Chương 1. Tổng quan về PLC và Fx5U của Mishubishi

1.1. Giới thiệu chung về bộ điều khiển logic khả trình (PLC – Programmable Logic Controller)

Bộ điều khiển logic khả trình PLC là thiết bị điện tử bán dẫn thực hiện các hàm điều khiển logic bằng chương trình thay thế cho các mạch logic kiểu rơ le (tiếp điểm và phi tiếp điểm).

Về bản chất, PLC là hệ vi xử lý được thiết kế tương tự máy tính số, với ngôn ngữ lập trình riêng gần gũi với người xử dụng, được ứng dụng trong các bài toán điều khiển logic. Hạt nhân của hệ là bộ vi xử lý thực hiện các phép tính số học và logic cùng với các thành phần cấu thành hệ như bộ nhớ, các cổng vào / ra,...

Về phạm vi ứng dụng, PLC là thiết bị đặt tại dây chuyền sản xuất, tích hợp với các thành phần của hệ thống điều khiển để thực hiện điều khiển trực tiếp công nghệ một quá trình kỹ thuật. PLC thường làm việc trong môi trườn rất khắc nghiệt (nhiệt độ cao, độ ẩm lớn, thời gian hoạt động liên tục) và gắn liền với người vận hành trực tiếp thiết bị. Vì vậy, PLC được thiết kế và chế tạo với các tiêu chuẩn đặc biệt về độ bền, tính module hóa cao, ngôn ngữ lập trình phù hợp và thân thiện với trình độ người sử dụng

Về cơ bản, PLC là thiết bị điều khiển ở hiện trường sản xuất, sát các thiết bị và cơ cấu chấp hành. Tuy nhiệ hiện nay các họ PLC hiện đại được tích hợp các tính năng xử lý thông minh, quản lý dữ liệu và mở rộng các chức năng xử lý ngắt. Ngoài chức năng điều khiển, PLC còn đóng vai trò là khâu thu nhập và xử lý dữ liệu trong các hệ SCADA và là một nút trong các hệ điều khiển phân tán (DCS). Vì vậy, với quan điểm hệ thống, PLC là thành phần cơ bản cấu thành hệ điều khiển.

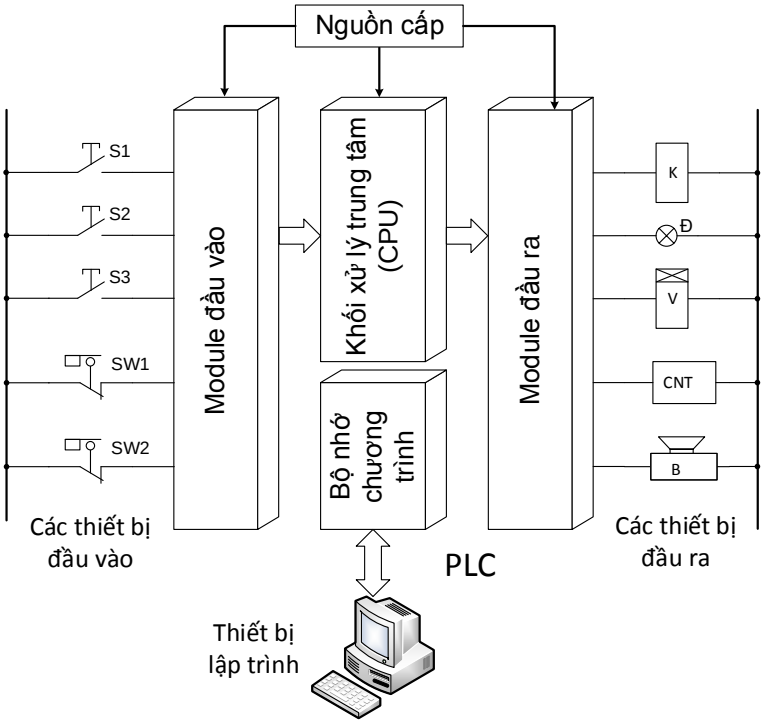
Như mọi thiết bị tính, PLC gồm phần cứng và phần mềm. Phần cứng là các thiết bị vật lý cấu thành hệ gồm: nguồn cung cấp, CPU, module vào/ra và các thiết bị phụ trợ... Các thiết bị vật lý được lắp ghép với nhau tạo thành một cấu hình vật lý của hệ thống. Phền mềm bao gồm hệ điều hành và chương trình ứng dụng. Hệ điều hành do nhà sản xuất cung cấp được cài sẵn trong bộ nhớ cảu PLC. Chương trình ứng dụng do người sử dụng lập bằng ngôn ngữ lập trình của PLC để thực hiện một thuật toán (algorithm) điều khiển xác định. Giữa phần cứng và phần mềm có mối liên hệ chặt chẽ với nhau. Một chương trình ứng dụng chỉ được thiết lập trên cơ sở một cấu hình vật lý cụ thể. Ngược lại, một hệ thống chỉ có thể thực hiện được đúng thuật toán điều khiển nếu chương trình đó được thiết kế phù hợp với cấu hình của nó.

Tính ưu việt của PLC.

Việc sử dụng PLC thay thế các bộ điều khiển logic nối dây đem lại các lợi ích căn bản.

Các bộ điều khiển logic nối dây có đặc điểm chung là các phần tử logic là các phần tử vật lý. Bộ điều khiển logic nối dây thực hiện hàm điều khiển bằng sơ đồ nối các phần tử logic bằng dây dẫn vật lý (dây dẫn điện, mạch in) đã được nối cứng. Vì vậy hệ này chỉ thực hiện một hàm điều khiển nhất định. Muốn thay đổi hàm điều khiển cần phải thay đổi cấu trúc của hệ. Đó là tính không mềm dẻo của bộ điều khiển logic nối dây. Đối với các hệ phức tạp, nhiều phần tử thì tính không mềm dẻo là một nhược điểm lớn. Tuy nhiên, ưu điểm của bộ điều khiển logic nối dây phù hợp với các hệ đơn giản, ít phần tử và công suất lớn.

Đặc điểm của PLC là các phần tử logic được định nghĩa bằng chương trình và thực hiện hàm điều khiển bằng chương trình (hình dưới).



Sơ đồ hệ điều khiển logic dùng PLC

Trong sơ đồ này các module vào và module ra là các thiết bị kết nối với các phần tử logic bên ngoài. Chương trình điều khiển được lưu giữ trong bộ nhớ. PLC thực hiện tuần tự các lệnh của chương trình để điều khiển các thiết bị tương tự như sơ đồ điều khiển kiểu nối dây..

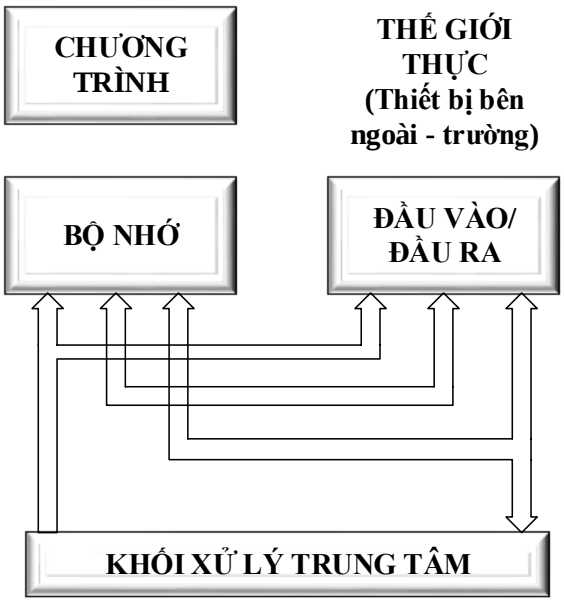
PLC đã thực hiện thay thế các mạch logic nối dây bằng các "mạch logic lập trình được". Trong các mạch logic này có thể cắt bỏ, chèn, thêm vào các phần tử một cách dễ dàng và đơn giản. Trong thực tế, việc thay đổi tham số điều khiển của chương trình, thậm chí thay đổi chương trình điều khiển thường xuyên xảy ra khi thay đổi sản phẩm, thay đổi công nghệ. Đối với hệ điều khiển logic dùng PLC, cùng một cấu trúc vật lý có thể thực hiện các hàm điều khiển khác nhau, tùy thuộc vào chương trình. Nghĩa là, có thể thay đổi hàm điều khiển mà không cần thay đổi cấu trúc của hệ. Đó là tính mềm dẻo của PLC. Tính mềm dẻo này đảm bảo PLC được sử dụng có hiệu quả cao trong các hệ phức tạp, có nhiều phần tử. Ngoài ra, ưu điểm của PLC là hoạt động tin cậy, tiêu thụ năng lượng ít, dễ dàng mở rộng hệ thống, việc chuyển giao công nghệ được nhanh và hiệu quả hơn so với các hệ logic nối dây. Hạn chế của PLC là tính tác động nhanh không cao và chỉ sử dụng tạo ra các tín hiệu điều khiển công suất nhỏ. Một ưu điểm cần nhấn mạnh khi mở rộng phạm vi ứng dụng của PLC là có thể tiến hành mô phỏng khi khảo sát và thiết kế hệ thống. PLC với các chức năng truyền thông có thể kết nối mạng với các bộ điều khiển khác, với các hệ thống máy tính và điều khiển để thực hiện các chức năng điều khiển quá trình, điều khiển phân tán, thu nhạp dữ liệu và giao diện máy- người.

1.2. Cấu trúc PLC

Thành phần cơ bản của PLC gồm có: khối xử lý trung tâm (CPU – Central Procesing Unit), các module vào/ra, nguồn cung cấp (Power Supply Unit) và thiết bị lập trình (Programming Device).

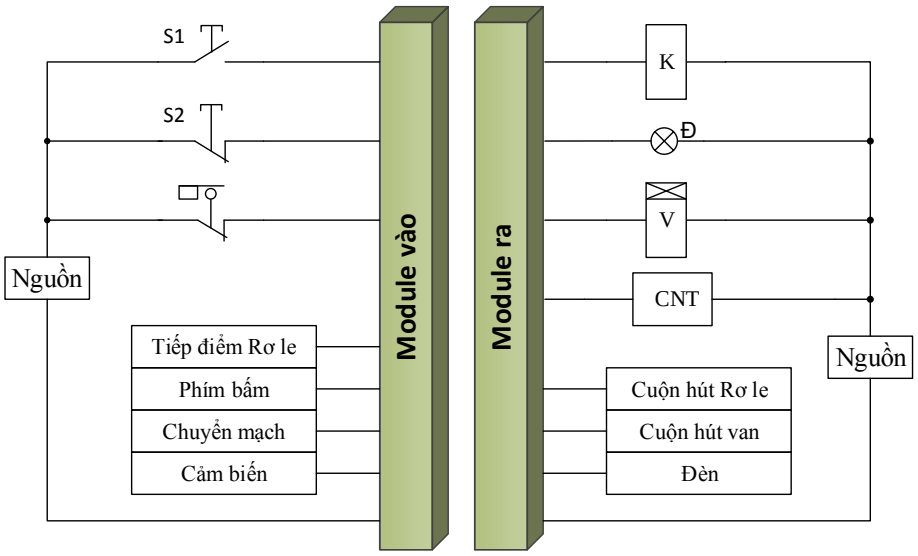
Chương trình được soạn thảo trong thiết bị lập trình và được nạp vào bộ nhớ của PLC. Các module vào/ra là các cổng phép nối PLC với thiết bị bên ngoài(gọi là thiết bị trường- Field Device). Các cổng vào/ ra có nhiệm vụ chuyển đổi thích ứng giữa các nguồn tín hiệu và PLC. Các module vào là các thiết bị nhận tín hiệu từ thiết bị vào, chuyển đổi thành dữ liệu, ví dụ: phím bấm, công tắc hành trình, cảm biến, chuyển mạch... Các module ra là thiết bị ghép nối PLC với các thiết bị ra, chuyển đổi dữ liệu thành tín hiệu điều khiển các cơ cấu chấp hành, ví

dụ: rơ le, van. Đèn... Sơ đồ nối các thiết bị vào/ra (I/O) với các module vào/ra được trình bày trên hình dưới.



Sơ đồ cấu trúc của PLC

Trong thực tế, các cổng vào/ra có hai loại: loại cố định (Fixed) và loại dạng module hóa (Modular). Loại cố định được sử dụng cho các PLC cỡ nhỏ, các cổng vào/ra gắn cố định vào khối CPU, không thay đổi được vị trí. Ưu điểm của loại này là giá thành thấp. Tuy nhiên nếu muốn mở rộng cổng vào/ra cần phải trang bị thêm khối mở rộng tương ứng. Loại module hóa được sử dụng trong đa số các trường hợp và là cấu trúc tiêu chuẩn của PLC. Các module vào/ra có thể tháo lắp, thay đổi vị trí dễ dàng trên các khe cắm (Slot) và các rãnh (Rack). Cấu trúc kiểu này (bao gồm cả các đầu nối) tạo thành bảng mạch Bus (Backplane), trên đó coe thể lắp các khối nguồn, CPU, module vào/ra, module mở rộng... và thực hiện trao đổi thông tin với nhau.



Sơ đồ module vào (a) và module ra (b)

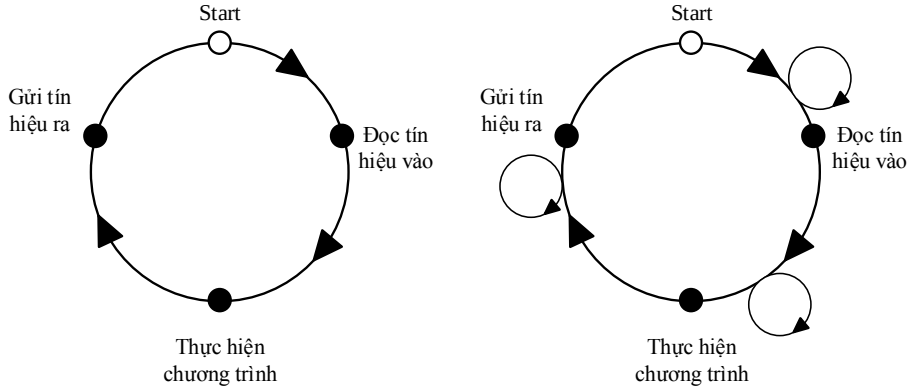
Khối nguồn cung cấp nguồn một chiều cho các khối được lắp đặt vào bảng mạch Bus. Công suất của khối nguồn được chọn tùy thuộc vào cấu hình của hệ. Trong đa số các trường hợp,

nguồn cung cấp này không phù hợp với các thiết bị trường. Vì vậy, các thiết bị trường thường được cung cấp bằng nguồn ngoài riêng.

Khối CPU là bộ não của PLC, hạt nhân là bộ vi xử lý quyết định tính chất và khả năng của PLC: tốc độ xử lý, khả năng quá trình vào/ra... CPU thực hiện chương trình trong bộ nhớ chương trình, đưa ra các quyết định và trao đổi thông tin với bên ngoài thông qua các cổng vào/ra.

1.3.Nguyên tắc hoạt động cơ bản của PLC: vòng quét chương trình

PLC hoạt động theo nguyên tắc quét vòng (Scan). Mỗi vòng quét (Scan Cycle) bao gồm ba giai đoạn cơ bản được trình bày trên hình dưới.



(a) Vòng quét cơ bản (b) Vòng quét với các vòng quét phụ

Sơ đồ vòng quét thực hiện chương trình của PLC

Ở giai đoạn thứ nhất, PLC đọc trạng thái tín hiệu ở các module vào, gửi vào vùng đầu vào để làm dữ liệu thực hiện chương trình.

Giai đoạn thứ hai là thực hiện chương trình trong bộ nhớ. Kết quả thực hiện chương trình là dữ liệu và các quyết định được lưu giữ trong bộ nhớ dùng cho vòng quét sau hay đưa module ra.

Giai đoạn thứ ba, PLC gửi dữ liệu đến vùng đầu ra và biến đổi thành tín hiệu điều khiển cơ cấu chấp hành nối với module ra. khi đó, một vòng quét được hoàn thành, vòng quét tiếp theo bắt đầu và quá trình được thực hiện liên tục không ngừng.

Quá trình đọc tín hiệu vào và gửi tín hiệu ra gọi là quá trình quét vào/ra. Quá trình thực hiện chương trình gọi là quét chương trình.

Thời gian để thực hiện một vòng quét gọi là chu kỳ quét. Chu kì quét có ảnh hưởng đến tốc độ xử lý của PLC và ảnh hưởng đến khả năng xử lý thời gian thực của PLC. Nói cách khác, việc sử dụng PLC trong các bài toán điều khiển chỉ được chấp nhận khi chu kì quét của PLC đủ nhỏ so với hằng số thời gian của hệ điều khiển. Khi đó, có thể chấp nhận xử lý đồng thời (thời gian thực) được thay thế bằng xử lý tuần tự.

Chu kỳ quét phụ thuộc vào các nhân tố sau: tốc độ của bộ vi xử lý của CPU, độ dài chương trình, số lượng các đầu vào/ra. Ngoài ra, chu kỳ quét còn phụ thuộc một số các chu kỳ quét phụ như: thời gian chuyển đổi song song – nối tiếp của hệ thống vào ra phân tán (Remote I/O), thời gian xử lý truyền thông nối tiếp, thời gian xử lý ngắt, thời gian đọc/ ghi đầu vào / ra tương tự, thời gian thực hiện các chương trình kiểm tra, cảnh báo hệ thống... Tuy nhiên, đối với một hệ cụ thể thì các nhân tố, trừ tốc độ của bộ vi xử lý, đều là cố định. Vì vậy để giảm chu kỳ quét thì phải chọn CPU có tốc độ xử lý cao.

Nguyên tắc hoạt động quét vòng của CPU hạn chế khả năng xử lý tức thời của PLC. Vì vậy, PLC chủ yếu được sử dụng trong các hệ điều khiển quá trình biến thiên chậm. Tuy nhiên,

các PLC hiện đại đã được trang bị và tăng cường các tính năng xử lý ngắt ngày càng hoàn thiện để xử lý nhanh và kịp thời.

Vấn đề xử lý vòng quét đầu tiên cần phải được quan tâm khi ứng dụng PLC. Điều này là do ở vòng quét đầu tiên, các dữ liệu đều chưa sẵn sàng, hệ đang ở quá trình khởi tạo. Đối với các hệ mà quá trình khởi tạo không ảnh hưởng đến quá trình điều khiển thì có thể bỏ qua. Ngược lại, các hệ thống khác cần lưu ý vòng quét này. Vì vậy, PLC đều cung cấp cờ trạng thái có giá trị bằng 1 ở vòng quét đầu tiên và bằng 0 ở các vòng quét khác, gọi là First Scan Flag. Người sử dụng có thể dùng cờ trạng thái này để tiến hành khởi tạo và thiết lập các điều kiện ban đầu cho hệ thống.

Sự khác biệt giữa PLC và PC

Về cấu trúc, PLC tương tự như máy tính số. Tuy nhiên, giữa PLC và máy tính số có sự khác nhau về căn bản.

Thứ nhất, PLC được thiết kế để hoạt động trong môi trường công nghiệp rất khắc nghiệt với sự thay đổi lớn về độ ẩm, nhiệt độ và nhiễu mạnh.

Thứ hai, phần cứng và phần mềm của PLC được thiết kế dễ sử dụng và phù hợp với trình độ của người vận hành trực tiếp tại dây chuyền sản xuất. Phần cứng được chế tạo ở dạng các module tiêu chuẩn dễ lắp ráp, bảo dưỡng. Chương trình của PLC được biểu diễn một cách tiêu chuẩn không chính thức ở dạng giản đồ thang (LAD) rất trực quan và dễ sử dụng. Mỗi họ PLC có hệ điều hành riêng và chỉ sử dụng một ngôn ngữ lập trình do nhà sản xuất cung cấp. Vì vậy, không thể chạy chương trình của PLC hãng này trên PLC của hang khác. Trong khi đó, máy tính có thể cài đặt nhiều hệ điều hành, có thể sử dụng nhiều ngôn ngữ lập trình. Có thể sử dụng máy tính vai trò như PLC, thậm chí có thể chạy chương trình mô phỏng PLC trên máy tính. máy tính được cài đặt phần mềm lập trình trở thành thiết bị lập trình cho PLC và điều khiển PLC. Máy tính có thể sử dụng như thiết bị giao diện người máy trong các hệ điều hành mà PLC vừa là các bộ điều khiển, vừa là thiết bị thu nhập dữ liệu.

Thứ ba, máy tính là thiết bị tính toán phức tạp, có chức năng đa nhiệm (Multitask). Bộ nhớ của máy tính có thể chứa đồng thời nhiều chương trình. Trong khi đó, PLC chỉ thực hiện một chương trình được lưu trữ trong bộ nhớ RAM.

Thứ tư, PLC hoạt động theo nguyên tắc quét vòng, máy tính hoạt động theo nguyên tắc xử lý ngắt.

Các PLC hiện đại được hoàn thiện về tốc độ tính toán, mở rộng bộ nhớ, tăng cường trang bị các chức năng tính toán và xử lý ngắt để thu hẹp khoảng cách so với máy tính. Vì vậy, các PLC hiện đại ngoài chức năng cơ bản là điều khiển logic, nó còn là một trạm tính toán đóng vai trò như các bộ điều khiển quá trình, điều khiển vị trí và xử lý thông tin.

1.4. Các chủng loại PLC và ứng dụng

PLC có rất nhiều chủng loại và do rất nhiều nhà sản xuất cung cấp. Một số nhà sản xuất và tích hợp hệ thống sử dụng PLC do chính họ chế tạo.Nó là một thành phần cấu thành hệ thống và được sử dụng trong phạm vi hẹp. Một số nhà sản xuất cung cấp PLC như là sản phẩm đa dụng cho người thiết kế và tích hợp hệ thống. Nhà sản xuất cung cấp thiết bị, phần mềm, hỗ trợ kỹ thuật và đào tạo để người sử dụng có điều kiện ứng dụng các sản phẩm này vào các hệ thống của mình. Có một số hãng sản xuất điển hình là: SIEMENS(Đức), ALLEN-BRADLEY, GE-FUNUC(Mỹ), MITSUBISHI, TOSHIBA(Nhật bản)....

Do PLC được sử dụng rất rộng rãi từ các bài toán đơn giản đến các bài toán phức tạp, nên PLC được chế tạo dưới nhiều loại khác nhau phù hợp với yêu cầu của thực tế. Việc phân loại PLC dựa trên cơ sở khả năng (tốc độ xử lý, dung lượng bộ nhớ, số lượng đầu vào/ ra) được chia thành các loại chính sau: loại nhỏ, loại vừa và loại lớn.

PLC loại nhỏ có nhiều tên gọi khác nhau tùy thuộc các hãng chế tạo (small, micro), có dung lượng bố nhớ dưới 2KB, quản lý số điểm vào/ra dưới 128 và được sử dụng trong các ứng dụng đơn giản, yêu cầu ít điểm ra/vào.

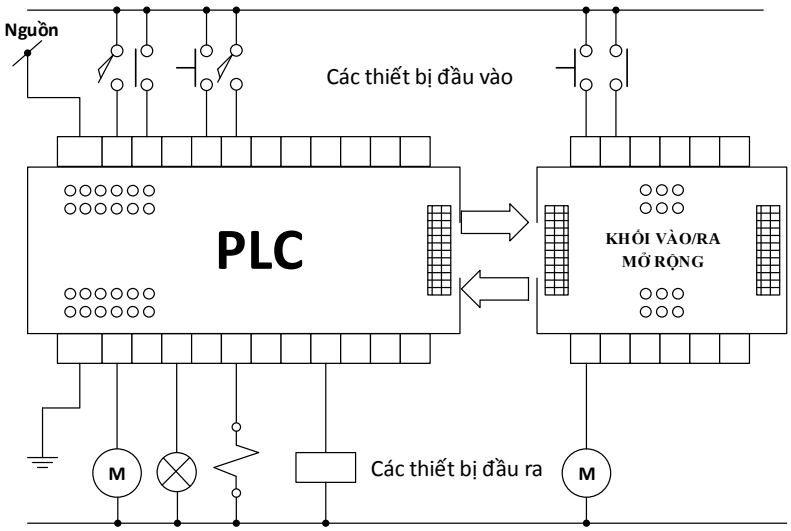
PLC cỡ vừa (Medium) có bộ nhớ đến 32KB, và quản lý số điểm vào/ra đến 2048. cấu hình của hệ có thể sử dụng các module vào/ra đặc biệt để thực hiện các chức năng điều khiển quá trình và xử lý thông tin.

PLC cỡ lớn (Large) là thiết bị phức tạp nhất có thể quản lý đến 2MB bộ nhớ và 16.000 điểm vào ra. PLC loại này có ứng dụng không hạn chế từ điều khiển một quá trình công nghệ đến điều khiển một phân xưởng, một nhà máy.

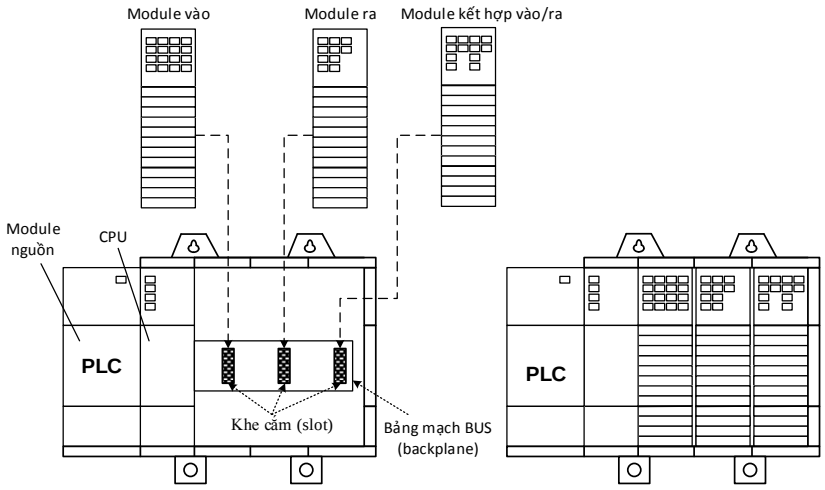
Phương pháp phân loại PLC ở trên kết hợp với kiểu dáng chế tạo sẽ đưa ra các chủng loại PLC sau đây.

Các PLC cỡ nhỏ thường được chế tạo ở dạng cố định (Compact, Fixed). Với loại này, nguồn cung cấp,CPU và một số điểm vào/ra được chế tạo trên cùng một (Onboard). Ưu điểm cơ bản của PLC loại này là giá thành thấp, nhỏ, gọn và thích hợp các ứng dụng nhỏ. Số các điểm vào/ra trên PLC theo tỷ lệ 3:2, ví dụ, loại 10 điểm (6 vào, 4 ra), loại 20 điểm (12 vào, 8 ra), loại 30 điểm (12 vào, 18 ra) và loại 48 điểm, 60 điểm.... Khi cần thiết có thể sử dụng các module vào/ra mở rộng. Tuy nhiên với PLC loại này ít khi sử dụng cách mở rộng như vậy. Nhược điểm chính là tính mềm dẻo không cao, tốc độ xử lý chậm, bộ nhớ nhỏ, hạn chế số điểm vào/ra. Sơ đồ tổ chức PLC loại nhỏ, dạng cố định được trình bày trên hình dưới.

Các PLC loại vừa và loại lớn được chế tạo ở dạng các module riêng biệt, có thể tháo, lắp dễ dàng (Modular). Các module cơ bản là: nguồn, CPU, vào/ra... Đây là cấu trúc tiêu chuẩn của PLC, đảm bảo cho PLC được sử dụng một cách mềm dẻo và người sử dụng có nhiều lựa chọn cho cấu hình của mình. Các module được lắp vào các khe cắm (Slot) trên bảng mạch Bus (Bus Module, Backplane).



PLC loại nhỏ, dạng cố định.



PLC loại vừa và lớn, dạng modul.

Ứng dụng của PLC được chia làm 3 nhóm chính là: Đơn nhiệm (Single), đa nhiệm (Multitask) và quản lý điều khiển (Control Manegment).

Ứng dụng đơn nhiệm là chỉ sử dụng một PLC duy nhất để điều khiển một quá trình kĩ thuật. Đó là một khối điều khiển độc lập, không có trao đổi thông tin với máy tính hoặc các PLC khác. Cấu hình của hệ có thể dùng PLC các loại nhỏ, vừa hoặc lớn.

Ứng dụng đa nhiệm thường sử dụng PLC cỡ vừa để điều khiển một công đoạn của dây truyền sản xuất hoặc để điều khiển một vài quá trình kỹ thuật với số lượng điểm vào/ra thích hợp. Mỗi PLC có thể thành một nút trong hệ điều khiển phức tạp (ví dụ: hệ điều khiển DCS). Khi đó, yêu cầu có sự trao đổi dữ liệu, thông tin giữa các PLC với nhau, hoặc giữa PLC và các thiết bị khác (như máy tính, trạm kĩ thuật...). Việc trao đổi dữ liệu, thông tin nhờ truyền thông mạng theo chuẩn công nghiệp.

Ứng dụng quản lý điều khiển thường sử dụng các PLC cỡ lớn, với cấu hình của hệ là một mạng LAN điều khiển thống nhất, có sự trao đổi dữ liệu và thông tin giữa các thành phần của hệ. Trong đó PLC đóng vai trò là bộ điều khiển, đồng thời quản lý hoạt động toàn bộ hệ là trạm chủ (Master). Các PLC khác là các bộ điều khiển và đồng thời là thiết bị thu nhập dữ liệu phục vụ cho công tác quản lý và theo dõi hệ thống gọi trạm tớ (Slave).

1.5. Các kiểu chương trình

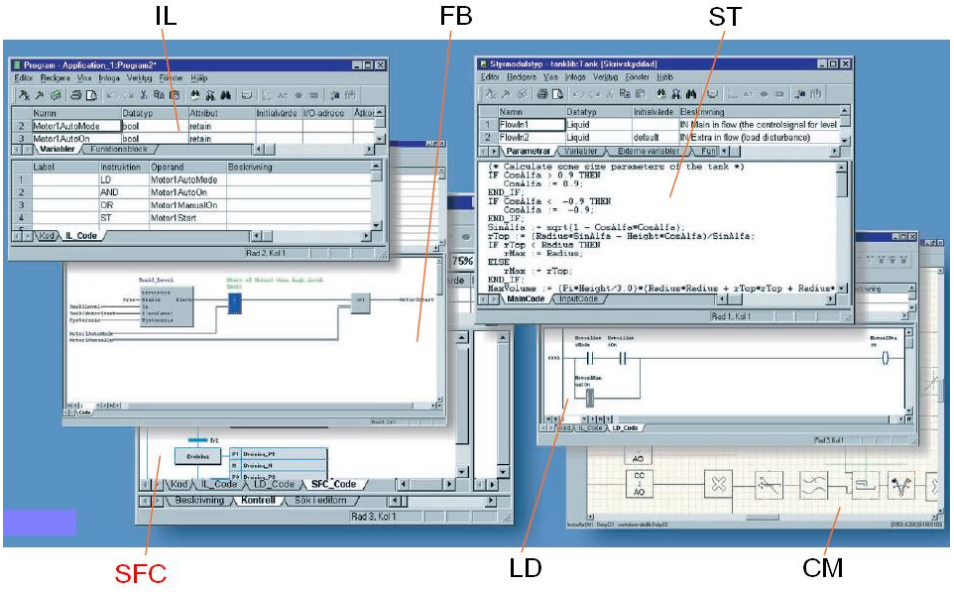
Hiện nay có một số kiểu chương trình được qui chuẩn hóa quốc tế và gọi là tiêu chuẩn lập trình IEC-61131. Đó là:

+ Chương trình kiểu danh sách lệnh - Instruction List (IL), đây là ngôn ngữ bậc thấp thể hiện dưới các câu lệnh và chương trình là tập hợp một dãy lệnh liên tiếp giống với Assembler. Khi thể hiện ở dạng đồ họa có hình thức giống như vẽ mạch điện kinh điển và gọi là Ladder program.

+ Chương trình kiểu cấu trúc – Structured Text (ST), đây là ngôn ngữ bậc cao như C, nên thực hiện các phép gán giá trị các biến, gọi hàm và khối hàm, các biểu thức, các câu lệnh điều kiện và các vòng lặp.

+ Chương trình kiểu khối hàm – Function Block (FB): là một ngôn ngữ đồ họa, diễn tả quá trình theo dòng tín hiệu giữa các phần tử, khá tương tự với sơ đồ mạch điện tử logic.

Function Block (FB) : là một ngôn ngữ đồ hoạ, diễn tả quá trình trên phương diện dòng tín hiệu giữa các phần tử; tương tự sơ đồ của các mạch điện tử.



Các kiểu chương trình PLC

+ Chương trình kiểu lưu đồ trạng thái tuần tự - Sequential Function Charts (SFC). SFC được phát triển từ ngôn ngữ GRAFCET (là một công cụ đồ họa miêu tả chuỗi hành động). SFC là m ột công cụ rất mạnh trong miêu tả cấu trúc hệ thống điều khiển tuần tự.

+ Chương trình kiểu khối điều khiển – Control Module (CM), là dạng lưu đồ điều khiển ở mức độ rất cao, trong đó không chỉ thể hiện logic điều khiển và các phép toán mà còn thể hiện cả các dữ liệu,, truyền thông…

Mỗi kiểu chương trình có ưu-nhược điểm riêng, để kết hợp được các ưu điểm của từng loại vào một chương trình, hiện này các hãng đã thiết kế để cho phép trong một chương trình có thể lập trình đồng thời theo nhiều kiểu. Thông thường lấy chương trình Ladder là cốt, trong từng đoạn có thể chuyển sang dùng FB, ST…

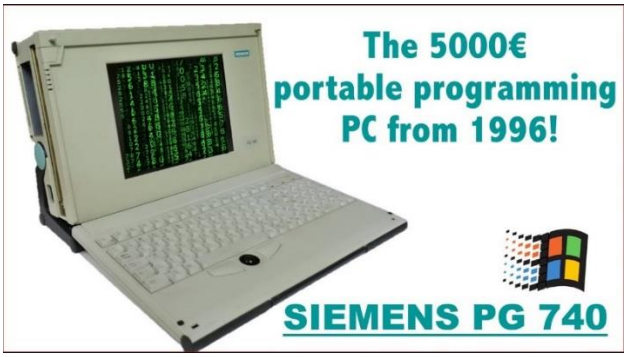
1.6. Thiết bị và công cụ lập trình

Để đưa chương trình vào PLC cần có công cụ lập trình tương ứng. Thiết bị lập trình được sử dụng để soạn thảo chương trình, nạp vào bộ nhớ của PLC. Ngoài ra, thiết bị lập trình còn được sử dụng để theo dõi, gỡ rối, thay đổi lệnh, lưu giữ chương trình và thực hiện các thao tác điều khiển PLC. Thiết bị lập trình có các loại sau:

+ Máy lập trình cầm tay do từng hãng chế tạo để lập trình cho riêng PLC của bản hãng và có tên gọi do hãng đặt như “Programmable console”, HandHeld Programmer… Thiết bị nhỏ gọn gồm cụm phím bấm với một màn hình nhỏ trên đó chỉ hiển thị các ký tự hạn chế, số lượng dòng trên màn hình cũng ít (dưới 6 dòng). Do vậy chỉ có thể lập trình kiểu danh sách lệnh STL. Do khả năng hạn chế nên hiện nay rất ít dùng.

+ Máy lập trình chuyên dụng có hình dạng giống với máy tính cũng do hãng chế tạo cho PLC của mình. Loại này lập trình được nhiều kiểu do màn hình lớn như máy tính, cho phép kiểm tra, theo dõi đầy đủ và dễ dàng hoạt động của PLC, có thể can thiệp sâu vào cấu trúc hệ thống. Điểm hạn chế là máy này chỉ áp dụng được cho PLC của một hãng.

+ Lập trình trên máy tính PC thông thường có cài đặt phần mềm lập trình do hãng chế tạo PLC thiết kế và giữ bản quyền. Lập trình được nhiều kiểu chương trình tùy theo phần mềm, cũng cho phép người sử dụng theo rõi đầy đủ cả quá trình lập trình (Off-line) và quan sát hoạt động của PLC (chế độ On-Line). Trên một máy tính PC có thể cài đặt nhiều phần mềm lập trình của các hãng khác nhau để làm việc được với các PLC của nhiều hãng. Do ưu điểm này nên phương pháp này được sử dụng rộng rãi trên khắp thế giới.



1.7. Các đầu vào/ra số

Module vào ra số.

Module vào ra số (digital I/O module) là loại module phổ biến nhất, là thành phần cấu hình cơ bản và phong phú nhất của plc. Module vào ra rời rạc là cổng giao tiếp với các thiết bị vào vào và thiết bị ra kiểu on/off.

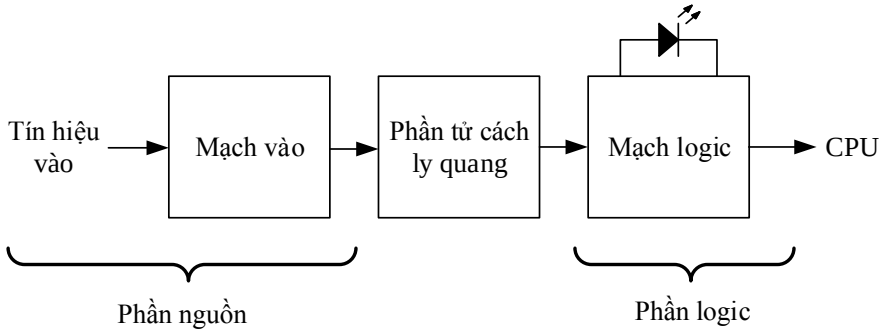
1 Module vào rời rạc.

Module vào rời rạc thực hiện các nhiệm vụ nhận tín hiệu từ các thiết bị vào, biến đổi thành các tín hiệu số gửi đến CPU. Modul rời rạc là thiết bị giao tiếp giữa PLC và thiết bị vào.

Các thiết bị vào ra rời rạc là các tín hiệu logic như chuyển mạch (selector switch), phím nhấn (push button), công tắc hành trình (limit switch), các tiếp điểm là đầu ra của các bộ điều khiển (contact type), các loại cảm biến tiệm cận (proximity sensor), cảm biến quang (photo sensor)...

Mỗi thiết bị vào nối với module vào tại 1 điểm có vị trị xác định gọi là điểm đầu vào (Input point). Mỗi điểm đầu vào tương ứng với 1 địa chỉ của Bit dữ liệu trong vùng đầu vào. Giá trị của bit dữ liệu phản ánh trạng thái của tín hiệu vào. Nếu tín hiệu vào ở mức cao (ví dụ, đối với tín hiệu xoay chiều là 24 VDC, đối với tín hiệu xoay chiều là 110 VAC hoặc 220 VAC thì Bit tương ứng có giá trị là 1. Nếu tín hiệu vào ở mức thấp (0V) thì bit tương ứng có giá trị bằng 0.

Trên các module vào đều có LED chỉ thị trạng thái của tín hiệu. Mạch điện của khối CPU và mạch ngoài được cách ly bằng phần tử quang (optocouple). Sơ đồ khối chức năng của module đầu vào được trình bày như hình dưới đây.

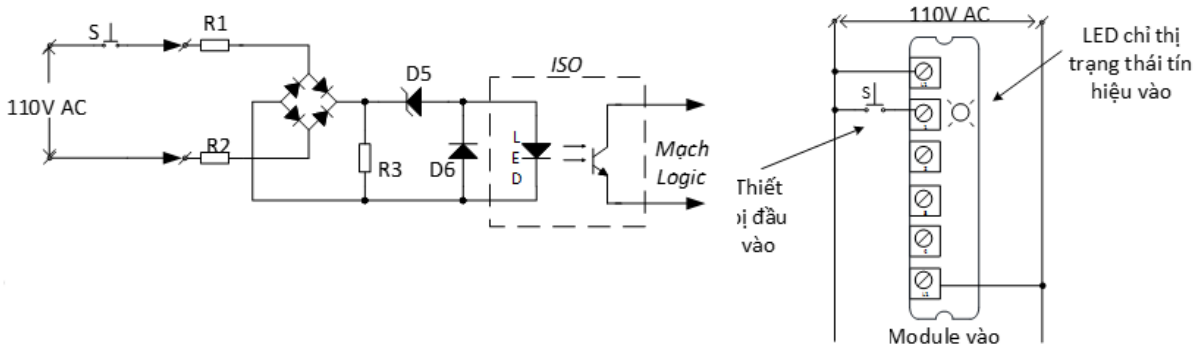


Sơ đồ khối chức năng của module vào rời rạc.

Sơ đồ chia làm 2 phần: phần nguồn cung cấp cho các thiết bị vào và phần và phần tạo ra tín hiệu logic phù hợp với PLC. Vì vậy tín hiệu vào có thể là tín hiệu xoay chiều,một chiều với các mức điện áp khác nhau. Mạch vào là các mạch biến đổi tín hiệu, mạch lọc nhiễu. Phần tử cách ly thường dùng là các phần tử các ly quang hoặc biến áp xung. Mạch logic tạo ra tín hiệu logic phù hợp với CPU và LED chỉ thị trạng thái của tín hiệu vào.

a). Đầu vào xoay chiều.

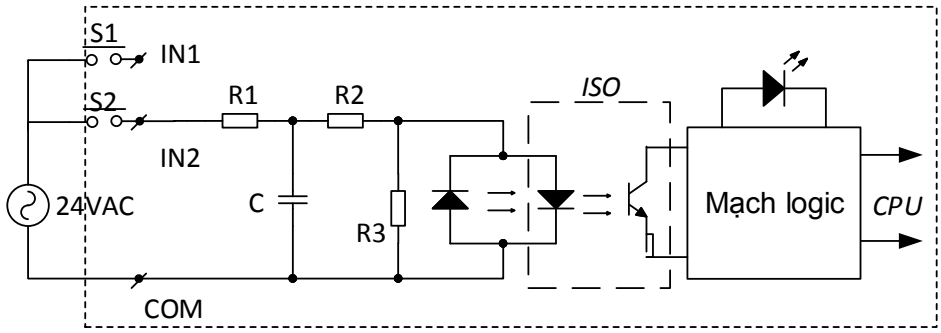
Sơ đồ nguyên lý một đầu vào của module vào xoay chiều với điện áp 110 VAC/ 220VAC như hình dưới.



Sơ đồ nguyên lý một đầu vào của module vào xoay chiều và cách đấu nối

Tín hiệu vào nhận được từ nguồn xoay chiều L1-L2 qua phím S được biến đổi thành tín hiệu 1 chiều nhờ bộ chỉnh lưu (D1-D4). Điện trở R1,R2 giúp hạn chế dòng vào, điện trở R3 là là điện trở tải của bộ chỉnh lưu, diot ổn áp D5 xách định ngưỡng nhỏ nhất của tín hiệu vào. Phần tử cách ly quang ISO chuyển đổi tín hiệu từ nguồn thành tín hiệu logic gửi đến CPU.

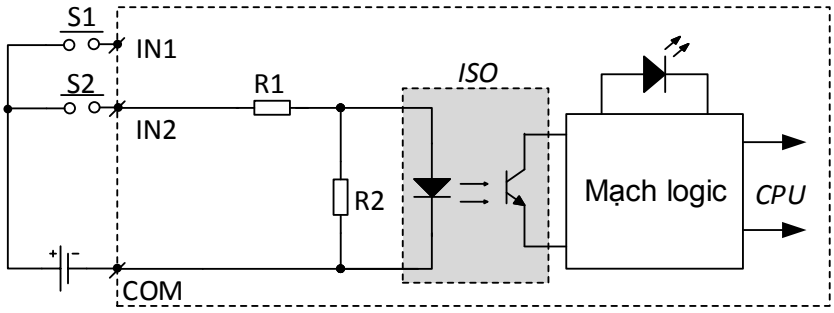
Trường hợp điện áp xoay chiều đầu vào nhỏ thường không sử dụng cầu chỉnh lưu. Sơ đồ này không sử dụng mạch chỉnh lưu mà dùng phần tử cách ly quang gồm 2 LED mắc song song ngược. Đầu vào sử dụng mạch lọc thông thấp RC. Tín hiệu từ LED chỉ thị trạng thái đầu vào PLC.



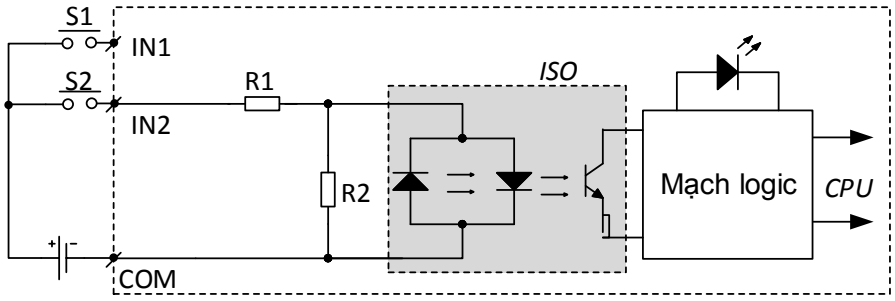
Sơ đồ nguyên lý và sơ đồ nối một đầu vào của module vào xoay chiều, điện áp 24V AC

b). Đầu vào một chiều.

Khi điện áp vào là một chiều thì module vào có cấu trúc đơn giản hơn, để thuận tiện cho người dùng có loại cho phép sử dụng tùy ý dấu của điện áp vào.



a) Sử dụng nguồn một chiều cố định.



b) Sử dụng nguồn 1 chiều tùy ý

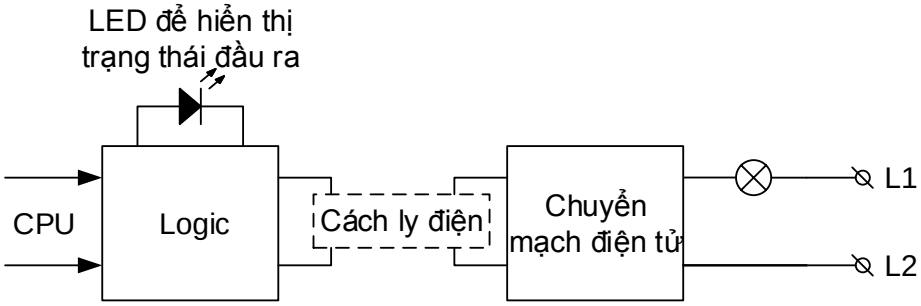
Sơ đồ nguyên lý một đầu vào của module vào một chiều điện áp 24V DC.

Việc phân loại module vào rời rạc dựa trên số lượng các điểm vào trên module và kiểu tín hiệu vào. Trên cơ sở số lượng đầu vào trên module có các loại 8 điểm 16 điểm 32 điểm….

Trên cơ sở kiểu tín hiệu vào có các loại 5VDC, 24V AC/DC, 48V AC/DC, 110V AC/DC, 220V AC/DC.

2. Nguyên lý đầu ra số

Các module ra rời rạc thực hiện các nhiệm vụ nhận dữ liệu từ CPU, biến đổi thành tín hiệu phù hợp điều khiển các thiết bị ra (cơ cấu chấp hành). Sơ đồ nối thiết bị ra với module ra rời rạc được trình bày như hình dưới.

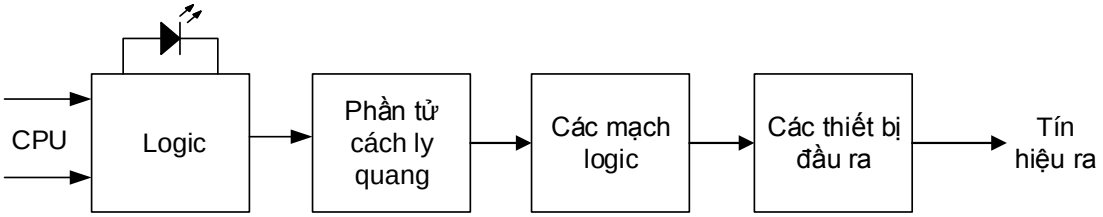


Sơ đồ nối thiết bị ra với module ra rời rạc

Các thiết bị ra là các thiết bị logic như: đèn, rơ le, contactor, van,… Đó là các thiết bị có hai trạng thái ON/OFF.

Mỗi thiết bị ra logic nối với module ra tại một điểm có vị trí xác định gọi là điểm đầu ra (Output Point). Mỗi điểm đầu ra tương ứng với một địa chỉ của Bit dữ liệu trong vùng đầu ra. Giá trị của Bit dữ liệu quyết định trạng thái của thiết bị ra. Nếu giá trị của Bit bằng 1 thì trạng thái của thiết bị ra là tích cực (Active). Nếu giá trị của Bit bằng 0, thì trạng thái của thiết bị ra là không tích cực (Inactive).

Trên các module ra đều có LED chỉ thị tín hiệu ra. Mạch điện của khối CPU và mạch ngoài được cách ly với nhau. Sơ đồ khối chức năng của moule ra rời rạc được trình bày ở dưới.

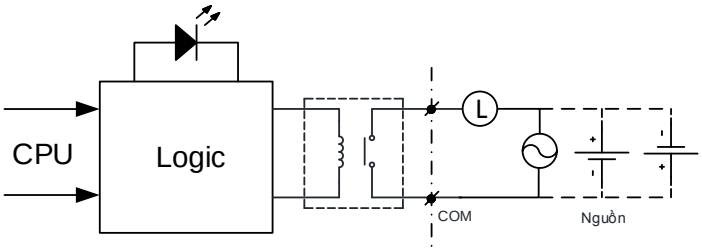


Sơ đồ khối chức năng của các module ra rời rạc

Sơ đồ được chia làm 2 phần: phần logic và phần nguồn. Các mạch logic xác định trạng thái đầu ra phụ thuộc tín hiệu nhận CPU. Trạng thái tín hiệu đầu ra được chỉ thị bằng LED. Phần tử đầu ra có hai trạng thái ON/OFF tương ứng với tín hiệu từ mạch logic

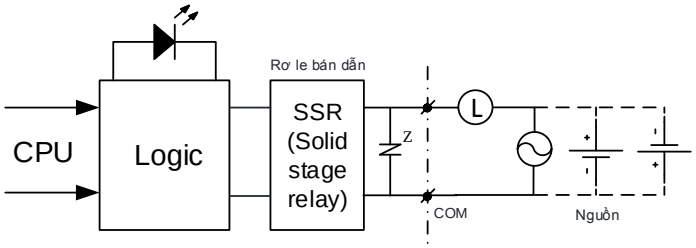
a). Đầu ra rơle

Hình dưới trình bày sơ đồ một đầu ra kiểu tiếp điểm. Phần tử L là tải. Đầu ra kiểu tiếp điểm rơ le điện từ sử dụng nguồn cung cấp một chiều hoặc xoay chiều.



(a) Đầu ra kiểu tiếp điểm rơ le điện từ

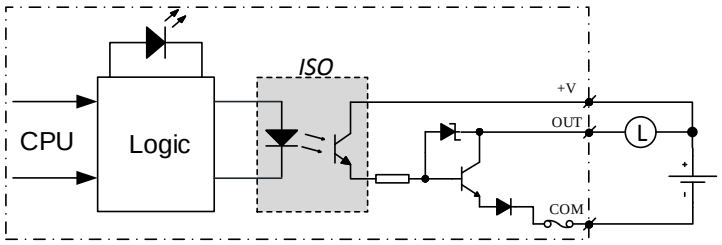
Đầu ra kiểu rơle bán dẫn sử dụng nguồn xoay chiều



(b) Đầu ra kiểu rơ le bán dẫn (SSR – Solid State Relay)

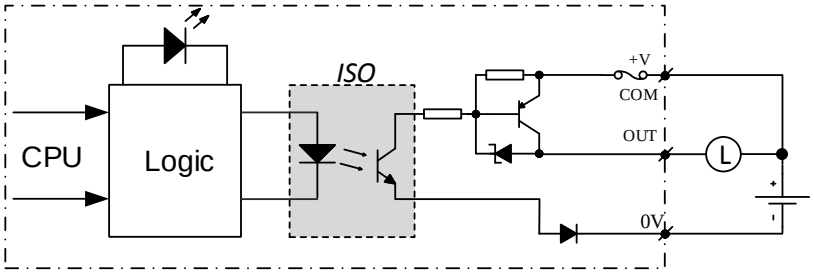
b). Đầu ra kiểu transistor

Mạch đầu ra có hai kiểu: NPN và PNP. Mạch đầu ra kiểu NPN có đặc điểm: điểm COM là 0V, tải L nối giữa đầu ra (Out) và cực dương của nguồn (+V)



(a) Mạch kiểu NPN

Mạch đầu ra kiểu PNP có đặc điểm: điểm COM là dương nguồn +V, tải L nối giữa đầu ra (OUT) và 0V.

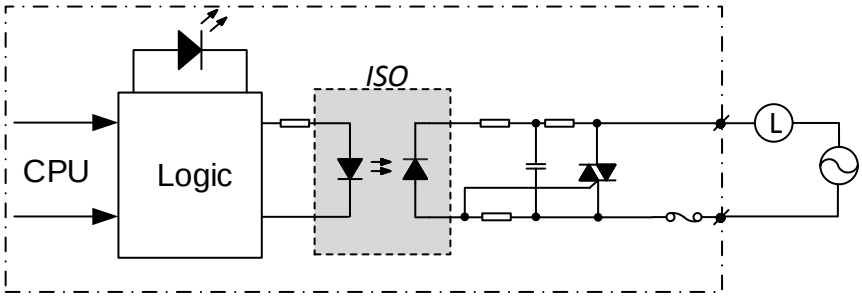


(b) Mạch kiểu PNP

Các đầu ra kiểu tín hiệu điện áp đều sử dụng mạch collector hở, cầu chì bảo vệ quá dòng F và chỉ sử dụng nguồn cung cấp một chiều

c). Đầu ra kiểu Triac

Hình trình bày sơ đồ đầu ra kiểu xoay chiều. Phần tử đóng cắt là Triac. Đầu ra xoay chiều này cho phép dòng qua tải lớn và điện áp nguồn 110VAC hoặc 220VAC



Sơ đồ đầu ra kiểu xoay chiều

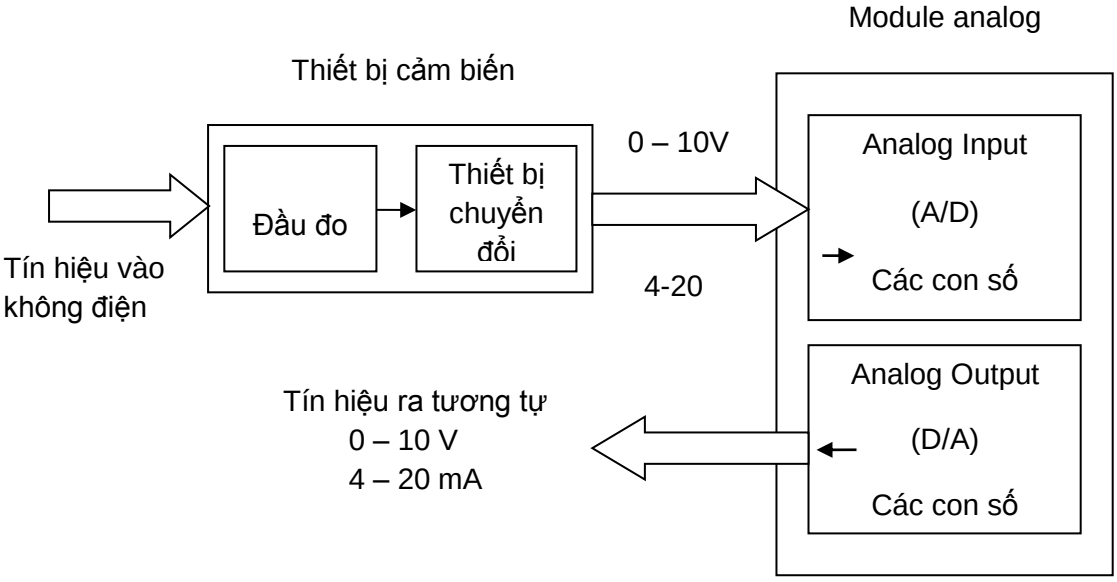
2.1. Các đầu vào/ra tương tự

Trên thực tế PLC được ứng dụng trong các bộ điều khiển quá trình với nguồn tín hiệu và đối tượng điều khiển là các thiết bị tương tự. Các module vào/ra tương tự là các mạch ghép nối PLC với các thiết bị này.

1. Đầu vào tương tự

Thực chất nó là một bộ biến đổi tương tự - số (A/D). Nó chuyển tín hiệu tương tự ở đầu vào thành các giá trị dưới dạng số ở đầu ra. Dùng để kết nối các thiết bị đo với bộ điều khiển: chẳng hạn như đo nhiệt độ, độ ẩm, lưu lượng, áp suất, lưu lượng, khối lượng….

Các module vào tương tự nhận tín hiệu tương tự (dòng điện, điện áp) từ thiết bị trường, từ các bộ chuyển đổi (Transducer), từ các bộ truyền tín hiệu (Transmitter) biến đổi thành tín hiệu số nhờ bộ biến đổi ADC.



Tín hiệu đầu vào analog theo chuẩn điện áp và chuẩn dòng điện.

Tín hiệu điện áp có thể là đơn cực và lưỡng cực: Tín hiệu đơn cực có các dải điện áp sau: (0V÷ +5V), (0V ÷ +10V), (1V ÷ +5V). Tín hiệu lưỡng cực có dải điện áp như sau:

(-5V ÷ +5V), (-10V ÷ +10V).

Tín hiệu dòng điện có dải: (0mA ÷ 20mA) và (4mA ÷ 20mA).

Trên module vào tương tự có thể có 2, 4, 8 đầu vào gọi là các kênh. Kiểu và dải tín hiệu (V/I) ở mỗi kênh được chọn nhờ các chuyển mạch chọn trên module. Nguồn cung cấp cho các module vào tương tự thông qua Bus nguồn của hệ thống. Cũng có một số họ PLC yêu cầu nguồn cung cấp riêng từ ngoài cho các module vào tương tự.

CPU nhận tín hiệu số từ các kênh của module vào tương tự nhờ lệnh đọc riêng và cất vào một vùng nhớ riêng do hệ thống quy định. Mỗi họ PLC có cách tổ chức riêng.

Các tham số đặc trưng cho module vào tương tự là:

- Số kênh
- Kiểu và dải tín hiệu vào
- Trở khánh vào
- Độ phân dải: 8 bit, 10 bit, 12 bit, …
- Tốc độ biến đổi
- Hệ số nén tín hiệu đồng pha

Ngoài các module vào tương tự với tín hiệu điện áp và dòng điện chuẩn như ở trên, còn có các module tương tự, mà tín hiệu vào nhận trực tiếp từ các sensor. Ví dụ, các sensor nhiệt độ (cặp nhiệt ngẫu, điện trở nhiệt Pt 100…), sensor áp suất… Mạch vào của các module này là các bộ khuếch đại tín hiệu nhỏ. Vì vậy, dây nối sensor với các đầu vào phải được bọc kim để chống nhiễu.

b). Đầu ra tương tự

Analog output cũng là một phần của module analog. Thực chất nó là một bộ biến đổi số - tương tự (D/A). Nó chuyển tín hiệu số ở đầu vào thành tín hiệu tương tự ở đầu ra. Dùng để điều khiển các thiết bị với dải đo tương tự. Chẳng hạn như điều khiển Van mở với góc từ 0-100%, hay điều khiển tốc độ biến tần 0-50Hz.

Các module ra tương tự nhận tín hiệu số từ CPU, biến đổi thành tín hiệu điện áp và dòng điện để điều khiển các thiết bị trường. Thành phần cơ bản của module ra tương tự là bộ DAC. Tín hiệu ra tương tự được chuẩn hóa theo các thiết bị trường. Ví dụ, tín hiệu ra điện áp có các dải: (0V÷ +5V), (0V ÷ +10V), (1V ÷ +5V). tín hiệu dòng điện có các dải: (0mA ÷ 20mA) và (4mA ÷ 20mA).

Trên module ra tương tự có thể có 2, 4, 8 đầu ra gọi là các kênh. Kiểu và dải tín hiệu (V/I) ở mỗi kênh được chọn nhờ các chuyển mạch chọn trên module.

Các tham số đặc trưng cho module ra tương tự là:

- Số kênh
- Kiểu và dải tín hiệu ra
- Trở khánh ra
- Độ phân dải: 8 Bit, 10 Bit, 12 Bit…..
- Tốc độ biến đổi

Trong thực tế, ngoài các module vào/ra tương tự riêng biệt, nhà sản xuất còn cung cấp các module vào/ra tương tự kiểu hỗn hợp. Ví dụ, module 2 kênh vào – 2 kênh ra tương tự, module 4 kênh vào – 1 kênh ra tương tự.

2.2. Dữ liệu và các kiểu dữ liệu trong PLC với Fx5U

Các kiểu dữ liệu và độ dài dữ liệu của FX5U

1. *Các kiểu dữ liệu*

Bảng danh sách các kiểu dữ liệu có thể sử dụng để chỉ thị trong modun CPU

Dữ liệu	Phân loại
Bit	Bit dữ liệu
16-bit (dữ liệu kí tự)	16-bit dạng nhị phân có dấu
	16-bit dạng nhị phân không dấu
32-bit (dữ liệu kí tự kép)	32-bit dạng nhị phân có dấu
	32-bit dạng nhị phân không dấu
Số thực (dữ liệu điểm nổi)	Số thực đơn chính xác
BCD	4 chữ số BCD
	8 chữ số BCD
String	string

2. Độ dài dữ liệu

a). Kiểu dữ liệu 16 bit (word data)

a) Kích thước dữ liệu và dải dữ liệu

Dữ liệu 16 bit bao gồm dữ liệu 16 bit có dấu và không dấu

Trong dữ liệu 16 bit có dấu, 1 số âm được biểu diễn thành phần bù của 2

Tên dữ liệu	Kích thước	Dải giá trị	
		Hệ thập phân	Hệ hexa
16 bit có dấu	16 bit (1 word)	Từ -32768 đến 32767	Từ 0000H đến FFFFH
16 bit không dấu		Từ 0 đến 65535	

Ký hiệu K*số bit thiết bị bắt đầu

* là số lượng chữ số (chỉ định số trong phạm vi từ 1 đến 4)

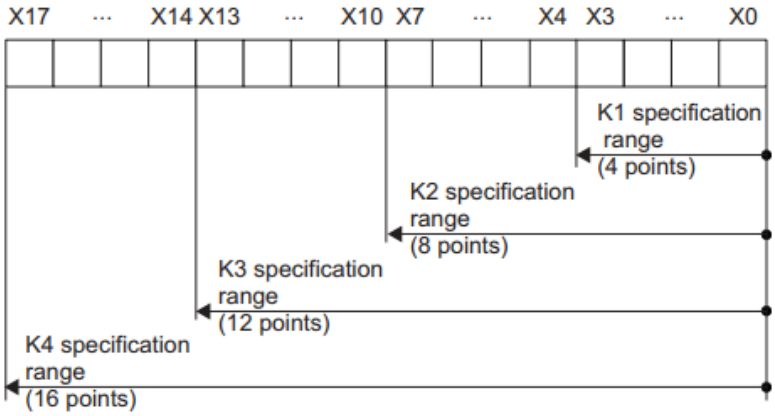
b) Phạm vi đặc điểm kỹ thuật nhóm 4 bit

Đặc điểm KT nhóm 4bit	Hệ thập phân	Hệ hexa
K1	0 đến 15	0H to FH
K2	0 đến 225	00H to FFH
K3	0 đến 4095	000H to FFFH
K4	Không dấu: 0 đến 65535	0000H to FFFFH
	Có dấu: -32768 đến 32767	

K là hằng số chỉ số nhóm 4 bit liên tiếp

Ví dụ:

- K1X0→4 points from X0 to X3
- K2X0→8 points from X0 to X7
- K3X0→12 points from X0 to X13
- K4X0→16 points from X0 to X17



b). Kiểu dữ liệu 32 bit (double – word data)

b) Kích thước dữ liệu và dải dữ liệu

Dữ liệu 32 bit bao gồm dữ liệu 32bit có dấu và không dấu

Trong dữ liệu 32bit có dấu, 1 số âm được biểu diễn thành phần bù của 2

Tên dữ liệu	Kích thước	Dải giá trị	
		Hệ thập phân	Hệ hexa
32 bit có dấu	32 bit (2 word)	Từ -2147483648 đến 2147483647	Từ 00000000H đến FFFFFFFFH
32 bit không dấu		Từ 0 đến 4294967295	

Xử lý dữ liệu 32 bit với bit thiết bị: 1 bit thiết bị có thể được xử lý như 32 bit bằng cách biểu diễn 4bit

Ký hiệu K*số bit thiết bị bắt đầu

* là số lượng chữ số (chỉ định số trong phạm vi từ 1 đến 8)

b) Phạm vi đặc điểm kỹ thuật nhóm 4 bit

Đặc điểm KT nhóm 4 bit	Hệ thập phân	Hệ hexa
K1	0 đến 15	0H to FH
K2	0 đến 225	00H to FFH
K3	0 đến 4095	000H to FFFH
K4	0 đến 65535	0000H to FFFFH
K5	0 đến 1048575	00000H to FFFFFH

K6	0 đến 16777215	000000H to FFFFFFH
K7	0 to 268435455	0000000H to FFFFFFFH
K8	32 bit có dấu: -2147483648 to 2147483647 32 bit không dấu: 0 to 4294967295	00000000H to FFFFFFFFH

K là hằng số chỉ số nhóm 4 bit liên tiếp

c). Kiểu BCD

a) BCD 4 chữ số

Dữ liệu 16 bit được phân chia bằng 4 chữ số và mỗi chữ số đươc định rõ từ 0 đến 9

b) BCD 8 chữ số

Dữ liệu 32 bit được phân chia bằng 8 chữ số và mỗi chữ số đươc định rõ từ 0 đến 9

d). Kiểu dữ liệu số thực

a) Kích thước và dải dữ liệu

Dữ liệu số thực bao gồm dữ liệu số thực đơn 32 bit chính xác đơn

Dữ liệu số thực có thể được lưu trữ trong các thiết bị khác với bit thiết bị hoặc trong dữ liệu thực đơn chính xác loại nhãn

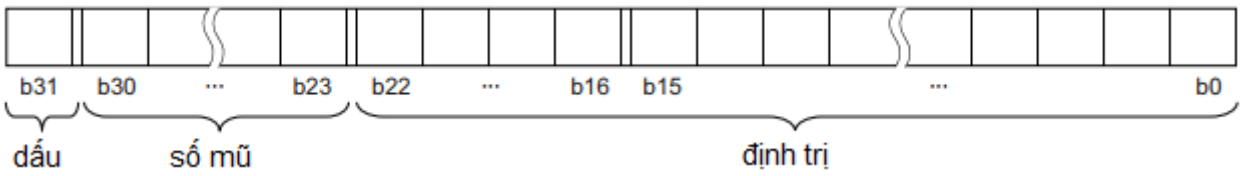
Kiểu dữ liệu		Kích thước	Dải giá trị
Số thực đơn	Số dương	32 bit (2 từ)	$2^{-126} \leq$ số thực $< 2^{128}$
	Số 0		0
	Số âm		$-2^{128} <$ số thực $\leq -2^{-126}$

Cấu hình của dữ liệu số thực chính xác đơn

Dữ liệu số thực chính xác đơn gồm có 1 dấu (sign), phần định trị (mantissa), số mũ (exponent) và được biểu thị như dưới đây:

Dấu 1. Phần định trị x $2^{mũ}$

Trong đó: phần dấu là 1 bit; phần định trị 23 bit và phần số mũ 8 bit.



- Dấu (sign) 1 bit

Bit này đặc trưng cho dấu dương hoặc âm của 1 số. “0” biểu thị một số dương hoặc 0. “1” biểu thị một số âm.

- Phần định trị (mantissa) 23 bits

Một phần định trị (logarit) nghĩa là XXXXX….of 1.XXXXX….$x2^N$ đại diện cho một số thực chính xác đơn trong hệ nhị phân.

- Phần mũ (exponent) 8 bits

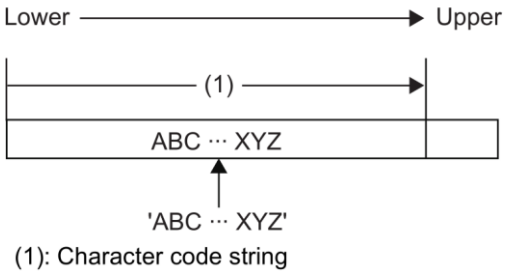
Một số mũ tức là N của 1.XXXXX….$x2^N$ đại diện cho một số thực chính xác đơn trong hệ nhị phân.

e). Kiểu dữ liệu chuỗi ký tự (Character string)

a) định dạng của kiểu dữ liệu chuỗi ký tự

Kiểu	Mã kỹ tự	Ký tự cuối
Chuỗi ký tự	Mã ASCII	NULL (00H)

Dữ liệu chuỗi ký tự được lưu trong thiết bị hoặc một mảng theo thứ tự tăng dần



Dải dữ liệu

Kiểu	Số lượng tối đa của chuỗi ký tự	Số lượng tối đa của chuỗi ký tự có thể được xử lý trong chương trình
Chuỗi ký tự	255 byte ký tự đơn (trừ ký tự NULL ở cuối)	16383 ký tự (không bao gồm ký tự NULL ở cuối)

Số words cần thiết để lưu trữ dữ liệu

Số byte chuôi ký tự	Số words cần thiết để lưu trữ chuỗi ký tự
0 byte	1 [word]
Số byte lẻ	(Số byte chuỗi ký tự +1) + 2 [words]
Số byte chẵn	(Số byte chuỗi ký tự :2) + 1 [words]

2.3. Bộ nhớ của PLC

1. Sơ lược về bộ nhớ của PLC

Bộ nhớ ROM là bộ nhớ cứng (nhớ vĩnh cửu) dùng để nhớ chương trình điều hành cơ bản do nhà sản xuất ghi.

Bộ nhớ EPROM, EEPROM, là bộ nhớ cứng có thể lập trình lại được bằng các công cụ lập trình, dùng để để lưu nhớ chương trình ứng dụng.

Bộ nhớ RAM là bộ nhớ động dùng để nhớ chương trình và các kết quả tính trung gian. Bộ nhớ này thường được nuôi bằng Pin, việc thay pin hoặc trong thời gian dài không sử dụng phải theo chỉ dẫn của nhà sản xuất.

Phân vùng bộ nhớ: Mỗi loại PLC có ký hiệu và cách phân vùng bộ nhớ cũng như dung lượng của mỗi vùng là khác nhau. Nói chung vùng nhớ của PLC được phân thành các vùng sau:

- Vùng nhớ vào/ ra;
- Vùng nhớ trung gian;
- Vùng nhớ giao tiếp;
- Vùng nhớ Timer, Counter;
- Vùng nhớ chuyên dụng.

2. Các vùng nhớ trong FX5U

Phân loại	Kiểu dữ liệu	Tên vùng	Ký hiệu – dải địa chỉ	Hệ cơ số
Thiết bị người dùng	Bit	Đầu vào (X)	1024 điểm *1	8
	Bit	Đầu ra (Y)	1024 điểm *1	8
	Bit	Rơle trung gian (M)	32768 điểm	10
	Bit	Rơle chốt (L)	32768 điểm	10
	Bit	Rơle liên kết (B)	32768 điểm	16
	Bit	Chỉ báo (F)	32768 điểm	10
	Bit	Rơle liên kết đặc biệt (SB)	32768 điểm	16
	Bit	Rơle bước (S)	4096 điểm (fixed)	10
	Bit/word	Timer (T)	1024 điểm	10
	Bit/word	Retentive Timer (ST)	1024 điểm	10
	Bit/word	Bộ đếm thường (C)	1024 điểm	10
	Bit/double word	Bộ đếm dài (LC)	1024 điểm	10
	Word	Thanh ghi dữ liệu (D)	800 điểm	10
	Word	Thanh ghi chỉ số (V)	16	16
	Word	Thanh ghi liên kết (W)	32768 điểm	16
	Word	Thanh ghi liên kết đặc biệt (SW)	32768 điểm	16
Thiết bị hệ thống	Bit	Rơ le đặc biệt (SM)	10000 điểm (Fixed)	10
	Word	Thanh ghi đặc biệt (SD)	12000 (fixed)	10
Thiết bị môđun truy cập	Word	Vùng môđun truy cập (G)	65536 điểm	10
Thanh ghi chỉ mục	Word	Thanh ghi chỉ mục (Z)	24 điểm	10
	Double word	Thanh ghi chỉ mục dài (LZ)	12 điểm	10

Tập thanh ghi	Word	Tập thanh ghi (R)	32768 có thể thay đổi	10
Nesting	_	Nesting (N)	15 điểm (fixed)	10
Con trỏ	_	Con trỏ (P)	4096 điểm	10
	_	Con trỏ ngắt (I)	178 điểm (fixed)	10
Hằng số	Hệ cơ số 10 (K)	Có dấu 16 bit	-32768 đến 32767	10
		Không dấu 16 bit	0 đến 65535	10
		Có dấu 32 bit	-2147483648 đến 2147483647	10
		Không dấu 32 bit	0 đến 4294967295	10
	Hệ cơ số 16 Hexa (H)	16 bit	0 đến FFFF	16
		32 bit	0 đến FFFFFFFF	16
	Số thực (E)	E-3.40282347+38 đến E-1.17549435-38; 0 ; E1.17549435 đến E3.40282347+38		_

*1. Tổng số điểm thực dùng cho cả hai vùng vào ra (X/Y) là 256 điểm.

3. Các vùng nhớ và chức năng của Fx5U

a) . Vùng vào

Input (X) là vùng nhớ cung cấp cho CPU các lệnh và dữ liệu bằng các thiết bị bên ngoài như nút nhấn, công tắc lựa chọn, công tắc giới hạn, công tắc số….

b). Vùng ra

Output Y là vùng nhớ lưu trữ kết quả đưa tín hiệu điều khiển cho các thiết bị bên ngoài như đèn báo, hiển thị số, công tắc tơ, cuộn dây điện từ v.v..

c). Vùng trung gian

Internal relay (M) là thiết bị được sử dụng như một rơ le phụ bên trong mô đun CPU. Tất cả các rơ le trung gian được tắt bằng các hoạt động sau:

- Tắt nguồn mô đun CPU
- Reset
- Xóa chốt

d). Vùng chốt (Latch relay (L))

Latch relay (L) là rơ le phụ có thể chốt (dự phòng bằng pin) trong mô đun CPU.

Kết quả tính toán (thông tin ON/OFF) được chốt khi thực hiện các hoạt động sau

- Tắt nguồn mô đun CPU
- Reset

3. Vùng liên kết

Thiết bị được sử dụng như một phần phụ của CPU khi làm mới dữ liệu bit giữa mô đun CPU và mô đun mạng.

Gửi/nhận dữ liệu qua lại giữa những mô đun mạng rơ le liên kết (LB) và rơ le liên kết (B) trong mô đun CPU. Đặt mới dải bằng thông số của mô đun mạng và không thể sử dụng rơ le liên kết cho những mục đích khác.

e). Vùng liên kết đặc biệt

Thông tin và trạng thái lỗi của các mô đun mạng là đầu ra đến các rơ le liên kết đặc biệt trong phạm vi mạng. Rơ le liên kiết đặc biệt (SB) là thiết bị được sử dụng như một điểm đến làm mới cho các rơ le liên kết đặc biệt trong mạng và không được sử dụng cho các mục đích khác.

f). Vùng trạng thái

Thiết bị được sử dụng với các lệnh Step ladder. Ở đâu Step ladder không được sử dụng nó có thể được dùng cho các mục đích như rơ le phụ.

g). Vùng Timer

Là thiết bị mà ở đó phép đo bắt đầu khi cuộn hút timer bật, thời gian tăng lên đến khi giá trị hiện tại đạt đến giá trị cài đặt, và khi đó tiếp điểm timer được bật. timer cũng là một kiểu bộ đếm thêm vào. Khi thời gian đếm tăng, giá trị hiện tại và giá trị đếm là như nhau. Trong PLC có nhiều dạng Timer khác nhau

h). **Vùng bộ đếm**

Có chức năng đếm số lần tăng các điều kiện đầu vào trong chương trình. Khi giá trị đếm đạt giá trị cài đặt thì tiếp điểm ngõ ra của Counter bật ON. Trong PLC cũng có nhiều loại Counter.

i). Vùng thanh ghi dữ liệu

Là thiết bị có khả năng lưu trữ dữ liệu số.

k). Vùng thanh ghi liên kết và thanh ghi liên kết đặc biệt

- Thanh ghi liên kết (LW)

Là thiết bị được sử dụng nhằm mục đích như một thiết bị phụ CPU khi làm mới dữ liệu từ giữa mô đun CPU và mô đun mạng.

Gửi/Nhận dữ liệu qua lại lẫn nhau giữa các thanh ghi liên kết trong mô đun mạng và thanh ghi liên kết trong mô đun CPU. Đặt mới phạm vi sử dụng bằng các thông số của mô đun mạng. Các thanh ghi liên kết không được sử dụng cho các mục đích khác.

- Thanh ghi liên kết đặc biệt (SW)

Dữ liệu từ như là thông tin và trạng thái lỗi của các mô đun mạng là đầu ra đến các rơ le liên kết đặc biệt trong phạm vi mạng. Các thanh ghi liên kết đặc biệt (SW) là các thiết bị được sử dụng như một điểm đến làm mới cho các thanh ghi liên kết đặc biệt trong mạng và không được sử dụng cho các mục đích khác.

l). Vùng rơ le đặc biệt

Plc chứa các rơ le nội với các thông số được cố định. Vì vậy nó không thể được sử dụng trong chương trình như một rơ le nội thông thường. Tuy nhiên nó có thể được bật/tắt để điều khiển CPU khi cần thiết.

m). Vùng thanh ghi đặc biệt

Plc chứa các rơ le nội với các thông số được cố định. Vì vậy nó không thể được sử dụng trong chương trình như một thanh ghi nội thông thường. Tuy nhiên dữ liệu có thể được viết để điều khiển CPU khi cần thiết.

n). Vùng mô đun truy cập

Thiết bị cho phép bạn truy cập trực tiếp bộ nhớ đệm của các mô đun chức năng thông minh đã được kết nối đến mô đun CPU.

- Đặc điểm kỹ thuật

Được xác định bằng U[số lượng các mô đun chức năng thông minh]/[địa chỉ bộ nhớ đệm]

- Tốc độ xử lý

Tốc độ xử lý của việc đọc và ghi bằng thiết bị mô đun truy cập là nhanh hơn việc sử dụng lệnh FROM/TO.

Khi việc đọc bộ nhớ đệm của một thiết bị mô đun truy cập và thực thi quá trình bằng một lệnh khác

o). Vùng thanh ghi chỉ mục

Là thiết bị được sử dụng cho việc đánh chỉ mục của các thiết bị.

- Các kiểu của các thanh ghi chỉ mục
- ➢ Thanh ghi chỉ mục Z (index register) được sử dụng để sử đổi chỉ số 16 bit

➢ Thanh ghi chỉ mục dài LZ (Long index register) được sử dụng để sử đổi chỉ số 32 bit

p). Vùng thanh ghi tập tin

Là thiết bị có khả năng lưu trữ dữ liệu số

q). Vùng Nesting

r). Vùng con trỏ

Là thiết bị được sử dụng như lệnh nhảy và chương trình con của lệnh gọi CALL.

Kiểu của con trỏ như sau:

Con trỏ	Mô tả
Con trỏ chung	Con trỏ có thể được gọi từ tất cả chương trình
Con trỏ gán nhãn	Con trỏ được sử dụng bằng việc gán đến các nhãn. Số lượng con trỏ được gán đến nhãn được xác định tự động bằng các công cụ kỹ thuật. Người dùng không thể chỉ định một số con trỏ đã được gãn nhãn.

Con trỏ được sử dụng cho các mục đích sau:

Xác định nhãn và nơi nhảy đến cho lệnh nhảy (CJ)

Xác định nhãn (chương trình đầu của chương trình con) và lệnh gọi đến đích của chương trình con (lệnh CALL)

+ Con trỏ chung. Con trỏ cho việc gọi chương trình con từ tất cả các chương trình đang được chạy.

Con trỏ gán nhãn. Con trỏ cho gán nhãn được chỉ định một cách tự động bởi các công cụ kỹ thuật và không được chỉ định trực tiếp.

s). Vùng con trỏ ngắt

Thiết bị được sử dụng như nhãn tại chương trình đầu của ngắt. Có thể được sử dụng cho tất cả các chương trình đang chạy.

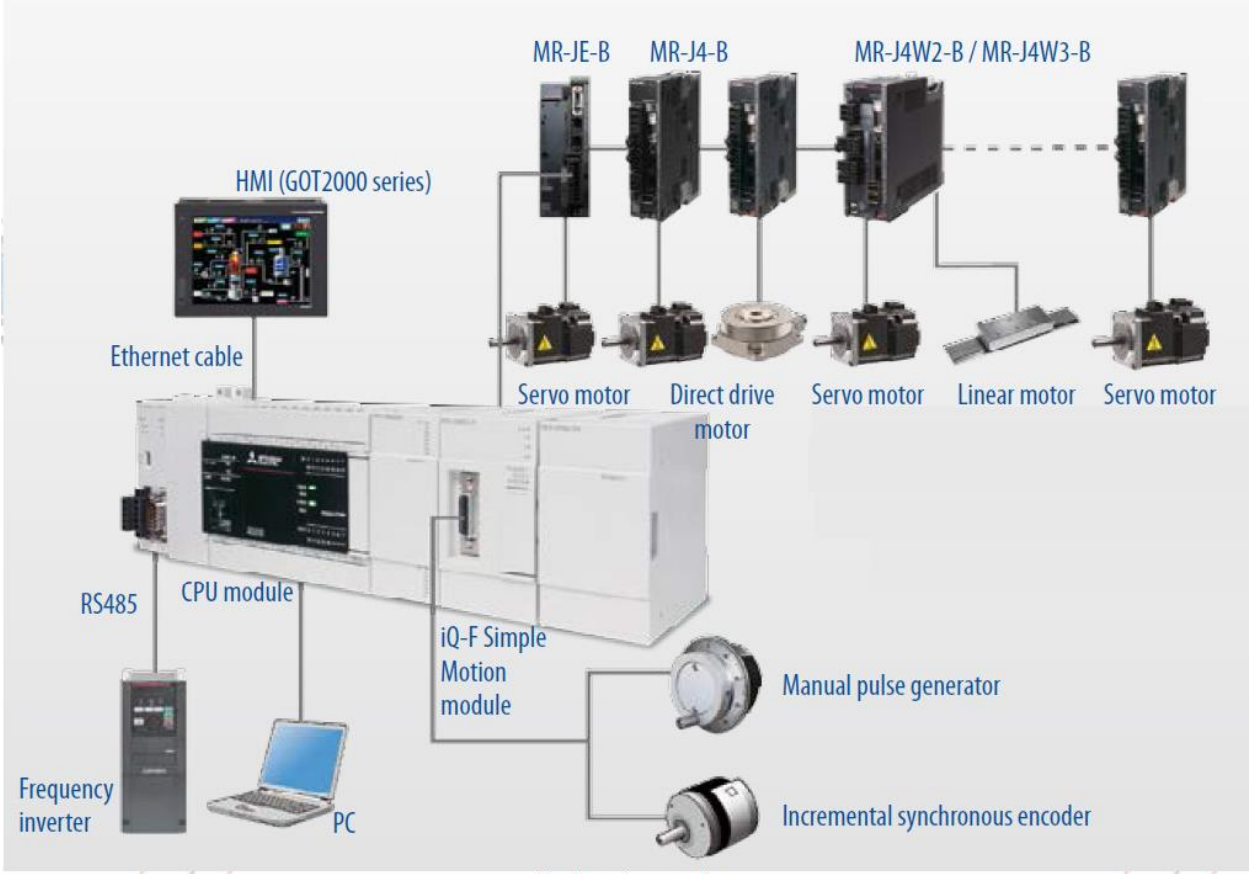
t). Vùng hằng số

Hằng số cơ số 10 (K)

Thiết bị chỉ rõ dữ liệu cơ số 10 cho chương trình. Được chỉ định là Km (vi dụ K1234)

Kiểu dữ liệu tham số của lệnh		Dải làm việc của các hằng số cơ số 10
Kích thước dữ liệu	Tên kiểu dữ liệu	
16 bits	Word(có dấu)	K-32768 đến K32767
	Word (không dấu)/ Bit string (16 bit)	K0 đến K65535
32 bits	Double word (có dấu)	K-2147483648 đến K2147483647
	Double word (không dấu)/ Bit string (32 bit)	K0 đến K4294967295

2.4. Giới thiệu về PLC thế hệ mới FX 5U của Mitshubishi



Configuration example

1. Modun CPU

FX5U

FX5UC

Các đặc điểm chung

Control Scale	từ 32 đến 256 điểm; tối đa 512 điểm bao gồm cả CC-Link, AnyWireALINK và remote I/O
Program memory	64000 bước (step)
Có khe cắm SD card	Tối đa 4GB
Cổng Ethernet có sẵn	+ Kết nối trực tiếp với các PLC khác + Ghi/Đọc chương trình dùng GX Work3 qua cổng VPN. + Ghi/đọc các dữ liệu (data) từ PLC
Cổng RS-485 có sẵn	+ Thích ứng cả hai chuẩn RS-485 và RS-422 + ứng dụng cho truyền thông với biến tần (chiều dài lớn nhất 50m. tối đa nối được 16 thiết bị. + truyền thông kiểu MODBUS cho phép kết nối với 32 thiết bị bao gồm các PLC khác, các cảm biến, các bộ điều chỉnh nhiệt độ.
I/O chức năng chuyên dụng	Điều khiển vị trí 4 trục với xung tần số 200kHz
	Bộ đếm tốc độ cao: tối đa 8 kênh với xung đầu vào 200kHz
Đầu ra	Kiểu rơ-le và transistor

Riêng loại FX5U còn có sẵn 2 kênh vào A/D 12 bit và 1 kênh ra D/A 12 bit..

Đặc điểm chi tiết các vùng nhớ để sử dụng đã đề cập chi tiết ở mục vùng nhớ.

CPU FX5U được chế tạo để dùng cả điện xoay chiều (85-260VAC) và một chiều 24VDC.

2. Các môdun mở rộng

Để mở rộng khả năng hoạt động có chế tạo sẵn các loại sau.

a). Khối I/O mở rộng dùng nguồn AC hoặc DC với 16 hoặc 32 I/O (nửa là Inputs + nửa là Outputs) có cả đầu ra role và transistor (soucing và sinking).

b) Khối đầu vào mở rộng có loại 8/16 và 32 Inputs dùng nguồn 24VDC’

c) Khối đầu ra mở rộng có loại 8/16 và 32 Outputs cả kiểu rơle và transistor (soucing và sinking).

d). Khối xung cao tốc (high speed pulse) với 8 đầu vào và 8 đầu ra.

e) Khối vào tương tự (A/D) với các loại 4 kênh và 8 kênh được chế tạo kiểu điện áp có các dải: 0-5V; 1-5V; 0-10V và dải (-10V đến +10V).

f). Khối ra tương tự (D/A) với loại 4 kênh cũng kiểu điện áp có dải như khối đầu vào tương tự

g). Khối đầu vào kết nối trực tiếp với cảm biến nhiệt độ; có loại 4 kênh và 8 kênh. Chủng loại sensor nhiệt độ từ -200^{o}C đến 1200^{o}C (Pt100; Ni100;… Căp nhiệt ngẫu loại K,J,T,B,R,S…).

h). Khối bộ đếm tốc độ cao (High Speed Counter): loại 2 kênh, tần số tối đa 200kHz.

i). Khối chuyên dụng cho điều khiển chuyển động (Motion) 4 trục và 8 trục.

k).Khối chuyên dụng điều khiển vị trí 1 trục và 2 trục (tần số 200kHz).

l). Các khối mở rộng về truyền thông loại:CC-LINK; Ethernet; MODBUS; Serial communication.

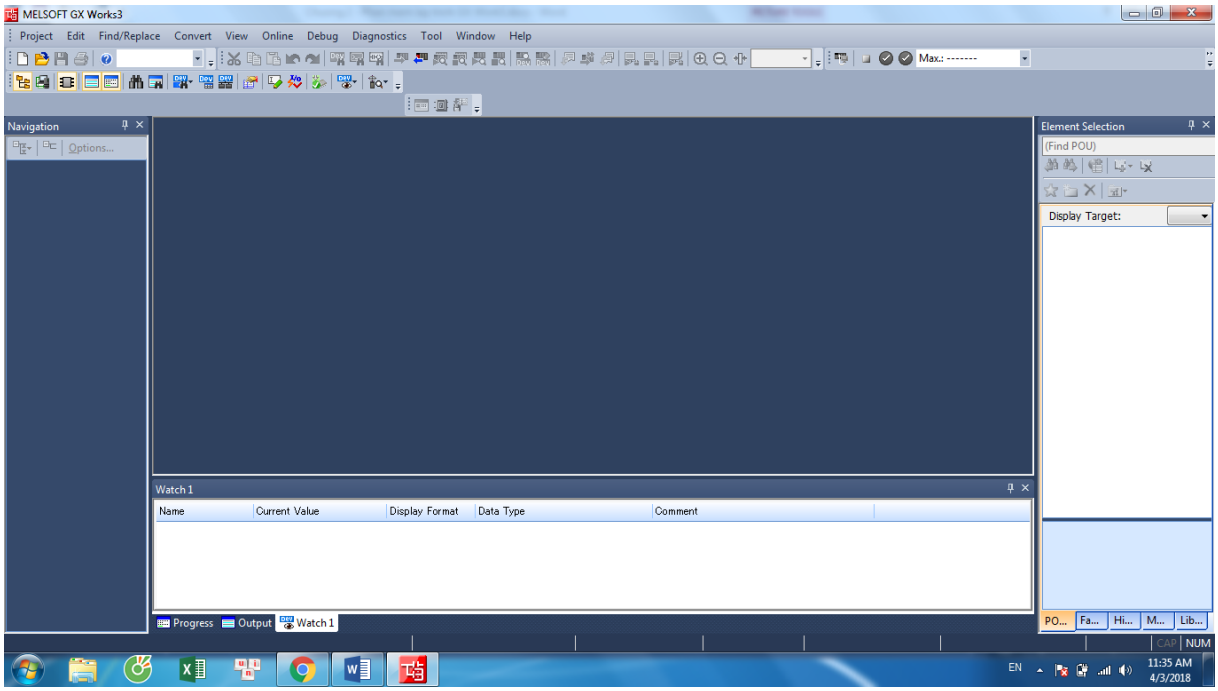
m). Các khối nguồn cung cấp 5VDC dòng tải xấp xỉ 1A.

Chương 2: Phần mềm lập trình PLC GX Works3

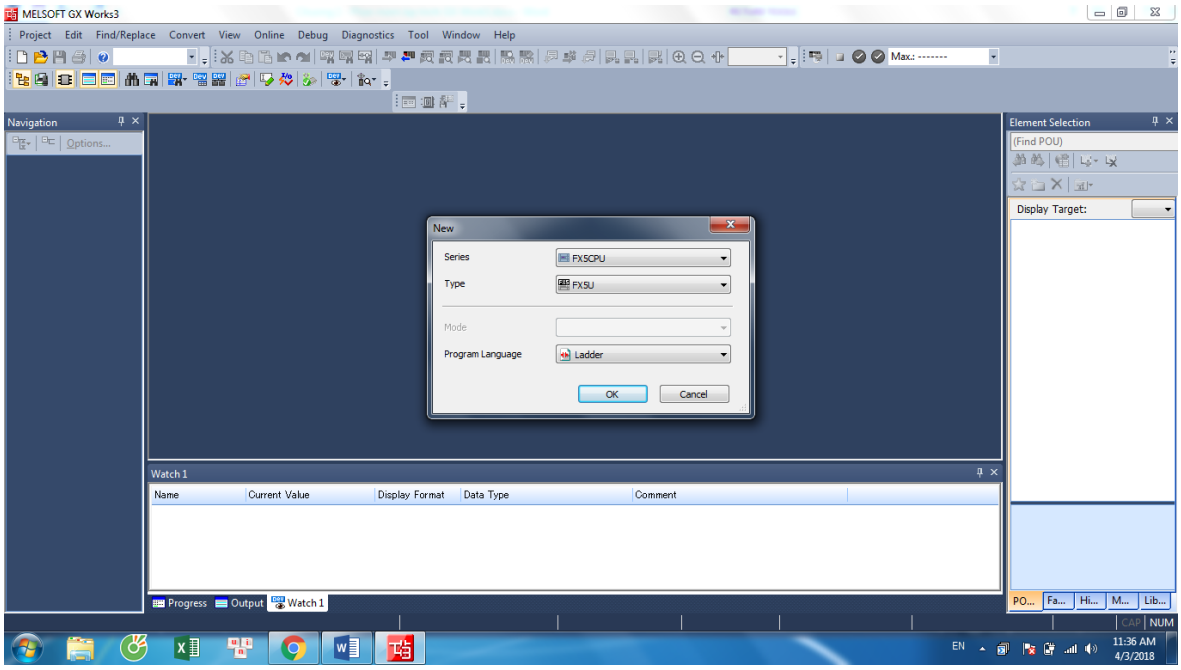
GX Works3 là phần mềm lập trình PLC mới nhất của Mitsubishi dành cho 2 dòng PLC mới của hãng là FX5U (iQ-F) và iQ-R. GX Works3 có rất nhiều tính năng ngoài thiết lập tham số cho từng module của PLC, lập trình bằng nhiều ngôn ngữ (LAD, FBD, SFC, ST), như là chuẩn đoán lỗi của từng module trong PLC, theo dõi chương trình trực tiếp khi PLC hoạt động, theo dõi các dữ liệu trong các vùng nhớ dữ liệu khác vùng nhớ chương trình, chuẩn đoán tình trạng của hệ thống mạng CC-Link, bổ sung các bản cập nhật firmware cho các module, vv…

Các bước tạo một project mới với FX5U:

Bước 1: Khởi động phần mềm GX Works3



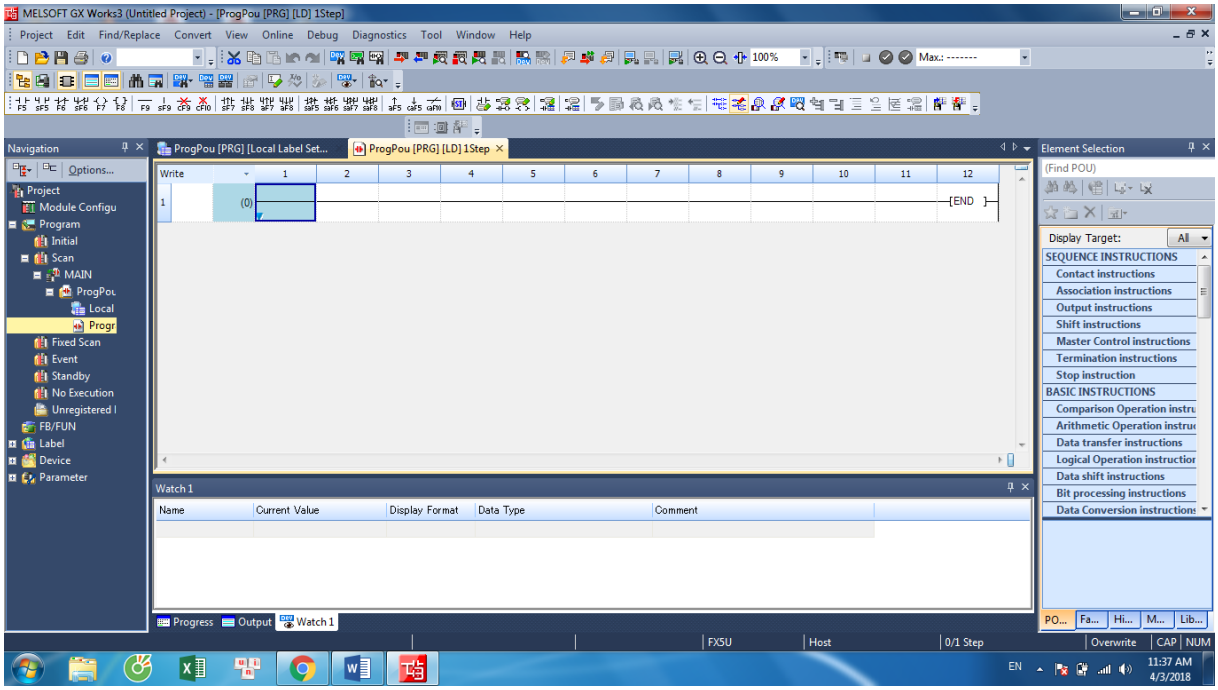
***Bước 2*:** Chọn Project/New để tạo project mới:



Trong cửa sổ New có các lựa chọn sau:

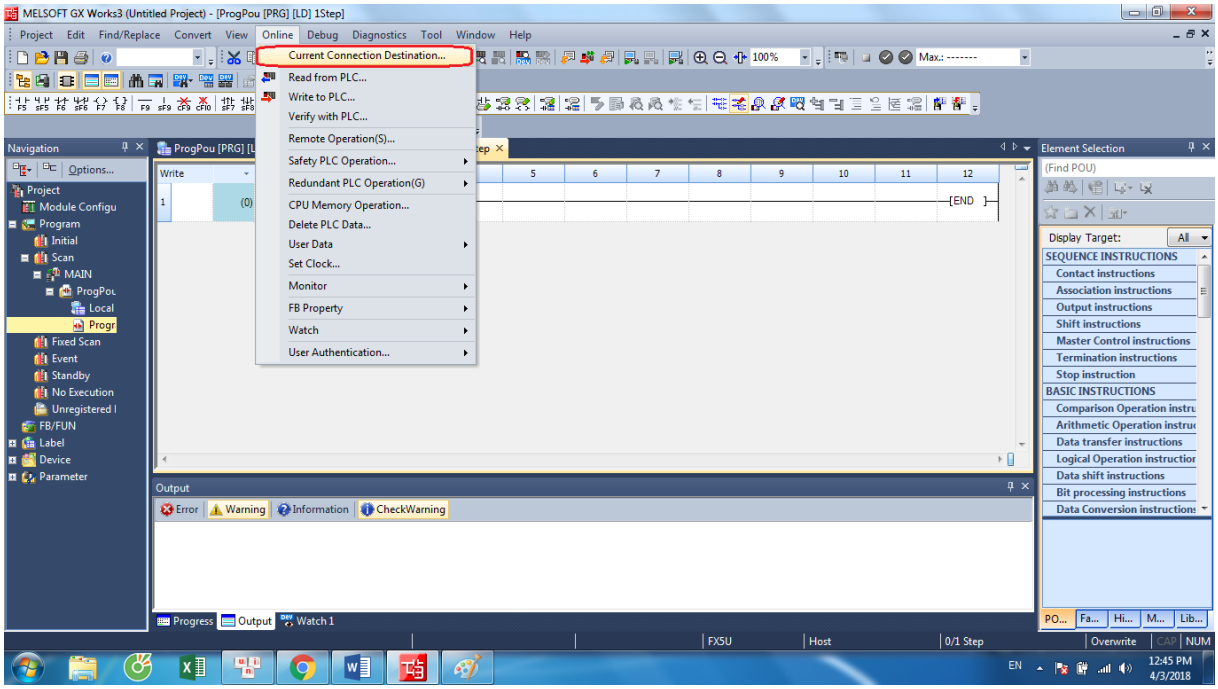
- Series: Chọn dòng PLC cần lập trình FX5CPU.
- Type: chọn loại PLC cần lập trình là FX5U.
- Program Language: Chọn ngôn ngữ lập trình cho PLC.

Khai báo xong các thông số chọn ***OK***.



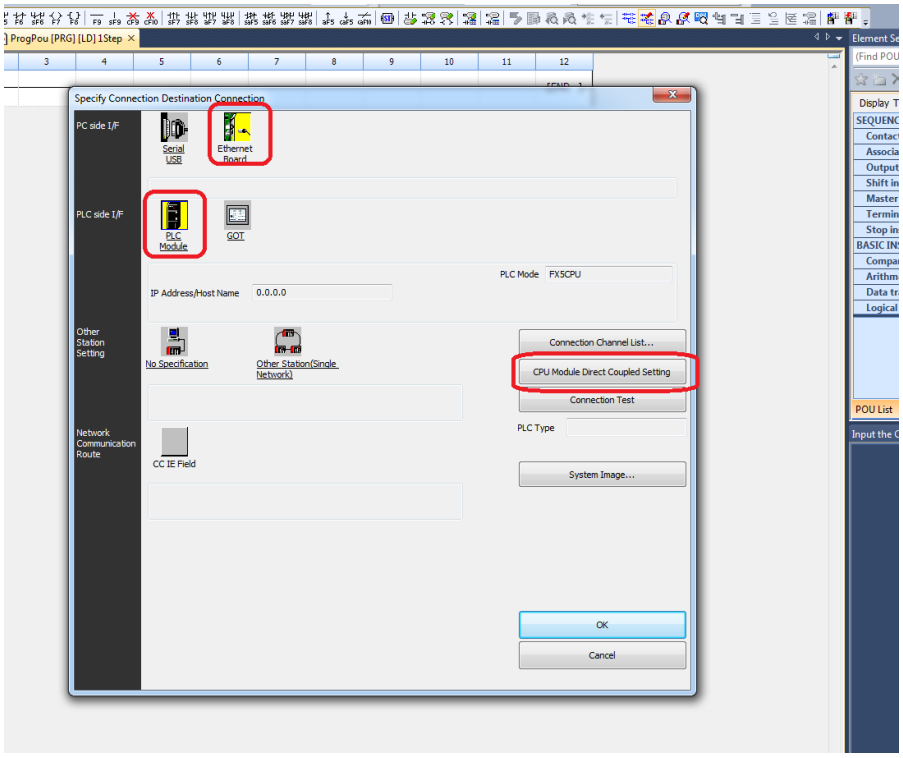
***Bước 3*:** Kiểm tra kết nối giữa PC và FX5CPU khi kết nối trực tiếp bằng cáp Ethernet.

- Trên ***Menu*** bar, click ***Online*** và chọn ***Current Connection Destination***.

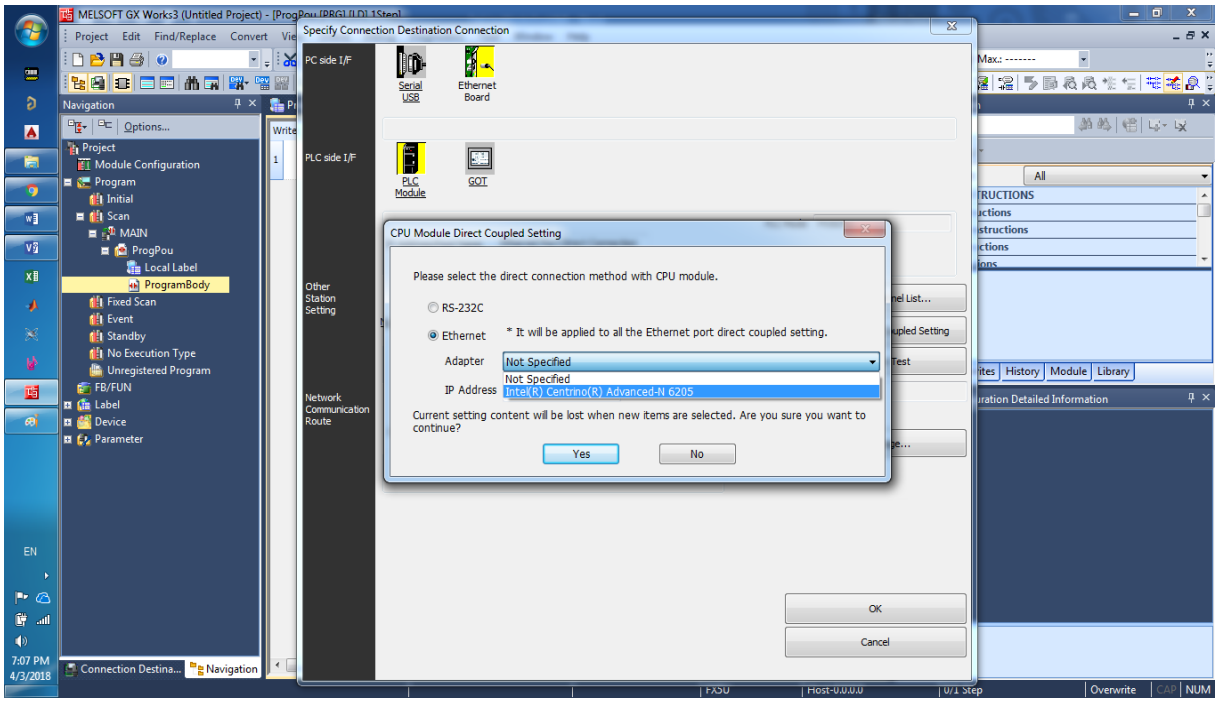


Xuất hiện cửa sổ thiết lập kết nối giữa PC và PLC.

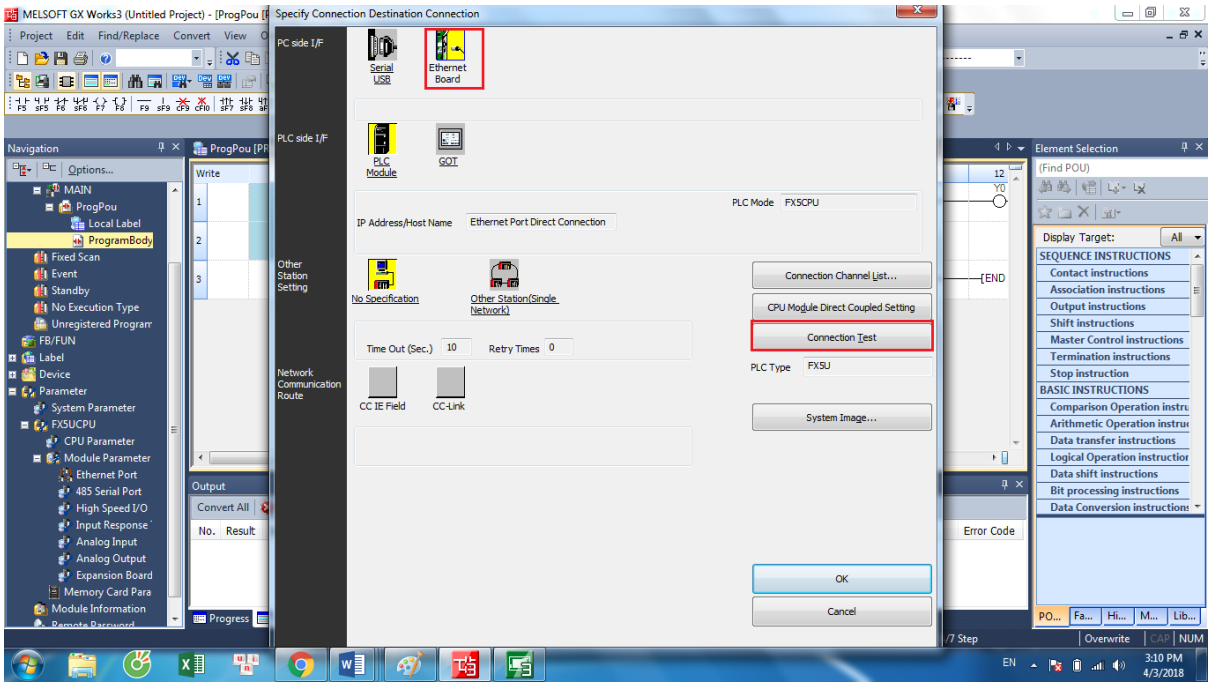
- PC side I/F: chọn kết nối phía máy tính. PLC side I/F chọn kết nối phía FX5U. Ở đây kết nối trực tiếp bằng cáp Ethernet nên hai mục này chọn như hình dưới.
- Nếu kết nối lần đầu thì chọn card mạng như hình dưới.



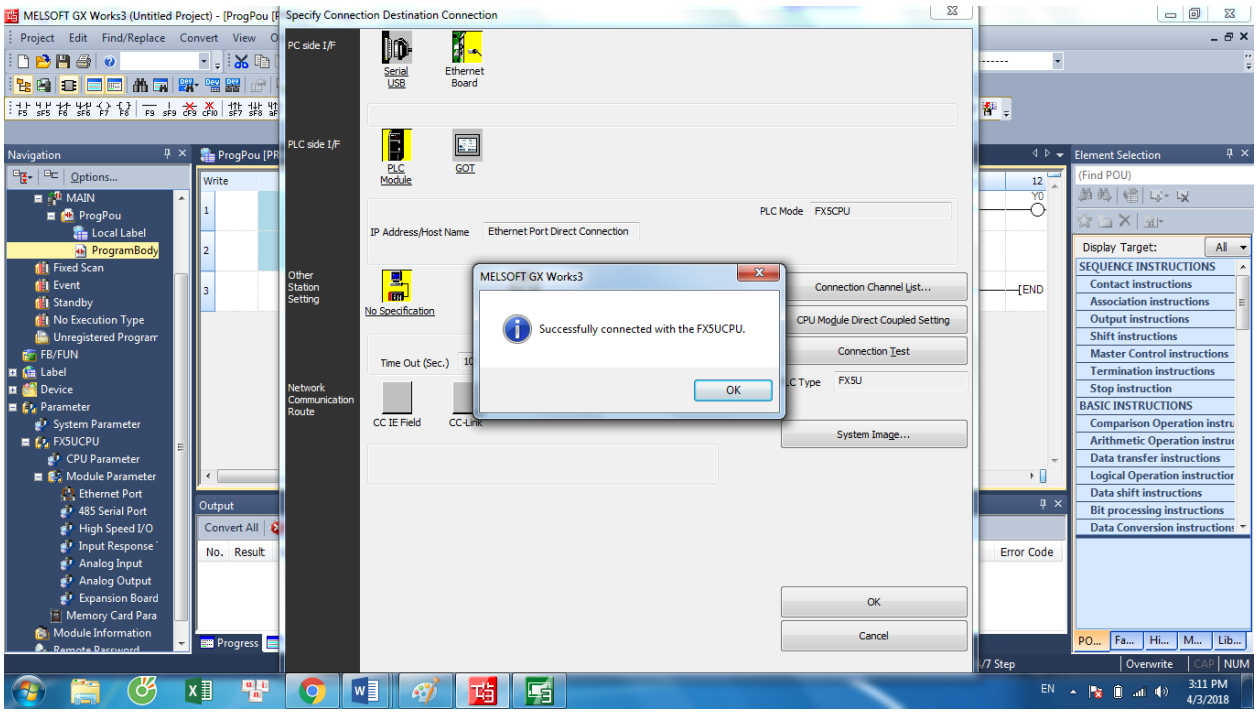
- Ban đầu thường mặc định là "Not Specified", chọn lại loại card mạng đang có trong máy tính như hình dưới. Nếu không có kiểm tra máy tính có card mạng hay không hoặc chưa cài driver cho card mạng.



- Sau khi chọn đường kết nối xong bấm ***Click Connection Test*** để kiểm tra kết nối.

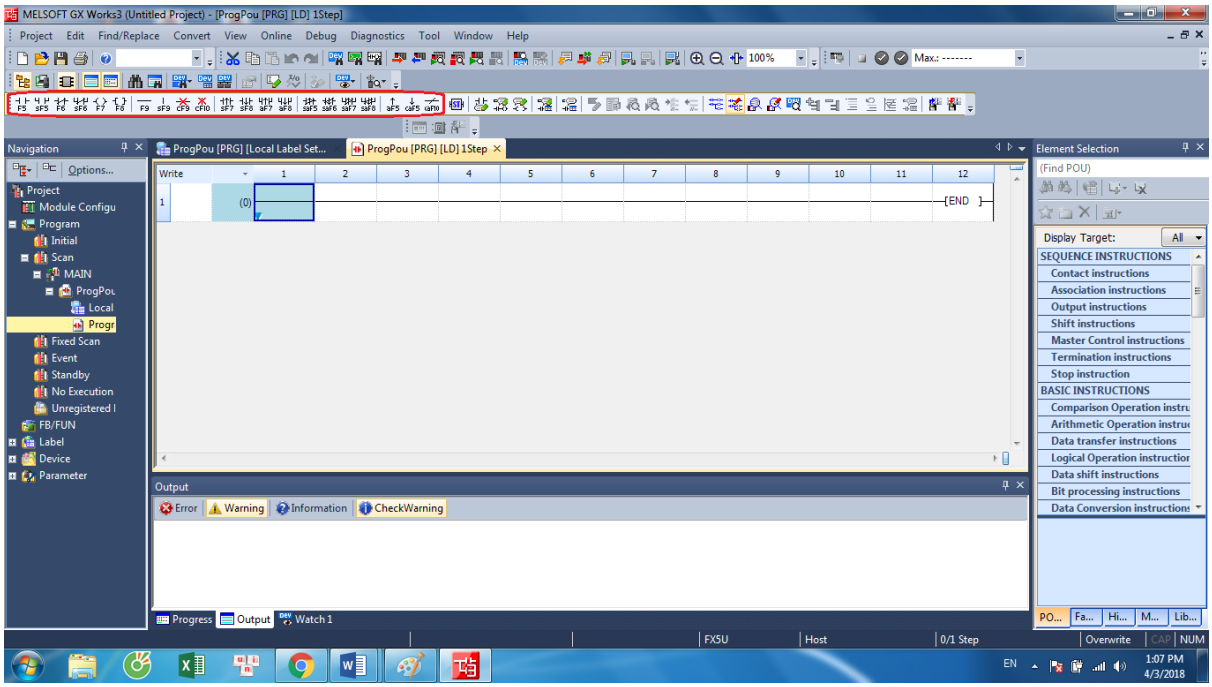


- Cửa sổ thông báo kết nối thành công như hình dưới. Nếu không thành công kiểm tra lại các bước hoặc kiểm tra thiết bị kết nối (đầu nối, dây nối …)

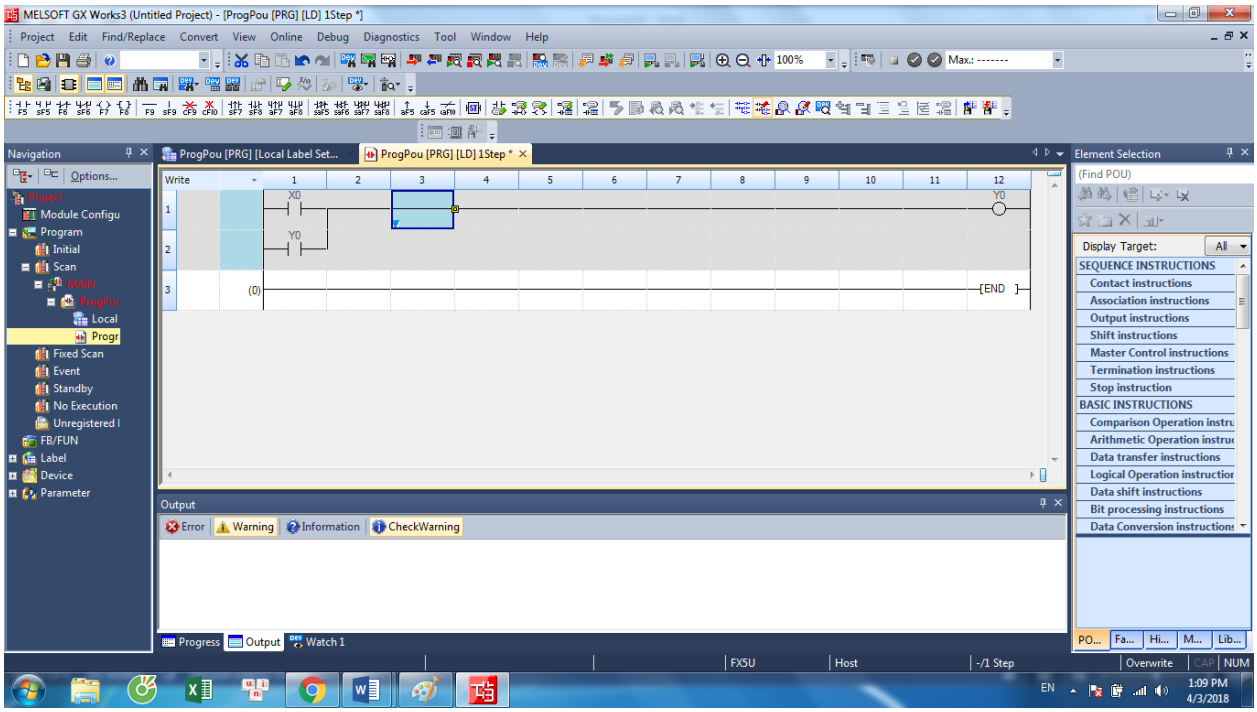

MELSOFT GX Works3 (Untitled Project) - [ProgPou
Project Edit Find/Replace Convert View
Navigation
Options...
MAIN
ProgPou
Local Label
ProgramBody
Fixed Scan
Event
Standby
No Execution Type
Unregistered Program
FB/FUN
Label
Device
Parameter
System Parameter
FX5UCPU
CPU Parameter
Module Parameter
Ethernet Port
485 Serial Port
High Speed I/O
Input Response
Analog Input
Analog Output
Expansion Board
Memory Card Para
Module Information
ProgPou [PR
Write
Output
Convert All
No. Result
Progress
Specify Connection Destination Connection
PC side I/F
Serial USB
Ethernet Board
PLC side I/F
PLC Module
GOT
PLC Mode FX5CPU
IP Address/Host Name
Ethernet Port Direct Connection
Other Station Setting
No Specification
Connection Channel List...
CPU Module Direct Coupled Setting
Connection Test
Time Out (Sec.)
LC Type FX5U
Network Communication Route
CC IE Field
CC-Link
System Image...
OK
Cancel
MELSOFT GX Works3
Successfully connected with the FX5UCPU.
OK
Element Selection
(Find POU)
Display Target: All
SEQUENCE INSTRUCTIONS
Contact instructions
Association instructions
Output instructions
Shift instructions
Master Control instructions
Termination instructions
Stop instruction
BASIC INSTRUCTIONS
Comparison Operation instru
Arithmetic Operation instruc
Data transfer instructions
Logical Operation instructior
Data shift instructions
Bit processing instructions
Data Conversion instructions
Error Code
Overwrite
CAP NUM
EN
3:11 PM
4/3/2018

Bước 4: Viết chương trình

Trên thanh công cụ lựa chọn các tiếp điểm, cuộn hút, bus phù hợp để lập trình.

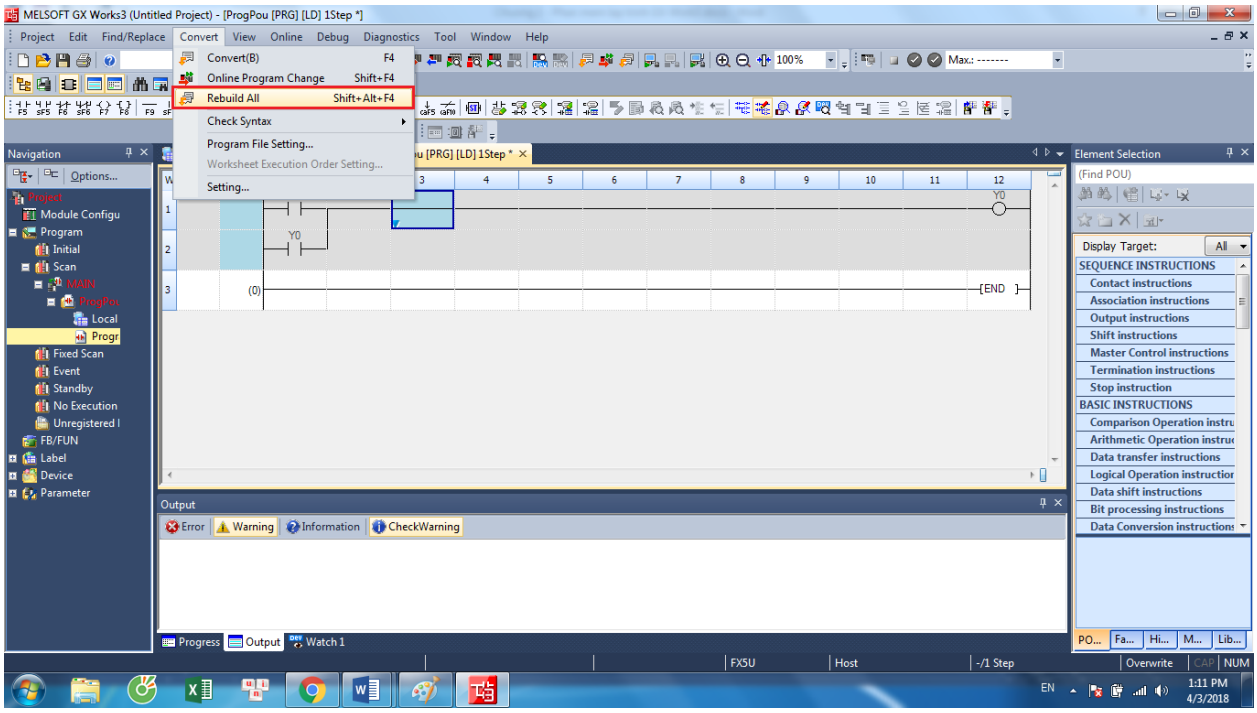


Ví dụ một chương trình đơn giản.

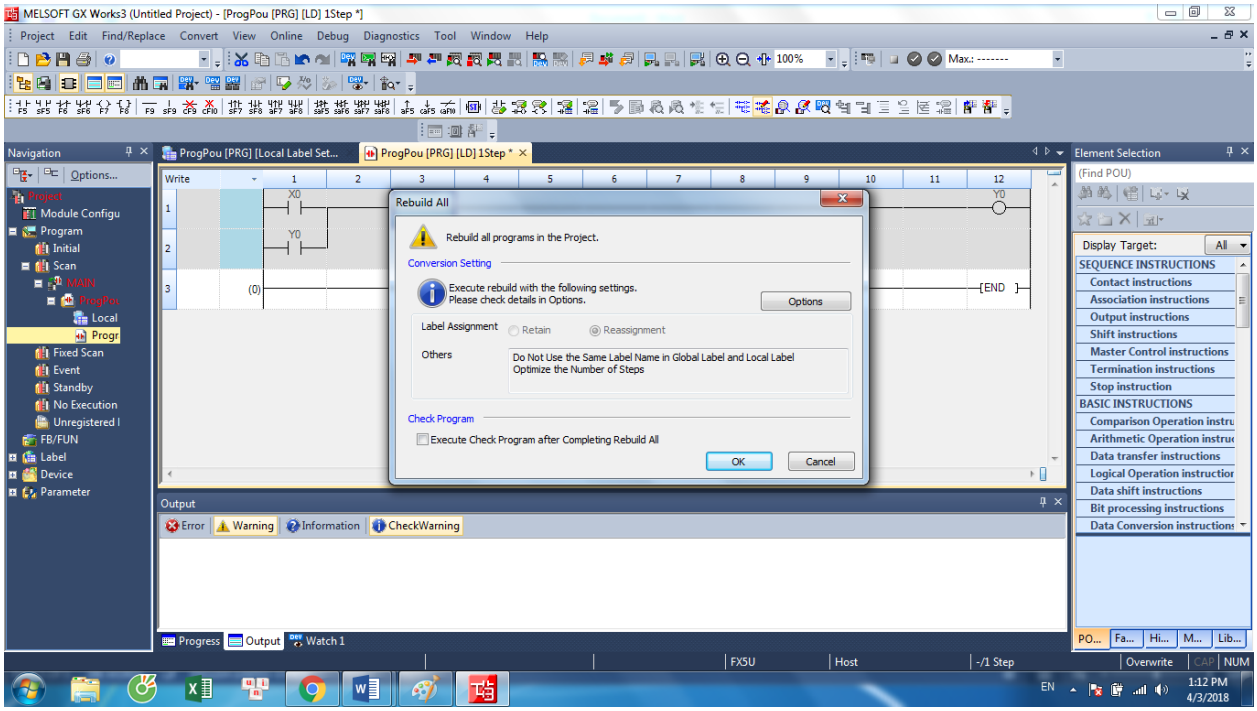


Bước 5: Biên dịch chương trình để kiểm tra lỗi soạn thảo và nạp vào PLC.

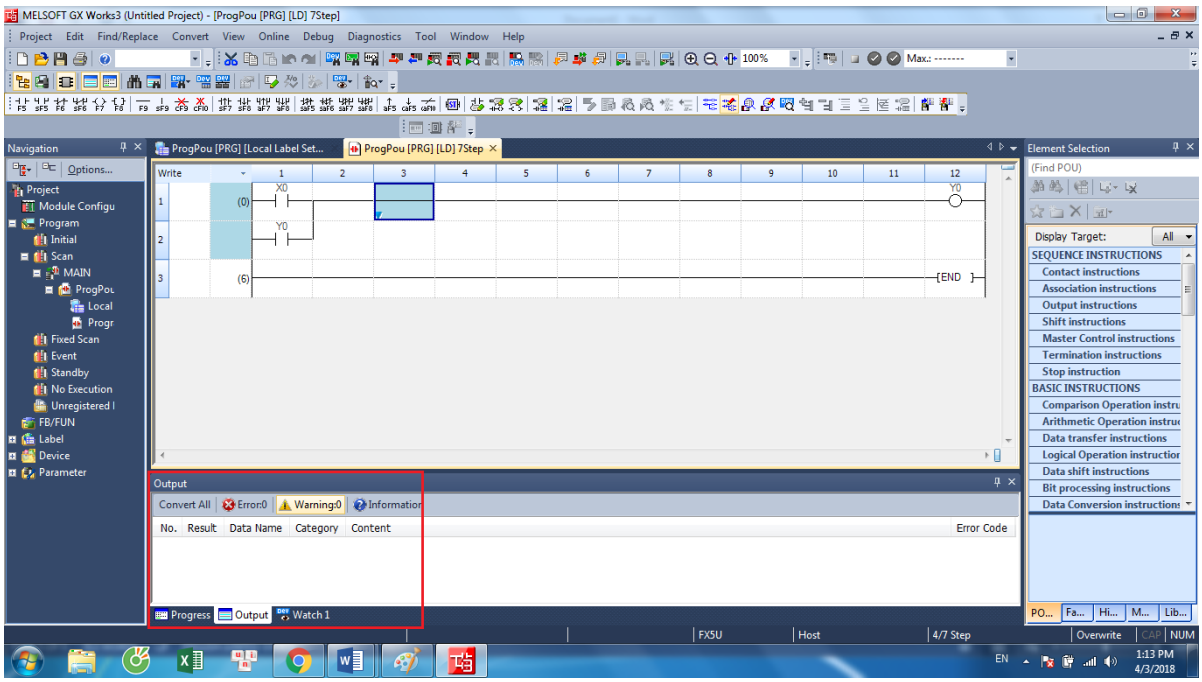
Trên ***Menu Bar***, click ***Convert*** và chọn ***Rebuild All.***



Bảng ***Rebuild All*** xuất hiện, chọn ***Ok*** .



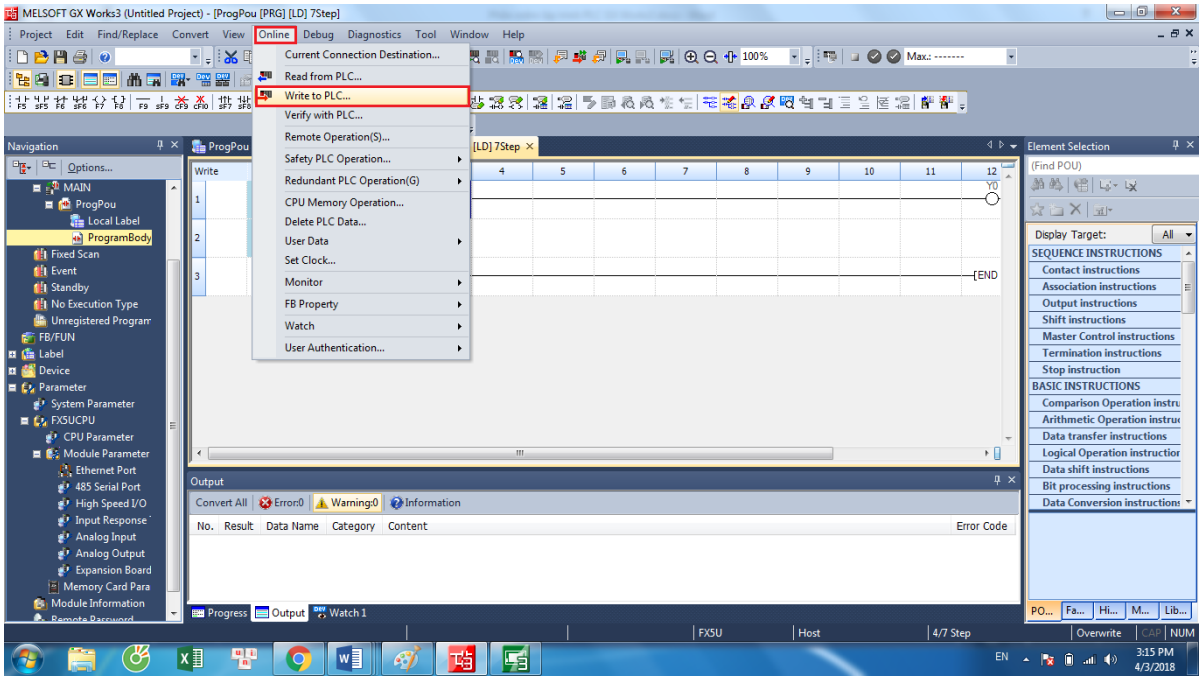
Sau đó kiểm tra có lỗi xảy ra hay không tại của sổ Output.



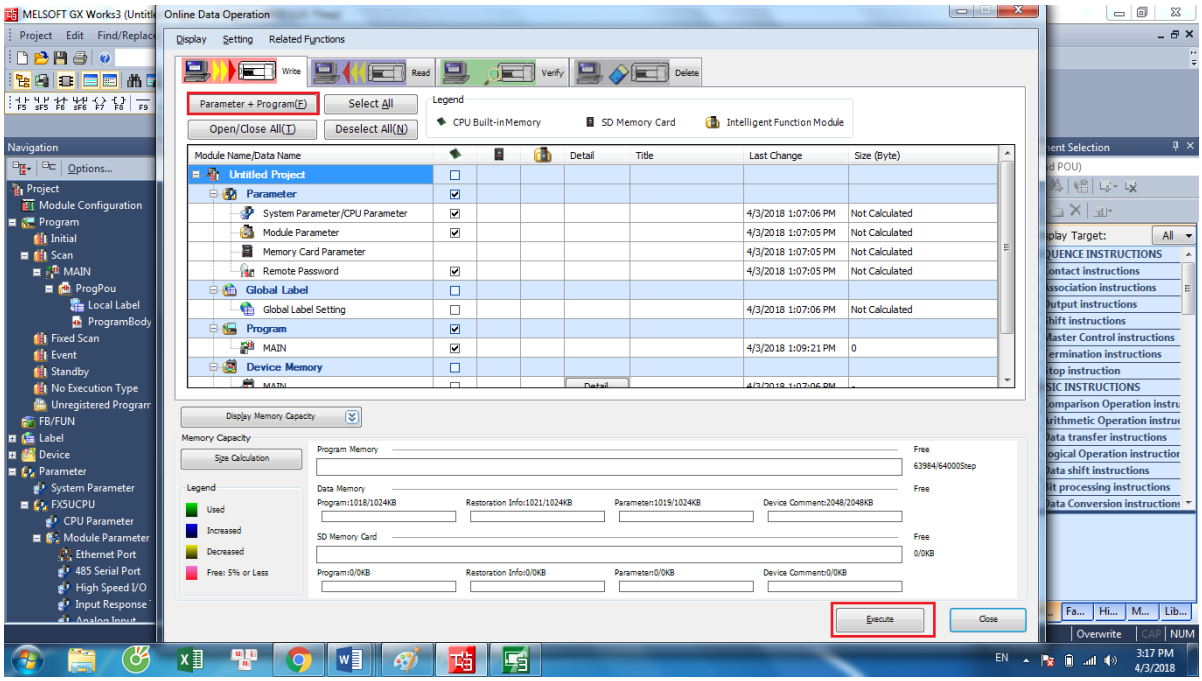
Nếu không có lỗi là Project đã biên dịch thành công.

Bước 6: Nạp chương trình vào FX5UCPU. Phải chắc chắn đã nối dây Ethernet giữa PLC và PC.

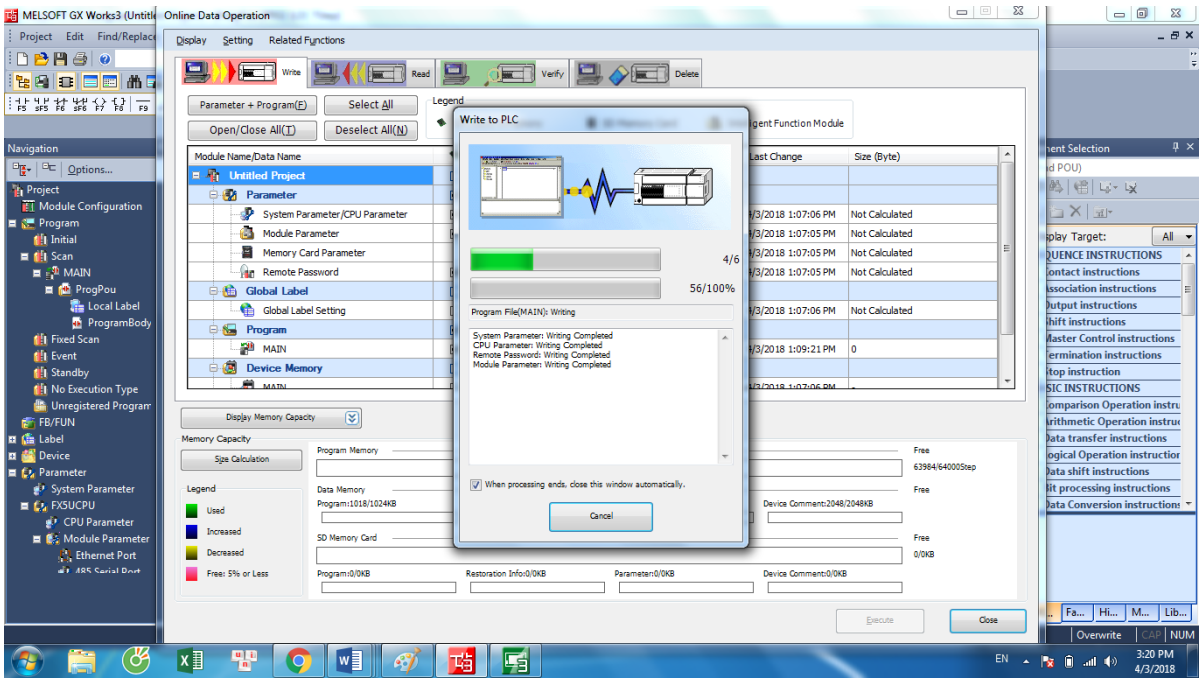
Chọn ***Online*** => ***Write to PLC***



Click ***Parameter + Program*** => Execute



Thông báo hiện ra chọn ***Yes to all*** và chương trình được nạp vào PLC.

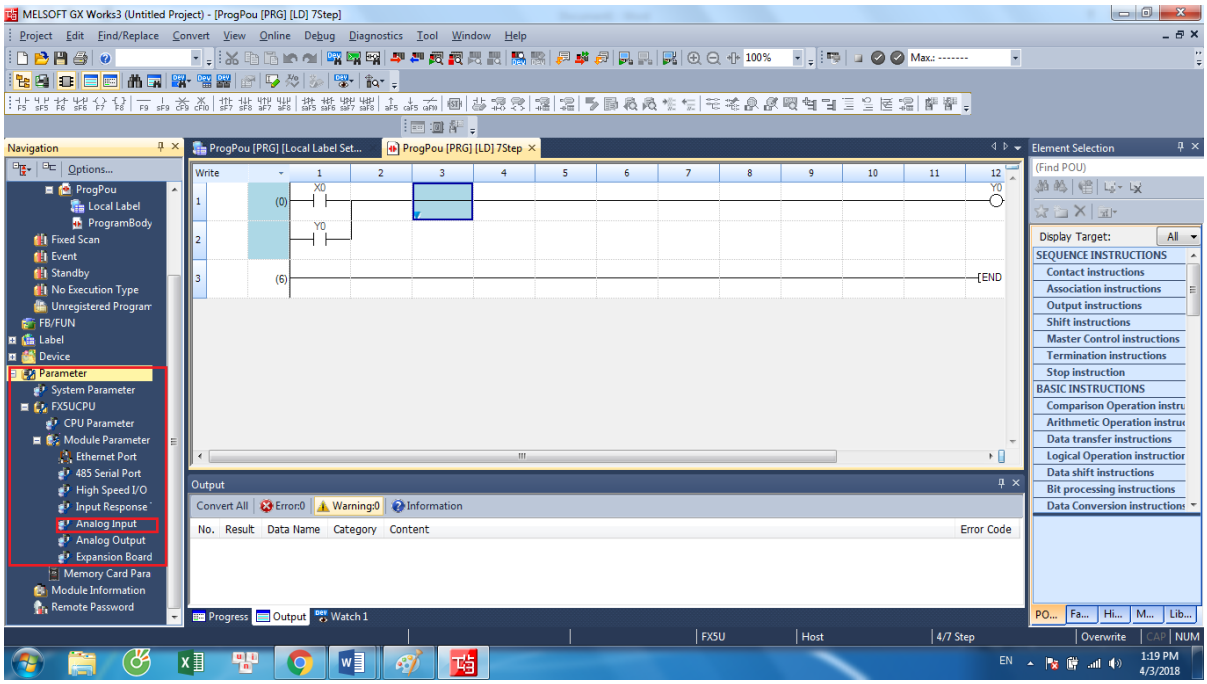


Sau đó chọn ***Close*** để đóng cửa sổ ***Online Data Operation.***

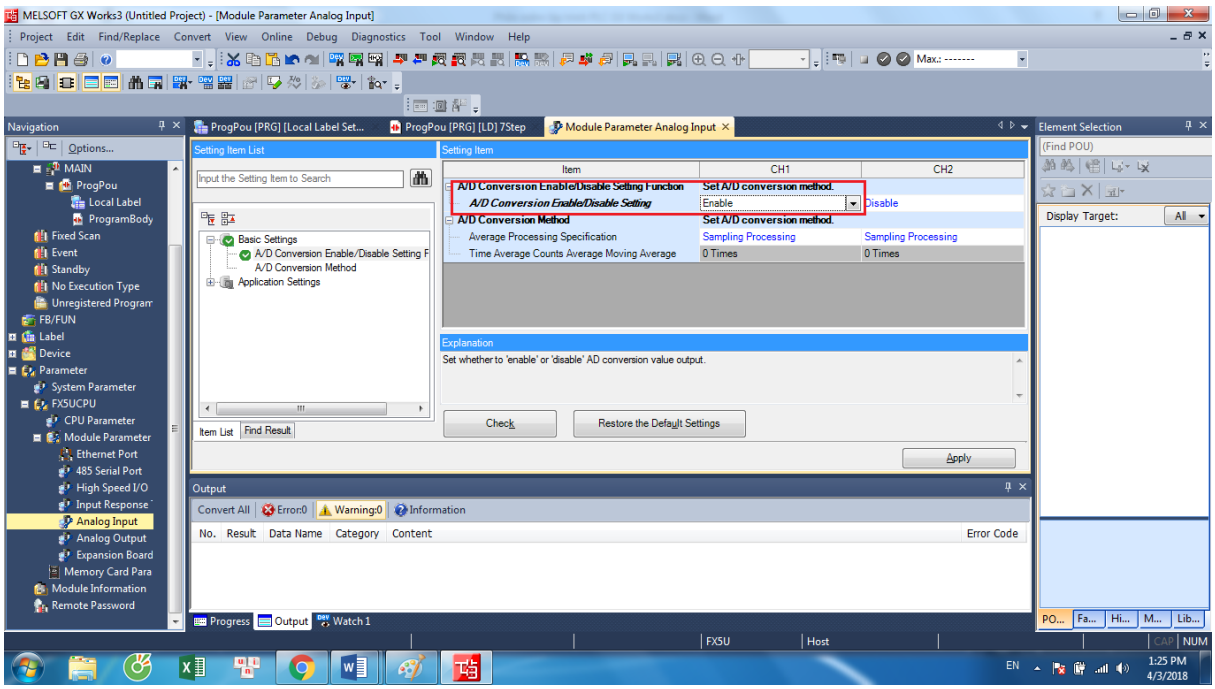
1. Cấu hình một số chức năng được tích hợp trong FX5CPU.

2.1 .Analog Input

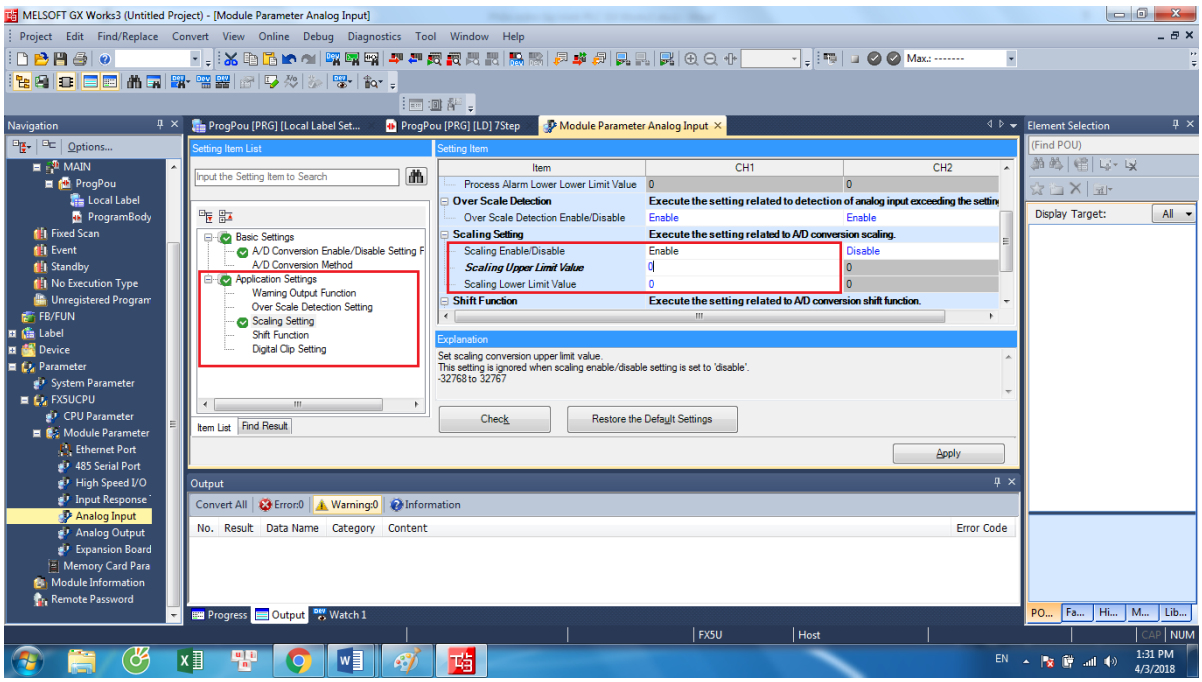
Trong cửa sổ Navigation chọn Parameter => FX5UCPU => Module Parameter và Click vào Analog Input.



Cửa số cài đặt Analog Input xuất hiện => chọn *A/D Conversion Enable/Disable Setting* thành Enable.

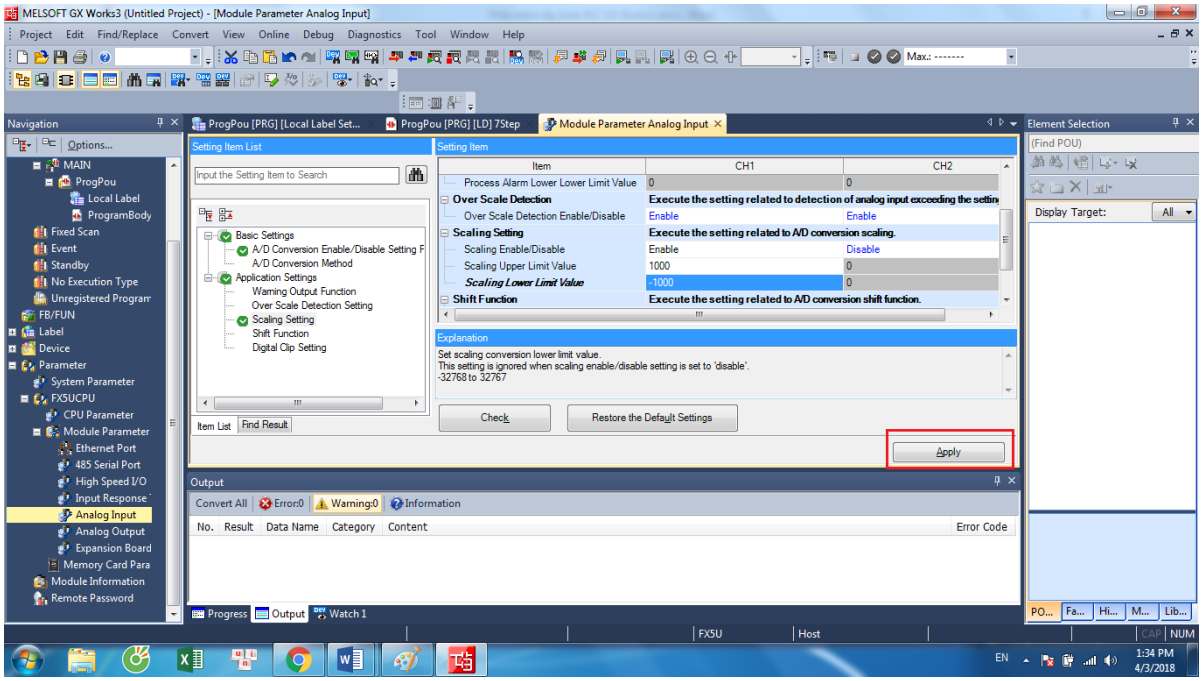


Để cài đặt Scaling, chọn Application Setting => Scaling Setting => *Scaling anable/disable* => enable.



Ta có thể cài đặt thông số giới hạn trên và dưới cho Scaling trong *Scaling Upper Limit Value* và *Scaling Lower Limit Value*.

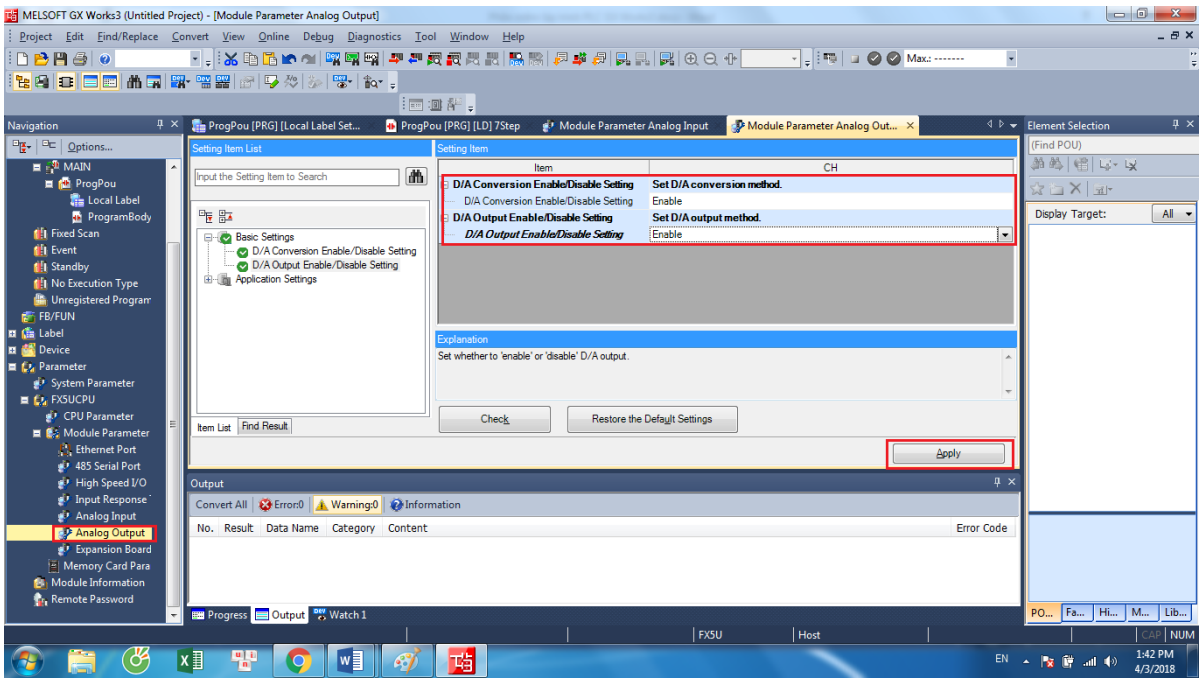
Click Apply để lưu cài đặt.



2.2 Analog Output

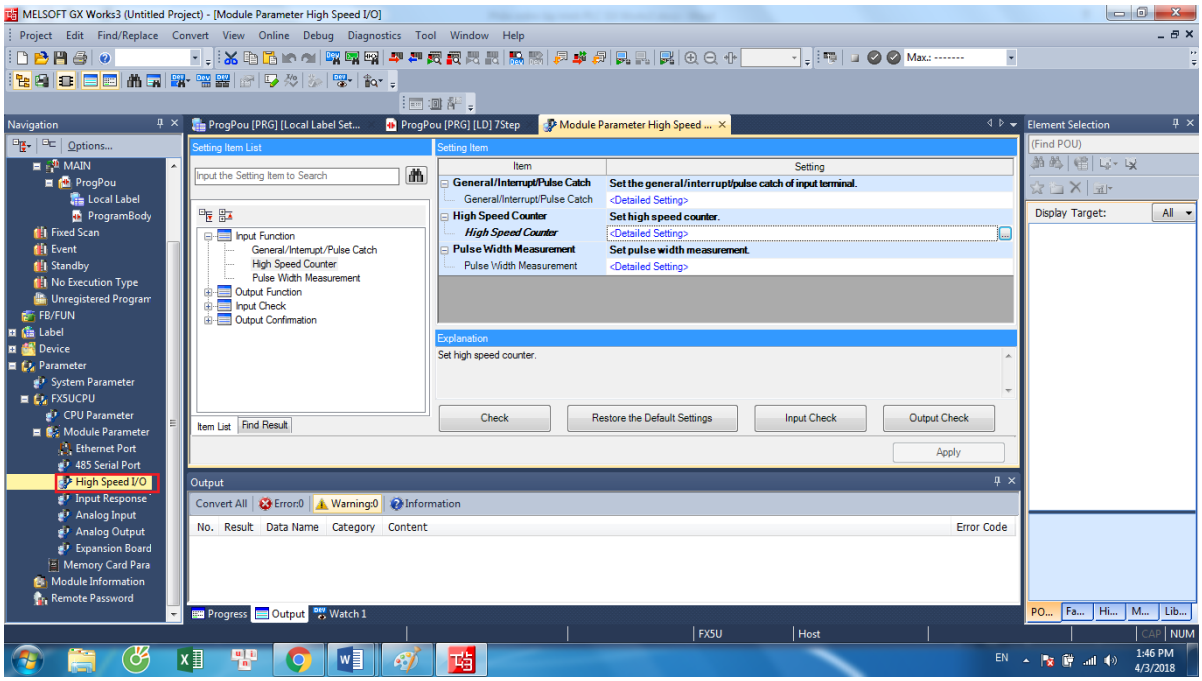
Trong cửa sổ Navigation chọn Parameter => FX5UCPU => Module Parameter và Click vào Analog Output.

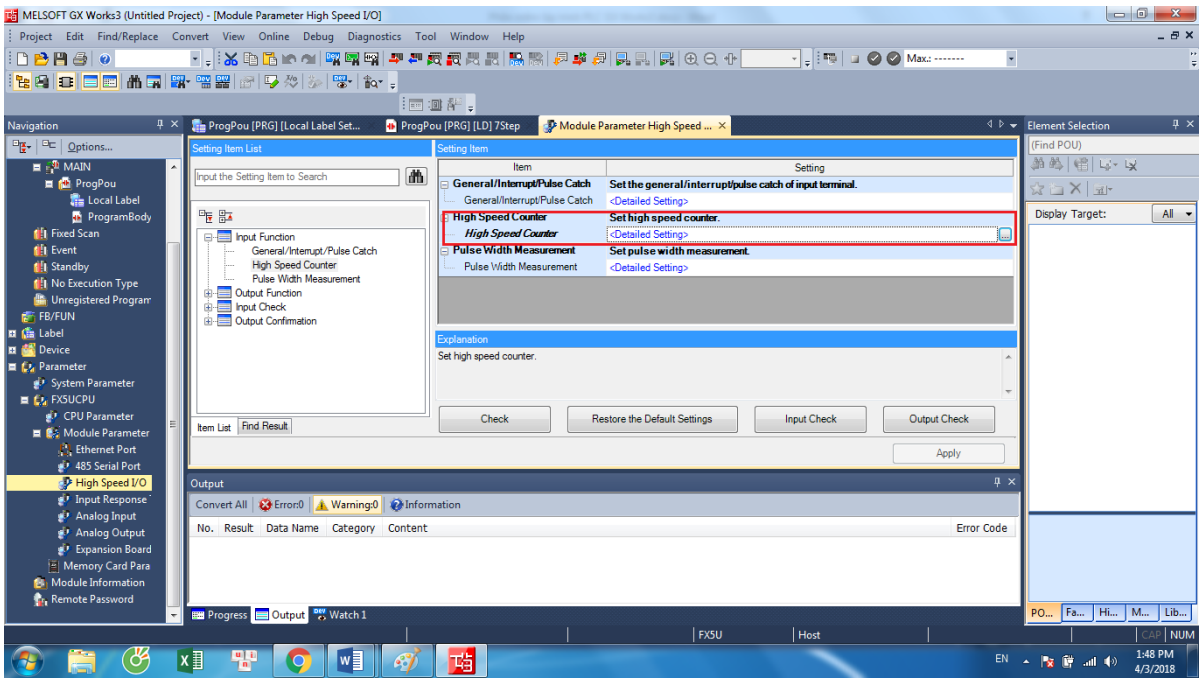
Cài đặt *D/A Conversion Enable/Disable Setting* => Enable và *D/A Output Enable/Disable* => Enable.Click Apply.



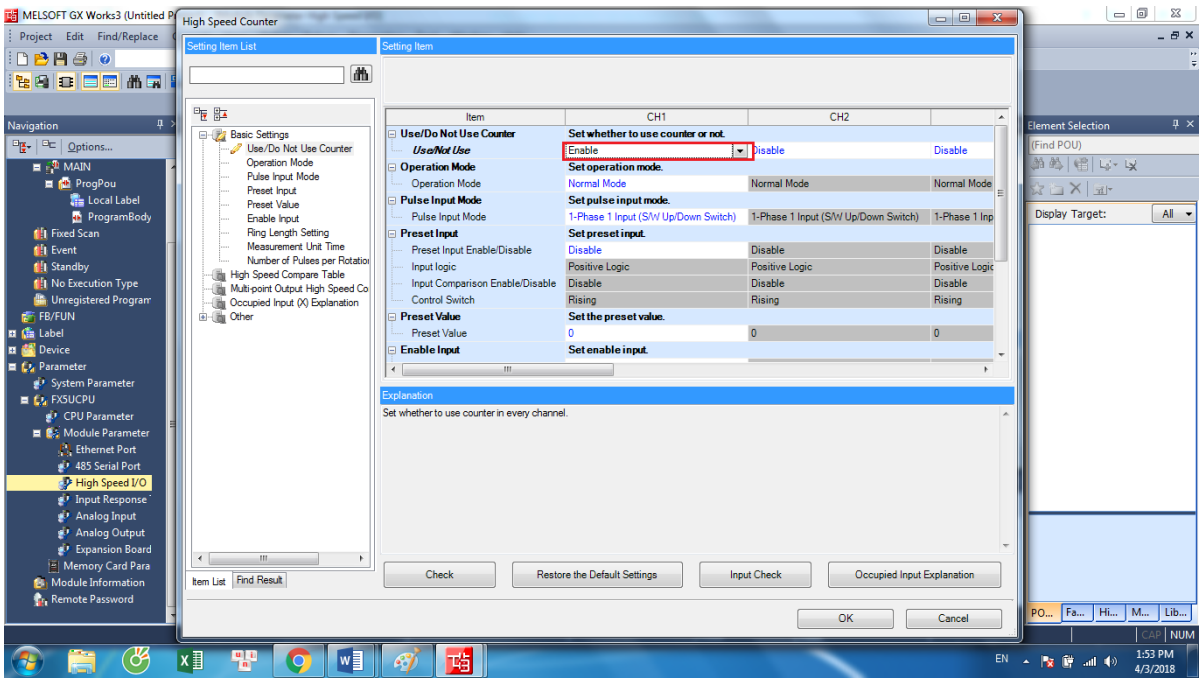
2.3 HighSpeed I/O

Trong cửa sổ Navigation chọn Parameter => FX5UCPU => Module Parameter và Click vào High Speed I/O.





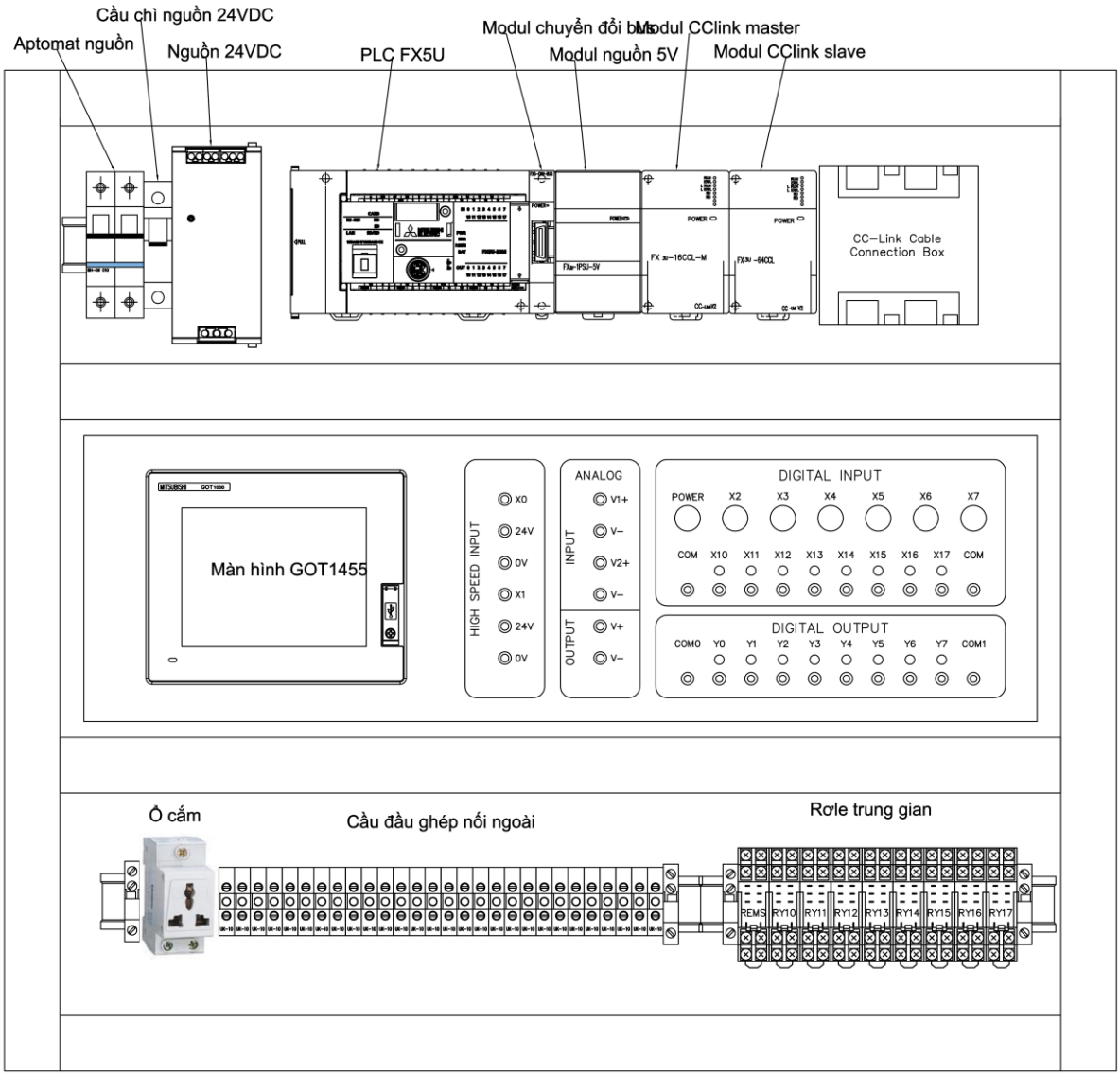
Click vào *High Speed Counter* và bảng cài đặt xuất hiện.Ta có thể cài đặt các chức năng chế độ cho High speed counter trong bảng này rồi Click Ok.



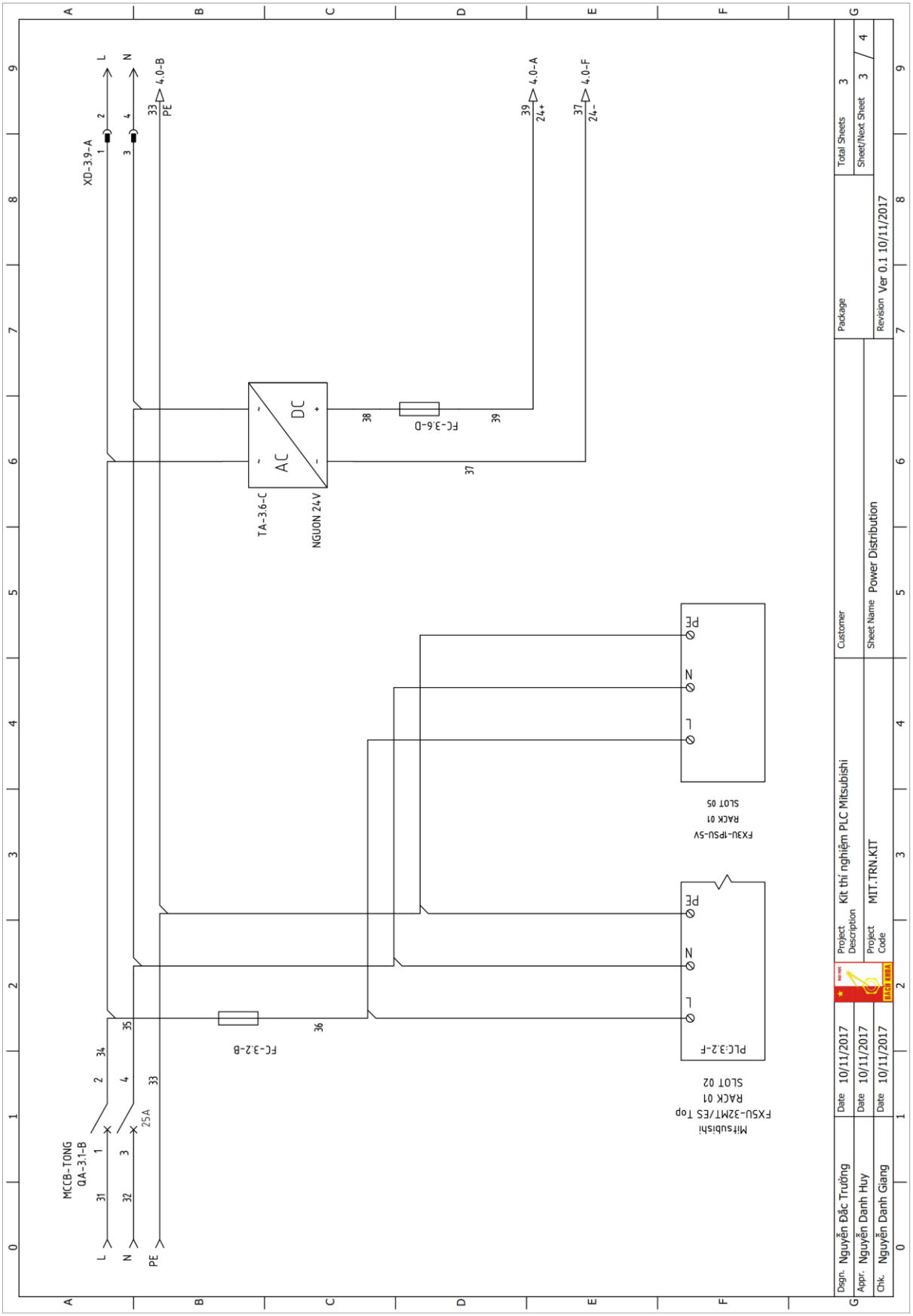
A. Các thiết bị:

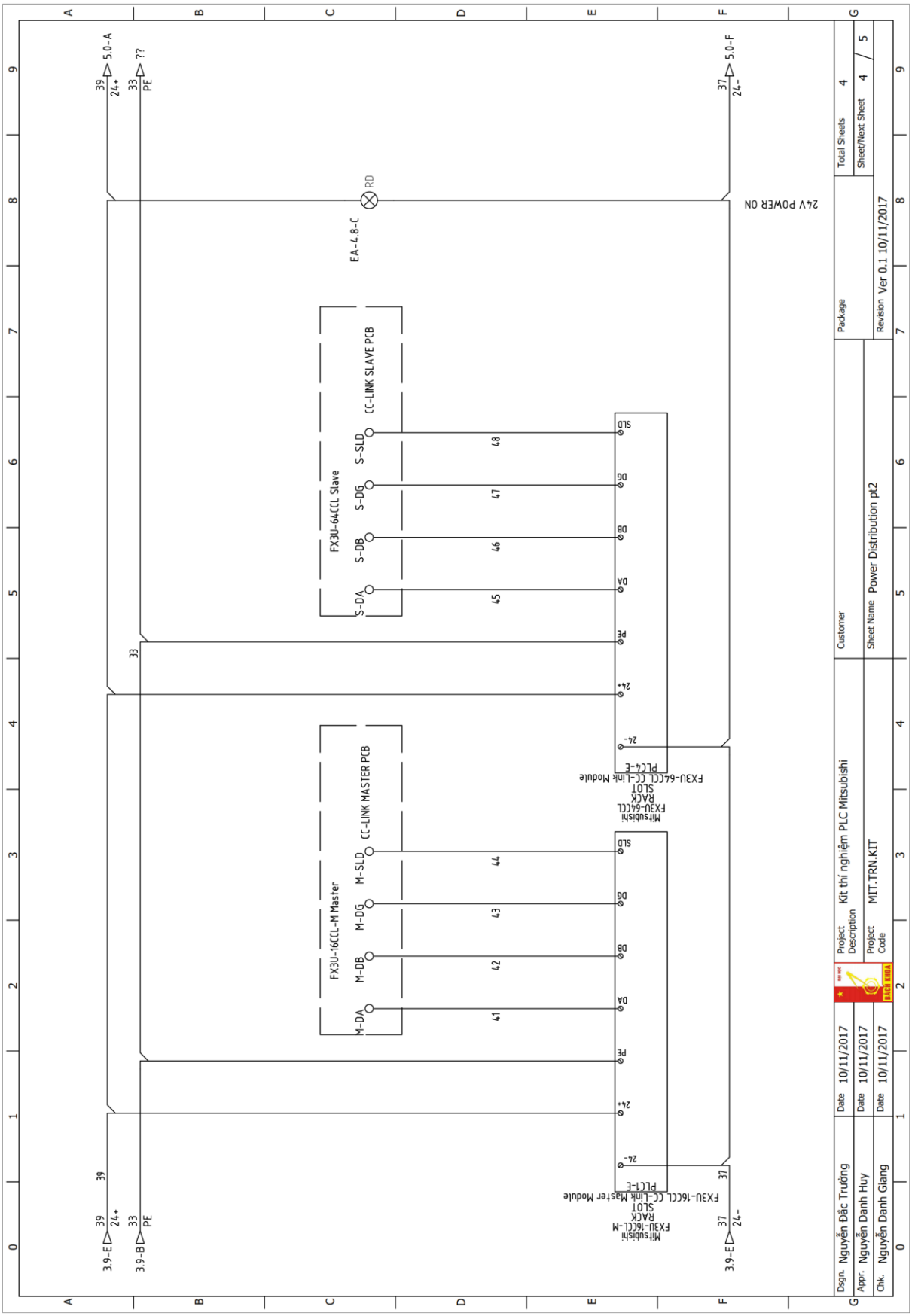
- Loại PLC: FX5U (Mitsubishi)
  - Số đầu vào: 16 đầu vào
  - Số đầu ra: 16 đầu ra
- Màn hình HMI: GOT1455 (Mitsubishi)

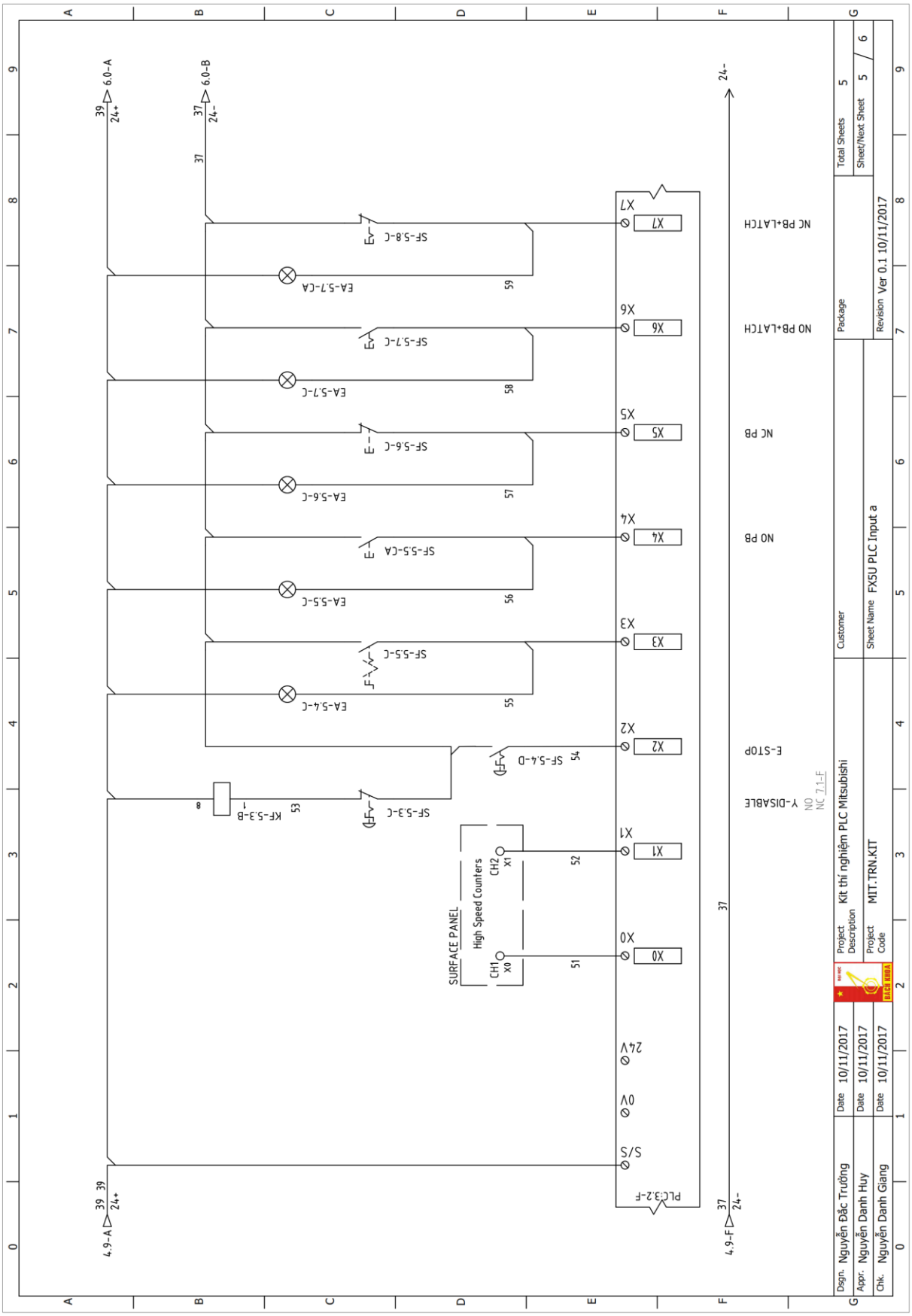
B. Mô tả thiết bị:

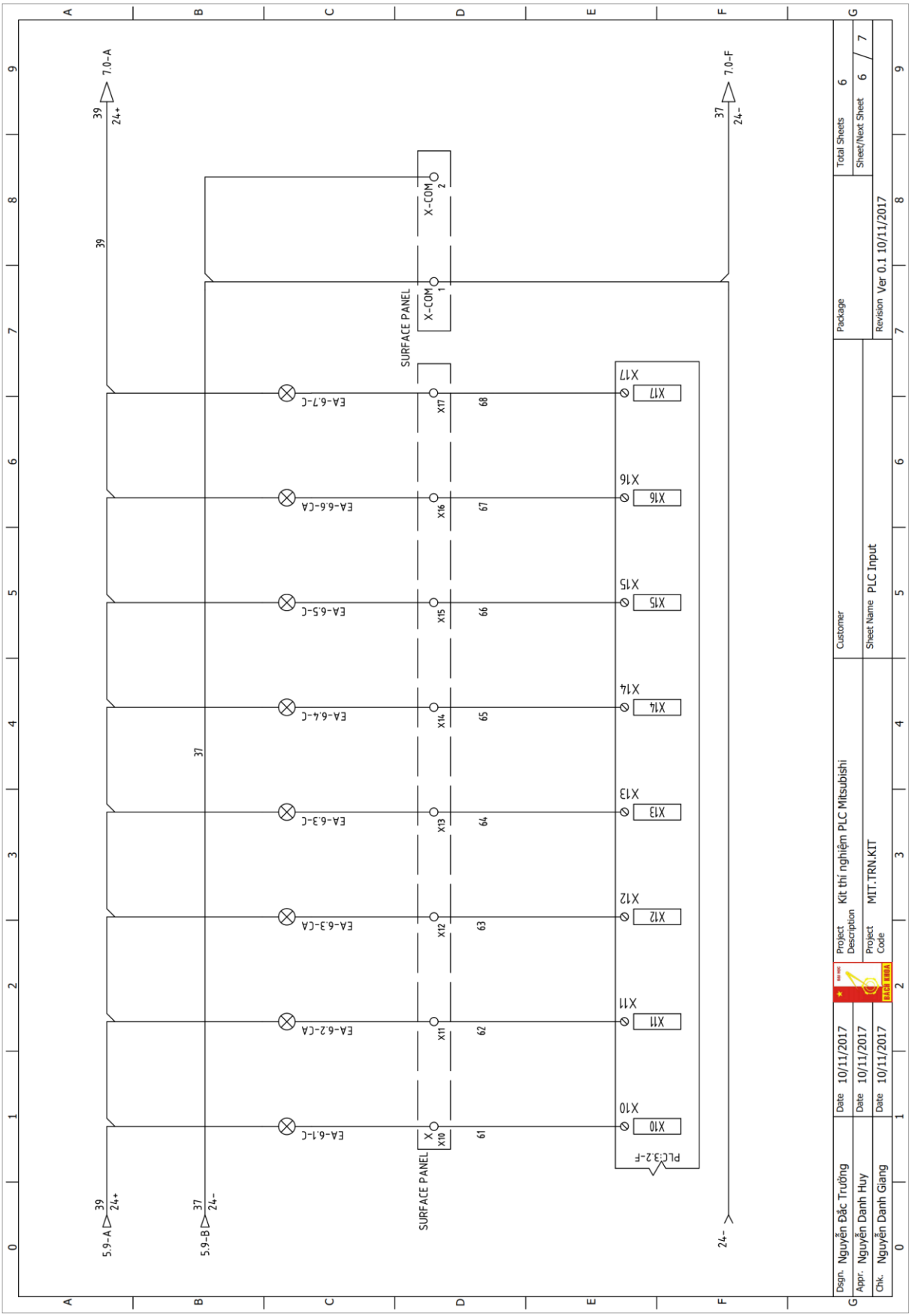


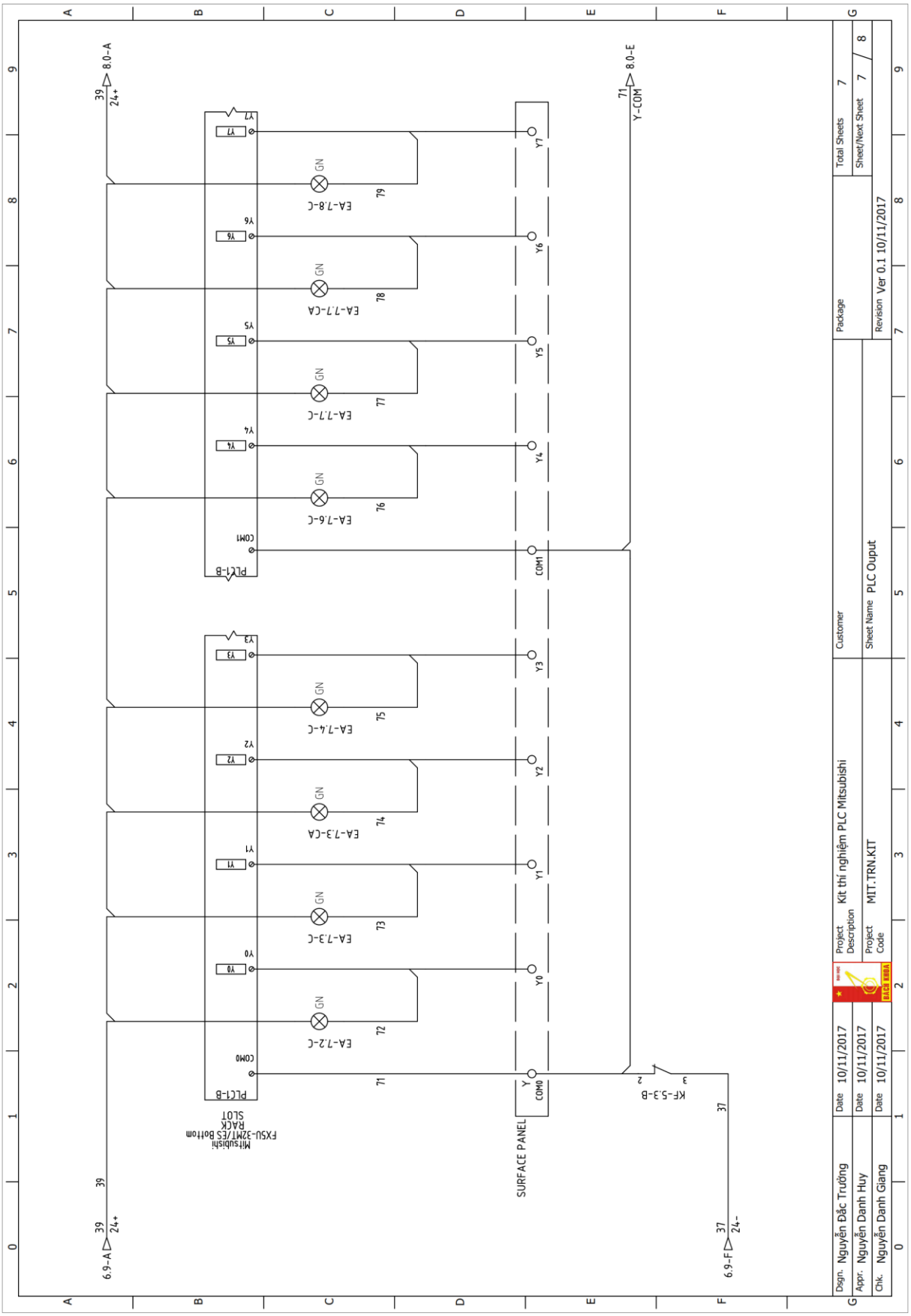
- Aptomat nguồn: dùng để bảo vệ và bật tắt nguồn cấp cho bàn thí nghiệm:
- Cầu chì nguồn 24V: bảo vệ nguồn 24V một chiều của bàn thí nghiệm
- Nguồn 24VDC: nguồn 24VDC cấp nguồn cho màn hình và các thiết bị ngoại vi.
- Ổ cắm: dùng để cắm các thiết bị khác khi làm việc với bàn thí nghiệm như máy tính, máy đo …
- Cầu đầu ghép nối ngoài: một số tín hiệu được đấu sẵn ra cầu đấu ngoài để ghép nối với các thiế bị khác như cảm biến, biến tần …
- Rơ le trung gian: gồm các rơ le ghép nối với vàog /ra của PLC để từ PLC trên bàn thì nghiệm có thể điều khiển các thiết bị ngoài khác

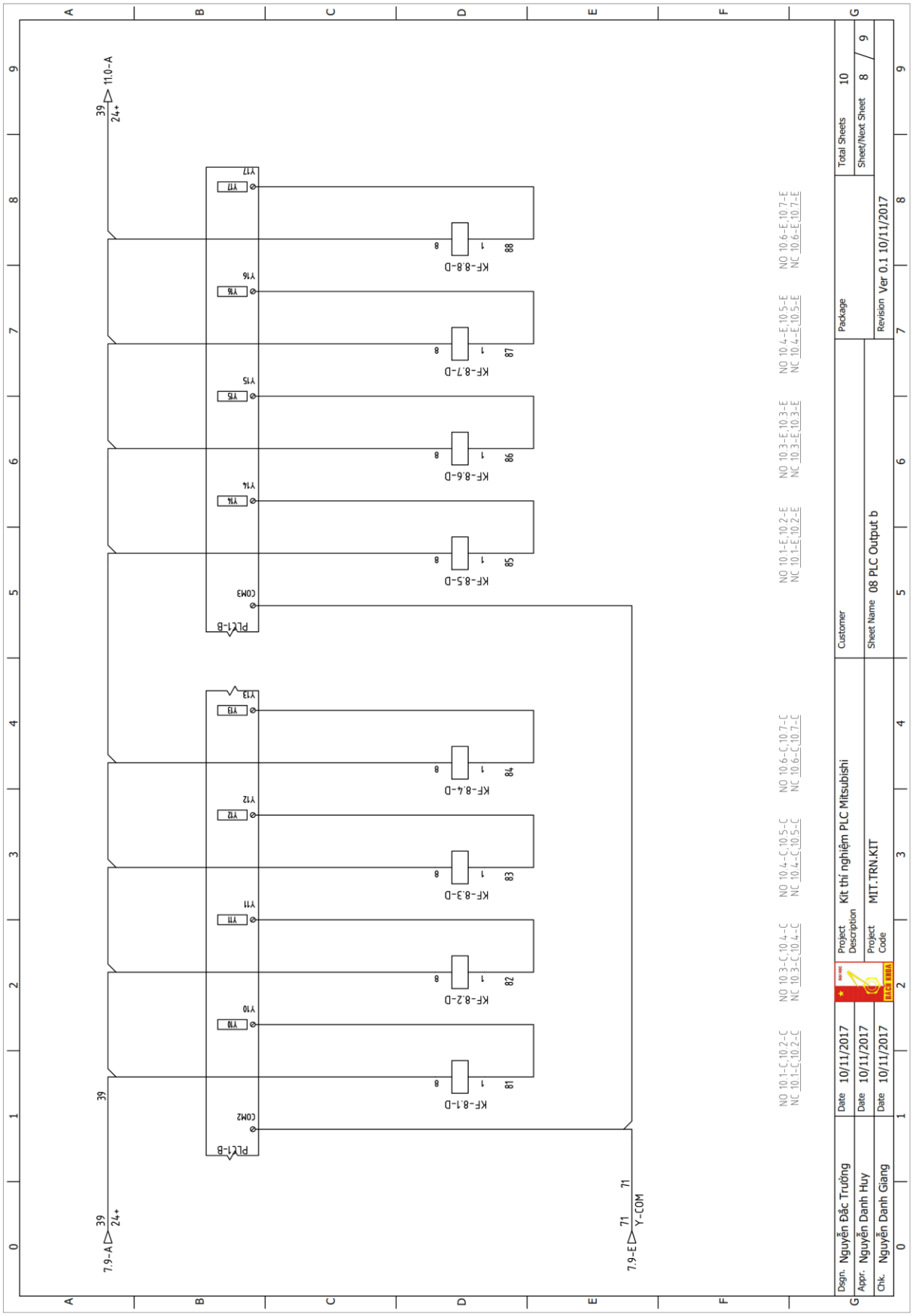

MCCB-TONG
QA-3.1-B
XD-3.9-A
4.0-B
4.0-A
4.0-F
TA-3.6-C
NGUON 24 V
FC-3.6-D
FC-3.2-B
PLC.3.2-F
FX3U-1PSU-5V
RACK 01
SLOT 05
Mitsubishi
FX5U-32MT/ES Top
RACK 01
SLOT 02
Kit thí nghiệm PLC Mitsubishi
MIT.TRN.KIT
Power Distribution
Nguyễn Đắc Trường
Nguyễn Danh Huy
Nguyễn Danh Giang
10/11/2017
Ver 0.1 10/11/2017

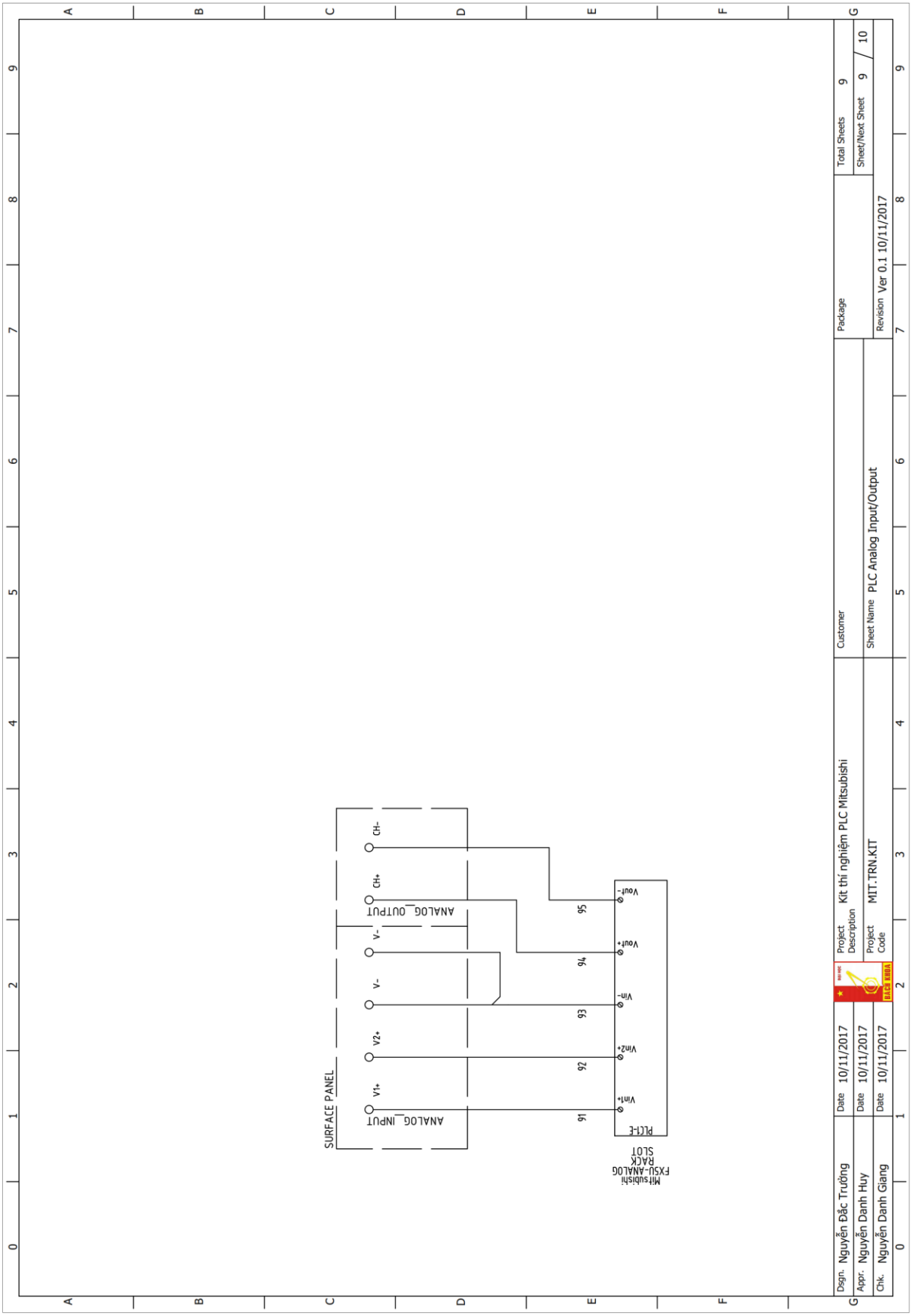

24V POWER ON
EA-4.8-C
RD
FX3U-64CCL Slave
CC-LINK SLAVE PCB
S-SLD
S-DG
S-DB
S-DA
FX3U-16CCL-M Master
CC-LINK MASTER PCB
M-SLD
M-DG
M-DB
M-DA
FX3U-64CCL CC-Link Module
FX3U-16CCL CC-Link Master Module
Mitsubishi
FX3U-64CCL
RACK
SLOT
PLC4-E
Mitsubishi
FX3U-16CCL-M
RACK
SLOT
PLC1-E
Dsgn. Nguyễn Đắc Trường
Date 10/11/2017
Appr. Nguyễn Danh Huy
Date 10/11/2017
Chk. Nguyễn Danh Giang
Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Package
Sheet Name Power Distribution pt2
Total Sheets 4
Sheet/Next Sheet 4 / 5
Revision Ver 0.1 10/11/2017

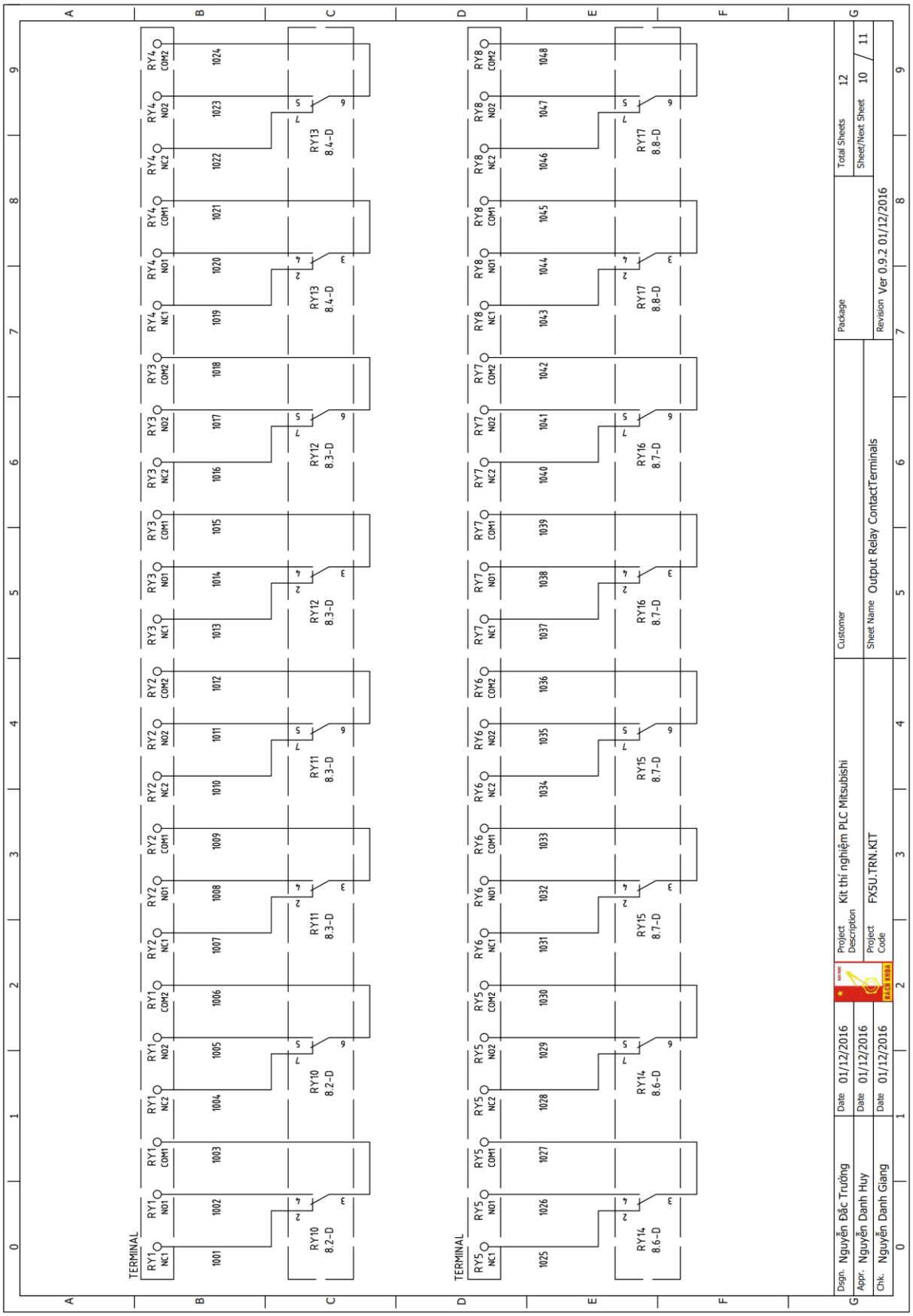

Kit thí nghiệm PLC Mitsubishi
MIT.TRN.KIT
FX5U PLC Input a
Ver 0.1 10/11/2017
Nguyễn Đắc Trường
Nguyễn Danh Huy
Nguyễn Danh Giang
SURFACE PANEL
High Speed Counters
E-STOP
Y-DISABLE
NC PB+LATCH
NO PB+LATCH
NC PB
NO PB

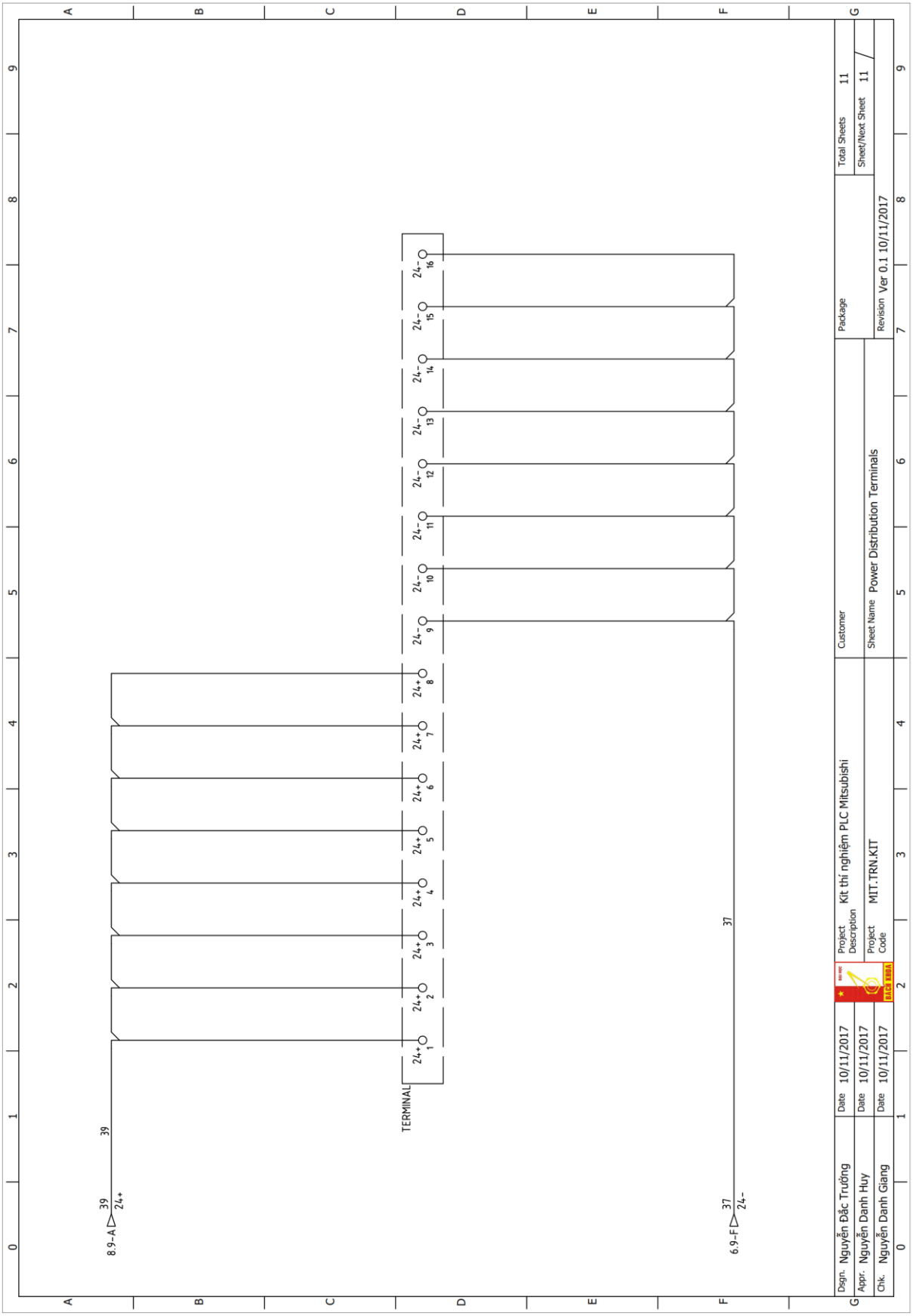

7.0-A
39
24+
7.0-F
37
24-
SURFACE PANEL
X-COM
2
X-COM
1
X17
X16
X15
X14
X13
X12
X11
X10
EA-6.7-C
EA-6.6-CA
EA-6.5-C
EA-6.4-C
EA-6.3-C
EA-6.3-CA
EA-6.2-CA
EA-6.1-C
68
67
66
65
64
63
62
61
PLC:3.2-F
5.9-A
5.9-B
Dsgn. Nguyễn Đắc Trường
Appr. Nguyễn Danh Huy
Chk. Nguyễn Danh Giang
Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Package
Sheet Name PLC Input
Revision Ver 0.1 10/11/2017
Total Sheets 6
Sheet/Next Sheet 6 / 7


Dsgn. Nguyễn Đắc Trường
Appr. Nguyễn Danh Huy
Chk. Nguyễn Danh Giang
Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Sheet Name PLC Ouput
Package
Revision Ver 0.1 10/11/2017
Total Sheets 7
Sheet/Next Sheet 7 / 8
SURFACE PANEL
PLC1-B
Mitsubishi FX5U-32MT/ES Bottom RACK SLOT
KF-5.3-B
EA-7.8-C
EA-7.7-CA
EA-7.7-C
EA-7.6-C
EA-7.4-C
EA-7.3-CA
EA-7.3-C
EA-7.2-C
8.0-A
8.0-E
6.9-A
6.9-F
Y-COM
24+
24-


39
24+
11.0-A
Y17
Y16
Y15
Y14
COM3
PLC1-B
Y13
Y12
Y11
Y10
COM2
PLC1-B
39
24+
7.9-A
71
Y-COM
7.9-E
KF-8.8-D
KF-8.7-D
KF-8.6-D
KF-8.5-D
KF-8.4-D
KF-8.3-D
KF-8.2-D
KF-8.1-D
NO 10.6-E,10.7-E
NC 10.6-E,10.7-E
NO 10.4-E,10.5-E
NC 10.4-E,10.5-E
NO 10.3-E,10.3-E
NC 10.3-E,10.3-E
NO 10.1-E,10.2-E
NC 10.1-E,10.2-E
NO 10.6-C,10.7-C
NC 10.6-C,10.7-C
NO 10.4-C,10.5-C
NC 10.4-C,10.5-C
NO 10.3-C,10.4-C
NC 10.3-C,10.4-C
NO 10.1-C,10.2-C
NC 10.1-C,10.2-C
Dsgn. Nguyễn Đắc Trường
Date 10/11/2017
Appr. Nguyễn Danh Huy
Date 10/11/2017
Chk. Nguyễn Danh Giang
Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Package
Sheet Name 08 PLC Output b
Revision Ver 0.1 10/11/2017
Total Sheets 10
Sheet/Next Sheet 8 / 9


SURFACE PANEL
ANALOG_INPUT
V1+
V2+
V-
V+-
ANALOG_OUTPUT
CH+
CH-
Mitsubishi
FX5U-ANALOG
RACK
SLOT
PLC1-F
Vin1+
Vin2+
Vin-
Vout+
Vout-
91
92
93
94
95
Dsgn. Nguyễn Đắc Trường
Date 10/11/2017
Appr. Nguyễn Danh Huy
Date 10/11/2017
Chk. Nguyễn Danh Giang
Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Package
Revision Ver 0.1 10/11/2017
Sheet Name PLC Analog Input/Output
Total Sheets 9
Sheet/Next Sheet 9 / 10


TERMINAL
RY1 NC1
RY1 NO1
RY1 COM1
RY1 NC2
RY1 NO2
RY1 COM2
RY2 NC1
RY2 NO1
RY2 COM1
RY2 NC2
RY2 NO2
RY2 COM2
RY3 NC1
RY3 NO1
RY3 COM1
RY3 NC2
RY3 NO2
RY3 COM2
RY4 NC1
RY4 NO1
RY4 COM1
RY4 NC2
RY4 NO2
RY4 COM2
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
RY10 8.2-D
RY11 8.3-D
RY12 8.3-D
RY13 8.4-D
TERMINAL
RY5 NC1
RY5 NO1
RY5 COM1
RY5 NC2
RY5 NO2
RY5 COM2
RY6 NC1
RY6 NO1
RY6 COM1
RY6 NC2
RY6 NO2
RY6 COM2
RY7 NC1
RY7 NO1
RY7 COM1
RY7 NC2
RY7 NO2
RY7 COM2
RY8 NC1
RY8 NO1
RY8 COM1
RY8 NC2
RY8 NO2
RY8 COM2
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
RY14 8.6-D
RY15 8.7-D
RY16 8.7-D
RY17 8.8-D
Dsgn. Nguyễn Đắc Trường
Date 01/12/2016
Appr. Nguyễn Danh Huy
Date 01/12/2016
Chk. Nguyễn Danh Giang
Date 01/12/2016
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code FX5U.TRN.KIT
Customer
Package
Sheet Name Output Relay ContactTerminals
Revision Ver 0.9.2 01/12/2016
Total Sheets 12
Sheet/Next Sheet 10 / 11


TERMINAL
24+ 1
24+ 2
24+ 3
24+ 4
24+ 5
24+ 6
24+ 7
24+ 8
24- 9
24- 10
24- 11
24- 12
24- 13
24- 14
24- 15
24- 16
39
8.9-A
24+
39
37
6.9-F
24-
37
Dsgn. Nguyễn Đắc Trường Date 10/11/2017
Appr. Nguyễn Danh Huy Date 10/11/2017
Chk. Nguyễn Danh Giang Date 10/11/2017
Project Description Kit thí nghiệm PLC Mitsubishi
Project Code MIT.TRN.KIT
Customer
Package
Sheet Name Power Distribution Terminals
Revision Ver 0.1 10/11/2017
Total Sheets 11
Sheet/Next Sheet 11

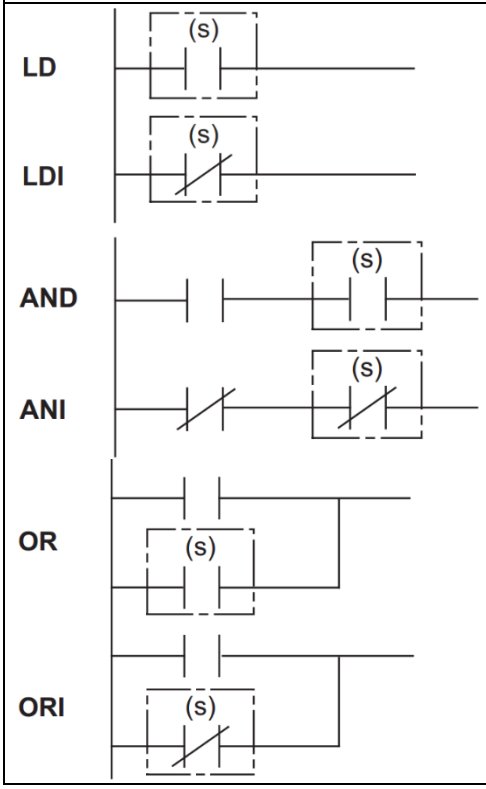
Chương 3. HỆ LỆNH PLC FX5U

3.1. Nhóm lệnh cho CPU

3.1.1. Lệnh tuần tự - Sequence Instruction

1. Lệnh khởi tạo, kết nối nối tiếp, kết nối song song

Tên lệnh: LD, LDI, AND, ANI, OR, ORI

Ladder	ST	FBD
		Như Ladder

Dữ liệu lệnh:

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(s)	Được dùng như một tiếp điểm	__	Bit	

- Kiểu toán hạng:

Toán hạng	Bit	Word		Double word		Hằng số	
	X, Y, M, L, SM, F, B, SB, S	**T, ST, C, D, W, SD, SW, R**	**Z**	**LC**	**LZ**	**K, H**	**E**
(s)							

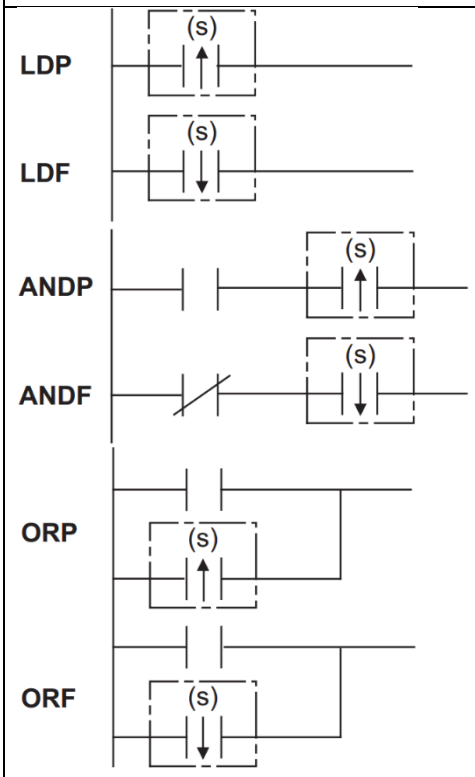
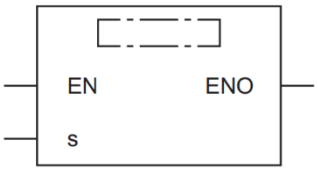
- **LD:** Lệnh khởi tạo giá trị logic loại tiếp điểm thường hở NO;
- **LDI**: Lệnh khởi tạo giá trị logic loại tiếp điểm thường đóng NC;

Những lệnh này sẽ bắt theo trạng tái ON/OFF của tiếp điểm và sử dụng kết quả này như kết quả hoạt để phục vụ cho các lệnh khác.

- **AND:** Lệnh nối tiếp với tiếp điểm thường hở NO. Sử dụng kết quả này như kết quả hoạt động.
- **ANDI**: Lệnh nối tiếp với tiếp diểm thường đóng. Sử dụng kết quả này như kết quả hoạt động.
- **OR:** Lệnh kết nối song song với tiếp điểm thường hở NO. Sử dụng kết quả này như kết quả hoạt động.
- **ORI**: Lệnh kết nối song song tiếp điểm thường đóng NC. Sử dụng kết quả này như kết quả hoạt động.

2. Lệnh khởi tạo xung, kết nối nối tiếp xung, kết nối song song xung.

Tên lệnh: LDP, LDF, ANDP, ANDF, ORP, ORF

Ladder	ST	FBD
	ENO:=LDP(EN,s); ENO:=LDF(EN,s); ENO:=ANDP(EN,s); ENO:=ANDF(EN,s); ENO:=ORP(EN,s); ENO:=ORF(EN,s);	

LDP: Lệnh khởi tạo bit logic bắt xung cạnh lên. Chỉ dẫn ngay khi có cạnh lên của thiết bị bit (s). Khi các thiết bị Word được xác định bởi các bit con, tiếp điểm này chỉ dẫn ngay khi trang thái của bit thay đổi thừ 0 → 1.

LDF: Lệnh khởi tạo bit logic bắt xung cạnh lên. Chỉ dẫn ngay khi có cạnh lên của thiết bị bit (s). Khi các thiết bị Word được xác định bởi các bit con, tiếp điểm này chỉ dẫn ngay khi trang thái của thiết bị bit thay đổi thừ 1 → 0.

ANDP: Lệnh nối tiếp với tiếp điểm bắt xung cạnh lên LDP xác định bởi thiết bị bit (s).

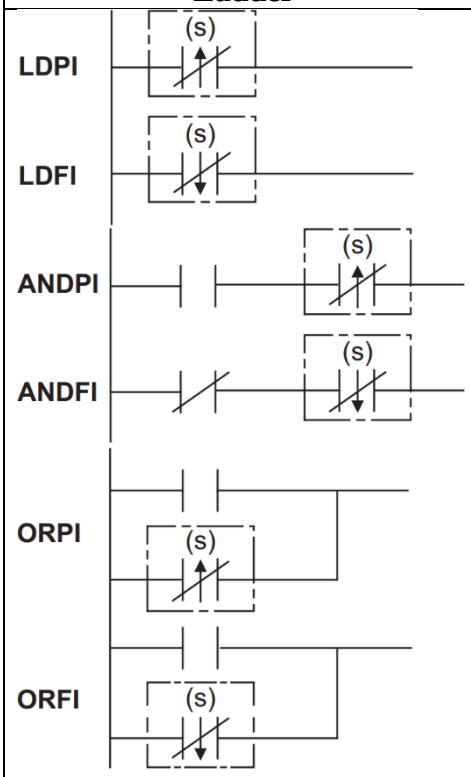
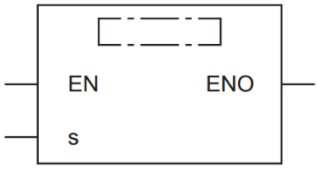
ANDF: Lệnh nối tiếp với tiếp điểm bắt xung cạnh xuống LDF xác định bởi thiết bị bit (s)

ORP: Lệnh kết nối song song với tiếp điểm bắt xung cạnh lên LDP xác định bởi thiết bị bit (s)

ORF: Lệnh kết nối song song với tiếp điểm bắt xung cạnh xuống LDF xác định bởi thiết bị bit (s)

3. Lệnh khởi tạo xung NOT, kết nối nối tiếp với xung NOT, kết nối song song với xung NOT

Tên lệnh: LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI

Ladder	ST	FBD
	ENO:=LDPI(EN,s); ENO:=LDFI(EN,s); ENO:=ANDPI(EN,s); ENO:=ANDFI(EN,s); ENO:=ORPI(EN,s); ENO:=ORFI(EN,s);	

LDPI: Lệnh không dẫn khi có xung cạnh lên. Tiếp điểm chỉ ngưng dẫn ngay khi có sự thay đổi trạng thái từ 0 → 1 của thiết bị bit (s)

LDFI: Lệnh không dẫn khi có xung cạnh xuống. Tiếp điểm chỉ ngưng dẫn ngay khi có sự thay đổi trạng thái từ 1 → 0 của thiết bị bit (s)

ANDPI: Lệnh kết nối nối tiếp với tiếp điểm LDPI

ANDFI: Lệnh kết nối nối tiếp với tiếp điểm LDFI

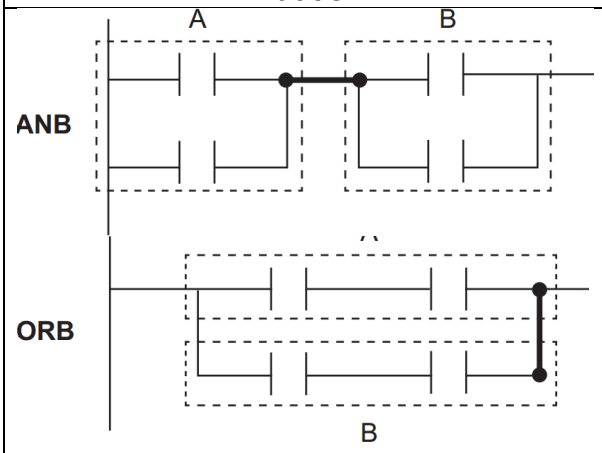
ORPI: Lệnh kết nối song song với tiếp điểm LDPI

ORFI: Lệnh kết nối nối tiếp với tiếp điểm LDFI

4. Lệnh nối nối tiếp/song song khối lệnh Ladder

Tên lệnh: **ANB, ORB**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
		

ANB: Lệnh AND hai khối A và B với nhau

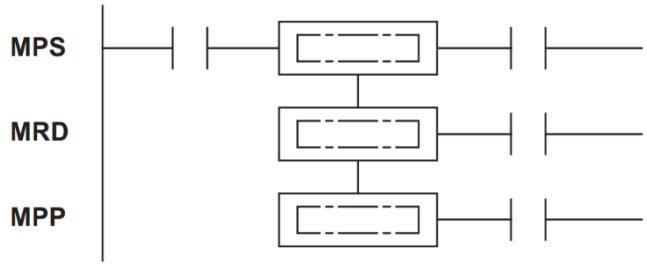
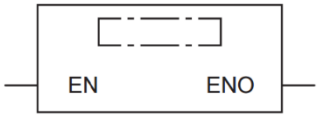
Ký hiệu của lệnh này không phải là ký hiệu NO mà là biểu tượng kết nối.

ORB: Lệnh OR hai khối A và B với nhau

5. Lệnh rẽ nhánh

Tên lệnh: **MPS, MRD, MPP**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=MPS(EN); ENO:=MRD(EN); ENO:=MPP(EN);	

MPS: Lưu trữ kết quả hiện hành của các hoạt động bên trong PLC.Lệnh này có thể sử dụng tối đa 16 lần lien tiếp.Khi lệnh MPP được sử dụng ở giữa, số lệnh có thể sử dụng của lệnh MPS bị giảm đi 1.

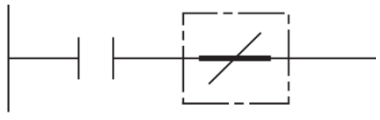
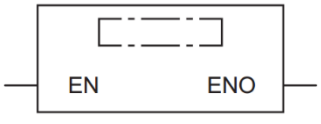
MRD: đọc kết quả hiện hành của các hoạt động bên trong PLC.

MPP: lấy lại và xóa kết quả lưu trữ hiện hành.

6. Lệnh nghịch đảo kết quả

Tên lệnh: **INV**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=INV(EN);	

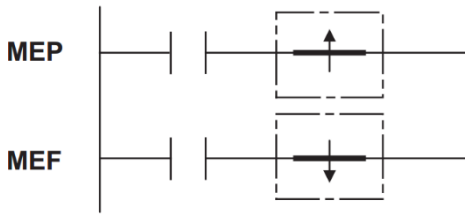
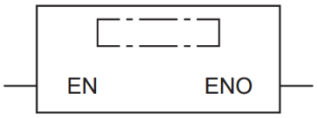
Lệnh này cho phép nghịch đảo kết quả hoạt động của phần lệnh phía trước.

Dữ liệu đưa vào lệnh INV	Dữ liệu đi ra lệnh INV
ON	OFF
OFF	ON

7. Biến đổi kết quả hoạt động thành xung

Tên lệnh: **MEP, MEF**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=MEP(EN); ENO:=MEF(EN);	

MEP: Lệnh này ON khi có cạnh lên của giá trị hiện hành và OFF trong các trường hợp khác

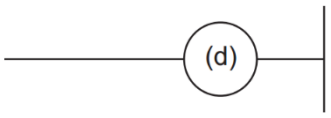
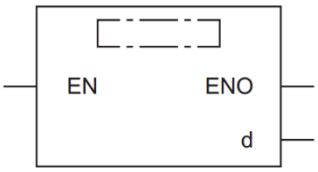
Sử dụng lệnh này để ngịch đảo xung dễ dàng khi nhiều tiếp điểm được nối với nhau.

MEF: Lệnh này ON khi có cạnh xuống của giá trị hiện hành và ON trong các trường hợp khác. ‘

8. Lệnh Out (ngoại trừ Timer, Counter và trình thông báo Annunciator)

Tên lệnh: **OUT**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=OUT(EN,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Số thiết bị bật tắt	__	Bit	ANY_BOOL
EN	Điều kiện thực thi	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Hoạt động

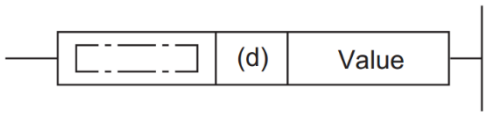
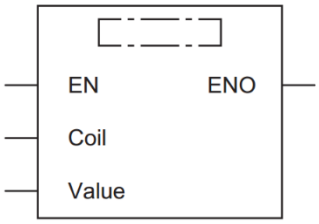
Lệnh này xuất giá trị hiện hành đến các thiết bị xác định

Điều kiện	**Giá trị hiện hành**	**Cuộn dây/ Bit xác định**
Khi thiết bị Bit được sử dụng	OFF	OFF
	ON	ON
Khi Bit của Word được sử dụng	OFF	0
	ON	1

9. Bộ định thời Timer (low-speed, high-speed, low-speed có nhớ, high-speed có nhớ)

Tên lệnh: **OUT T, OUTH T, OUTHS T, OUT ST, OUTH ST, OUTHS ST**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**	**FBD**
	ENO:=OUT_T(EN,Coil,Value); ENO:=OUTH(EN,Coil,Value); ENO:=OUTHS(EN,Coil,Value);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Tên Timer	__	Timer/ timer có nhớ	ANY
value (set value)	Thời gian đặt	0 to 32767	16-bit không dấu	
EN	Điều khiện thực thi	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Bộ đếm thời gian sẽ đếm tiến theo giá trị đặt (set value). Khi giá trị hiện hành đạt tới giá trị đặt thì cuộn dây của Timer/Timer có giữ (d) sẽ được bật lên. Khi đó, tiếp điểm thường mở NO sẽ dẫn và tiếp điểm thường đóng NC sẽ ngưng dẫn

Khi giá trị hiện hành của lệnh OUT sẽ thay đổi từ ON sang OFF, hoạt động của các tiếp điểm sẽ như bảng sau

Loại Timer	Timer coil	Giá trị hiện tại của Timer	Trước time-out		Sau timer-out	
			NO contact	NC contact	NO contact	NC contact
Timer	Off	0	Không dẫn	Dẫn	Không dẫn	Dẫn
Retentive có nhớ	off	Giữ giá trị hiện tại	Không dẫn	Dẫn	Dẫn	Không dẫn

Times-up: Lúc Timer có giá trị đếm bằng giá trị đặt.

Sau khi Timer Times-up, xóa giá trị hiện hành của Timer có giữ và tắt tiếp điểm bằng cách sử dụng lệnh reset RST.

Khi giá trị đặt là 0, Times-up xảy ra ngay khi lệnh OUT được thực thi.

Giá trị được sử dụng cho Timer có thể được cài đặt trong dải 1 → 32767. Vì các lệnh OUT, OUTH và OUTHS hoạt động tương ứng với các bộ đếm thời gian 100 ms, 10 ms và 1 ms, Các hằng số thời gian thực sẽ như sau:

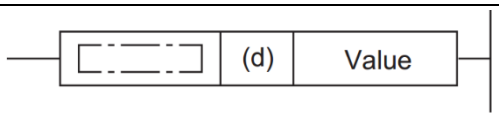
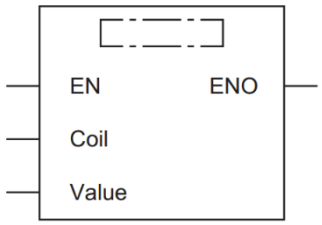
• OUT : 0.1 to 3276.7 seconds
• OUTH : 0.01 to 327.67 seconds
• OUTHS : 0.001 to 32.767 seconds

10. Bộ đếm Counter, Long Counter

a. Counter

Tên lệnh: **OUT C**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=OUT_C(EN,Coil,Value);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Timer số (tên Timer)	__	Counter	ANY
(set value)	Giá trị đặt	0 to 32767	16-bit không dấu	
EN	Điều kiện thực thi	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Lệnh **OUT C** tăng giá trị hiện tại của bộ đếm xác định bởi (d) lên 1 khi giá trị hiện hành theo lệnh OUT thay đổi từ OFF thành ON, và khi bộ đếm chạm đến giá trị đặt thì tiếp điểm NO sẽ dẫn và tiếp điểm NC sẽ ngưng dẫn.

Bộ đếm sẽ không đếm khi giá trị hiện hành đang được giữ ở trạng thái ON (Đầu vào đếm không nhất thiết phải biến đổi thành dạng xung).

Sau khi đếm đến giá trị đặt, giá trị đếm và trạng thái tiếp điểm sẽ không thay đổi cho đến khi lệnh RST được thực thi.

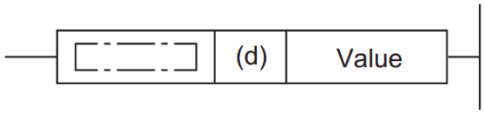
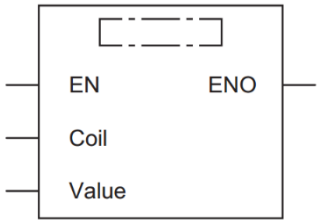
Khi đặt giá trị đặt là 0, xử lý tương tự như khi giá trị đặt 1 được thực hiện.

b. Long Counter:

Tên lệnh OUT LC. Lệnh tăng giá trị hiện tại của bộ đếm xác định bởi (d) lên 1 khi giá trị hiện hành theo lệnh OUT thay đổi từ OFF thành ON, và khi bộ đếm chạm đến giá trị đặt thì

tiếp điểm NO sẽ dẫn và tiếp điểm NC sẽ ngưng dẫn.Tương tự như Counter thông thường chỉ khác ở giá trị đếm theo kiểu 32bit, dải giá trị của LC là (-2147483648 to + 2147483647)

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=OUT_C(EN,Coil,Value);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Counter số (d)	__	Long counter	ANY
(set value)	Giá trị đặt	0 to 4294967295	32-bit unsigned binary	
EN	Điều kiện thực thi	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Hoạt động tương tự Counter thường.

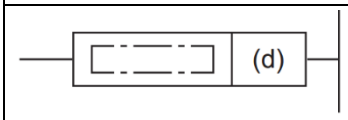
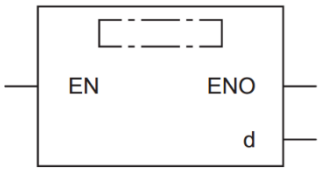
11. Lệnh SET thiết bị (ngoại trừ trình thông báo annunciator), RESET thiết bị (ngoại trừ trình thông báo)

Tên lệnh **SET.** Trạng thái của thiết bị bit (d) thay đổi như sau khi yêu cầu thực thi được ON

+ Bit: Chuyển các cuộn dây và tiếp điêm sang trạng thái ON

+ Bit của Word: đặt giá trị của bit lên 1.

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=SET(EN,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Bit hoặc Bit của Word được bật ON	__	Bit	ANY_BOOL
EN	Execution condition	__	Bit	BOOL
ENO	Execution result	__	Bit	BOOL

Hoạt động.

Trạng thái của thiết bị (d) thay đổi như sau khi yêu cầu thực thi được ON

Thiết bị	Trạng thái thiết bị
Bit	Bật cuộn dây và tiếp điểm ON
Bit của Word	Đặt giá trị bit là 1

Thiết bị khi đã được bật ON thì sẽ được giữ ON ngay cả khi yêu cầu thực thi chuyển sang OFF. Những thiết bị mà đã được bật bởi lệnh SET chỉ có thể OFF bằng lệnh RST.

Chú ý:

Khi lệnh SET và RST được thực thi ở cùng một đầu ra relay (Y), kết quả của lệnh gần lệnh END hơn sẽ là đầu ra của reley Y.

12. Lệnh **RESET** thiết bị (ngoại trừ trình thông báo)

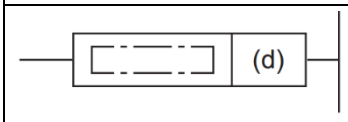
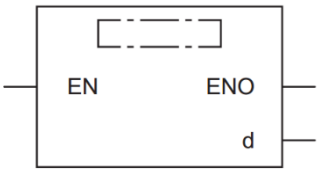
Tên lệnh: **RST.**

+ Thiết bị Bit: Chuyển cuộn dây và tiếp điểm sang trạng thái OFF
+ Bit của Word: Đặt giá trị 0 cho bit đó

Trạng thái của thiết bị (d) thay đổi như sau khi lệnh RST được bật ON:

+ Thiết bị Bit: Cuộn dây và tiếp điểm chuyển sang OFF
+ Timers, Counters: Đặt lại giá trị hiện thời về 0, cuộn dây và tiếp điểm chuyển sang trạng thái OFF
+ Bit trong Word: Đặt giá trị bit là 0
+ Thiết bị Word, thiết bị truy cập module, thanh ghi chỉ số: đặt nội dung là 0.

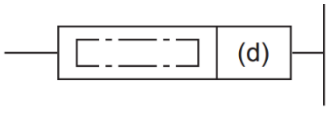
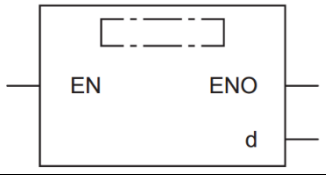
Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=RST(EN,d);	

13. Lệnh xuất kết quả khi có sườn lên/sườn xuống

Xuất kết quả khi có canh lên. Tên lệnh: **PLS**

Biểu diễn trong các ngôn ngữ lập trình

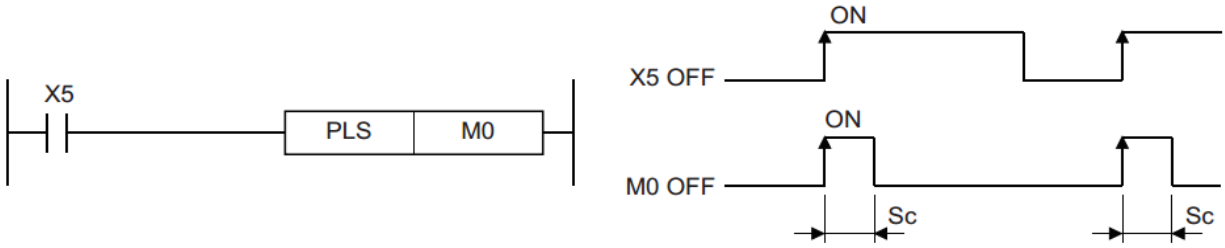
Ladder	ST	FBD
	ENO:=PLS(EN,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị được chuyển đổi sang xung	__	Bit	ANY_BOOL
EN	Điều kiện thực hti	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Hoạt động

Lệnh này bật ON thiết bị trong 1 chu kì khi yêu cầu thực thi lênh PLS được bật từ OFF lên ON và chuyển sang tắt trong các trường hợp còn lại. Khi có 1 lệnh PLS được lập trình cho thiết bị (d) trong suốt 1 chu kỳ quét, thiết bị sẽ bật ON trong 1 chu kì.

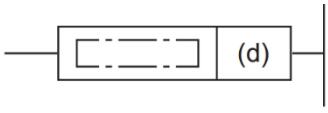
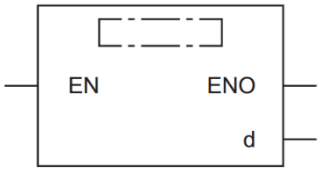


Chú ý:

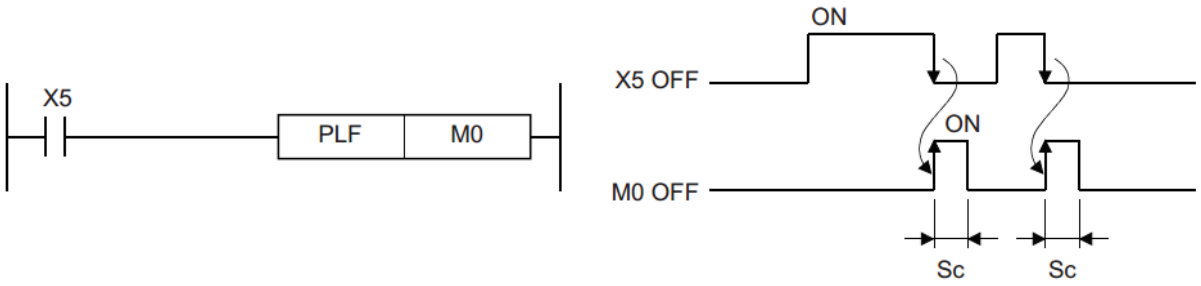
+ Khi ghi trong suốt quá trình RUN được hoàn thành cho mạch bao gồm lệnh xung cạnh lên (lệnh LDP/ANDP/ORP), lệnh này sẽ không thực thi bất kể trạng thái của thiết bị mục tiêu của lệnh xung cạnh lênh. Tương tự, trong trường hợp lệnh cạnh lên (PLS), lệnh này sẽ không thực thi bất kể trạng thái ON/OFF của thiết bị được đặt như là điều kiện thực thi. Lệnh này sẽ được thực thi chỉ khi thiết bị mục tiêu và thiết bị trong điều kiện thực thi được đặt từ OFF sang ON trở lại.
+ Chú ý rằng, thiết bị (d) nhiều khi bật ON trong 1 hoặc nhiều chu kì khi lênh PLS được tạo ra để nhảy bởi lệnh CJ hoặc chương tình con thực thi không được gọi bởi lệnh CALL(P)

Xuất kết quả khi có cạnh xuống. Tên lệnh: **PLF**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**	**FBD**
	ENO:=PLF(EN,d);	

Hoạt động. Lệnh này chỉ bật ON thiết bị trong 1 chu kỳ quét khi yêu cầu thực thi lệnh PLF được thay đổi từ OFF sang ON và tắt trong các trường hợp còn lại. Khi có 1 lệnh PLF được lập tình cho thiết bị (d) trong suốt chu kỳ quét, thiết bị bật ON trong 1 chu kỳ.

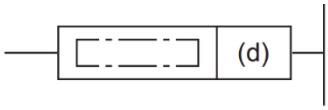
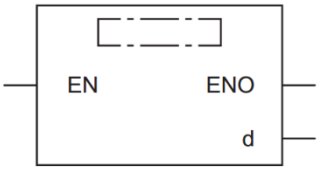


Chú ý: Tương tự như chú ý lệnh PLS

14. Lệnh nghịch đảo đầu ra của thiết bị Bit.

Tên lệnh: **FF**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**	**FBD**
	ENO:=FF(EN,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị được đảo trạng thái	__	Bit	ANY_BOOL
EN	Điều kiện thực thi	__	Bit	BOOL
ENO	Kết quả thực thi	__	Bit	BOOL

Hoạt động

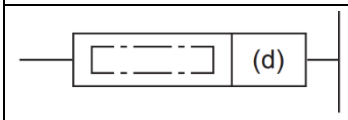
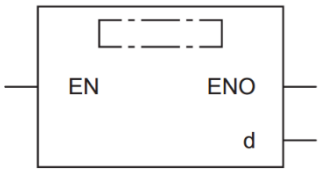
• Lệnh này đảo trạng thái đầu ra của thiết bị (d) khi yêu cầu thực thi thay đổi từ OFF sang ON

Thiết bị	Trạng thái thiết bị	
	Trước khi thực hiện lệnh FF	**Sau khi thực hiện lệnh FF**
Thiết bị Bit	OFF	ON
	ON	OFF
Bit trong Word	0	1
	1	0

15. Lệnh dịch Bit

Tên lệnh: **SFT(P)**

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=SFT(EN,d); ENO:=SFTP(EN,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị nhận sự dịch	__	Bit	

Hoạt động:

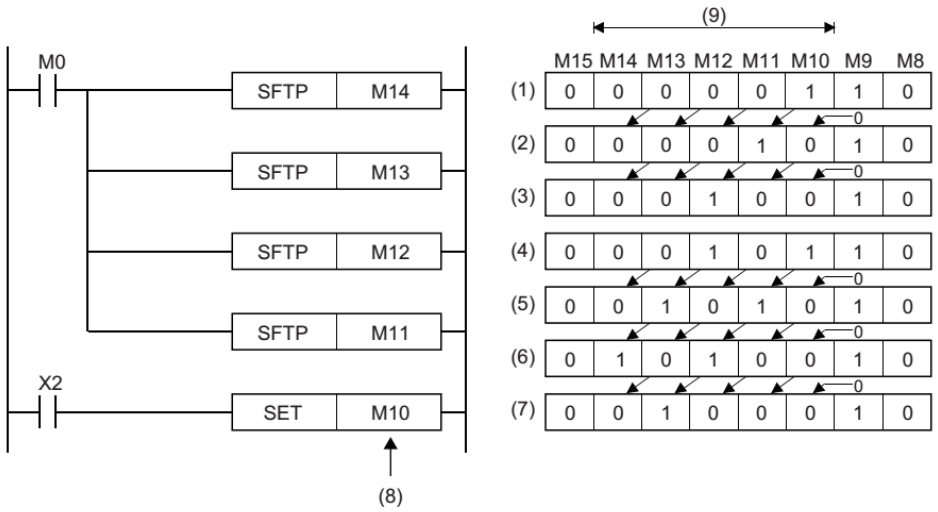
+ Trường hợp thiết bị Bit:

Lệnh này dịch trạng thái ON/OFF của thiết bị Bit nằm trước thiết bị Bit (d) thành thiết bị (d)

Ví dụ

Khi bit M11 được xác định bằng lệnh SFTP và lệnh SFTP được thực thi, trạng thái của bit M10 sẽ dịch sang cho bit M11 và M10 sẽ chuyển sang trạng thái OFF:

+ Bật ON thiết bị đầu tiên đã được dịch bằng lệnh SET
+ Khi lệnh SFT(P) được sử dụng sau đó, tạo đoạn chương trình để bắt đầu từ thiết bị có số lớn nhất.



(1): X2 bật ON

(2): Sau lần dịch đầu tiên

(3): Sau lần dịch thứ 2

(4): X2 bật ON

(5): Sau lần dịch thứ 3

(6): Sau lần dịch thứ 4

(7): Sau lần dịch thứ 5

(8): Thiết bị đầu tiên của quá trình dịch

(9): Phạm vi dịch.

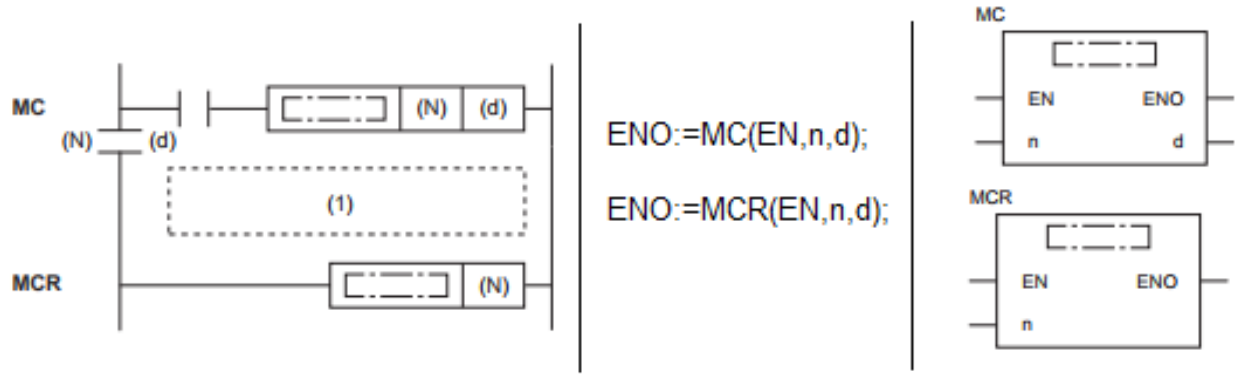
Ngoài lệnh dịch bit SFT còn có nhiều lệnh dịch khác như:

+ Dịch dữ liệu 16-bit sang phải/trái n-bit: **SFR(P)** và **SFL(P).**
+ Dịch n-bit dữ liệu sang phải/trái 1-bit: **BSFR(P)** và **BSFL(P).**
+ Dịch n-word dữ liệu sang phải/trái 1-word: **DSFR(P)** và **DSFL(P).**
+ Dịch dữ liệu n-bit sang phải/trái (n) bit: **SFTR(P)** và **SFTL(P).**
+ Dịch n-word dữ liệu sang phải/trái (n) word: **WSFR(P) và WSFL(P).**

16. Lệnh điều khiển đoạn chương trình

Tên lệnh: **MC** (Master Control) bật điều khiển tách đoạn

MCR (Master Control Reset) kết thúc điều khiển tách đoạn



- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
N	Nestin	0 đến 14	Tên phần tử	ANY16_S
(d)	Số phần tử được bật ON	-	Bit	ANY_BOOL
EN	Điều kiện thực thi	-	Bit	BOOL
ENO	Kết quả thực thi	-	Bit	BOOL

- Dữ liệu phần tử ứng dụng được

Toán hạng	Bit		Word			Double word		Hằng số	
	X, Y, M, L, SM, F, B, SB, S	**T, ST, C, LC**	**T, ST, C, D, W, SD, SW, R**	**U□ \G□**	**Z**	**LC**	**LZ**	**K, H**	**E**
(n)	x		x (1)						

(1) T, ST, C không được sử dung

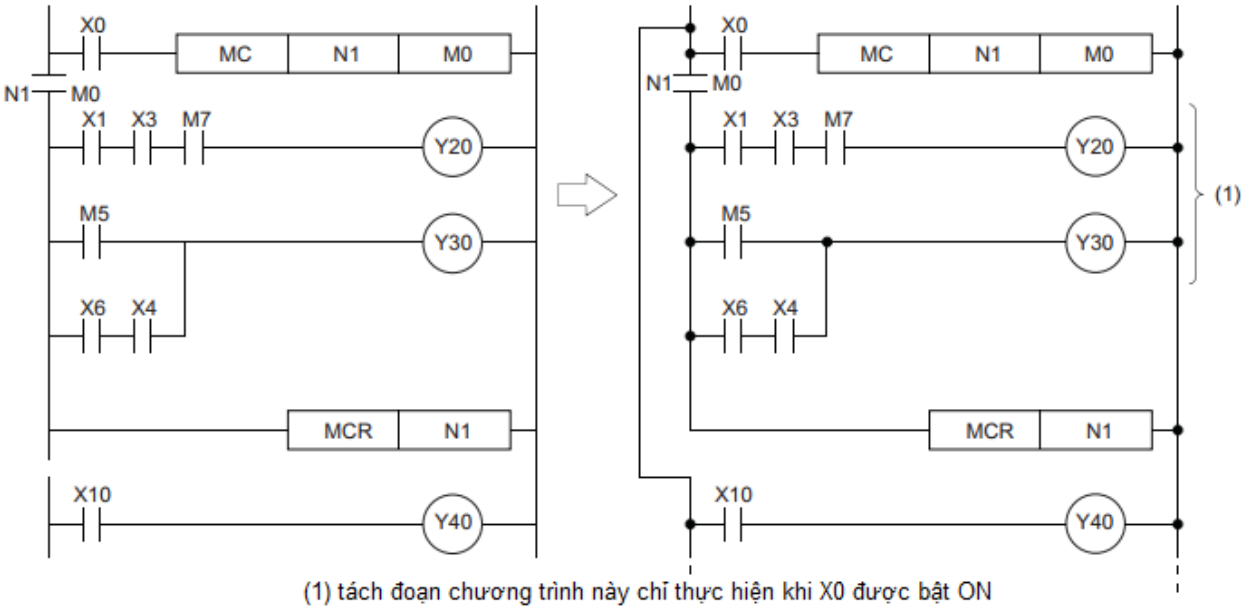
Hoạt động.

Hai lệnh này tạo ra một tách đoạn chương trình trên chương trình Ladder chính.

Hình dưới đây minh họa: ở bên trái là hình ảnh Ladder khi lập trình, còn ở bên phải là quá trình thực hiện thực tế chương trình.

Như vậy khi điều kiện của lệnh MC chưa được thỏa mãn (X0=OFF) thì đoạn chương trình này bị ngắt khỏi chương trình chính do M0=OFF.

Khi điều kiện của lệnh MC được thỏa mãn (X0=ON) thì tách đoạn chương trình được thông (nhờ M1=ON) và đoạn chương trình này sẽ hoạt động như đoạn Ladder bình thường.



Sau khi điều kiện của MC lại ngắt (ở đây là X0=OFF trở lại) thì trạng thái các phần tử nằm trong tách đoạn chương trình này (đoạn nằm giữa MC và MCR) sẽ như sau:

+ Các đầu ra Y đều bị ngắt Y=OFF.

+ Các phần tử nằm trong các lệnh SET, RST, SFT, các lệnh cơ bản về lệnh ứng dụng giữ nguyên trạng thái của nó.

+ Giá trị hiện thời của các bộ đếm và Timer nhớ (retentive) giữ nguyên, tuy nhiên các phần tử này bị dừng hoạt động

+ Bộ Timer thường bị ngắt đồng thời giá trị đếm bị xóa về 0.

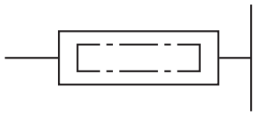
Chú ý. Lệnh này không được dùng trong đoạn chương trinh với lệnh FOR – NEXT; STL – RET (SRET); I – IRET.

17. Lệnh kết thúc.

a). Kết thúc chương trình chính (main routine program)

Tên lệnh **FEND.** Lệnh này được sử dụng để nhánh hoạt động của chương trình tuần tự bởi lệnh nhảy hoặc để chia lại chương trình chính thành chương trình con hoặc chương trình ngắt.

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	Không hỗ trợ	Không hỗ trợ

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu
(s)	Được dùng như một tiếp điểm	__	Bit

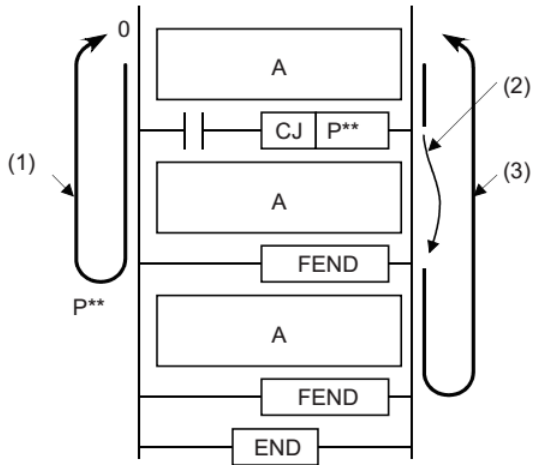
Hoạt động.

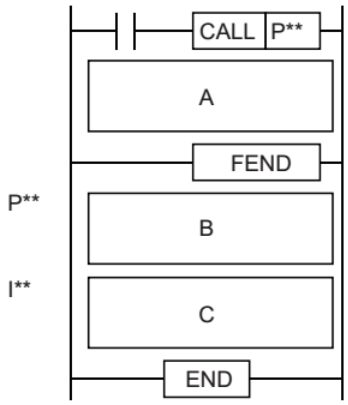
Lệnh này được sử dụng để nhánh hoạt động của chương trình tuần tự bởi lệnh CJ hoặc để chia lại chương trình chính thành chương trình con hoặc chương trình ngắt

Khi lệnh này được thực thi, việc thực thi chương trình sẽ quay trở lại tới đoạn chương trình tại bước 0 sau khi xử lý đầu ra, xử lý đầu vào và làm mới Watchdog timer.

Chương trình tuần tự từ lệnh này về phía trước có thể cũng được trình bày như Ladder bởi công cụ kỹ thuật

(Trái: Khi lệnh CJ được sử dụng, Phải: Khi có chương trình con và chuowng trình ngắt)

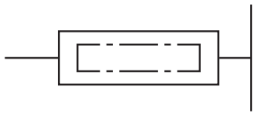




A: Chương tình chính
B: Chương trình con
C: Chuowng trình ngắt
(1): Vận hành khi lệnh CJ không được thực thi
(2): Nhảy bởi lệnh CJ
(3): Vận hành khi lệnh CJ được thực thi

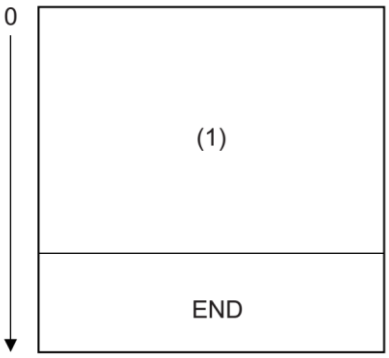
b). Kết thúc chương trình tuần tự

Tên lệnh **END.** Lệnh này chỉ ra điểm kết thúc của toàn chương trình. b. Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	Không hỗ trợ	Không hỗ trợ

Hoạt động.

Lệnh này chỉ ra điểm kết thúc của toàn chương trình bao gồm chương trình chính, chương trình con và chương trình ngắt. Khi lệnh này được thực thi, module CPU kết thúc việc thực thi của chương trình đang thực thi hiện thời.



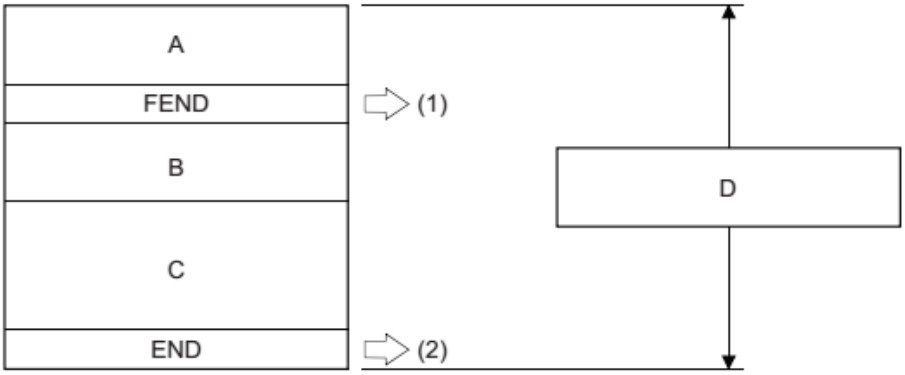
(1) Chương trình tuần tự

Lần đầu thiên lệnh RUN được khởi động, việc thực thi bắt đầu từ lệnh này.

Lệnh này khoong thể được lập trình ở giữa trong suốt chương trình tuần tự chính. Khi quá trình xử lý được yêu cầu ở giữa trong suất chương trình, sử dụng lềnh FEND

Khi chương trình được thực hiện sử dụng công cụ kỹ thuất trong chế độ chỉnh sửa Ladder, lệnh END tự động được nhập vào và không thể chỉnh sửa

Phần sau minh họa cách lệnh END và FEND được sử dụng phù hợp khi chương trình bao gồm chương trình chính, chương trình con và chương trình ngắt.



A: Chương trình chính
B: Chương trình con
C: Chương trình ngắt
D: Khu vực chương trình chính tuần tự
(1): Lệnh FEND được yêu cầu
(2): lệnh END được yêu cầu

3.1.2. Nhóm lệnh cơ bản

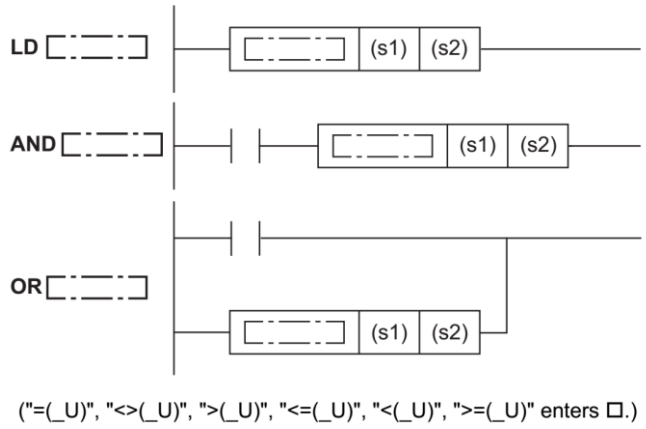
Nhóm lệnh cơ bản là tập hợp các loại lệnh như so sánh, các phép toán số học, toán logic, các lệnh về xử lý bit, lệnh chuyển đổi qua lại giữa các dạng dữ liệu, lệnh đọc chuyển mạch đầu vào kiểu số (digital switch) và các lệnh về truyền dữ liệu. Sau đây giới thiệu một số lệnh thường gặp khi lập trình ở mức ban đầu.

1. Lệnh so sánh

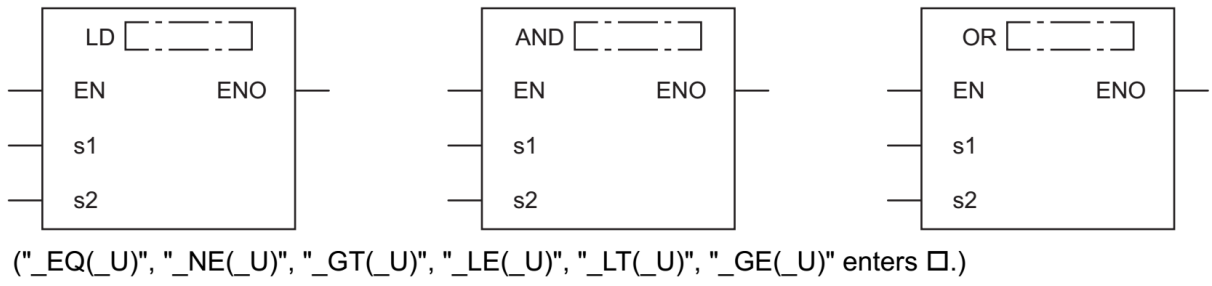
a). So sánh dữ liệu 16-bit

Tên lệnh: **LD□(_U), AND□(_U), OR□(_U).** Những lệnh này thực hiện hoạt động so sánh giữa dữ liệu 16-bit trong thiết bị xác định bởi (s1) và dữ liệu 16-bit trong thiết bị xác định bởi (s2). Thiết bị ở đây được sử dụng như tiếp điểm NO.

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**
	Không hỗ trợ

FBD



- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng		Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(s1)	LD□, AND□, OR□	Dữ liệu so sánh hoặc thiết bị nơi dữ liệu so sánh được lưu trữ.	-32768 to +32767	16-bit signed binary	ANY16_S
	LD□_U, AND□_U,		0 to 65535	16-bit không dấu	ANY16_U

	OR□_U				
(s2)	LD□, AND□, OR□	Dữ liệu so sánh hoặc thiết bị nơi dữ liệu so sánh được lưu trữ	-32768 to +32767	16-bit signed binary	ANY16_S
	LD□_U, AND□_U, OR□_U		0 to 65535	16-bit không dấu	ANY16_U

- Dữ liệu phần tử ứng dụng được

Toán hạng	Bit			Word			Double word		Hằng số	
	X, Y, M, L, SM, F, B, SB, S	**T, ST, C, LC**	**U□ \G□**	**T, ST, C, D, W, SD, SW, R**	**U□ \G□**	**Z**	**LC**	**LZ**	**K, H**	**E**
(s1)	x			x	x	x			x	
(s2)	x			x	x	x			x	

Hoạt động

Những lệnh này thực hiện hoạt động so sánh giữa dữ liệu 16-bit trong thiết bị xác định bởi (s1) và dữ liệu 16-bit trong thiết bị xác định bởi (s2). (Những thiết bị được sử dụng như tiếp điểm NO)

Bảng sau liệt kê kết quả hoạt động so sánh của mỗi lệnh:

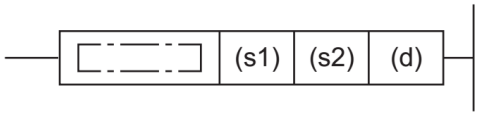
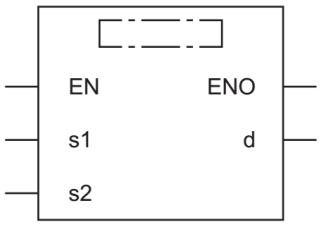
Kí hiệu lệnh	Điều kiện	Kết quả
=(_U)	(s1) = (s2)	Dẫn
<>(_U)	(s1) # (s2)	
>(_U)	(s1) > (s2)	
<=(_U)	(s1) ≤ (s2)	
<(_U)	(s1) < (s2)	
>=(_U)	(s1) ≥ (s2)	
=(_U)	s1) # (s2)	Ngưng dẫn
<>(_U)	(s1) = (s2)	
>(_U)	(s1) ≤ (s2)	
<=(_U)	(s1) > (s2)	
<(_U)	(s1) ≥ (s2)	
>=(_U)	(s1) < (s2)	

So sánh dữ liệu 32-bit. Về cơ bản giống 16-bit, chỉ cần thay 32-bit cho 16-bit

b). So sánh đầu ra 16-bit

Tên lệnh **CMP(P)(_U).** Lệnh này thực hiện hoạt động so sánh giữa dữ liệu 16-bit trong thiết bị xác định bởi (s1) và (s2).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=CMP(EN,s1,s2,d); ENO:=CMPP(EN,s1,s2,d); ENO:=CMP_U(EN,s1,s2,d); ENO:=CMPP_U(EN,s1,s2,d);	

So sánh đầu ra 32-bit. Về cơ bản giống 16-bit, chỉ cần thay 32-bit cho 16-bit

Ngoài ra còn một số lệnh so sánh khác

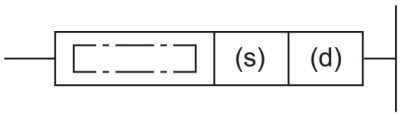
2. Lệnh về các phép toán số học

a). Phép cộng dữ liệu 16bit/32bit

Cộng dữ liệu 16-bit

Lệnh +(P)(_U) và ADD(P)(_U) có thể được sử dụng cho cộng dữ liệu 16-bit

Tên lệnh **+(P)(_U) [using two operands].** Những lệnh này cộng dữ liệu 16-bit trong thiết bị xác định bởi (d) và dữ liệu 16-bit trong thiết bị xác định bởi (s), và lưu dữ kết quả trong thiết bị xác định bởi (d). b. Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	Không hỗ trợ	Không hỗ trợ

c. Dữ liệu lệnh:

- Chức năng, dải giá trị, kiểu dữ liệu:

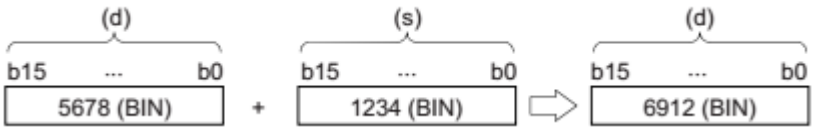
Toán hạng		Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(s)	+(P)	Số hạng hoặc thiết bị nơi lưu dữ liệu được cộng với một số hạng khác được lưu trữ	-32768 to +32767	16-bit signed binary	ANY16_S
	+(P)_U		0 to 65535	16-bit không dấu	ANY16_U
(d)	+(P)	Thiết bị nơi dữ liệu khác được cộng vào	-32768 to +32767	16-bit signed binary	ANY16_S

	+(P)_U	và lưu trữ vào trong đó	0 to 65535	16-bit không dấu	ANY16_U

- Kiểu toán hạng: tương tự nhóm lệnh đầu.

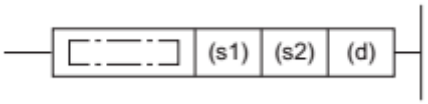
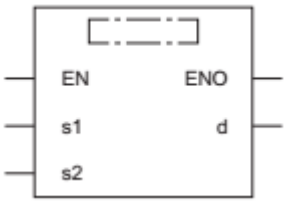
Hoạt động.

Những lệnh này cộng dữ liệu 16-bit trong thiết bị xác định bởi (d) và dữ liệu 16-bit trong thiết bị xác định bởi (s), và lưu dữ kết quả trong thiết bị xác định bởi (d)



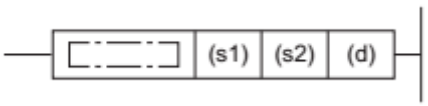
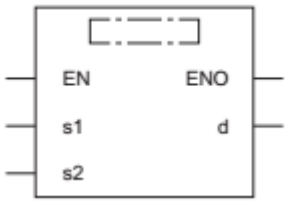
Tên lệnh **+(P)(_U) [using three operands].** Những lệnh này cộng dữ liệu 16-bit trong thiết bị xác định bởi (s1) và dữ liệu 16-bit xác định bởi thiết bị (s2) và lưu trữ kết quả trong thiết bị xác định bởi (d).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=PLUS(EN,s1,s2,d); ENO:=PLUSP(EN,s1,s2,d); ENO:=PLUS_U(EN,s1,s2,d); ENO:=PLUSP_U(EN,s1,s2,d);	

("PLUS", "PLUSP", "PLUS_U", "PLUSP_U" enters ☐.)

Tên lệnh **ADD(P)(_U).** Những lệnh này cộng dữ liệu 16-bit trong thiết bị xác định bởi (s1) và dữ liệu 16-bit trong thiết bị xác định bởi (s2) và kết quả được lưu trong thiết bị xác định bởi (d). bBiểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=ADDP(EN,s1,s2,d); ENO:=ADD_U(EN,s1,s2,d); ENO:=ADDP_U(EN,s1,s2,d);	 ("ADDP", "ADD_U", "ADDP_U" enters ☐.)

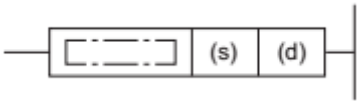
b). Phép trừ. Cũng có các lệnh tương tự như phép cộng

<u>Trừ dữ liệu 16-bit</u>

Lệnh -(P)(_U) and lệnh SUB(P)(_U) có thể sử dụng để trừ dữ liệu 16-bit

+ Tên lệnh: **-(P)(_U) [using two operands].** Những lệnh này trừ dữ liệu 16-bit trong thiết bị xác định bởi (d) và dữ liệu 16-bit trong thiết bị xác định bởi (s), và lưu trữ kết quả trong thiết bị xác định bởi (d).

Biểu diễn trong các ngôn ngữ lập trình

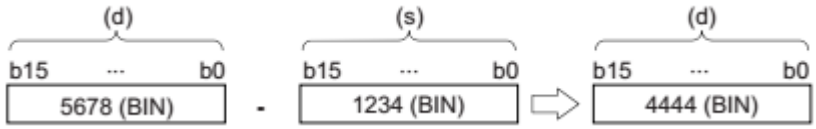
Ladder	ST	FBD
	Không hỗ trợ	Không hỗ trợ

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng		Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(s)	-(P)	Dữ liệu hoặc thiết bị nơi lưu trữ số trừ	-32768 to +32767	16-bit signed binary	ANY16_S
	-(P)_U		0 to 65535	16-bit không dấu	ANY16_U
(d)	-(P)	Thiết bị nơi lưu trữ số bị trừ và kết quả của phép trừ	-32768 to +32767	16-bit signed binary	ANY16_S
	-(P)_U		0 to 65535	16-bit không dấu	ANY16_U

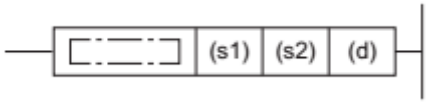
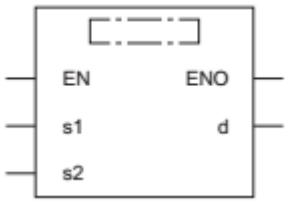
Hoạt động.

Những lệnh này trừ dữ liệu 16-bit trong thiết bị xác định bởi (d) và dữ liệu 16-bit trong thiết bị xác định bởi (s), và lưu trữ kết quả trong thiết bị xác định bởi (d)



+ Tên lệnh: **-(P)(_U) [using three operands].** Những lệnh này trừ dữ liệu 16-bit trong thiết bị xác định với (s1) cho dữ liệu 16-bit trong thiết bị xác định bởi (s2) và lưu kết quả vào trong thiết bị xác định bởi (d).

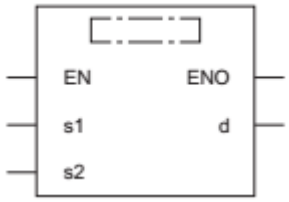
Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=MINUS(EN,s1,s2,d); ENO:=MINUSP(EN,s1,s2,d); ENO:=MINUS_U(EN,s1,s2,d); ENO:=MINUSP_U(EN,s1,s2,d);	

("MINUS", "MINUSP", "MINUS_U", "MINUSP_U" enters □.)

+ Tên lệnh **SUB(P)(_U).** Những lệnh này trừ dữ liệu 16-bit trong thiết bị xác định bởi (s1) và dữ liệu 16-bit trong thiết bị xác định bởi (s2), và lưu kết quả trong thiết bị xác định bởi (d).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=SUBP(EN,s1,s2,d); ENO:=SUB_U(EN,s1,s2,d); ENO:=SUBP_U(EN,s1,s2,d);	

("SUBP", "SUB_U", "SUBP_U" □.)

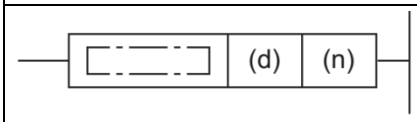
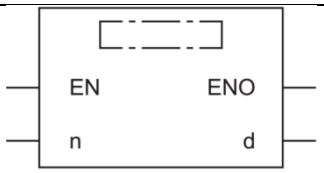
Các phép cộng/trừ dữ liệu 32-bit cũng tương tự như 16-bit, chỉ khác ở dự liệu là 32-bit, (thêm D trước mỗi tên lệnh)

3. Các lệnh xử lý bit

a). Setting 1 bit trong 1 word dữ liệu

Lệnh **BSET(P)** cho phép set 1 bit(lên 1)(n) trong 1 word chỉ định (d).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	Setting : ENO:=BSET(EN,n,d); ENO:=BSETP(EN,n,d); Resetting : ENO:=BRST(EN,n,d); ENO:=BRSTP(EN,n,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

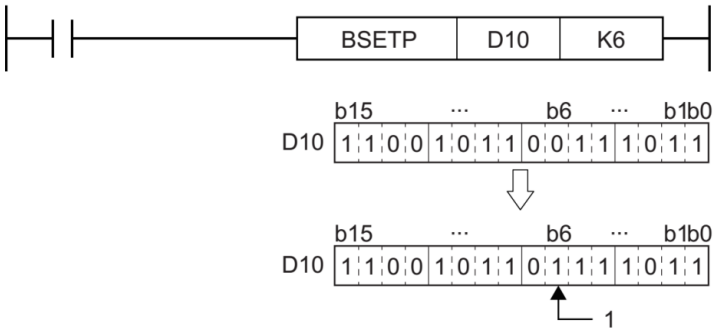
Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị word nơi chứa bit cần set/reset	__	16-bit signed binary	ANY16
(n)	Vị trí của bit cần set/reset trong word	0 to 15	16-bit không dấu	ANY16
EN	Điều kiện thực hiện	__	Bit	BOOL
ENO	Kết quả thực hiện	__	Bit	BOOL

- Dữ liệu phần tử ứng dụng được

Toán hạng	Bit			Word			Double word		Hằng số	
	X, Y, M, L, SM, F, B, SB, S	**T, ST, C, LC**	**U□ \G□**	**T, ST, C, D, W, SD, SW, R**	**U□ \G□**	**Z**	**LC**	**LZ**	**K, H**	**E**
(d)	x			x	x	x			x	
(n)	x			x	x	x				

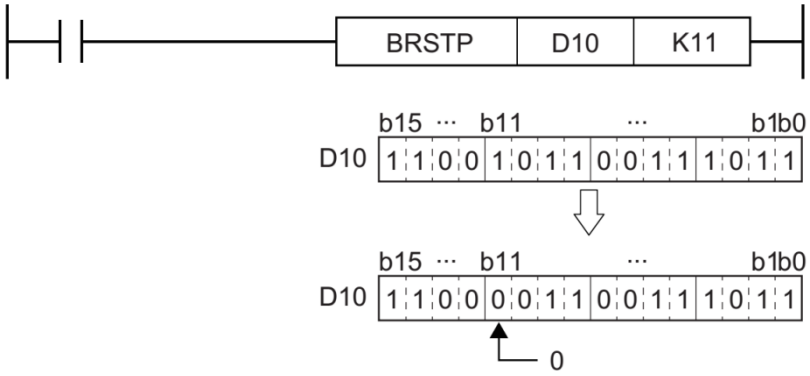
Hoạt động. -Lệnh BSET(P):

• Lệnh cho phép set 1 bit(lên 1)(n) trong 1 word chỉ định (d) .
• Nếu (n) vượt quá 15, việc xử lý sẽ được thực hiện dựa trên 4 bit thấp của (n).



b). **Reset 1 bit trong 1 word dữ liệu**

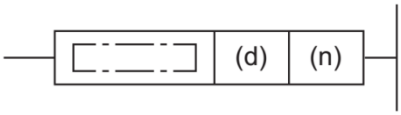
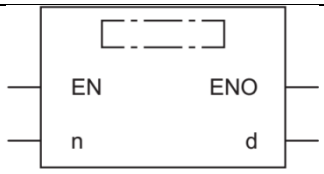
Lệnh **BRST(P)** cho phép reset 1 bit(xuống 0)(n) trong 1 word chỉ định (d) .
• Nếu (n) vượt quá 15, việc xử lý sẽ được thực hiện dựa trên 4 bit thấp của (n).



c). Reset bit hàng loạt

Tên lệnh **BKRST(P).** Cho phép reset (n) thiết bị bit tính từ thiết bị bit được chỉ định bởi (d).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**	**FBD**
	ENO:=BKRST(EN,n,d); ENO:=BKRSTP(EN,n,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị đầu được reset	__	Bit	ANY_BOOL
(n)	Số thiết bị được reset	__	16-bit không dấu	ANY16
EN	Điều kiện thực hiện	__	Bit	BOOL
ENO	Kết quả thực hiện	__	Bit	BOOL

Hoạt động. Cho phép reset (n) thiết bị bit tính từ thiết bị bit được chỉ định bởi (d).

• Reset trạng thái của thiết bị bit như sau.

Thiết bị	Trạng thái
Annunciator (F)	•(n) điểm tính từ số F được chỉ định bởi (d) được tắt. • Các số thông báo từ SD64 to SD79 đã tắt thì bị xóa và các số tiếp theo thì được chuyển về phía trước. • Số lượng số thông báo được lưu trữ trong SD63
Timer (T), Counter (C)	• Giá trị hiện tại của (n) điểm từ timer (T) hoặc counter (C) được chỉ định bởi (d) được đặt thành 0, và tiếp điểm với cuộn dây được tắt.
Thiết bị bit khác	• Cuộn dây và tiếp điểm của (n) điểm từ thiết bị được chỉ định bởi (d) được tắt

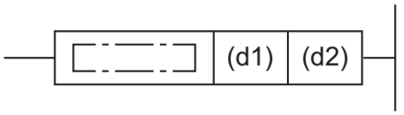
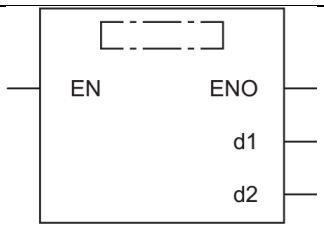
• Nếu thiết bị được chỉ định OFF,. trạng thái thiết bị sẽ không thay đổi.

d). Reset một vùng dữ liệu

Tên lệnh **ZRST(P).** Lệnh này reset các thiết bị được định vị trong một vùng giữa hai thiết bị xác định cùng một lúc.

Sử dụng lệnh này cho hoạt động khởi động lại từ điểm bắt đầu sau khi tạm dừng hoặc sau khi dữ liệu điều khiển được reset.

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
		ENO:=ZRST(EN, d1, d2); ENO:=ZRSTP(EN, d1, d2);

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d1)	Tên thiết bị bit/ word bắt đầu được reset	__	Bit/16-bit signed binary	ANY_ELEMENTARY
(d2)	Tên thiết bị bit/word cuối được reset	__	Bit/16-bit signed binary	ANY_ELEMENTARY
EN	Điều kiện thực hiện	__	Bit	BOOL
ENO	Kết quả thực hiện	__	Bit	BOOL

Hoạt động.

• Lệnh này cho phép reset lại toàn bộ dữ liệu của các thiết bị cùng loại nằm trong dải được giới hạn bởi (d1) và (d2).

• OFF (reset) được ghi vào toàn bộ dải thiết bị từ (d1) đến (d2) cùng 1 lúc nếu (d1) hoặc (d2)

3.1.3. Lệnh ứng dụng – Application Instruction

Đây là nhóm lệnh thực hiện các xử lý như quay (rotate), lệnh về xung, lệnh nhảy về điểm cuối chương trình, các lệnh cho lập trình kiểu cấu trúc (Structure Text), lệnh về bảng số liệu, về số thực, số ngẫu nhiên …

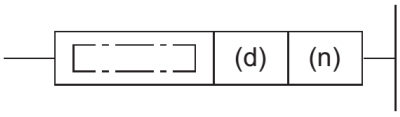
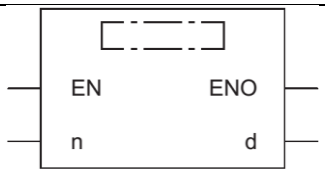
1. Lệnh quay dữ liệu

a). Lệnh quay dữ liệu 16-bit sang phải

Lệnh ROR(P). Lệnh này dùng để dịch chuyển và quay dữ liệu 16-bit nhị phân được lưu trữ trong (d) sang phải với số bit xác định(n) (không sử dụng cờ carry).

Lệnh RCR(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 16-bit nhị phân được lưu trữ trong (d) sang phải với số bit xác định(n) (có sử dụng cờ carry).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	**ST**	**FBD**
	ENO:=RORP(EN,n,d); ENO:=RCR(EN,n,d); ENO:=RCRP(EN,n,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

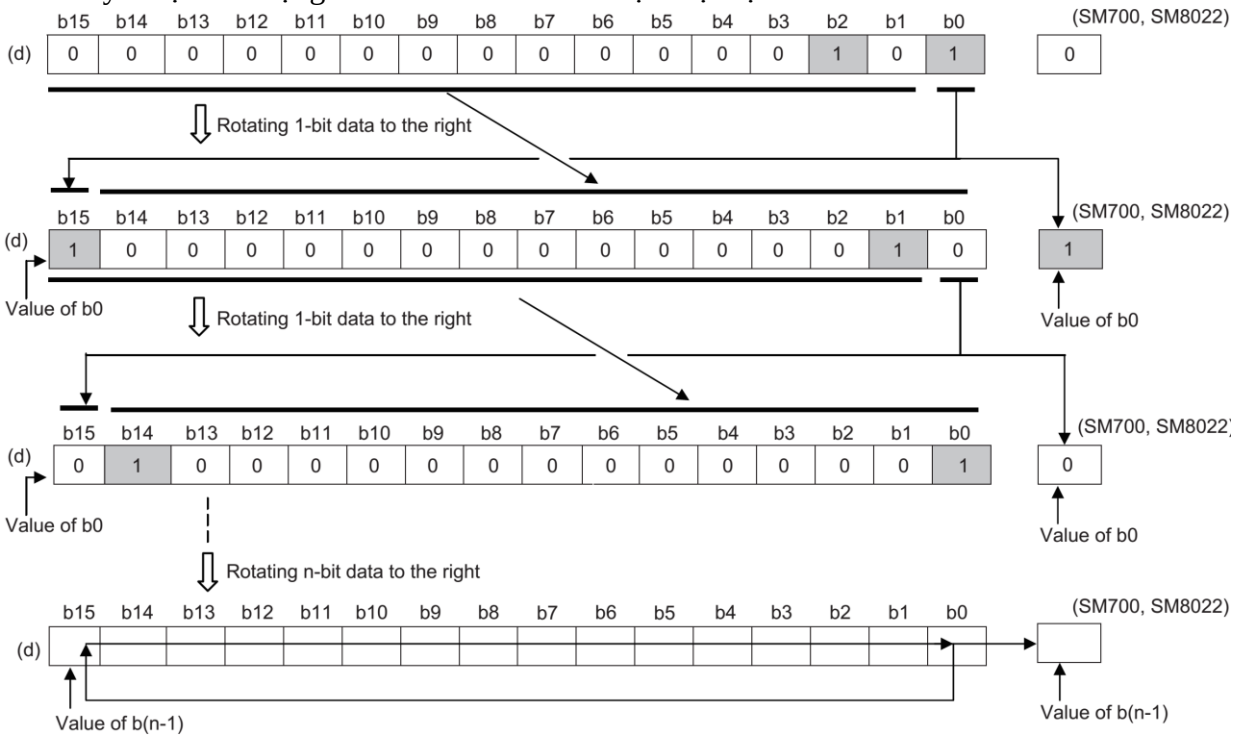
Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(d)	Thiết bị đầu lưu trữ dữ liệu cần quay	__	16-bit signed binary	ANY16
(n)	Số bit được quay	0 to 15	16-bit không dấu	ANY16
EN	Điều kiện thực hiện	__	Bit	BOOL
ENO	Kết quả thực hiện	__	Bit	BOOL

- Kiểu toán hạng: tương tự nhóm lệnh đầu.

Hoạt động.

* ROR(P):

• Lệnh này dùng để dịch chuyển và quay dữ liệu 16-bit nhị phân được lưu trữ trong (d) sang phải với số bit xác định(n) (không sử dụng cờ carry). Cờ carry được tắt hoặc bật tùy thuộc vào trạng thái của nó trước khi thực hiện lệnh.



• Khi (d) là một thiết bị bit, các bit được xoay sang phải trong phạm vi thiết bị cho phép. Số bit thực sự được quay là phần dư của phép chia (n) ÷ (số thiết bị bit khả dụng). Ví dụ, khi (n) là 15 và số thiết bị bit khả dụng là 12, 3 bit được xoay vì 15 chia cho 12 bằng 1 dư 3.

• Chỉ định bất kỳ giá trị nào giữa 0 và 15 cho (n). Nếu giá trị 16 hoặc lớn hơn được chỉ định, các bit được quay bằng phần dư của phép chia n ÷ 16. Ví dụ, khi (n) là 18, 2 bit được xoay vì 18 chia cho 16 bằng 1 dư 2.

b). Lệnh quay dữ liệu 16-bit sang trái:

Lệnh ROL(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 16-bit nhị phân được lưu trữ trong (d) sang trái với số bit xác định(n) (không sử dụng cờ carry).

Lệnh RCL(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 16-bit nhị phân được lưu trữ trong (d) sang trái với số bit xác định(n) (có sử dụng cờ carry).

Tham khảo mục 1

c). Lệnh quay dữ liệu 32-bit sang phải

DROR(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 32-bit nhị phân được lưu trữ trong (d) sang phải với số bit xác định(n) (không sử dụng cờ carry).

DRCR(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 32-bit nhị phân được lưu trữ trong (d) sang phải với số bit xác định(n) (có sử dụng cờ carry).

d). Lệnh quay dữ liệu 32-bit sang trái

DROL(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 32-bit nhị phân được lưu trữ trong (d) sang trái với số bit xác định(n) (không sử dụng cờ carry).

DRCL(P): Lệnh này dùng để dịch chuyển và quay dữ liệu 32-bit nhị phân được lưu trữ trong (d) sang trái với số bit xác định(n) (có sử dụng cờ carry).

2. Lệnh về chương trình con

Tên lệnh:

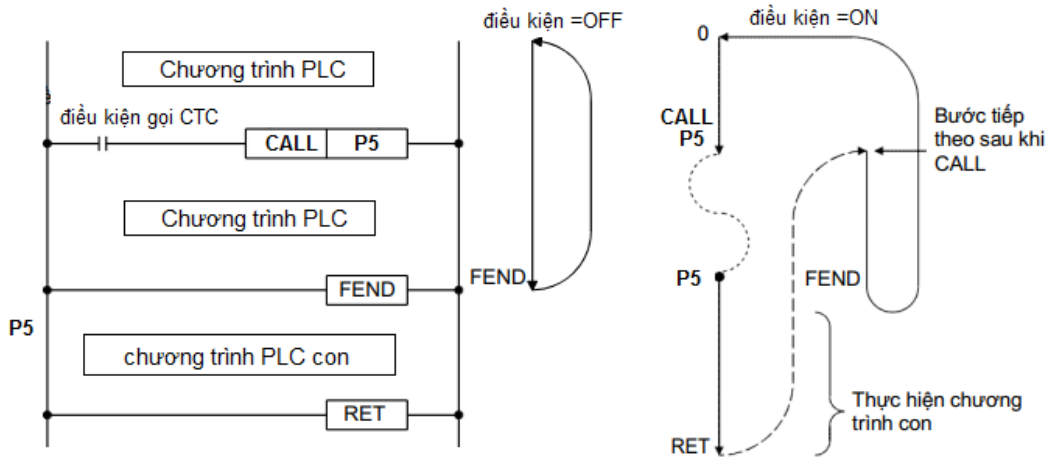
CALL(P) : lệnh gọi một chương trình con (subroutine program)

RET : lệnh kết thúc chương trình con

Các lệnh này chỉ có ở dạng lập trình Ladder, lập trình kiểu ST hay FB không áp dụng được.

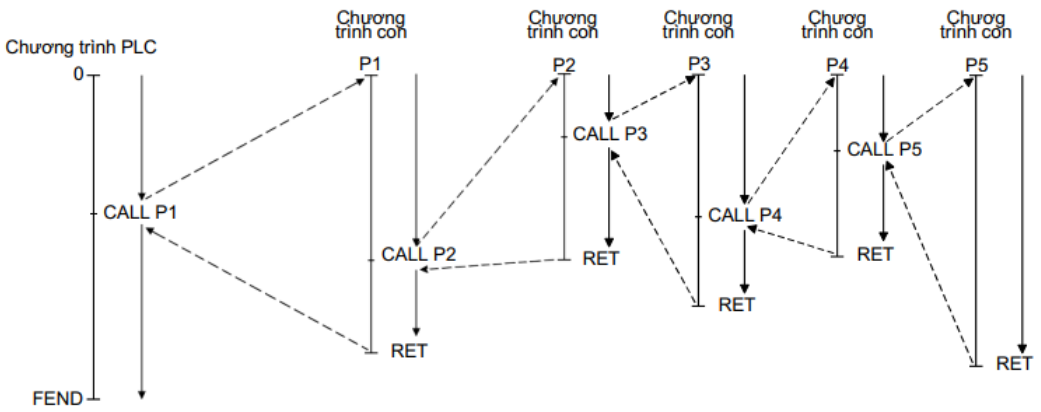
P là kiểu dữ liệu con trỏ và có dải từ 0 đến 4095.

Phương thức thực hiện vòng quét khi có chương trình con như hình dưới đây.



+ Một chương trình con có thể được gọi nhiều lần trong chương trình chính.

+ Có thể lồng các chương trình con trong nhau, tối đa đến 16 cấp.

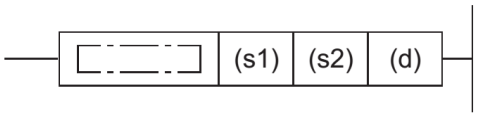
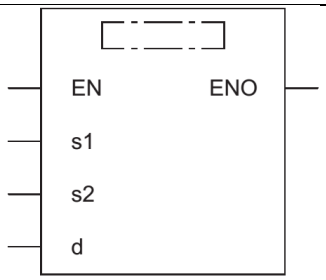


3. Lệnh về xung

Điều chế độ rộng xung 16/32-bit:

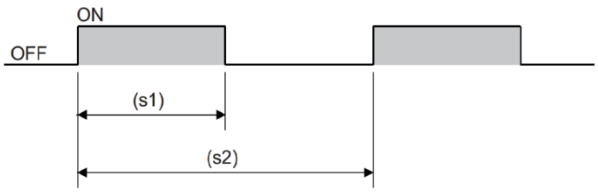
Tên lệnh **PWM** (16 bit) và **DPWM** (32 bit). Lệnh này xuất ra xung (trong các đơn vị dữ liệu 16/32-bit) của thời gian ON (trong các đơn vị dữ liệu 16/32-bit) được chỉ định bởi (s1) có chu kỳ chỉ định bởi (s2) đến đích đầu ra được chỉ định bởi (d).

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=PWM(EN,s1,s2,d); ENO:=DPWM(EN,s1,s2,d);	

Hoạt động PWM:

Lệnh này xuất ra các xung trong thời gian ON đã được xác định bởi (s1) và chu kỳ phát xung đã được xác định bởi (s2) để xuất đến vị trí đã được xác định bởi (d)



DPWM: tương tự như lênh PWM.

4. Lệnh về Clock

TRD(P) - lệnh đọc thời gian từ thời gian thực của modul CPU.

TWD(P) - lệnh viết thời gian vào vùng nhớ thời gian thực của modul CPU.

TADD - lệnh cộng thời gian được lưu trong thanh ghi (d1) cộng với thời gian trong thanh ghi (d2) và kết quả được lưu trong thanh ghi (d).

TSUB - lệnh trừ thời gian được lưu trong thanh ghi (d1) cộng với thời gian trong thanh ghi (d2) và kết quả được lưu trong thanh ghi (d).

DSTOH- là lệnh chuyển đổi thời gian được lưu trong thiết bị (s) từ giây sang giờ,ngày,tháng và lưu giá trị đó vào thiết bị (d) kiểu nhị phân 16 bit hoặc 32bit.

LDDT,ANDDT, ORDT - Các lệnh so sánh ngày được lưu trong thanh ghi (d1) với ngày trong thanh ghi (d2) và kết quả nếu bằng nhau sẽ set thanh ghi (s3).

LDDT,ANDDT, ORDT – các lệnh so sánh thời gian được lưu trong thanh ghi (d1) với thời gian trong thanh ghi (d2) và kết quả nếu bằng nhau sẽ set thanh ghi (s3).

DSTOH là lệnh so sánh thời gian đặc biệt trong (s1),(s2),(s3) với dữ liệu thời gian đặc biệt trong (s4), và bật tắt bit trong (d) với kết quả tương ứng.

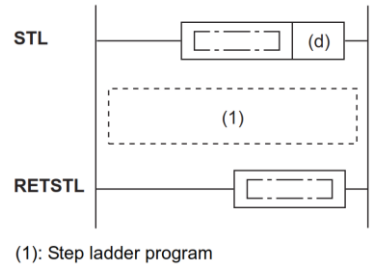
DSTOH - là lệnh so sánh thời gian đặc biệt giữa (s1),(s2) với dữ liệu thời gian đặc biệt trong (s3), và bật tắt bit trong (d) với kết quả tương ứng.

Generating timing pulses - lệnh cộng thời gian được lưu trong thanh ghi (d1) cộng với thời gian trong thanh ghi (d2) và kết quả được lưu trong thanh ghi (d).

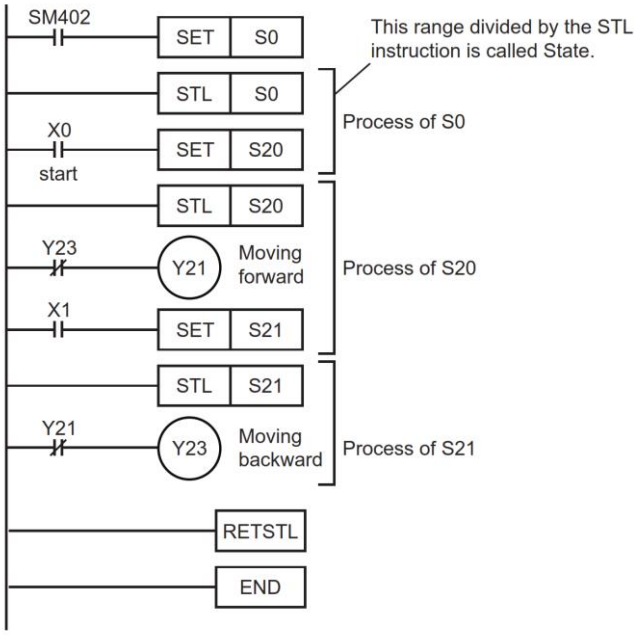
3.1.4. Lệnh bước – Step Ladder Instruction

Starts/Ends step ladder

STL: lệnh bắt đầu step ladder
RETSTL: lệnh kết thúc step ladder

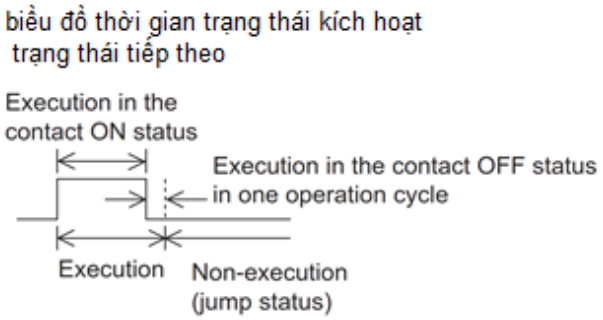
Ladder	ST	FBD
	Không được hỗ trợ	Không được hỗ trợ

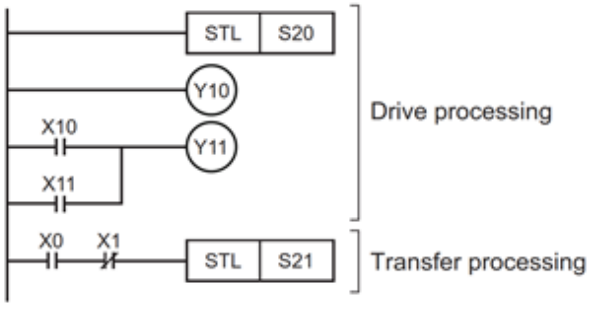
(d) là kiểu dữ liệu bit trong dải từ S0 đến S4095
Các bước chương trinh bậc thang thực tế như sau:



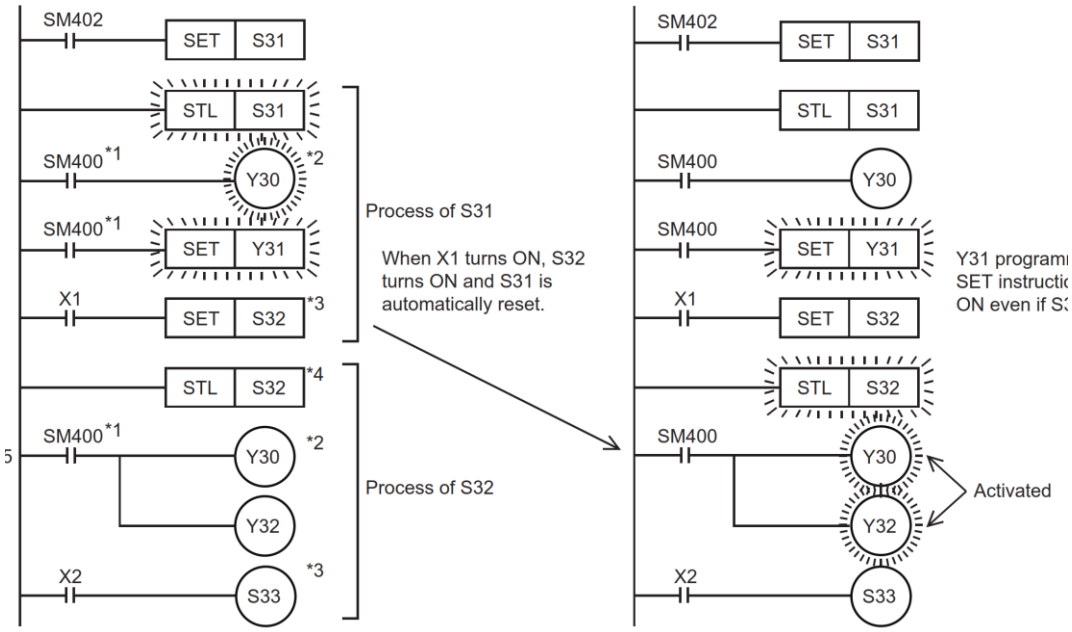
- 1 chương trình step ladder được biểu thị như một rơ le ladder, nhưng nó có thể được tạo ra theo máy kiểm soát dòng chảy sử dụng các rơ le trạng thái. Một rơ le trạng thái gồm một cuộn hút và tiếp điểm (STL contact) trong cùng một cách với các rơ le khác. Sử dụng lệnh SET hoặc OUT để điều khiển một cuộn hút và sử dụng lệnh STL cho tiếp điểm
- Bảng dưới đây cho biết hoạt động mạch điện được kết nối với rơ le trạng thái

Thực hiện trong các trạng thái tiếp điểm ON	Khi 1 rơ le trạng thái bật ON, 1 mạch kết nối được kích hoạt với một tiếp điểm STL
Thực hiện trong các trạng thái tiếp điểm OFF (cho một chu trình vận hành)	Khi 1 điều kiện được cung cấp giữa các rơ le trạng thái được thỏa mãn, rơ le trạng thái tiếp theo được bật ON và rơ le trạng thái cái mà được ON trước được tắt(chuyển hoạt động)
Non- execution	Một lệnh không được thực thi trong trạng thái tiếp điểm OFF sau chu trình hoạt động nơi mà lệnh được thực thi trong trang thái tiếp điểm OFF(trạng thái nhảy)





Sự hoạt động của chương trình



*1 Nó luôn luôn là cần thiết để chương trình điều khiển 1 đầu ra

*2 Các cuộn hút đầu ra có thể được sử dụng lại trong các rơ le trạng thái khác

*3 Mỗi lệnh OUT và SET cho các rơ le trạng thái tự động reset chuyển nguồn, và có chức năng tự giữ

*4 một số lượng rơ le trạng thái chỉ có thể được sử dụng một lần

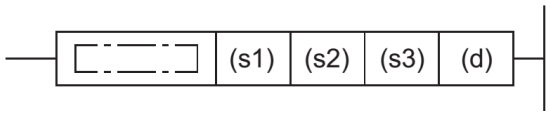
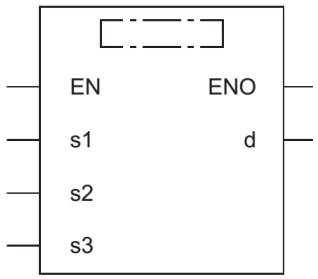
*5 Nó không thể đặt con trỏ ngay lập tức sau lệnh STL, nếu một con trỏ được đặt, một chương trình lỗi xảy ra(33E2H)

3.1.5. Lệnh điều khiển kiểu PID – PID Control Instruction

PID control loop

Tên lệnh **PID -** lệnh tiến hành PID cho đối tượng nhằm thay đổi đầu ra theo đầu vào.

Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	ENO:=PID(EN,s1,s2,s3,d);	

- Chức năng, dải giá trị, kiểu dữ liệu:

Toán hạng	Chức năng	Dải giá trị	Kiểu dữ liệu	Kiểu dữ liệu (nhãn)
(s1)	Địa chỉ giá trị thực	-32768- 32767	16-bit signed binary	ANY16

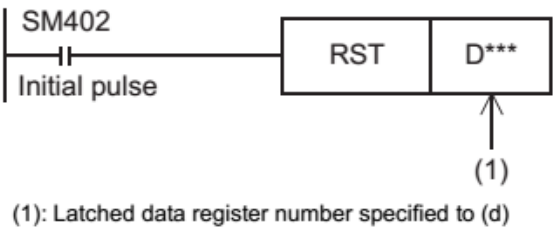
(s2)	Địa chỉ giá trị đặt	-32768- 32767	16-bit signed binary	ANY16
(s3)	Tham số	1- 32767	16-bit signed binary	ANY16
(d)	Gía trị lưu đầu ra	-32768- 32767	16-bit signed binary	ANY16
EN	Điều kiện thực hiện	---	Bit	BOOL
ENO	kết quả thực hiện	---	Bit	BOOL

- Kiểu toán hạng:

Toán hạng	Bit	Word		Double word		Hằng số	
	X, Y, M, L, SM, F, B, SB, S	**T, ST, C, D, W, SD, SW, R**	**Z**	**LC**	**LZ**	**K, H**	**E**
(s)							

- Hoạt động

  Đây là lệnh đọc dữ liệu thời gian từ địa chỉ (SD210 -> SD216) từ bộ thời gian thực trong modul CPU



(1): Latched data register number specified to (d)

Chú ý: Đây là lệnh chiếm 3 vùng địa chỉ bắt đầu từ địa chỉ SD210. Chắc chắn rằng địa chỉ này không dùng cho mục đích khác

3.2. Các lệnh cho modul đặc biệt

1. Built-in Ethernet Function Instructions

SP.SOCOPEN: Lệnh này sẽ mở ra một kết nối.

SP.SOCCLOSE lệnh đóng một kết nối.

SP.SOCRCV - lệnh đọc nhận dữ liệu

SP.SOCRCV Reading connection information

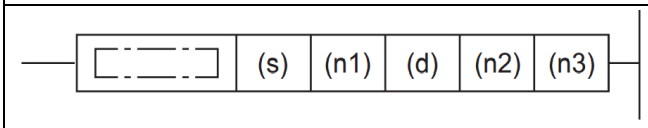
SP.SOCRCV-, Reading Socket communication receice data

SP.ECPRTCL - Executing the protocols registered for the predefined protocol support function

SP.ECPRTCL - Sending the SLMP frame

2. Serial Communication 2

Tên lệnh **RS2** - Lệnh này cho phép gửi hoặc nhận giữ liệu thông qua cổng truyền thông nối tiếp RS-232 hoặc RS-485. Biểu diễn trong các ngôn ngữ lập trình

Ladder	ST	FBD
	EN0:=RS2(EN,s,n1,n2,n3,d)	

3. Inverter Communication Instruction

a). Inverter operation monitoring (Status check)

Tên lệnh **IVCK** - Lệnh này đọc trạng thái hoạt động của 1 biến tần nhờ module CPU

b). Inverter operations control (Drive)

Tên lệnh. **IVDR** - là lệnh viết 1 giá trị điều khiển cần thiết cho hoạt động của biến tần từ 1 module CPU sử dụng máy tính kết nối với chắc năng hoạt động của biến tần.

c). Inverter parameter read

Tên lệnh **IVRD** Lệnh này đọc thông số của 1 biến tần thông qua module CPU

d). Inverter parameter write

Tên lệnh **IVWR** - lệnh này ghi giá trị 1 tham số của 1 biến tần từ module của CPU.

e). Inverter parameter block write

Tên lệnh **IVBWR** - Lệnh này viết hàng loạt tham số của 1 biến tần từ module CPU.

g). Inverter multi command

Tên lệnh **IVMC** – lệnh này thực hiện 2 loại cài đặt (lệnh hoạt động và cài đặt tần số) cho biến tần và đọc 2 loại dữ liệu (màn hình trạng thái biến tần và giá trị tần số ra của biến tần…) từ biến tần trong cùng 1 lúc.

4. MODBUS Communication Instruction

Tên lệnh. **ADPRW** - lệnh này cho phép truyền thông MODBUS giữa Master (đọc/ ghi dữ liệu) với Slaves.

Predefined Protocol Support Function Instruction

Tên lệnh: **S(P).CPRTCL** - lệnh này thực hiện các giao thức truyền thông đã được ghi lại sử dụng công cụ kỹ thuật.

5. Positioning Module

a). Restoring the absolute positon.

Tên lệnh **G.ABRST** - lệnh này khôi phục lại vị trí của trục đã xác định.

b). Starting the positioning

Tên lệnh **GP.PSTRT□** - lệnh này cho phép bắt đầu xác định vị trí của trục đã được chỉ định.

c). Teaching

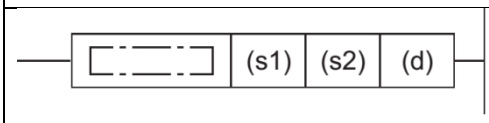
Tên lệnh **GP.TEACH□** - lệnh này huấn luyện trục đã được chỉ định.

6. HIGH-SPEED COUNTER INSTRUCTION

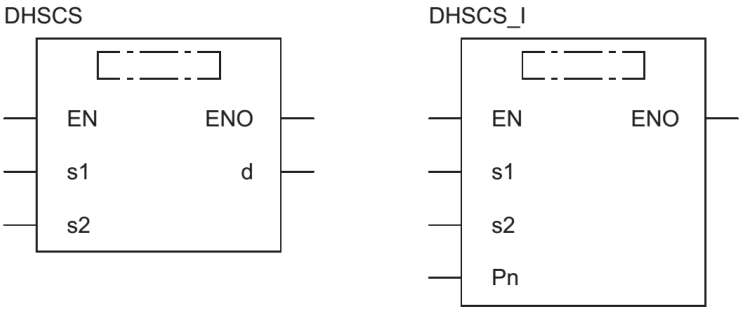
Xử lý bộ đếm tốc độ cao

a). Thiết lập so sánh dữ liệu 32bit

DHSCS. Lệnh này so sánh giá trị được đếm bởi bộ đếm tốc độ cao với một giá trị được chỉ định và ngay lập tức set một bit nếu hai giá trị bằng nhau.

Ladder	ST*1
	ENO:=DHSCS(EN,s1,s2,d); ENO:=DHSCS_I(EN,s1,s2,Pn);

FBD*1



*1 Khi con trỏ ngắt (I) được định nghĩa trong toán hạng (d) bằng ngôn ngữ ST và ngôn ngữ FBD / LD, sử dụng lệnh DHSCS_I.

Cài đặt dữ liệu

■Mô tả, phạm vi, và kiểu dữ liệu

Toán hạng		Mô tả	Phạm vi	Loại dữ liệu	Loại dữ liệu (Nhãn)
(s1)		Dữ liệu được so sánh với giá trị hiện tại của bộ đếm tốc độ cao hoặc từ thiết bị từ lưu trữ dữ liệu	-2147483648 to +2147483647	32-bit signed binary	ANY32
(s2)		Số kênh của một bộ đếm tốc độ cao	_	32-bit signed binary	ANY32
(d)	DHSCS	Bit được đặt thành ON khi hai giá trị so sánh bằng nhau	_	Bit	ANY_BOOL
	DHSCS_I *1			_	POINTER

EN	Điều kiện thực hiện	_		BOOL
ENO	Kết quả thực hiện	_		BOOL

*1 Trong trường hợp ngôn ngữ ST và ngôn ngữ FBD / LD, d sẽ hiển thị như Pn.

Đối tượng có thể áp dụng

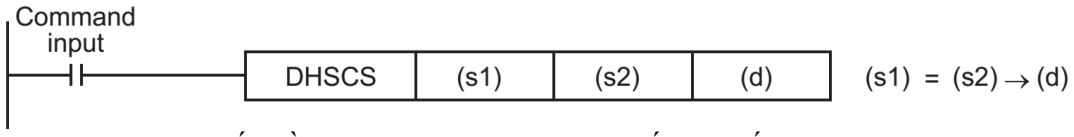
Toán hạng	**Bit**	**Word**			**Double word**		**Constant**			**Others**
	X, Y, M, L, SM, **F, B, SB, S**	**T, ST, C, D, W,** **SD, SW, R**	**U□\G□**	**Z**	**LC**	**LZ**	**K,H**	**E**	**$**	
(s1)	o	o	o	o	o	o	o	–	–	–
(s2)	o	o	o	o	o ***1**	o	o	–	–	–
(d)	o	–	–	–	–	–	–	–	–	o ***2**

*1 Bật chức năng tương thích FX3 và chỉ định thiết bị giữa LC35 và 55 được chỉ định là bộ đếm tốc độ cao tương thích FX3. Đối với chức năng tương thích FX3, hãy tham khảo🕮MSYS iQ-FX FX5 User's Manual (Application).

*2 I16 tới I23 có thể được sử dụng.

Hoạt động

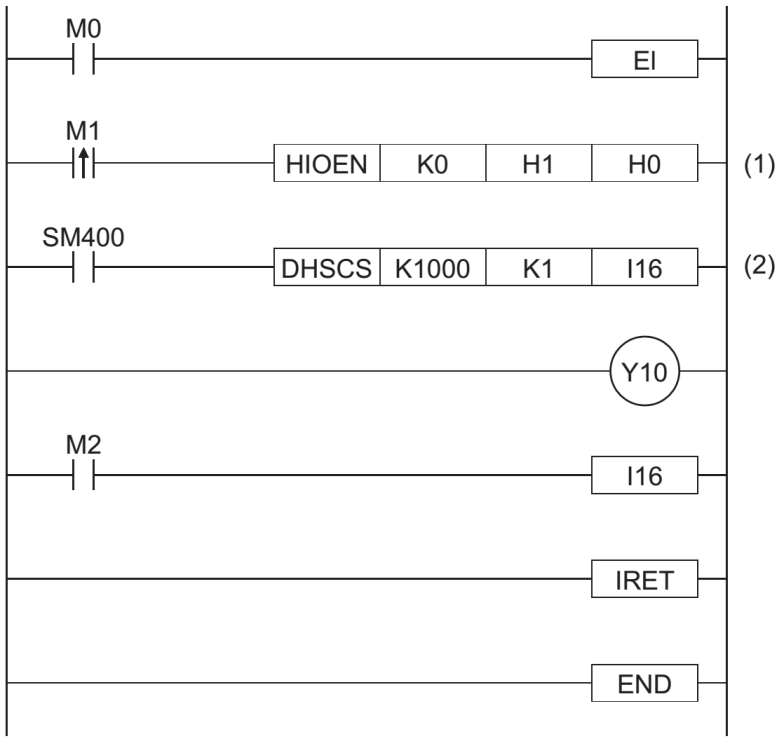
• Khi giá trị hiện tại của bộ đếm tốc độ cao của các kênh được chỉ định trong (s2) trở thành giá trị so sánh (s1) (ví dụ: khi giá trị hiện tại thay đổi từ "199" đến "200" hoặc "201" thành "200" nếu giá trị so sánh là K200), thiết bị bit (d) được đặt thành ON bất kể thời gian quét. Trong hướng dẫn này, quá trình so sánh được thực hiện sau khi xử lý đếm trong bộ đếm tốc độ cao. Để biết chi tiết, hãy tham khảo 🕮MSYS iQ-FX FX5 User's Manual (Application).



Sử dụng DHSCS nếu đầu ra phải được đưa ra khi kết quả đếm tương đương với giá trị so sánh bất kể thời gian quét của module CPU.

Khi số lượng chỉ dẫn được sử dụng đồng thời bị vượt quá giới hạn, hãy sử dụng một chỉ dẫn so sánh chung.

Nếu sử dụng ngắt kết hợp so sánh tốc độ cao, các chương trình ngắt tương ứng có thể được thực hiện bằng cách đặt các con trỏ ngắt (I16 đến I23) thành (d) như được hiển thị bên dưới.



A: Chương trình chính
B: I16 Chương trình ngắt
(1): Bật bộ đếm tốc độ cao CH1.
(2): Khi giá trị hiện tại của bộ đếm tốc độ cao CH1 đạt đến 1000, chương trình gián đoạn (I16) được thực hiện.

Chú ý

Giá trị được xác định trong (s2) chỉ nên là kênh số lượng truy cập tốc độ cao (1 đến 8) được đặt bởi tham số.

• Lỗi xảy ra trong các trường hợp sau.

- Khi một kênh không được đặt bởi tham số hoặc một giá trị được chỉ định khác K1 đến 8
- Khi một số thiết bị LC không được đặt bởi tham số được chỉ định

Với chú ý khác, xem tại 🕮MELSEC iQ-F FX5 User's Manual (Application).

b). Reset HSC 32bit

DHSCR. So sánh giá trị được đếm bởi bộ đếm tốc độ cao với một giá trị được chỉ định và đặt lại bit nếu hai giá trị tương đương với nhau hoặc đặt lại bộ đếm tốc độ cao.
Hoạt động. • Khi giá trị hiện tại của bộ đếm tốc độ cao của các kênh được chỉ định trong (s2) bằng giá trị so sánh (s1) thì bit (d) được đặt lại sang OFF bất kể thời gian quét.

Point

Sử dụng **DHSCR** nếu đầu ra phải được đưa ra khi kết quả tính toán tương đương với giá trị so sánh bất kể thời gian quét của Module CPU.

c). Comparison of 32-bit data band

DHSZ. Lệnh này so sánh giá trị hiện tại của bộ đếm tốc độ cao với hai giá trị (một vùng), và xuất ra kết quả so sánh (làm mới).

Hoạt động. Giá trị hiện tại của bộ đếm tốc độ cao được chỉ ra trong (s3) được so sánh với hai điểm so sánh (giá trị so sánh 1 và giá trị so sánh 2). Dựa trên kết quả so sánh

vùng, "nhỏ hơn so với giá trị so sánh thấp hơn", "bên trong vùng so sánh" hoặc "lớn hơn giá trị so sánh trên", một trong số (d), (d) +1 và (d) +2 là Thiết lập ON bất kể thời gian quét.

e). Start/stop of the 16-bit/32bit data high-speed I/O function

HIOEN(P). Kiểm soát sự bật/tắt của một chức năng I/O tốc độ cao 16 bit.
DHIOEN(P). Kiểm soát sự bật và tắt của một chức năng I/O tốc độ cao 32 bitt.

g). High-speed Current Value Transfer Instruction

HCMOV(P). High-speed current value transfer of 16-bit data. Hướng dẫn đọc và ghi (cập nhật) thanh ghi đặc biệt để đo tốc độ cao, đo xung độ rộng, PWM, và vị trí.
DHCMOV(P). High-speed current value transfer of 32-bit data. Các hướng dẫn này đọc và ghi (cập nhật) thanh ghi đặc biệt để truy cập tốc độ cao, đo chiều rộng xung, PWM, và vị trí.

3.3. Các hàm chuẩn và khối hàm

1. Các hàm tiêu chuẩn

Trong PLC Fx5U có khoảng xấp xỉ 100 hàm tiêu chuẩn được thể hiện dạng FB hoặc ST giúp cho việc laapjp trình được thuận tiện. Có thể liệt kê một số nhóm hàm tiêu chuẩn sau:

+ Các hàm chuyển đổi (Convertion Functions) giữa các kiểu dữ liệu sang nhau. Như đã trình bày có các kiểu dữ liệu là: Bool; Word (W); Double Word (DW); Integer (Int); Double Integer (Dint); Real (E); String; BCD. Ngoài ra còn có hàm chuyển đổi sang kiểu Time. Tuy nhiên không phải bất kỳ kiểu dữ liệu nào cũng có thể chuyển đổi sang đầy đủ các kiểu còn lại. Thí dụ dữ liệu kiểu Word chuyển được sang các kiểu DW; Bool; Int; Dint và Time. Dữ liệu kiểu BCD chỉ chuyển đổi sang kiểu Int và Dint. .v.v..

+ Các hàm chuẩn toán học: cộng, trừ, nhân, chia, hàm lượng giác, loogarit, hàm mũ....

+ Các hàm dịch bit (Shift).

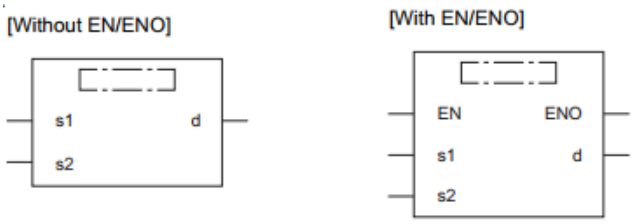
+ Các hàm thực hiện phép lôgic của đại số Bool.

+ Các hàm so sánh.

...

Dưới đây mô tả thí dụ về hàm chuẩn cho chọn lọc – Selection Functions.

a). Hàm chọn giá trị Max/Min



Hàm Max/Min dạng FB

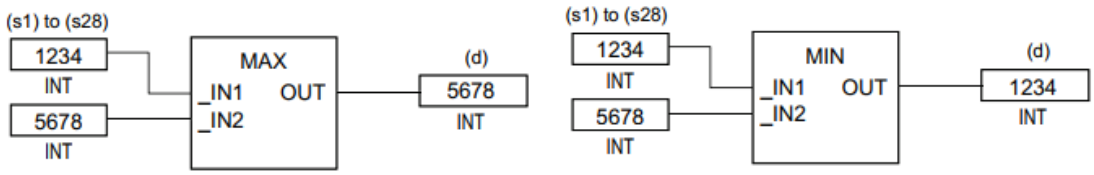
Tên hàm:

+ **Max(_E)** đưa ra ở (d) giá trị lớn nhất trong các giá trị của các đầu vào (s).

+ **Min(_E)** đưa ra ở (d) giá trị nhỏ nhất trong các giá trị của các đầu vào (s).

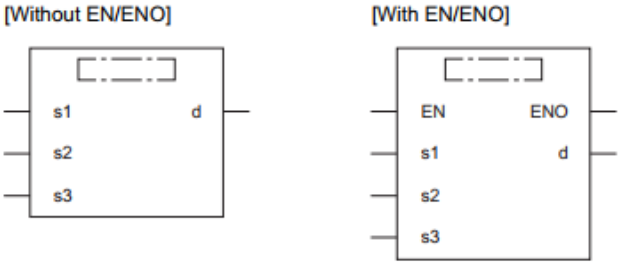
Số lượng các cổng vào (s) nằm trong khoảng từ 2 đến 28. Kiểu dữ liệu các đầu vào (s) và ra (d) là cùng loại và có thể là Bool; Int; Dint; W; DW; Real; String; Time.

Thí dụ



Thí dụ về hàm Max/Min

b). Hàm Limit



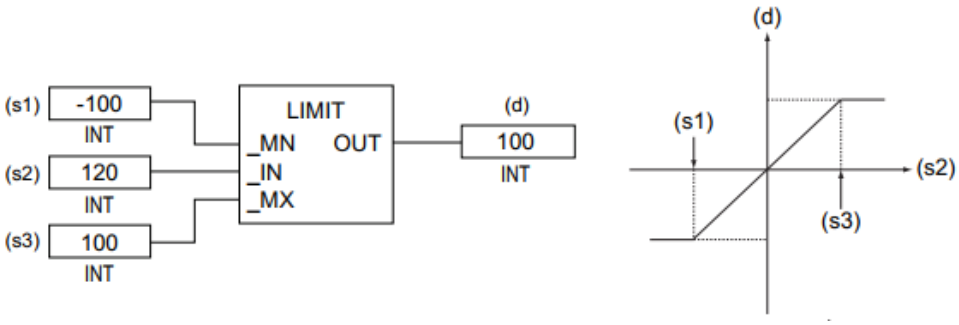
Hàm Limit dạng FB

Tên hàm: **Limit(_E).** Hàm này đưa ra đầu (d) trị số của đầu vào (s2) nhưng chỉ trong phạm vi bị hạn chế dưới bởi giá trị (s1) và hạn chế trên bởi giá trị (s3).

+ Nếu giá trị (s2)< s(1) thì (d) đưa ra trị số bằng (s1).

+ Nếu giá trị (s2) > s(3) thì (d) đưa ra trị số bằng (s3).

Kiểu dữ liệu các đầu vào (s) và ra (d) là cùng loại và có thể là Bool; Int; Dint; W; DW; Real; String; Time.



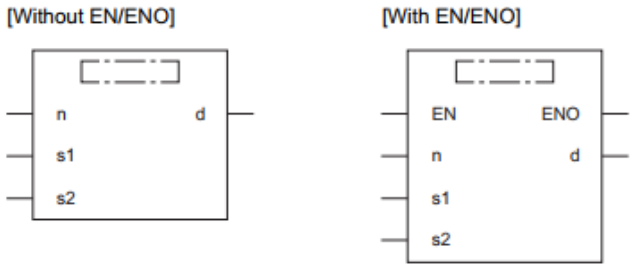
Thí dụ và đặc tính vào/ra của hàm Limit.

c). Hàm Multiplexer

Tên hàm: **Mux(_E).** Hàm này đưa ra ở (d) giá trị của vào (s) theo điều khiển từ trị số của cổng (n). Số lượng tối đa cổng (s) là 28 và (n) có trị số từ 0 đến (s_n-1) tức là trị số lớn nhất của n=27.

+ Khi (n)=0 thì (d) đưa ra giá trị ở (s1)

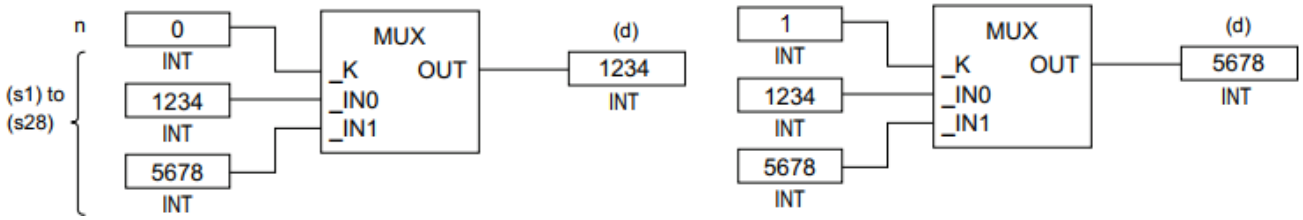
+ Khi (n) = (s_n-1) thì (d) đưa ra giá trị ở (s) cuối cùng.



Khối hàm Multiplexer

Kiểu dữ liệu của (n) là Int. Kiểu dữ liệu của (s) có thể là là Bool; Int; Dint; W; DW; Real; String; Time; Structure hoặc Array.

Thí dụ



2. Các khối hàm

Tổng số có 10 khối hàm chia thành 4 nhóm là:

+ khối hàm 2 trạng thái kiểu RS và SR.

+ khối phát hiện sườn dương và sườn âm tín hiệu.

+ khối hàm Timer gồm 4 khối

+ khối hàm Couter cũng gồm 4 khối.

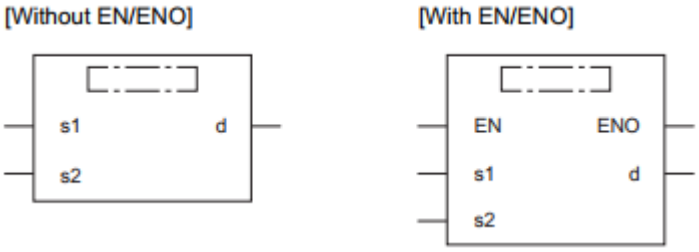
Nhóm cuối đã khá quen thuộc nên ở đây chỉ trình bày nhóm đầu tiên gồm

a). Khối **SR(_E)** ưu tiên set.

Khối này đầu ra (d) có hai trạng thái phụ thuộc trạng thái của hai đầu vào (s) với một đầu vào (s1) có tác dụng Set đầu ra (d) và đầu vào thứ hai (s2) có tác dụng Reset đầu ra (d). Hoạt động như vậy tương tự như loại Flip-Flop RS trong mạch điện tử lôgic. Điều khác biệt là ở đây cửa (s1) có mức ưu tiên cao hơn cửa (s2) thể hiện như sau.

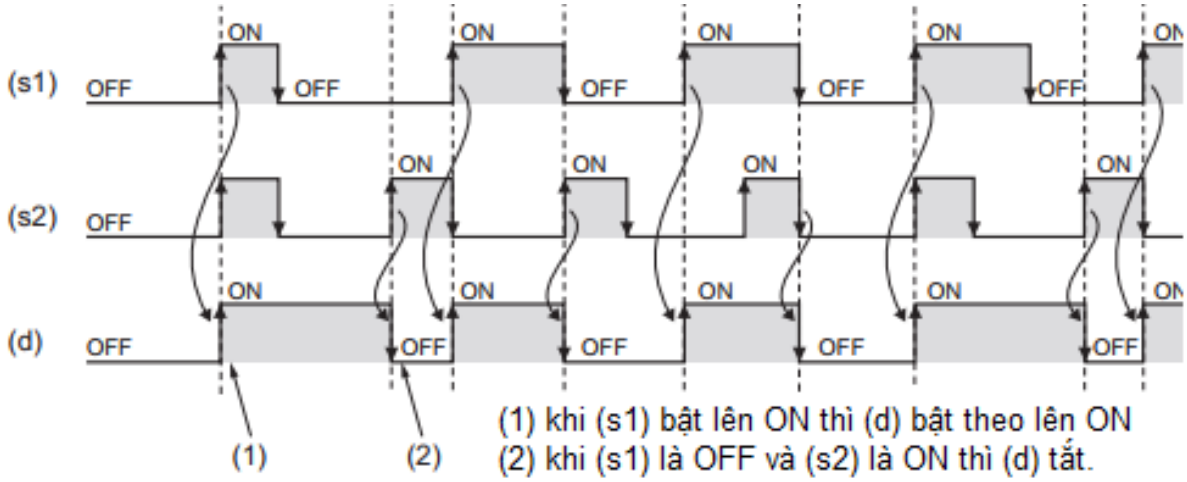
- Khi (s1) bật ON thì (d) được bật (Set). Nếu (s2) bật ON trong khi (s1) đang OFF thì (d) sẽ bị tắt (Reset); tuy nhiên:
- Nếu (s2) bật ON trong khi (s1) đang ON thì (d) sẽ không bị tắt (không Reset được).

Kiểu dữ liệu của tất cả là Bool.



Khối hàm SR và RS

Biểu đồ hoạt động theo thời gian



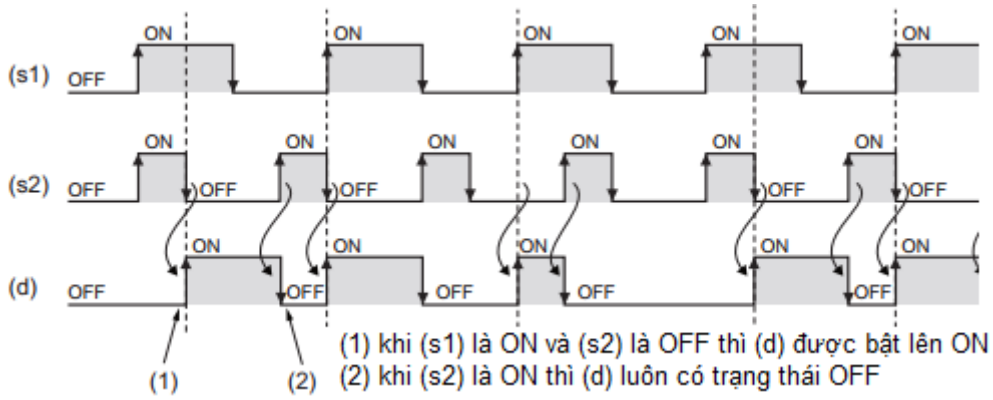
Với khối có sử dụng cổng EN thì mạch hoạt động bình thường như trên nếu EN=ON; con khi đầu EN=OFF thì đầu ra (d) giữ nguyên trạng thái trước đó.

b). Khối **RS(_E)** ưu tiên Reset.

Khối này có các đặc điểm hoạt động tương tự như khối SR(_E) ở trên, tuy nhiên ở đây phần ưu tiên thuộc về cổng (s2)

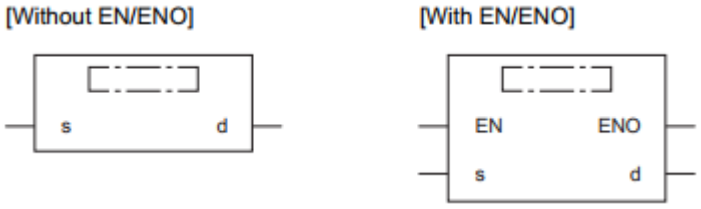
- Khi (s1) bật ON thì (d) được bật (Set), nếu (s2) bật ON thì (d) sẽ bị tắt (Reset),
- Nếu (ss) là ON trong khi s(1) vẫn là ON thì (d) không bật lên ON được (vẫn bị Reset).

Biểu đồ hoạt động theo thời gian

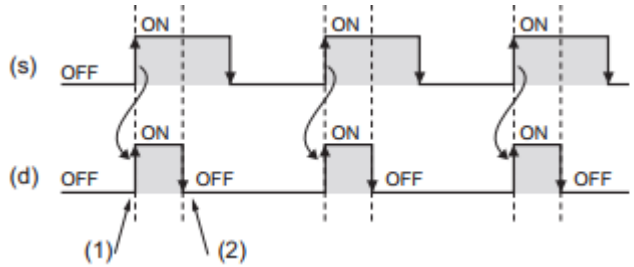


Tác dụng của cổng EN cũng tương tự như ở khối SR(_E).

c). Khối phát hiện sườn lên **R_TRIG(_E).** Khối này phát hiện sự xuất hiện của tín hiệu đầu vào (s1) và đưa ra (d) một xung kéo dài một chu kỳ quét.

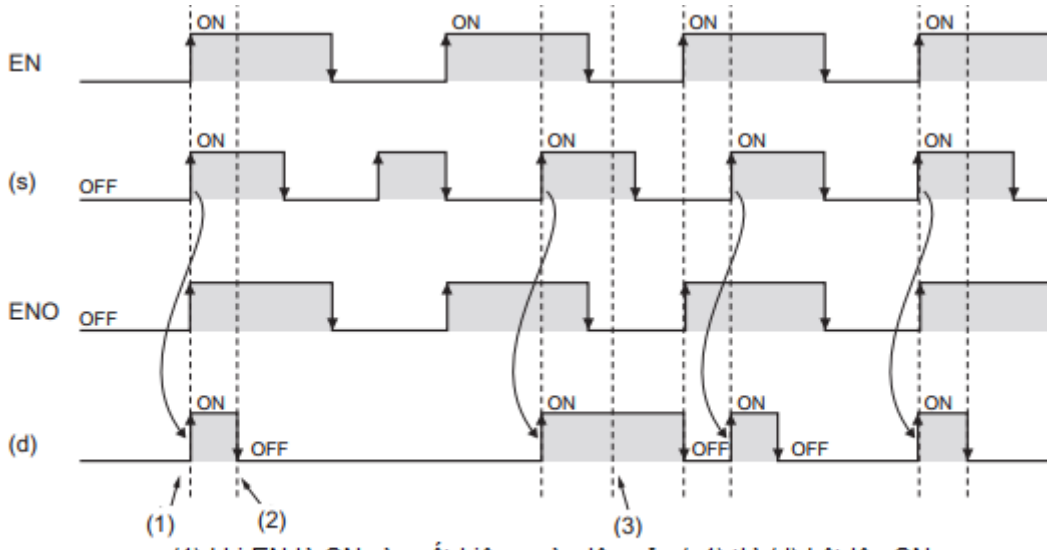


Biểu đồ hoạt động theo thời gian



(1) Khi đầu vào (s1) xuất hiện sườn lên thì (d) bật ON.
(2) Sau một chu kỳ quát thì (d) tự trở về trạng thái OFF.

Trường hợp dùngcổng EN thì ảnh hưởng của cổng này thể hiện ở biểu đồ thời gian sau đây.



(1) khi EN là ON và xuất hiện sườn lên của (s1) thì (d) bật lên ON
(2) Sau một chu kỳ quét thì (d) về OFF
(3) Khi EN =OFF thì (d) giữ nguyên trạng thái cũ.

d). Khối phát hiện sườn xuống **F_TRIG(_E).** Hoạt động hoàn toàn tương tự như khối phát hiện sườn dương nhưng là phát xung ra với độ dài một chu kỳ quét khi phát hiện tín hiệu (s) đang có thì mất đi.