phép của CPU 214:

Phương pháp truy nhập	Giới hạn cho phép của tốn hạng của CPU 214

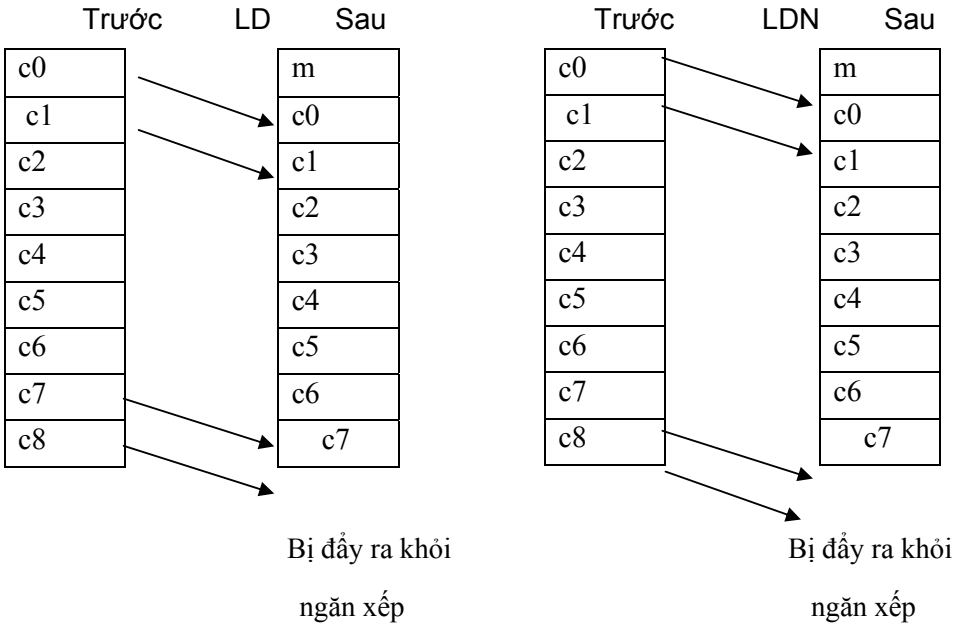
Truy nhập theo bit (địa chỉ byte, chỉ số bit)	V I Q M SM T C	(0.0 đến 4095.7) (0.0 đến 7.7) (0.0 đến 7.7) (0.0 đến 31.7) (0.0 đến 85.7) (0 đến 7.7) (0.0 đến 7.7)
Truy nhập theo byte	VB IB MB SMB AC Hằng số	(0 đến 4095) (0 đến 7) (0 đến 31) (0 đến 85) (0 đến 3)
Truy nhập theo từ đơn (word) (địa chỉ byte cao)	VW T C IW QW MW SMW AC AIW AQW Hằng số	(0 đến 4094) (0 đến 127) (0 đến 127) (0 đến 6) (0 đến 6) (0 đến 30) (0 đến 84) (0 đến 3) (0 đến 30) (0 đến 30)
Thuy nhập theo từ kép (địa chỉ byte cao)	VD ID QD MD SMD AC HC Hằng số	(0 đến 4092) (0 đến 4) (0 đến 4) (0 đến 28) (0 đến 82) (0 đến 3) (0 đến 2)

Một số lệnh cơ bản:

- **Lệnh vào/ra:**

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.

LOAD NOT (LDN): 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.



Các dạng khác nhau của lệnh LD, LDN cho LAD như sau:

LAD	Mô tả	Tốn hạng
n ┤├	Tiếp điểm thường mở sẽ đóng nếu n=1	n: I, Q, M, SM, (bit) T, C
n ┤\├	Tiếp điểm thường đóng sẽ mở khi n=1	
n ┤I├	Tiếp điểm thường mở sẽ đóng tức thời khi n=1	n:1
n ┤\I├	Tiếp điểm thường đóng sẽ mở tức thời khi n=1	

Các dạng khác nhau của lệnh LD, LDN cho STL như sau:

LAD	Mô tả	Tốn hạng
LD n	Lệnh nạp giá trị logic của điểm n vào bit đầu tiên trong ngăn xếp.	n: I, Q, M, SM, (bit) T, C
LDN n	Lệnh nạp giá trị logic nghịch đảo của điểm n vào bit đầu tiên trong ngăn xếp.	
LDI n	Lệnh nạp tức thời giá trị logic của điểm n vào bit đầu tiên trong ngăn xếp.	n:1
LDNI n	Lệnh nạp tức thời giá trị logic nghịch đảo của điểm n vào bit đầu tiên trong ngăn xếp.	

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.

Mô tả lệnh OUTPUT bằng LAD như sau:

LAD	Mô tả	Tốn hạng
n ─( )	Cuộn dây đầu ra ở trạng thái kích thích khi có dòng điều khiển đi qua	n:I,Q,M,SM,T,C (bit)
n ─(I)	Cuộn dây đầu ra được kích thích tức thời khi có dòng điều khiển đi qua	n:Q (bit)

- **Các lệnh ghi/xóa giá trị cho tiếp điểm:**

SET (S)

RESET (R): Lệnh dùng để đóng và ngắt các điểm gián đoạn đã được thiết kế. Trong LAD, logic điều khiển dòng điện đóng hay ngắt các cuộn dây đầu ra. Khi dòng điều khiển đến các cuộn dây thì các cuôn dây đóng hoặc 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 S hoặc R sẽ đóng ngắt 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.

Mô tả lệnh S (Set) và R (Reset) bằng LAD:

LAD	Mô tả	Tốn hạng
S bit n ──(S)	Đóng một mảng gồm n các tiếp điểm kể từ địa chỉ S-bit	S-bit: I, Q, M,SM,T, C,V (bit) n (byte): IB, QB, MB, SMB, VB,AC, hằng số, *VD, *AC
S bit n ──(R)	Ngắt một mảng gồm n các tiếp điểm kể từ S-bit. Nếu S-bit lại chỉ vào Timer hoặc Counter thì lệnh sẽ xố bit đầu ra của Timer/Counter đó.	
S bit n ──(SI)	Đóng tức thời một mảng gồm n các tiếp điểm kể từ địa chỉ S-bit	S-bit: Q (bit) n(byte):IB,QB, MB, SMB, VB,AC, hằng số, *VD, *AC
S bit n ──(RI)	Ngắt tức thời một mảng gồm n các tiếp điểm kể từ địa chỉ S-bit	

Mô tả lệnh S (Set) và R (Reset) bằng STL:

STL	Mô tả	Tốn hạng
S S-bit n	Ghi giá trị logic vào một mảng gồm n bit kể từ địa chỉ S-bit	S-bit: I, Q, M,SM,T, C,V (bit)
R S-bit n	Xóa một mảng gồm n bit kể từ địa chỉ S-bit. Nếu S-bit lại chỉ vào Timer hoặc Counter thì lệnh sẽ xố bit đầu ra của Timer/Counter đó.	
SI S-bit n	Ghi tức thời giá trị logic vào một mảng gồm n bit kể từ địa chỉ S-bit	S-bit: Q (bit) n (byte):IB,QB,MB, SMB, VB,AC, hằng số, *VD, *AC
RI S-bit n	Xóa tức thời một mảng gồm n bit kể từ địa chỉ S-bit.	

- **Các lệnh logic đại số Boolean:**

Các lệnh tiếp điểm đại số Boolean 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. Trong STL có thể sử dụng lệnh A (And) và O (Or) cho các hàm hở hoặc các lệnh AN (And Not), ON (Or Not) cho các hàm kín. Giá trị của ngăn xếp thay đổi phụ thuộc vào từng lệnh.

Lệnh	Mô tả	Tốn hạng
ALD	Lệnh tổ hợp giá trị của bit đầu tiên và thứ hai của ngăn xếp bằng phép tính logic AND. Kết quả ghi lại vào bit đầu tiên. Giá trị còn lại của ngăn xếp được kéo lên một bit.	Không có
OLD	Lệnh tổ hợp giá trị của bit đầu tiên và thứ hai của ngăn xếp bằng phép tính logic OR. Kết quả ghi lại vào bit đầu tiên. Giá trị còn lại của ngăn xếp được kéo lên một bit.	Không có
LPS	Lệnh Logic Push (LPS) sao chụp giá trị của bit đầu tiên vào bit thứ hai trong ngăn xếp. Giá trị còn lại bị đẩy xuống một bit. Bit cuối cùng bị đẩy ra khỏi ngăn xếp.	Không có
LRD	Lệnh sao chép giá trị của bit thứ hai vào bit đầu tiên trong ngăn xếp.Các giá trị còn lại của ngăn xếp giữ nguyên vị trí	Không có
LPP	Lệnh kéo ngăn xếp lên một bit. Giá trị của bit sau được chuyển cho bit trước.	Không có

Ngồi những lệnh làm việc trực tiếp với tiếp điểm, S7 – 200 còn có 5 lệnh đặc biệt biểu diễn cho các phép tính của đại số Boolean cho các bit trong ngăn xếp, được gọi là lệnh stack logic. Đó là các lệnh ALD (And Load), OLD (Or Load), LPS (Logic Push), LRD (Logic Read) và LPP (Logic Pop). Lệnh stack logic được dùng để tổ hợp, sao chụp hoặc xố các mệnh đề logic. LAD không có bộ đếm dành cho Stack logic. STL sử dụng các lệnh stack logic để thực hiện phương trình tổng thể có nhiều biểu thức con.

AND (A) Lệnh A và O phối hợp giá trị logic của một tiếp điểm n với

OR (O) giá trị bit đầu tiên của ngăn xếp. Kết quả phép tính được đặt lại vào bit đầu tiên trong ngăn xếp. Giá trị của các bit còn lại trong ngăn xếp không bị thay đổi.

Luật tinh tốn của các phép tính logic And và Or :

x	y	x ^ y(And)	x∨y (Or)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Tác động của các phép tính A (And) và O (Or)

A

Trước Sau

c0	m
c1	c1
c2	c2
c3	c3
c4	c4
c5	c5
c6	c6
c7	c7
c8	c8

m = c0 ^ n

O

Trước Sau

c0	m
c1	c1
c2	c2
c3	c3
c4	c4
c5	c5
c6	c6
c7	c7
c8	c8

m=c0∨n

m = c0

AND LOAD (ALD)

OR LOAD (OR) : Lệnh ALD và OLD thực hiện phép tính logic And và Or giữa hai bit đầu tiên của ngăn xếp. Kết quả của logic này sẽ được ghi lại vào bit đầu trong ngăn xếp. Nội dung còn lại của ngăn xếp được kéo lên một bit.

Tác động của lệnh ALD và OLD VÀO ngăn xếp như sau:

ALD

Trước	Sau	
c0	m	m = c0 ^ c1
c1	c2	
c2	c3	
c3	c4	
c4	c5	
c5	c6	
c6	c7	
c7	c8	
c8		

OLD

Trước	Sau	
c0	m	m = c0 ∨ c1
c1	c2	
c2	c3	
c3	c4	
c4	c5	
c5	c6	
c6	c7	
c7	c8	
c8		

LOGIC PUSH (LPS)

LOGIC READ (LRD)

LOGIC POP (LPP): Lệnh LPS, LRD và LPP là những lệnh thay đổi nội dung bit đầu tiên của ngăn xếp. Lệnh LPS sao chép nội dung bit đầu tiên vào bit thứ hai trong ngăn xếp, nội dung ngăn xếp sau đó bị đẩy xuống một bit. Lệnh LRD lấy giá trị bit thứ hai ghi vào bit đầu tiên của ngăn xếp, nội dung ngăn xếp sau đó được kéo lên một bit. Lệnh LPP kéo ngăn xếp lên một bit.

- **Các lệnh tiếp điểm đặc biệt ┤ NOT ├ ┤ P ├ ┤ N ├**

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 tốn hạng riêng của chính chúng vì thế 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 nhu cầu về bộ nhớ bởi vậy đối với CPU 214 có thể sử dụng nhiều nhất là 256 lệnh.

- **Các 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 hay Dword của S7 – 200.

AD sử dụng lệnh so sánh để so sánh các giá trị của byte, word hay Dword (giá trị thực hoặc nguyên). Những lệnh so sánh thường là: so sánh nhỏ hơn hoặc bằng (<=); so sánh bằng (==) và so sánh lớn hơn hoặc bằng (>=).

Khi so sánh giá trị của byte thí không cần phải để ý đến dấu của tốn hạng, ngược lại khi so sánh các từ hay từ kép với nhau thì phải để ý đến dấu của tốn hạng là bit cao nhất trong từ hoặc từ kép. Ví dụ 7FFF > 8000 và 7FFFFFFF > 80000000.

LAD	Mô tả	Tốn hạng
n1 n2 ┤==B├ n1 n2 ┤==I├ n1 n2 ┤==D├ n1 n2 ┤==R├	Tiếp điểm đóng khi n1=n2 B = byte I = Integer = Word D = Double Integer R = Real	n1, n2(*byte*): VB, IB, QB, MB, SMB, AC, Const, *VD, *AC
n1 n2 ┤>=B├ n1 n2 ┤>=I├ n1 n2 ┤>=D├ n1 n2 ┤>=R├	Tiếp điểm đóng khi n1≥ n2 B = byte I = Integer = Word D = Double Integer R = Real	n1, n2(*word*): VW, T, C, QW, MW, SMW, AC, AIW, hằng số, *VD, *AC
n1 n2 ┤<=B├ n1 n2 ┤<=I├ n1 n2 ┤<=D├ n1 n2 ┤<=R├	Tiếp điểm đóng khi n1≤ n2 B = byte I = Integer = Word D = Double Integer R = Real	n1, n2(D*word*): VD, ID, QD, MD, SMD, AC, HC, hằng số, *VD, *AC

Trong STL những lệnh so sánh thực hiện phép so sánh byte, từ hay từ kép. Căn cứ vào kiểu so sánh (<=, ==, >=), kết quả của phép so sánh có giá trị bằng 0 (nếu đúng) hoặc bằng 1 (nếu sai) nên nó có thể sử dụng kết hợp cùng các lệnh LD, A, O. Để tạo ra được các phép so sánh mà S7 – 200 không có lệnh so sánh tương ứng như: so sánh không bằng nhau (<>), so sánh nhỏ hơn (<) hoặc so sánh lớn hơn (>), có thể tạo ra được nhờ kết hợp lệnh **NOT** với các lệnh đã có (==, >=, <=)

- **Lệnh nhảy và lệnh gọi chương trình con**

Các lệnh của chương trình, nếu không có những lệnh điều khiển riêng, sẽ được thực hiện theo thứ tự từ trên xuống dưới trong một vòng quét. Lệnh điều khiển chương trình cho phép thay đổi thứ tự thực hiện lệnh. Chúng cho phép chuyển thứ tự thực hiện, đáng lẽ ra là lệnh tiếp theo, tới một lệnh bất cứ nào khác của chương trình, trong đó nơi điều khiển chuyển đến được đánh dấu trước bằng một *nhãn chỉ đích*. Thuộc nhóm lệnh điều khiển chương trình gồm: *lệnh nhảy, lệnh gïọi chương trình con*. Nhãn chỉ đích, hay gọi đơn giản là nhãn, phải được đánh dấu trước khi thực hiện nhảy hay lệnh gọi chương trình con.

Việc đặt nhãn cho *lệnh nhảy* phải nằm trong chương trình. Nhãn của chương trình con, hoặc của chương trình xử lý ngắt được khai báo ở đầu chương trình. Không thể dùng *lệnh nhảy* JMP để chuyển điều khiển từ chương trình chính vào một vào một nhãn bất kỳ trong chương trình con hoặc trong chương trình xử lý ngắt. Tương tự như vậy cũng không thể từ một chương trình con hay chương trình xử lý ngắt nhảy vào bất cứ một nhãn nào nằm ngồi các chương trình đó.

Lệnh gọi chương trình con là lệnh chuyển điều khiển đến chương trình con. Khi chương trình con thực hiện các phép tính của mình thì việc điều khiển lại được chuyển trở về lệnh tiếp theo trong chương trình chính ngay sau lệnh gọi chương trình con. Từ một chương trình con có thể gọi được một chương trình con khác trong nó, có thể gọi như vậy nhiều nhất là 8 lần trong S7 – 200. Đệ qui (trong một chương trình con có lệnh gọi đến chính nó) về nguyên tắc không bị cấm song phải chú ý đến giới hạn trên.

Nếu lệnh nhảy hay lệnh gọi chương trình con được thực hiện thì đỉnh ngăn xếp luôn có giá trị logic bằng 1. Bởi vậy trong chương trình con các lệnh có điều khiển được thực hiện như các lệnh không điều kiện. Sau các lệnh LBL (đặt nhãn) và SBR, lệnh LD trong STL sẽ bị vô hiệu hóa.

Khi một chương trình con được gọi, tồn bộ nội dung của ngăn xếp sẽ được cất đi, đỉnh của ngăn xếp nhận một giá trị mới là 1, các bit khác còn lại của ngăn xếp nhận giá trị logic 0 và chương trình được chuyển tiếp đến chương trình con đã được gọi. Khi thực hiện xong chương trình con và trước khi điều khiển được chuyển trở lại chương trình đã gọi nó, nội dung ngăn xếp đã được cất giữ trước đó sẽ được chuyển trở lại ngăn xếp.

Nội dung của thanh ghi AC không được cất giữ khi gọi chương trình con, nhưng khi một chương trình xử lý ngắt được gọi, nội dung của thanh ghi AC sẽ được cất giữ trước khi thực hiện chương trình xử lý ngắt và nạp lại khi chương trình xử lý ngắt đã được thực hiện xong. Bởi vậy chương trình xử lý ngắt có thể tự do sử dụng bốn thanh ghi AC của S7 – 200.

JMP, CALL

LBL, SBR : Lệnh nhảy JMP và lệnh gọi chương trình con SBR cho phép chuyển điều khiển từ vị trí này đến một vị trí khác trong chương trình. Cú pháp lệnh nhảy và lệnh gọi chương trình

con trong LAD và STL đều có tốn hạng là nhãn chỉ đích (nơi nhảy đến, nơi chứa chương trình con)

LAD	STL	Mô tả	Tốn hạng
n ─(JMP)	JMP Kn	Lệnh nhảy thực hiện việc chuyển điều khiển đến nhãn n trong một chương trình.	n: CPU 212: 0÷63 CPU 214: 0÷255
LBL: n	JMP Kn	Lệnh khai báo nhãn n trong một chương trình.	
n ─(CALL)	CALL Kn	Lệnh gọi chương trình con, thực hiện phép chuyển điều khiển đến chương trình con có nhãn n.	n: CPU 212: 0÷15 CPU 214: 0÷255
SBR:n	SBR Kn	Lệnh gán nhãn cho một chương trình con.	
─(CRET)	CRET	Lệnh trở về chương trình đã gọi chương trình con có điều kiện (bit đầu của ngăn xếp có giá trị logic bằng 1)	Không có
─(RET)	RET	Lệnh trở về chương trình đã gọi chương trình con không điều kiện.	

- **Các lệnh can thiệp vào thời gian vòng quét**

MEND, END, STOP, NOP, WDR

Các lệnh này được dùng để kết thúc chương trình đang thực hiện, và kéo dài một khoảng thời gian của một vòng quét.

Trong LAD và STL chương trình phải được kết thúc bằng *lệnh kết thúc không điều kiện* MEND. Có thể sử dụng lệnh kết thúc có điều kiện END trước lệnh kết thúc không điều kiện.

Lệnh STOP kết thúc chương trình, nó chuyển điều khiển chương trình đến chế độ STOP. Nếu như gặp lệnh STOP trong chương trình chính, hoặc trong chương trình con thì chương trình đang được thực hiện sẽ kết thúc ngay lập tức.

Lệnh rỗng NOP không có tác dụng gì trong việc thực hiện chương trình. Cần lưu ý lệnh NOP phải được đặt bên trong chương trình chính, chương trình con hoặc trong chương trình xử lý ngắt.

Lệnh WDR sẽ khởi động lại đồng hồ quan sát (*watchdog timer*), và chương trình tiếp tục được thực hiện trong vòng quét ở chế độ quan sát nên cẩn thận khi sử dụng lệnh WDR.

Việc chuyển công tắc cứng của S7 – 200 vào vị trí STOP hoặc thực hiện lệnh STOP trong chương trình sẽ là nguyên nhân đặt điều khiển vào chế độ dừng trong khoảng thời gian 1,4s …

- **Các lệnh điều khiển Timer**

Timer là bộ tạo thời gian giữa tín hiệu ra nên trong điều khiển vẫn 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 hai loại khác nhau là:

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

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

Hai kiểu Timer của S7 – 200 (TON và TONR) phân biệt với nhau ở phản ứng của nó đối với trạng thái đầu vào.

Cả hai Timer kiểu TON và TONR cùng bắt đầu tạo thời gian trễ tín hiệu kể từ thời điểm có sườn lên ở tín hiệu đầu vào, tức là khi tín hiệu đầu vào chuyển trạng thái logic từ 0 lên 1, được gọi là *thời gian Timer được kích*, và không tính khoảng thời gian khi đầu vào có giá trị logic 0 vào thời gian trễ tín hiệu đặt trước.

Khi đầu vào có giá trị logic bằng 0, TON tự động reset còn TONR thì không tự động reset. 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 TON và TONR bao gồm 3 loại với ba 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 là 50 thì thời gian trễ sẽ là τ = 500ms.

Timer của S7 – 200 có những tính chất cơ bản sau:

-Các bộ Timer được điều khiển bởi một cổng vào và giá trị đếm tức thời. Giá trị đếm tức thời của Timer được nhớ trong thanh ghi 2 byte (gọi là T-word) của Timer, xác định khoảng thời gian trễ kể từ khi Timer được kích. Giá trị đặt trước của các bộ Timer được ký hiệu trong LAD và STL là PT. Giá trị đếm tức thời của thanh ghi T-word thường xuyên được so sánh với giá trị đặt trước của Timer.

-Mỗi bộ Timer, ngồi thanh ghi 2 byte T-word lưu giá trị đếm tức thời, còn có một bit ký hiệu là T-bit, chỉ thị trạng thái logic đầu ra. Giá trị logic của bit này phụ thuộc vào kết quả so sánh giữa giá trị đếm tức thời với giá trị đặt trước.

-Trong khoảng thời gian tín hiệu x(t) có giá trị logic 1, giá trị đếm tức thời trong T-word luôn được cập nhật và thay đổi tăng dần cho đến khi nó đạt giá trị cực đại. Khi giá trị đếm tức thời lớn hơn hay bằng giá trị đặt trước, T-bit có giá trị logic 1.

Độ phân giải các loại Timer của S7 – 200, CPU 214

Lệnh	Độ phân giải	Giá trị cực đại	CPU 214
TON	1 ms	32,767 s	T32 và T96
	10 ms	327,67 s	T33 ÷ T36, T97 ÷ T100
	100 ms	3276,7 s	T32 ÷ T96, 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

Cú pháp khai báo sử dụng Timer như sau:

LAD	Mô tả	Tốn hạng
TON—Txx —IN	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.	Txx (*word*) CPU214:32÷63 96÷127 PT: VW, T,

— PT	có thể reset Timer kiểu TON bằng lệnh R hoặc bằng giá trị logic 0 tại đầu vào IN.	(*word*) C, IW, QW,MW,SMW, AC,AIW,hằngsố
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 (*word*) CPU214: 0 ÷31 64 ÷95 PT: VW, T, (*word*) C, IW, QW,MW,SMW, AC,AIW,hằngsố

Khi sử dụng Timer kiểu 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 hồn tồn 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 tin hiệu đầu ra cũng có trạng thái logic bằng 0.

- **Các lệnh điều khiển Counter**

Counter là bộ đếm thực hiện chức năng đếm sườn xung trong S7 – 200. Các bộ đếm của S7 – 200 được chia làm hai loại: bộ đếm tiến (CTU) và bộ đếm tiến/lùi (CTUD).

Bộ đếm tiến 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, 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 ngồi 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 C-bit có giá trị logic là 0.

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 khởi phát 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 xố này có mức logic là 1 hoặc khi lệnh R (reset) được thực hiện với C-bit. Khi bộ đếm được reset, cả C-word và C-bit đều nhận giá trị 0.

Bộ đếm tiến/lùi CTUD đếm tiến khi gặp sườn lên của xung vào cổng đếm tiến, ký hiệu là CU hoặc bit thứ 3 của ngăn xếp trong STL, và đếm lùi khi gặp sườn lên của xung vào cổng đếm lùi, ký hiệu là CD trong LAD hoặc bit thứ 2 của ngăn xếp trong STL.

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

LAD	Mô tả	Tốn hạng
CTU – Cxx —CU —PV —R	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 : (*word*) CPU 214: 0 ÷47 80 ÷127 PV :(*word*): VW, T,C,IW,QW,MW, SMW, AC, AIW, hằngsố,*VD,*AC
CTUD – Cx —CU —PV —CD —R	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 : (*word*) CPU 214: 48 ÷79 PV :(*word*): VW, T,C,IW,QW,MW, SMW, AC, AIW, hằngsố,*VD,*AC

- **Đồng hồ thời gian thực**

Đồng hồ thời gian thực chỉ có với CPU 214 .Để có thể làm việc với đồng hồ thời gian thực CPU 214 cung cấp 2 lệnh đọc và ghi giá trị cho đồng hồ. Những giá trị đọc được hoặc ghi được với đồng hồ thời gian thực là các giá trị về ngày, tháng, năm và các giá trị về giờ, phút, giây.

Các dữ liệu đọc, ghi với đồng hồ thời gian thực trong LAD và trong STL có độ dài một byte và phải được mã hố theo kiểu số nhị phân BCD(thí dụ 16#95 cho năm 95).Chúng nằm trong *bộ đếm gồm 8 byte liền nhau* theo thứ tự.

byte 0	Năm (0÷99)

byte 1	Tháng (0÷12)
byte 2	Ngày (0÷31)
byte 3	Giờ (0÷23)
byte4	Phút (0÷59)
byte 5	Giây (0÷59)
byte 6	0
byte 8	0 ngày trong tuần

Các kiểu dữ liệu hợp lệ là :

Năm (yy)	Tháng (mm)	Ngày (dd)	Giờ (hh)	Phút (mm)	Giây (ss)
0÷99	1÷12	1÷31	0÷23	0÷59	0÷59

Riêng giá trị về ngày trong tuần là một số tương ứng với nội dung của nibble thấp (4 bit) trong byte teo kiểu

Chủ nhật	Thứ hai	thứ ba	Thứ tư	Thứ năm	Thứ sáu	Thứ bẩy
1	2	3	4	5	6	7

READ_RTC (LAD)

TODR (STL): Lệnh đọc nội dung của đồng hồ thời gian thực vào bộ đệm 8 byte được chỉ thị trong bảng tốn hạng T

SET_RTC (LAD)

TODW (STL): Lệnh ghi nội dung của bộ đệm 8 byte được chỉ thị bằng lệnh tốn hạng T vào đồng hồ thời gian thực.

Cú pháp sử dụng lệnh đọc, ghi với đồng hồ thời gian thực trong LAD và STL:

LAD	STL	Tốn hạng
TON—Txx EN T	TODR T	T VB, IB, QB, MB,SMB, *VD, *AC (byte)
TONR—Txx IN T	TODW T	

Tuyệt đối không sử dụng lệnh TODR và lệnh TODW đồng thời vừa trong chương trình chính, vừa trong chương trình xử lý ngắt. Khi một lệnh TODR hay TODW đã được thực hiện, thì khi gọi chương trình xử lý ngắt, các lệnh làm việc với đồng hồ thời gian thực trong chương trình xử lý ngắt sẽ không được thực hiện nữa. Bít SM4.5 sẽ có logic 1 trong những trường hợp như vậy.

2.2 Microwin

2.2.1 Cài đặt STEP7 – Micro/ Win

STEP7 – Micro/ Win là phần mềm hỗ trợ:

- Khai báo cấu hình cứng cho trạm PLC thuộc họ Simatic S7 – 200.
- Soạn thảo và cài đặt chương trình điều khiển cho một hoặc nhiều trạm.
- Quan sát việc thực hiện chương trình điều khiển trong một trạm PLC và gỡ rối chương trình.

Ngồi ra Step7- Micro/Win còn có cả một thu viện đầy đủ với các hàm chuẩn hữu ích, phần trợ giúp, online rất mạnh có khả năng trả lời mọi câu hỏi của người sử dụng về cách sử dụng Step7 – Micro/Win, về cú pháp lệnh trong lập trình, về xây dựng cấu hình cứng của một trạm cũng như của mạng gồm nhiều trạm PLC…

Cài đặt

Hiện nay, ở nước ta phần mềm về Step7 – Micro/Win được sử dụng nhiều nhất là phiên bản Step7 – Micro/Win 32.

Phần lớn các đĩa gốc của Step7 đều có khả năng tự thực hiện chương trình cài đặt (autorum). Bởi vậy ta chỉ cần cho đĩa vào ổ CD và thực hiện theo đúng các chỉ dẫn hiện trên màn hình. Ta cũng có thể chủ động thực hiện việc cài đặt bằng cách gọi chương trình setup. exe có trên đĩa. Công việc cài đặt Step7 về cơ bản không khác nhiều so với việc cài đặt các phần mềm ứng dụng khác (Window, Office…), tức là cũng bắt đầu bằng việc chọn ngôn ngữ trong cài đặt (mặc định là tiếng Anh), chọn thư mục đích trên ổ cứng (mặc định là C:\simens), kiểm tra dung tích còn lại trên ổ đích, chọn ngôn ngữ sẽ được sử dụng trong quá trình làm việc với Step7 sau này…

Đặt tham số làm việc

Sau khi cài đặt xong Step7, trên màn hình (desktop) sẽ xuất hiện biểu tượng (icon) của nó như hình dưới. Đồng thời trong Menu của Window cũng có thư mục Simatic với tất cả các tên của những thành phần liên quan, từ các phần trợ giúp đến các phần mềm đặt cấu hình, chế độ làm việc của Step7…

STEP7
MicroWin32

2.2.2 Soạn thảo một Project

Khái niệm Project trong Simatic không đơn thuần chỉ là chương trình ứng dụng mà rộng hơn bao gồm tất cả những gì liên quan đến việc thiết kế phần mềm ứng dụng để điều khiển, giám sát một hay nhiều trạm PLC.Theo khái niệm như vậy, trong một Project sẽ có:

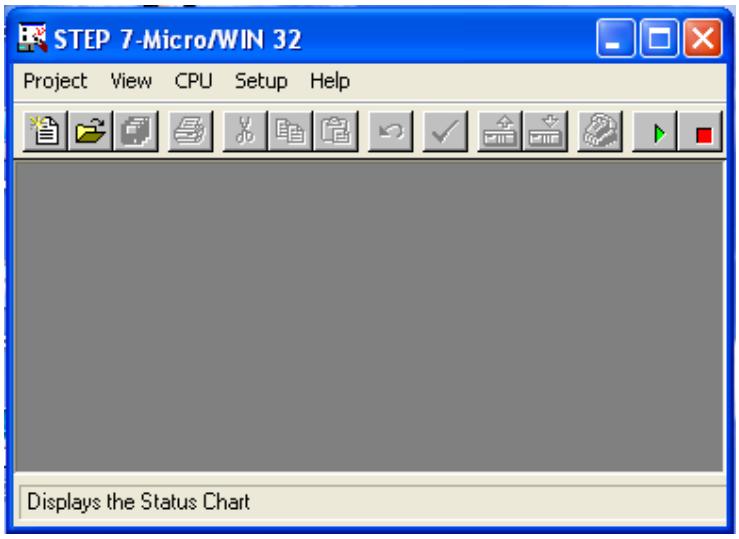
- bảng Ladder Editor gồm một thư viện (tiếp điểm,cuộn dây, hộp…) và phần dùng để viết chương trình điều khiển.
- bảng Statur Chart
- bảng Data Block Editor
- bảng Symbol Table dùng để khai báo biến hình thức.

Khai báo và mở một Project

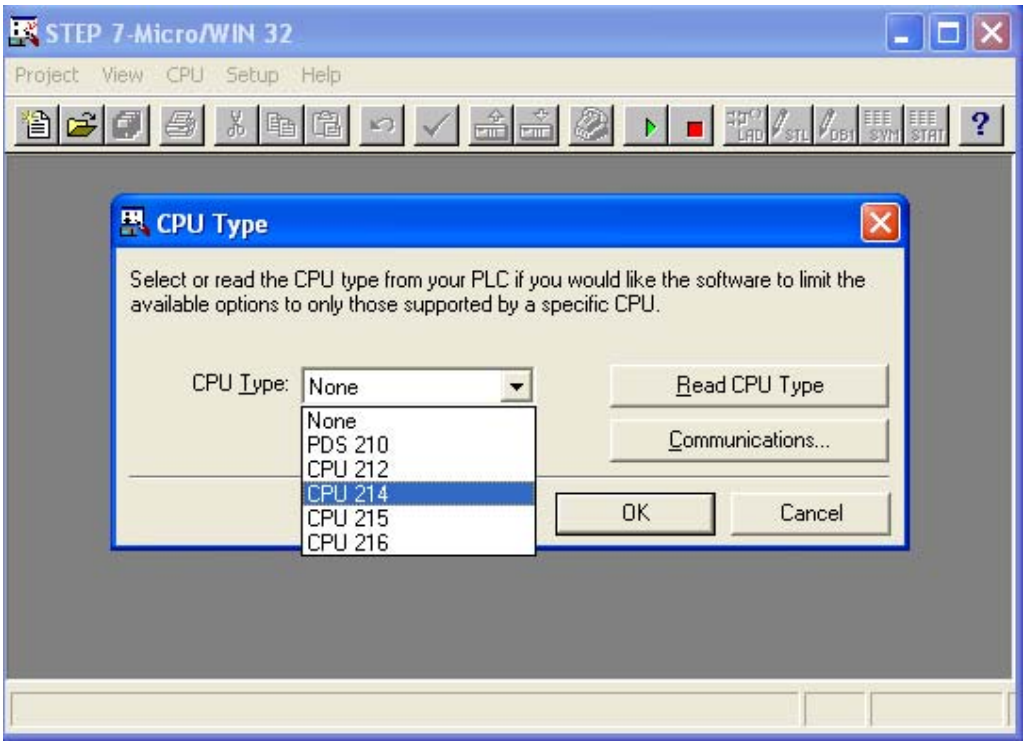
Để khai báo một Project, từ màn hình chính của Step7 ta chọn **Project** rồi chọn **New** nếu là **file** mới còn chọn **open** nếu chọn **file** đã có.

Khai báo một Project mới

Mở một Project đã có



Sau khi tạo lập Project mới màn hình sẽ hiện ra CPU Type để ta chọn loại CPU mong muốn bằng cách ấn **OK**



Khi đã chọn OK, file mới được thiết lập với đầy đủ thư viện và các công cụ trợ giúp thực hiện bài tốn.

Stop Mode

Ladder

Statement List

Run Mode

Download

Data Block

Upload

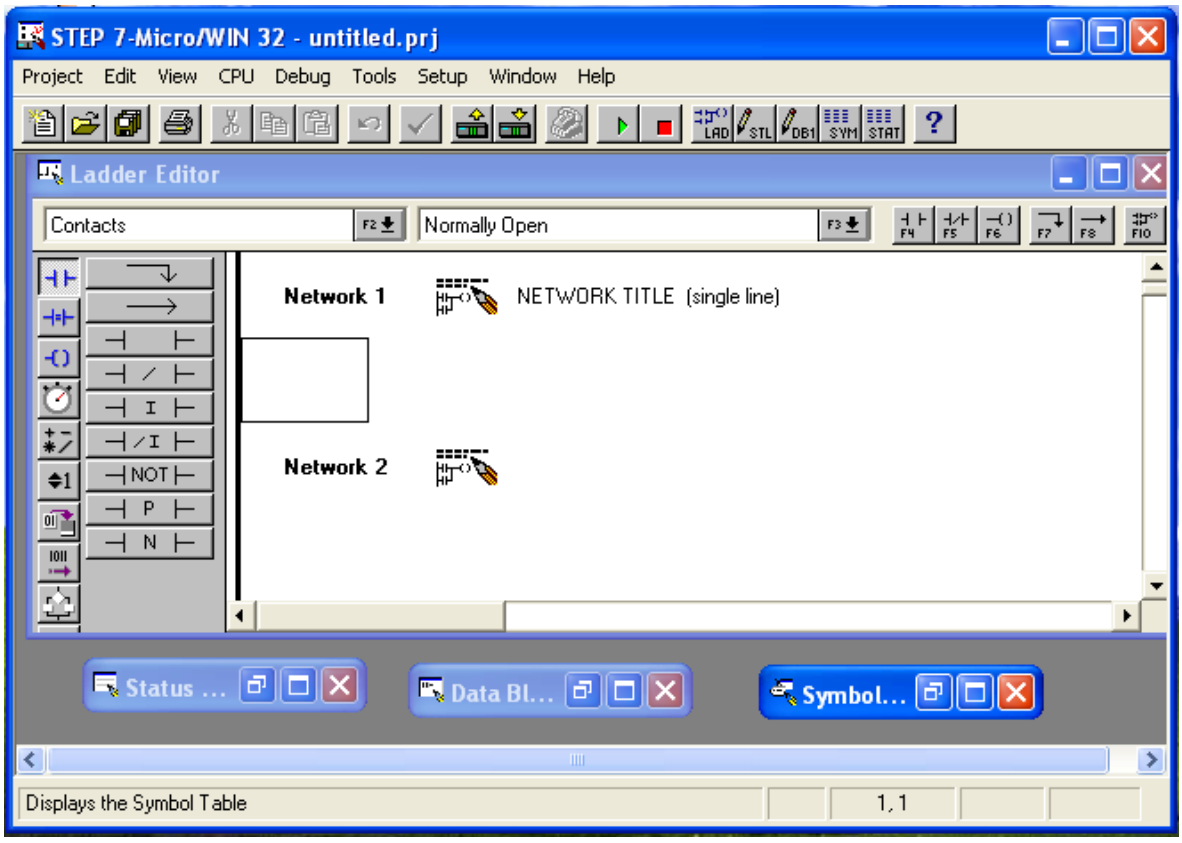
Complie

Symbol Table

Status Chart

Help

Thư viện



Chương 3 : CHƯƠNG TRÌNH ĐIỀU KHIỂN ĐÈN GIAO THÔNG BẰNG S7 – 200

3.1 Bài tốn

Viết chương trình điều khiển đèn giao thông ở ngã tư bằng hệ S7-200.Với điều kiện như sau:

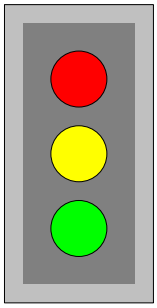
Từ 22g ÷ 5g: Tự xử hai đèn vàng nhấp nháy với chu kỳ đèn là T=4s.

6g ÷ 9g: Chu kỳ đèn T=60s.

10g÷ 15g: Chu kỳ đèn T=46s.

16g ÷ 18g: Chu kỳ đèn T=60s.

19g ÷ 21g: Chu kỳ đèn T=46s.



Đèn đỏ

Đèn vàng

Đèn xanh

Chọn thiết bị và cổng vào ra

- Chọn thiết bị là CPU 214 gồm:

  - Một đông hồ thời gian thực.
  - 14 DI.
  - 10 DO.
  - Một modul 0 (4 vào/4 ra) mở rộng.

-Chọn cổng vào/ra:

- Cổng vào:

  . Bằng tay I0.2

  . Tự động : Start I0.0

  Stop I0.1

Đèn xanh 1(đx1) =1; đèn đỏ 2 (đđ2) =1 : I1.1

Đèn vàng 1 (đv1) =1 :I1.2

Đèn đỏ 1(đđ1) =1; đèn xanh 2(đx2)=1:I1.3

Đèn vàng 2(đv2)=1 :I1.4

. Chỉnh đồng hồ: I1.0

- Cổng ra

  - Làn 1: đx1: Q0.0

đv1: Q0.1

đđ1: Q0.2

đèn xanh người 1(đxn1): Q0.3

đèn đỏ người 1(đđn1): Q0.4

- Làn 2: đx2: Q0.5

đv2: Q0.6

đđ2: Q0.7

đèn xanh người 2(đxn2): Q2.0

đèn đỏ người 2(đđn2): Q2.1

Main Program(OB1)

- Chỉnh đồng hồ

  TODW Table

- Đọc đồng hồ

  TODR Table

- Ra quyết định điều khiển

  22g ÷ 5g : M4.0

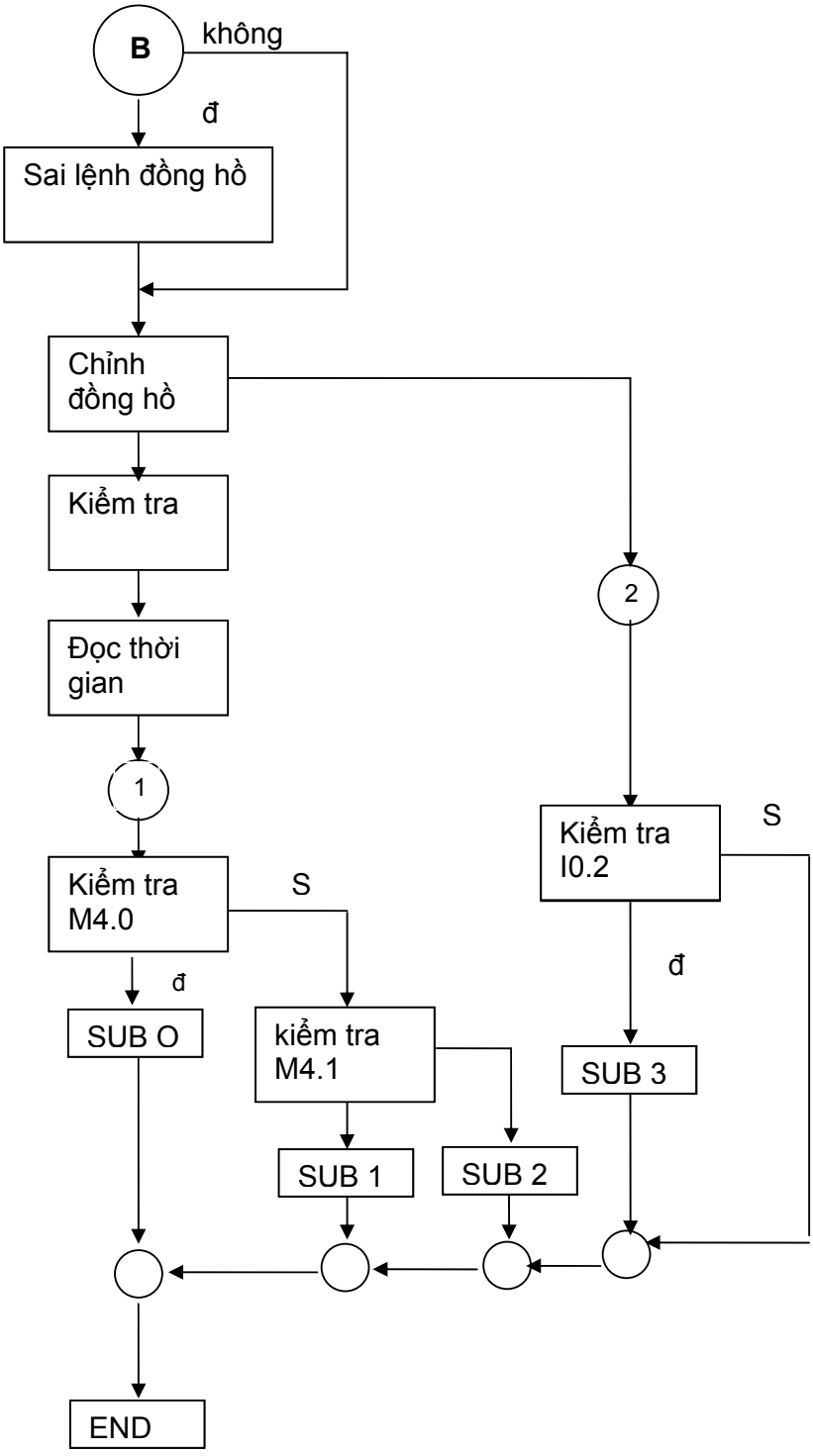
  6g ÷ 9g và 16g ÷ 18g : M4.1

10g ÷ 15g và 19g ÷ 21g: M4.2

- Gọi chương trình điều khiển

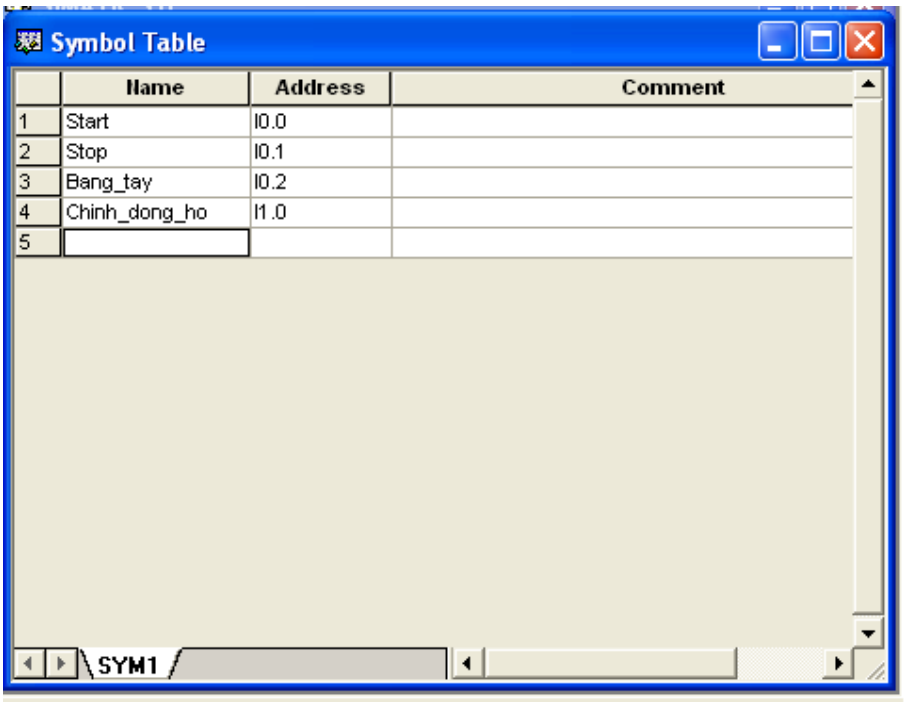
I0.1 = 1 và M4.0 = 1	SUB 0 - Tự xử
I0.1 = 1 và M4.1 = 1	SUB 1 – T=60s
I0.1 = 1 và M4.2 = 1	SUB 2 – T=46s
I0.2 =1	SUB 3 – Bằng tay
I0.0 =1	SUB 4 - Tự động

3.2 Sơ đồ khối của chương trình

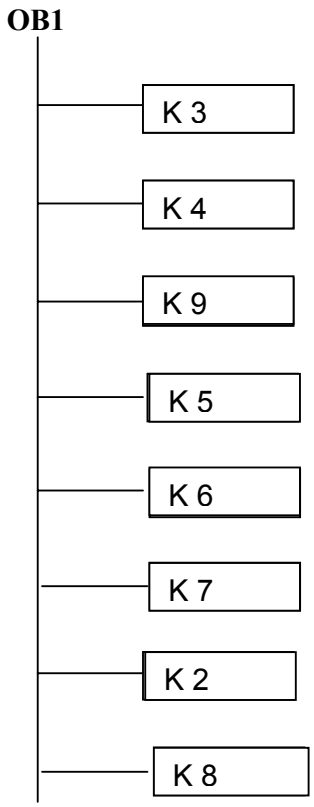


3.3 Cài đặt chương trình cho S7 – 200

Để chọn các cổng vào ra ta lập bảng Symbol sau:



Cấu trúc chương trình



Chương trình điều khiển

```
Var _ input
    Chu_ky:   word
    ss1:      INT;
    ss2:      INT;
    ss3:      INT;
    ss4:      INT;
    Timer:    T1;
    Congtacdk:    Bool;
End_Var
Var_out put
    dx 1:     bool;
    dx2:      bool;
    dd1:      bool;

    dd2:      bool;
```

```
    dv1:        bool;
    dv2:        bool;
    dxn1:       bool;
    dxn2:       bool;
    ddn1:       bool;
    ddn2:       bool;
End_Var
Begin;
```

OB1:

```
Network 1// Cài đặt tham số cho chu kỳ đèn T = 60s
        LD      M1.0
        EU
        Call    K4
        LD      I1.3
        O       I1.4
        JMP     N2
Network 2// Cài đặt tham số cho chu kỳ đèn T =46s
        LD      M1.1
        EU
        Call    K3
Network 3// Định nghĩa có cho phép M2.0
        LD      I0.0    // start
        AN      I0.1    // stop
            = M0.0
        LD      M0.0
        A       M1.0
        O       M1.1
            = M2.0

Network 4 // Đèn báo hệ thống làm việc ở chế độ tự động
```

LD M2.0

S Q2.2

Network 5 // Gọi chương trình điều khiển tự động

Call K9

Cho_phep: = M2.0

TON : = T1

Chu_ky : = VM10

So sanh 1 : = VM12

So sanh 2 : = VM14

So sanh 3 : = VM16

So sanh 4 : = VM18

So sanh 5 : = VM20

dx 1 : Q0.0

dv 1 : Q0.1

dd 1 : Q0.2

dxn 1: Q0.3

ddn 1: Q0.4

dx 2 : Q0.5

dv 2 : Q0.6

dd 2 : Q0.7

dxn 2 : Q2.0

ddn 2 : Q2.1

LD M2.0

JMP N1

Network 6 // Chế độ đèn vàng nhấp nháy

Call K5

Cho_phep : = M1.3 // Ô nhớ cho ‘’cho phep’’

dv 1 : = Q0.1

dv 2 : = Q0.6

```
Network 7 // Chỉnh đồng hồ
    N1 :  LD    I1.1
          EU
          Call  K6
Network 8 // Đọc đồng hồ
          Call  K7
          Gio : = VB403
Network 9 // Đọc đồng hồ đưa ra quyết định điều khiển
          LD    VB403
          LDB   = VB403,6
          LPS
          ALD
             = M4.1// gọi chu kỳ 60s
          LDB   = VB403,10
          LPS
          ALD
                = M4.2// gọi chu kỳ 46s
          LDB   = VB403,16
          LPS
          ALD
                = M4.1// gọi chu kỳ 60s
          LDB   =VB403,19
          LPS
          ALD
                = M4.2// gọi chu kỳ 46s
          LDB   = VB403,22
          LPS
          ALD



             = M4.0// gọi chu kỳ tự xử
```

```
Network 10 // Chu kỳ 60s
        LD      M4.1
        AN      M4.2
           = M1.1
Network 11 // Chu kỳ 46s
        LD      M4.2
        AN      M4.1
           = M1.2
Network 12 // Đèn vàng nhấp nháy
        LD      M4.0
        ONM4.1
           = M1.3
Network 13 // Tắt bỏ thời gian
        LD      M1.0
        O       M1.1
        O       M1.2
        O       M1.3
        EU
        Call    K2
    N2: LD      I1.0    // dx1 và dd2
         O      I1.3    // dd1 và dx2
         Call   K8
Network 14
        LD      I1.0
        Call    K8
Network 15
        END
        MEND
```

Khối K: Chương trình con

```
Network 1 // Định nghĩa điểm làm việc
        SBR    K2
        SET
        R      T38
        R      T39
        L      0
        MOVM   0   Qw4
Network 2 // Tạo chu kỳ đèn T = 46s
        SBR    K3
        LD     SM0.1
        MOVW   K460  VW10     // Chu kỳ
        MOVW   K200  VW12     // So sánh 1
        MOVW   K230  VW14     // So sánh 2
        MOVW   K240  VW16     // So sánh 3
        MOVW   K420  VW18     // So sánh 4
        MOVW   K450  VW20     // So sánh5
Network 3 // Tạo chu kỳ đèn T = 60s
        SBR    K4
        LD     SM0.1
        MOVW   K600  VW10     // Chu kỳ
        MOVW   K300  VW12     // So sanh 1
        MOVW   K330  VW14     // So sánh 2
        MOVW   K340  VW16     // So sánh 3
        MOVW   K560  VW18     // So sánh 4
        MOVW   K590  VW20     // So sánh 5
Network 4 //Tạo chu kỳ đèn nhấp nháy
        SBR    K5
        LD     #cho_phep

        MOVW   K40   VW10
```

```
        LDN   Q0.1
        LDN   Q0.6
        TON   T38,20
        LD    T38
        S     Q0.1,1 // đv1
        S     Q0.6,1 // đv2
        LD    Q0.1
        LD    Q0.6
        TON   T39,20
        LD    T39
        R     Q0.1,1
        R     Q0.6,1
Network 5 // Chỉnh đồng hồ(chuyển dữ liệu vào từng ô)
        MOVB      06,VB400
        MOVB      04,VB401
        MOVB      20,VB402
        MOVB      00,VB403
        MOVB      00,VB404
        MOVB      00,VB405
        MOVB      00,VB406
        MOVB      00,VB407
Network 6 // Đọc đồng hồ (Chuyển bảng tới đồng hồ)
        TODR      VB400
        LDB<=     6,VB403
              =M0.1
        LDB<=     VB403,10
        JMP       K4
        LDB<=     10,VB403

              =M0.1
```

```
LDB>=   VB403,16
JMP     K3
LDB<=   16,VB403
        =M0.1
LDB<=   VB403,19
JMP     K4
LDB<=   19,VB403
        =M0.1
LDB<=   VB403,22
JMP     K3
LDB<=   22,VB403
JMP     K5
MEND
```

Network 7 // Chương trình điều khiển khi tắt bỏ thời gian

```
SBR   K8
LD    SM0.1     //Đặt chế độ thời gian cho từng đèn
MOVW  K100  VW100   // Đèn xanh
MOVW  K110  VW102   // Đèn vàng
MOVW  K180  VW104   // Đèn đỏ
MOVW  K190  VW106   // Đèn vàng

LD    SM0.0
TON   T38   K190  // Tạo thời gian trễ
S     Q0.0  K1    // dx1
S     Q0.7  K1    // dd2
S     Q0.4  K1    //ddn1
S     Q2.0  K1    //dxn2

LDW>=  T38   VW100   // Đèn vàng
```

```
        R       Q0.0    K1

        R       Q0.7    K1
        S       Q0.1    K1      // dv1
        S       Q0.6    K1      // Dv2

        LDW>=   T38     VW104
        R       Q0.2    K1
        R       Q0.5    K1
        S       QO.1    K1
        S       Q0.6    K1

        LDW>=   T38     VW106   // Hết 1 chu kỳ
        R       T38     K1
        R       Q0.3    K1
        R       Q2.1    K1
        R       Q0.1    K1
        R       Q0.6    K1
Network 8 //Chế độ tự động
        SBR     K9
        LD      #cho_phep
        AN      T1
        TON     T1, #chu_ky

        LD      #cho_phep
        AW<=    T1, so_sanh1
                = # dx1
                = # ddn1

        LDW>    T1, #so_sanh1
```

```
AW<=    T1, #so_sanh2

        = # dv1
        = # ddn1
LDW>    T1, #so_sanh2
        = # dd1
        = # dxn2

LDW>    T1, #so_sanh3
AW<=    T1,#so_sanh4
        = #dx2
        = # ddn2

LDW>    T1,#so_sanh4
AW<=    T1,#so_sanh5
        = #dv2
        = #ddn2

LD      #cho_phep
AW<=    T1, #so_sanh3
LD>     T1, #so_sanh4
OLD
        = #dd2
        = #dxn2
```

KẾT LUẬN

Đồ án tìm hiểu về PLC S7 – 200 và chương trình ứng dụng của nó vào điều khiển đèn giao thông chỉ đề cập qua một số vấn đề như: nguyên lý làm việc, tổ chức bộ nhớ,các cú pháp lệnh của S7 – 200, nhiệm vụ của đèn giao thông …mà không có điều kiện để tìm hiểu cụ thể.

Phần tìm hiểu nguyên tắc hoạt đông của đèn giao thông và chương trình điều khiển của nó nói chung đã trình bầy được một số nội dung: về cấu tạo của đèn giao thông, nguyên tắc hoạt động, khai báo phần cứng, chương trình viết trên Step7 Microwin32…Qua đó ta thấy được rằng để thiết kế một hệ thống đèn giao thông là tương đối phức tạp nên trong đồ án này dù đã cố gắng nhưng vẫn chưa thật đầy đủ và còn thiếu sót rất nhiều.Ngay như khi trình bày về cấu tạo nguyên tắc của đèn cũng mới chỉ dừng lại ở trình bầy sơ lược chưa đi sâu về mạch lực, ghép nối, các thiết bị khác…

Phần trình bầy về phần mềm Micro PLC S7 – 200(công cụ chính để thực hiện bài tốn) được phân thành phần như: cấu hình cứng, cấu trúc bộ nhớ, mở rộng ngõ vào /ra, thực hiện chương trình, ngôn ngữ lập trình, microwin.Từ đó ta thấy PLC Simatic S7 – 200 có phạm vi kiến thức và sự hiểu biết tương đối lớn,trong một khoảng thời gian ngắn ngủi không thể tìm hiểu hết được.

Khi tìm hiểu cũng không có sự so sánh với các công nghệ khác, ưu điểm và nhược điểm của công nghệ còn chưa được chỉ rõ.Thực chất của quá trình là giới thiệu qua qua về nó , biến nó trở thành bước đệm để tìm hiểu về Micro PLC và ứng dụng của Micro PLC trong cuộc sống, đó là nội dung chính của đồ án tốt nghiệp này.

Trong quá trình thực hiện vì khó khăn của thiết bị và trình độ còn hạn chế nên chưa thể xây dựng một mô hình minh hoạ cụ thể, vì vậy bài tốn mới chỉ dừng lại ở việc viết trên Microwin32, điều đó đã nói nên phần nào những thiếu sót trong đồ án.Và đây cũng là sự mong muốn phát triển tiếp theo của đồ án này.

Đồ án được thực hiện trong một thời gian ngắn nên không tránh khỏi những sai sót mong các thầy cô thông cảm và giúp đỡ em hồn thiện đồ án này.

TÀI LIỆU THAM KHẢO

Tự Động Hố Với Simatic S7 – 200

- Dr- Ing NGUYỄN DỖN PHƯỚC
- Dr –Ing PHAN XUÂN MINH