# CHƯƠNG 1: GIỚI THIỆU PLC S7-1200

## 1.1. Tổng quan về PLC S7-1200

PLC viết tắt của Programmable Logic Controller là thiết bị điều khiển lập trình được cho phép thực hiện linh hoạt các thực toán điều khiển logic thong qua một ngôn ngữ lập trình. người sử dụng có thể lập trình để thực hiện môt loạt trình tự các sự kiện. Các sự kiện này được kích hoạt bởi tác nhân kích thích tác động vào plc hoặc qua các hoạt động có trễ như thời gian định kì hay thời gian được đếm. Một khi sự kiện được kích hoạt thật sự, nó bật ON hay OFF các thiết bị điều khiển bên ngoài được gọi là thiết bị vật lý. Một bộ điều khiển lập trình sẽ liên tục lặp trong chương trình do người sử dụng lập ra chờ tín hiệu ở ngõ vào và xuất tín hiệu ở ngõ ra tại các thời điểm đã lập trình.

Để khắc phục những nhược điểm của bộ điều khiển dung dây nối, người ta đã chế tao bộ điều khiển plc nhằm thoả mãn các yêu cầu sau:

+Lập trình dễ dàng, ngôn ngữ lập trình dễ học

+Gọn nhẹ, dễ bảo quản, sửa chữa

+Dung lượng bộ nhớ lớn để có thể chứa được những chương trình phức tạp

+Hoàn toàn tin cậy trong môi trường công nghiệp

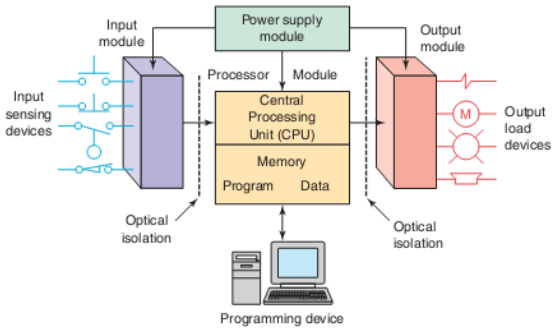
+Giao tiếp được với các thiết bị thông minh khác như máy tính, nối mạng, các module mở rộng

Các thiết kế đầu tiên là nhằm thay cho các phần cứng Relay dây nối và các logic thời gian. Tuy nhiên bên canh đó việc đòi hỏi tăng cường dung lượng nhớ và tính dễ dàng cho PLC mà vẫn đảm bảo tốc độ xử lí cũng như giá cả….

Chính điều này đã tạo ra sự quan tâm sâu sắc đến việc sử dụng PLC trong công nghiệp, các tập lệnh nhanh chống đi từ các lệnh logic đơn giản đến các lệnh đếm, định thời, thanh ghi dịch…Sự phát triển các máy tính dẫn đến các bộ PLC có dung lượng lớn, số lượng I/O nhiều hơn.

Trong PLC phần cứng CPU và chương trình là đơn vị cơ bản cho quá trình điều khiển và sử lí hệ thống, chức năng mà bộ điều khiển cần thực hiện sẽ được xác định bằng một chương trình. Chương trình này sẽ được nạp sẵn vào bộ nhớ của

PLC, PLC sẽ thực hiện việc điều khiển dựa vào chương trình này. Như vậy nếu muốn thay đổi hay mở rộng chức năng cửa quy trình công nghệ. Ta chỉ cần thay đổi chương trình bên trong bộ nhớ PLC. Việc thay đổi hay mở rộng chức năng sẽ được thực hiện một cách dễ dàng mà không cần một sự can thiệp vật lí nào so với các bộ dây nối hay Relay.




## 1.2. Các dòng sản phẩm của SIEMENS

LOGO

S7-200

S7-1200

S7-300

S7-400

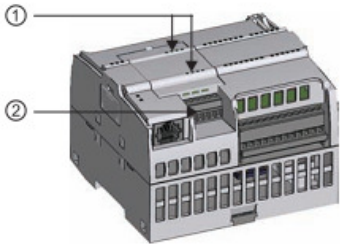
HMI

## 1.3. Cấu hình và điều hành SIMATIC S7-1200

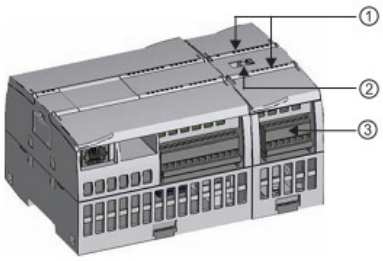
### 1.3.1. Signal boards

Signal Board

Kết nối hệ thống dây dẫn




### 1.3.2. Signal modules




Trạng thái đèn LED của I / O module tín hiệu

Bus kết nối

Kết nối hệ thống dây

### 1.3.3. Các mođun truyền thông

Trạng thái đèn LED cho các module giao tiếp

Truyền thông kết nối


## 1.4. Những đặc điểm nổi bật của Simatic S7 – 1200.

### 1.4.1.Thiết kế dạng Module.

+ Tích hợp cổng truyền thông Profinet (Ethernet) tạo sự dễ dàng trong kết nối.

+ Simatic S7 – 1200 với Simatic HMI Basic được lập trình chung trên một nền phần mềm là TIA Portal V10.5 (Simatic Step 7 Basic, WinCC Basic) hoặc version cao hơn. Các thao tác lập trình thực hiện theo cách kéo – thả, do đó tạo sự dễ dàng cho người sử dụng, lập trình nhanh chóng, đơn giản, chính xác trong sự truyền thông kết nối theo tags.

+ Tích hợp sẵn các đầu vào ra, cùng với các board tín hiệu, khi cần mở rộng ứng dụng với số lượng đầu vào ra ít sẽ tiết kiệm được chi phí, không gian và phần cứng.

+ Dễ dàng cho người sử dụng sản phẩm trong việc mua gói thiết bị.

**1.4.2 Phạm vi ứng dụng của Simatic S2 1200:**

+ S7 – 1200 bao gồm các họ CPU 1211C, 1212C, 1214C. Mỗi loại CPU có những tính năng khác nhau, thích hợp cho từng loại ứng dụng.

+ Các kiểu cấp nguồn và đầu vào ra có thể là DC/DC/DC hay DC/DC/Rly

+ Đều có khe cắm thẻ nhớ, dùng cho khi mở rộng bộ nhớ cho CPU, copy chương trình ứng dụng hay cập nhật firmware.

+ Chẩn đoán lỗi online/offline.

+ Một đồng hồ thời gian thực cho các ứng dụng thời gian thực

**2.4.1. Các chức năng nổi bật của CPU 1214C**

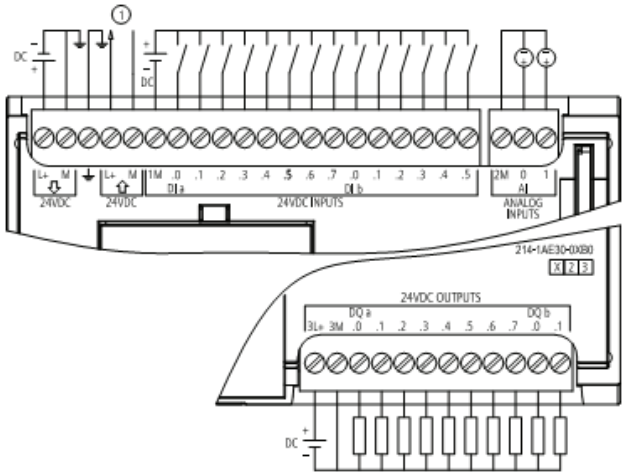
+ Có 6 bộ đếm tốc độ cao HSC dùng cho các ứng dụng đếm và đo lường.

+ Có 2 ngõ ra PTO 100kHz để điều khiển tốc độ, động cơ bước hay servo.

+ Có ngõ ra PWM điều chế độ rộng xung cho các ứng dụng điều khiển tốc độ động cơ, valve, nhiệt độ.

+ Có 16 bộ điều khiển PID với tính năng tự động xác định thông số cho bộ điều khiển (Autotuning)

### 1.4.2 Sơ đồ đấu dây PLC CPU 1214C DC/DC/DC




① 24 VDC Sensor Power Out

Figure A-9 CPU 1214C DC/DC/DC (6ES7 214-1AE30-0XB0)

### 1.4.3.Board tín hiệu của S7-1200

+ Board tín hiệu – một dạng module mở rộng tín hiệu vào/ra với số lượng tín hiệu ít, giúp tiết kiệm chi phí cho các ứng dụng yêu cầu mở rộng số lượng tín hiệu ít

Gồm các board:

1 cổng tín hiệu ra analog 12 bit (0-10VDC, 0-20mA)

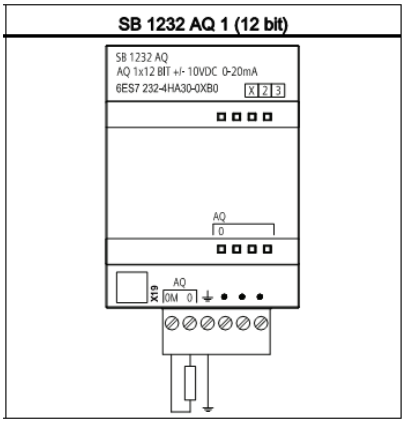


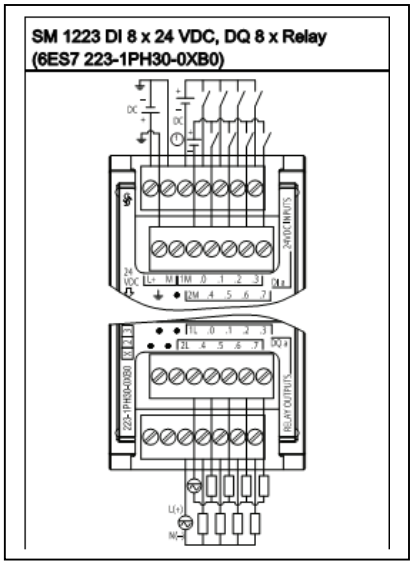

Table A- 127 Analog outputs

| Technical data | SB 1232 AQ 1 x 12 bit |
|---|---|
| Number of outputs | 1 |
| Type | Voltage or current |
| Range | ±10 V or 0 to 20 mA |
| Resolution | Voltage: 12 bits<br>Current: 11 bits |

| Technical data | SB 1232 AQ 1 x 12 bit |
|---|---|
| Full scale range (data word) | Voltage: -27,648 to 27,648 |
| Refer to the output ranges for voltage and current | Current: 0 to 27,648 |

### 1.4.4. Modules mở rộng tín hiệu vào/ra

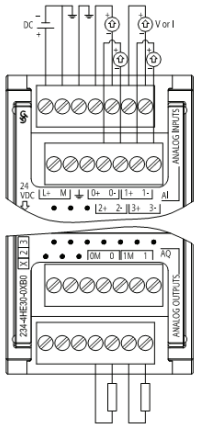
Các module mở rộng tín hiệu vào/ra được gắn trực tiếp vào phía bên phải của CPU. Với dải rộng các loại module tín hiệu vào/ra số và analog, giúp linh hoạt trong sử dụng S7-1200. Tính đa dạng của các module tín hiệu vào/ra sẽ được tiếp tục phát triển.




### 1.4.2 Module Analog

+ SM – tín hiệu module cho các đầu vào và đầu ra Analog

(cho CPU 1212C tối đa của 2 SM có thể sử dụng, cho 1214C tối đa là 8)



### 1.4.2.Module truyền thông

+ Giao tiếp với RS 232/RS 485



### 1.4.3.Thẻ nhớ

+ SIMATIC thẻ nhớ 2MB hoặc 24MB cho các chương trình lưu trữ dữ liệu và thay thế CPU đơn giản để bảo trì



### Module nguồn

+ Sử dụng module nguồn PM 1207 có các thông số: Input: 120/230V AC 50/60Hz, 1.2A/0.7A Output: 24V DC / 2.5A

### Switch

+ Module CSM1277 có 4 cổng cắm RJ45, tốc độ 10/100Mb/s

## 1.5 Cấu trúc và nguyên lý hoạt động

### 1.5.1.Cấu trúc

Tất cả PLC đều có thành phần chính là một bộ nhớ chương trình RAM bên trong, một bộ vi xử lý có cổng giao tiếp dùng cho việc ghép nối với PLC, các module I/O.

Bên cạnh đó, một số PLC hoàn chỉnh còn đi kèm theo một đơn vị lập trình bằng tay hay bằng máy tính. Hầu hết các đơn vị lập trình đơn giản đều có đủ RAM để chứa đựng chương trinh dưới dạng hoàn thiện hay bổ sung. Nếu đơn vị lập trình là đơn vị sách tay, RAM thường là loại CMOS có pin dự phòng, chỉ khi nào chương trình đã được kiểm tra và sẵn sang sử dụng thì nó mới truyền sang bộ nhớ PLC. Đối với các PLC lớn thường lập trình trên máy tính nhằm hỗ trợ cho viết, đọc và kiểm tra chương trình. Các đơn vị lập trình nối với PLC qua cổng RS232, RS422, RS458….

### 1.5.2 Nguyên lý hoạt động của PLC

CPU điều khiển các hoạt động bên trong PLC. Bộ xử lý sẽ đọc và kiểm tra chương trình được chứa trong bộ nhớ, sau đó sẽ thực hiện từng lệnh trong chương trình, sẽ đóng hay ngắt các đầu ra. Các trạng thái ngõ ra ấy được phát tới các thiết bị liên kết để thực thi và toàn bộ các hoat động thực thi đó đều phụ thuộc vào chương trình điều khiển được giữ trong bộ nhớ.

Hệ thống bus là tuyến dùng để truyền tín hiệu, hệ thống gồm nhiều đường tín hiệu song song:

+Address bus:bus địa chỉ dùng để truyền địa chỉ tới các module khác nhau

+Data bus:bus dùng để truyền dữ liệu

+Control bus:bus điều khiển dung để truyen các tín hiệu định thì và điều khiển đồng bộ các hoạt động trong PLC

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

### 1.5.3. Đèn tín hiệu PLC

Có 3 loại đèn báo hoạt động:

- Run/stop: đèn xanh/đèn vàng báo hiệu PLC đang hoạt động/dừng hoạt động
- Error: đèn báo lỗi
- Maint: đèn báo khi ta buộc (Force) địa chỉ nào đó lên 1

Có 2 loại đèn chỉ thị:

- Ix.x: chỉ trạng thái logic ngõ vào.
- Qx.x: chỉ trạng thái logic ngõ ra.

### 1.5.4. Bộ nhớ PLC

PLC thường yêu cầu bộ nhớ trong các trường hợp: làm bộ định thời cho các kênh trạng thái I/O. Làm bộ đệm trạng thái các chức năng trong PLC như định thời, đếm, gọi các Relay.

Mỗi lệnh của chương trình có một vị trí riêng trong bộ nhớ, tất cả các vị trí trong bộ nhớ đều được đánh số, những số này chính là địa chỉ trong bộ nhớ. Địa chỉ của từng ô nhớ sẽ được trỏ đến bởi một bộ đếm địa chỉ nằm bên trong bộ vi xử lý. Bộ vi xử lý sẽ có giá trị trong bộ đếm này thêm một trước khi xử lý lệnh tiếp theo. Với một địa chỉ mới, nội dung của ô nhớ tương ứng sẽ xuất hiện ở đầu ra, quá trình này gọi là quá trình đọc.

Bộ nhớ bên trong của PLC được tạo bởi vi mạch bán dẫn, mỗi vi mạch này có khả năng chứa 2000-16000 dòng lệnh tuỳ theo loại vi mạch trong PLC các bộ nhớ như RAM và EPROM đều được sử dụng

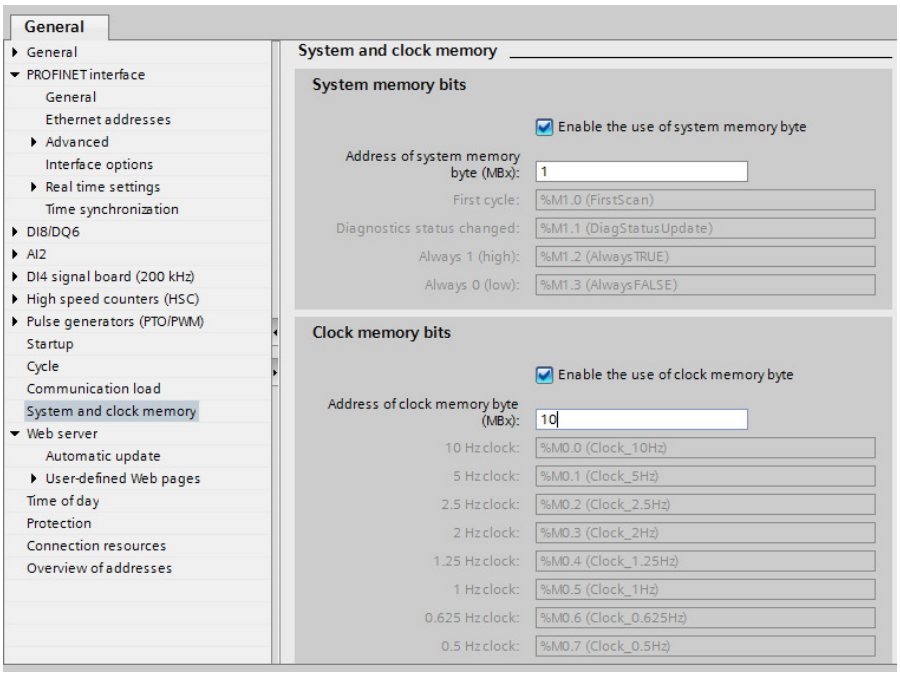
+RAM có thể nạp chương trình, thay đổi hay xoá bỏ nội dung bất kì lúc nào, nội dung của RAM sẽ bị mất nếu nguồn điện nuôi bị mất. Để tránh tình trạng này các PLC đều được trang bị pin khô có khả năng cung cấp năng lượng dự trữ cho

RAM từ vài tháng đến vài năm. Trong thực tế RAM được dung khởi tạo và kiểm tra chương trình. Khuynh hướng hiện nay dung CMOSRAM do khả năng tiêu thụ thấp và tuổi thọ cao

+EPROM là bộ nhớ mà người sử dụng bình thường có thể đọc chứ không ghi nội dung vào được, nội dung của EPROM không bị mất khi mất nguồn, nó được gắn sẵn trong máy, đã dược nhà sản xuất nạp và chứa sẵn hệ điều hành. Nếu người sử dụng không muốn sử dụng bộ nhớ thì chỉ dùng EPROM gắn bên trong PLC. Trên PG có sẵn chổ ghi và xoá EPROM

+EEEPROM liên kết với những truy xuất linh động của RAM và có tính ổn định. Nội dung của nó có thể xoá và lập trình bằng điện tuy nhiên số lần là có giới hạn

## 1.6. Hệ thống và bộ nhớ đồng hồ



**Biến nhớ tạo xung clock**

M10. 0 (Clock_10Hz)

M10. 1 (Clock_5Hz)

M10. 2 (Clock_2. 5Hz)

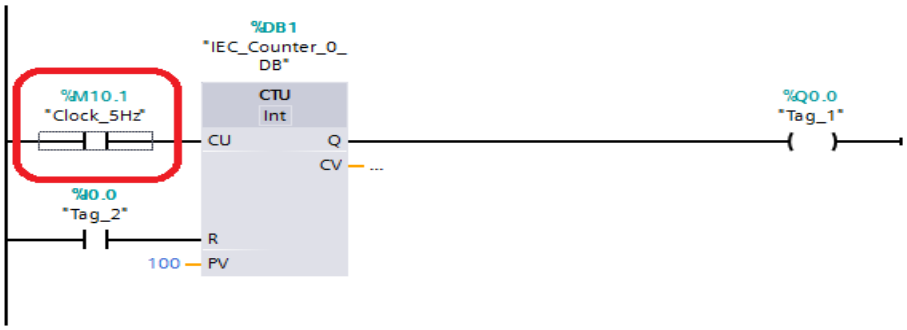
M10. 3 (Clock_2Hz)

M10. 4 (Clock_1. 25Hz)

M10. 5 (Clock_1Hz)

M10. 6 (Clock_0. 625Hz)

M10. 7 (Clock_0. 5Hz



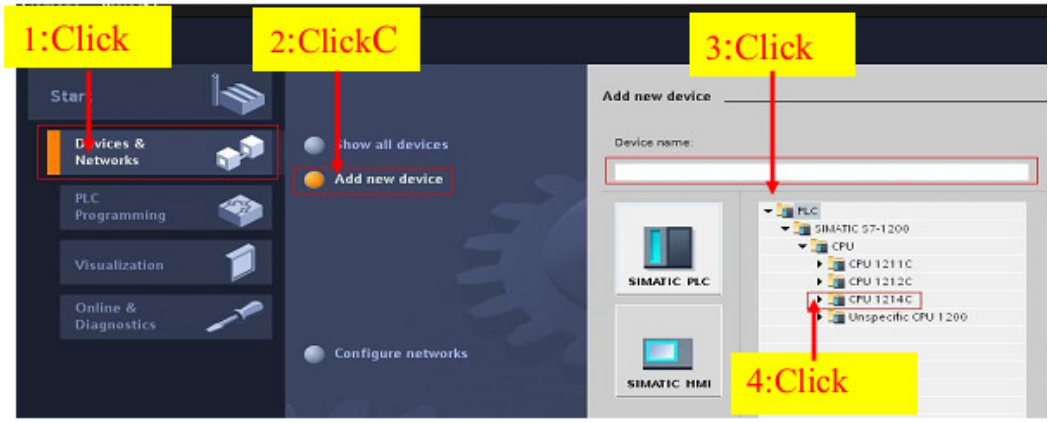
# CHƯƠNG 2: GIỚI THIỆU PHẦN MỀM TIA PORTAL V11

## 2.1. Kết nối CPU qua giao thức TCP/IP

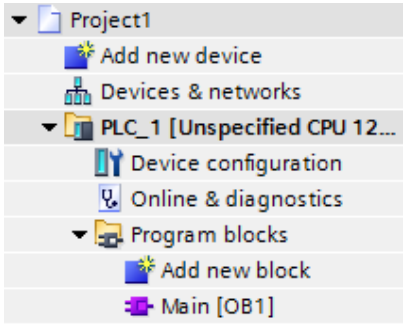
**Tạo ra 1 PROJECT S7-1200**:

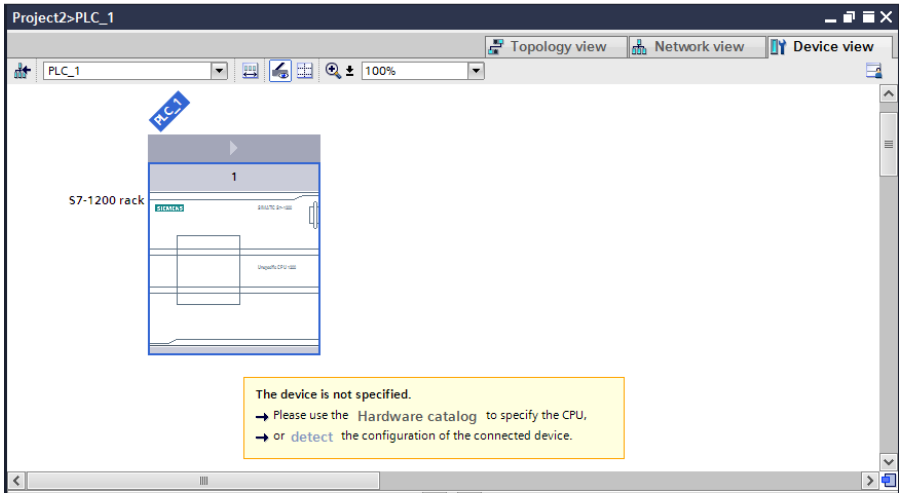
- 1 : Tạo 1 project mới
- 2 : Đặt tên cho Project
- 3 : Xác nhận tạo Project.

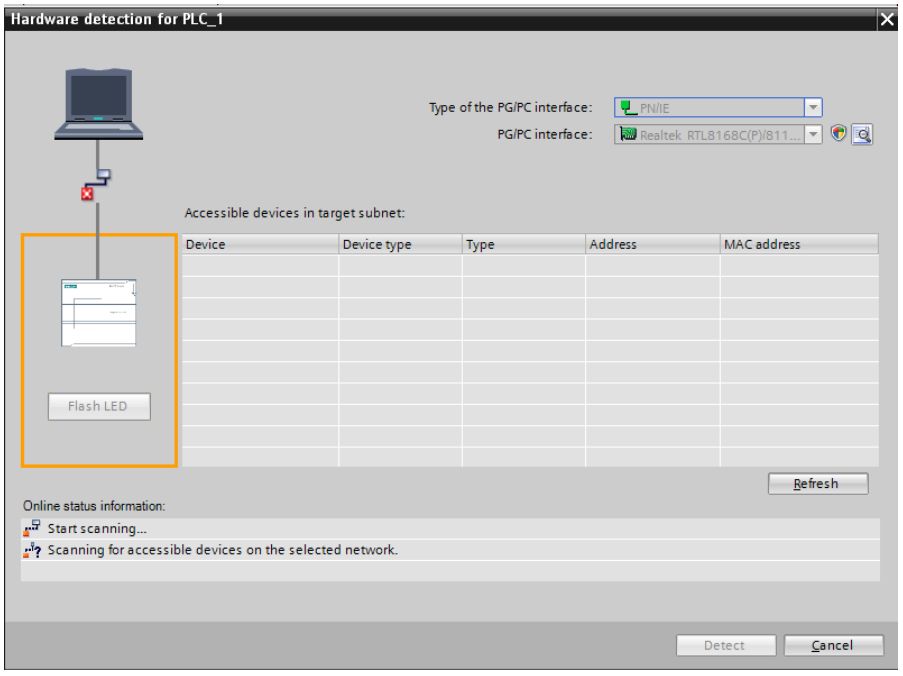

**Chọn loại CPU S7-1200**.CPU được chon : **CPU 1214C**

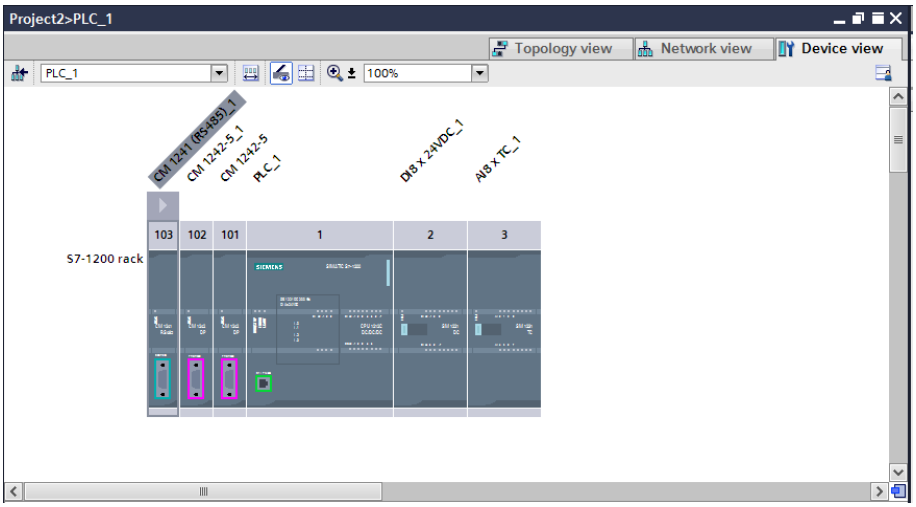


## 2.2. Cấu hình CPU

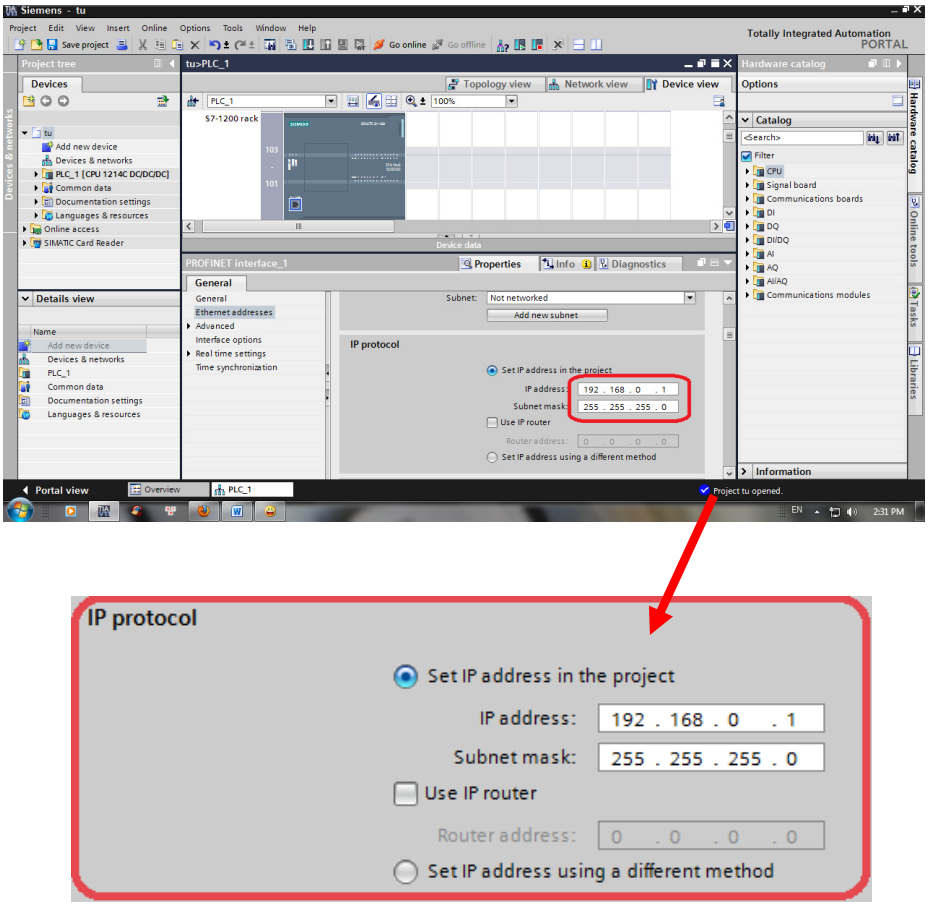


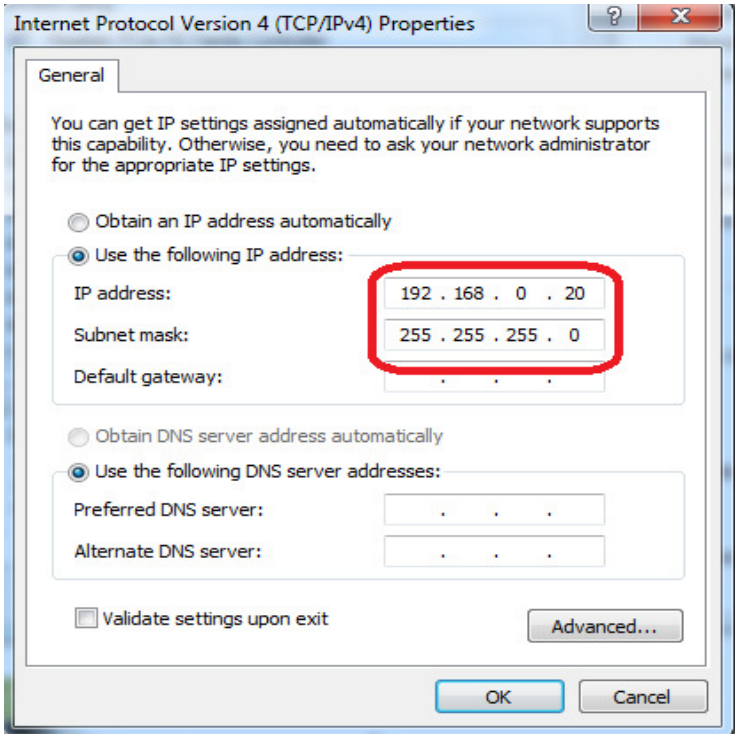



## 2.3. Địa chỉ IP mặc định của PLC S7 1200

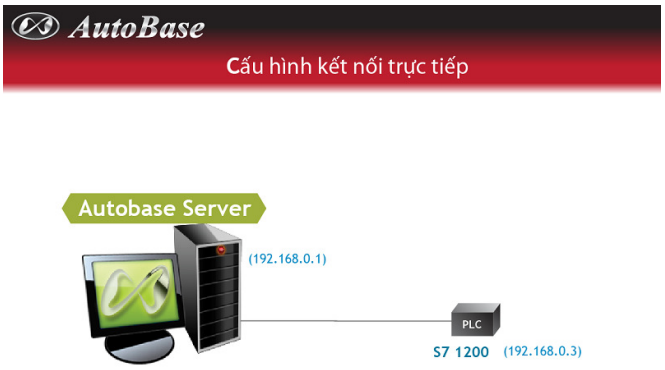


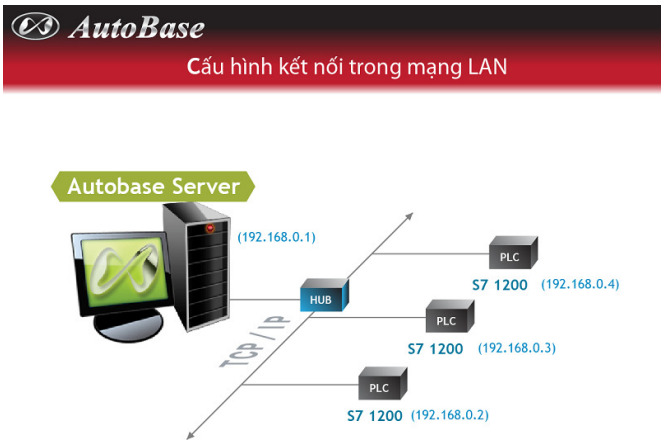
## 2.4. Cấp địa chỉ IP cho PC hoặc Laptop

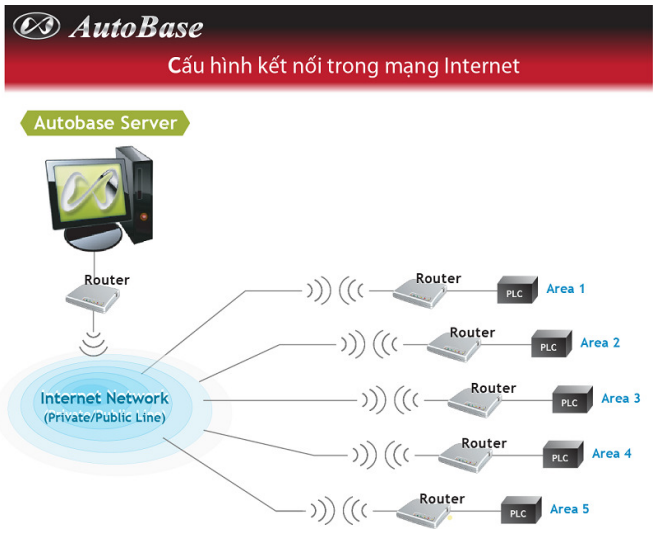
- Để lập trình SIMATIC S7-1200 từ PC, PG hay Laptop, cần một kết nối TCP/IP.
- Để PC và SIMATIC S7-1200 có thể giao tiếp với nhau, điều quan trọng là các địa chỉ IP của cả hai thiết bị điều phải phù hợp với nhau.
- Các bước thiết lập IP cho máy tính:
  - Chọn Network connections/ Properties của kết nối mạng LAN (Start → Setting → System control → Network connections → Local Area Connection → Properties)
  - Chọn 'Properties' từ 'Internet Protocol (TCP/IP)' → (Internet Protocol (TCP/IP) → Properties).
  - Thiết lập IP address và Subnet screen form, và chấp nhận với OK (→ Use the following IP address → IP address: 192. 168. 0. 99 → Subnet screen 255. 255. 255. 0 → OK → Close).



## 2.5. Kết nối máy tính với PLC

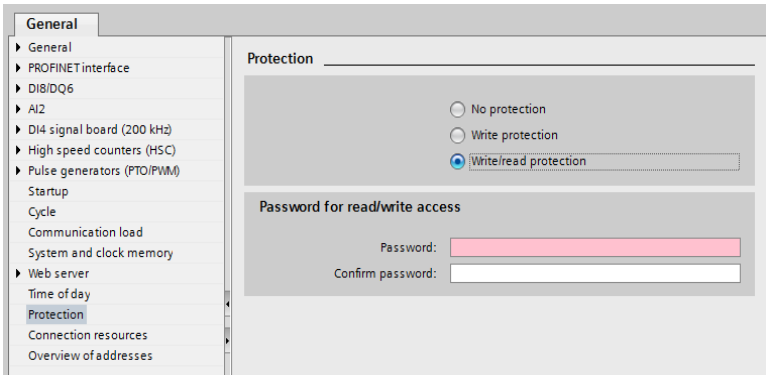


## 2.6. Kết nối Profinet

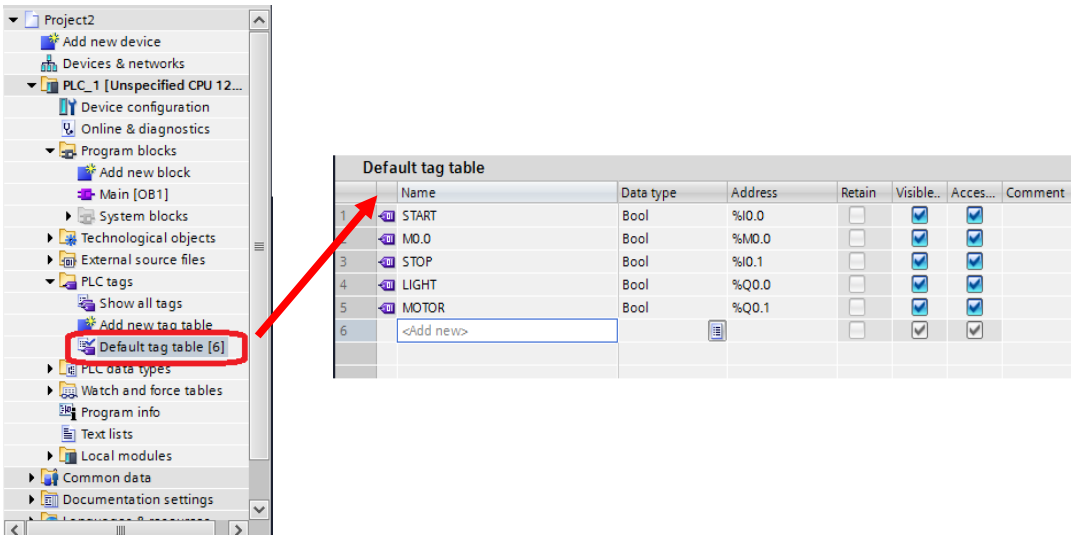
## 2.7. Mật khẩu bảo vệ cho CPU S7-1200

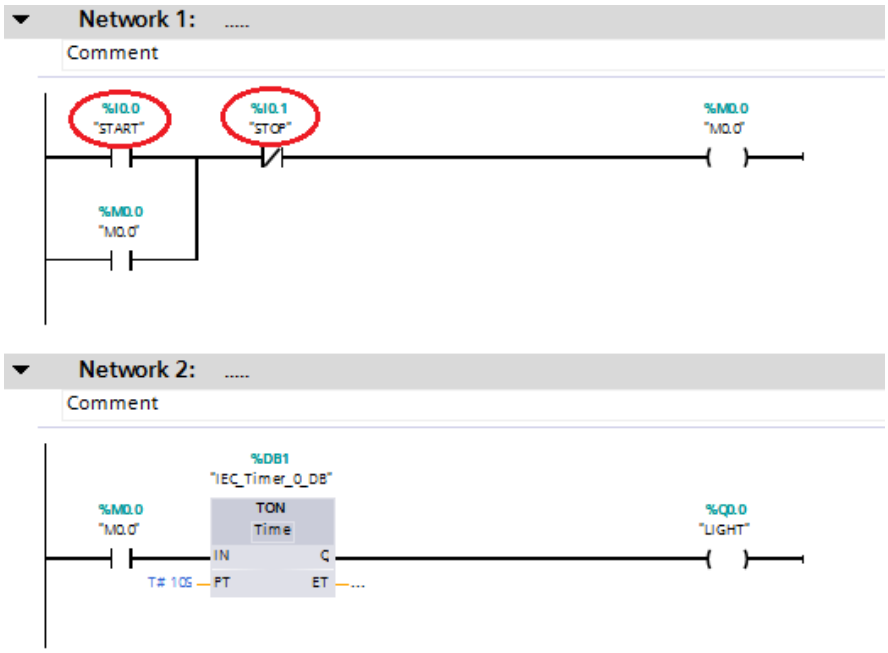



## 2.8. Khởi tạo bảng tag mới

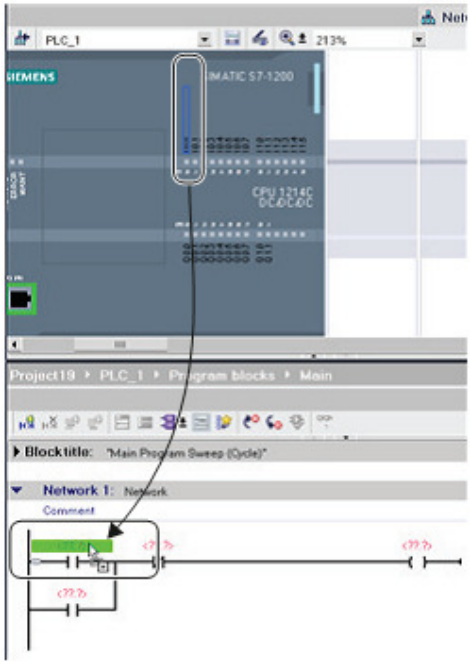
**Có 2 cách:**

**1.Tạo bảng tag**

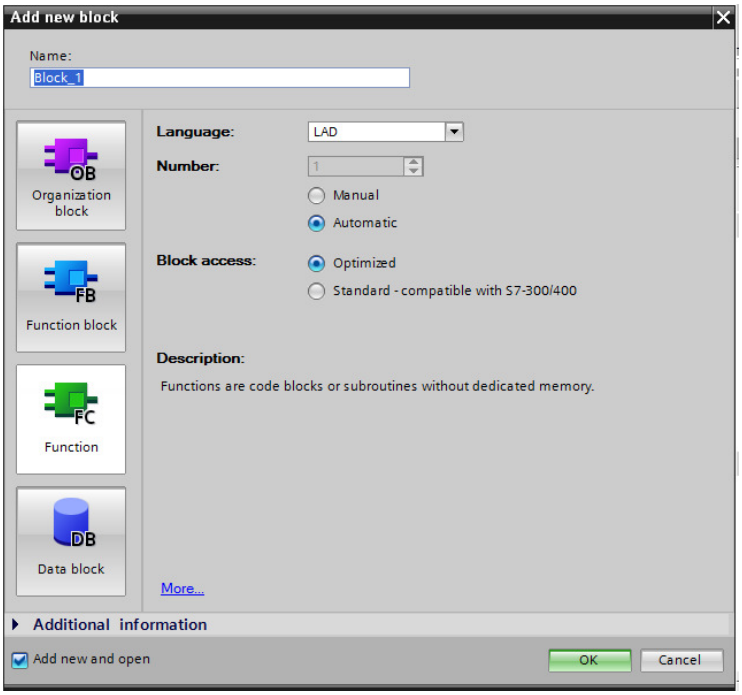





1. **Kéo thả vào địa chỉ plc**



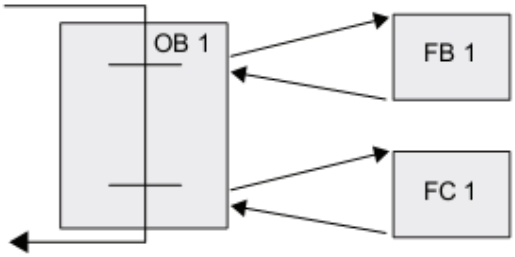
## 2.9. Khối chương trình

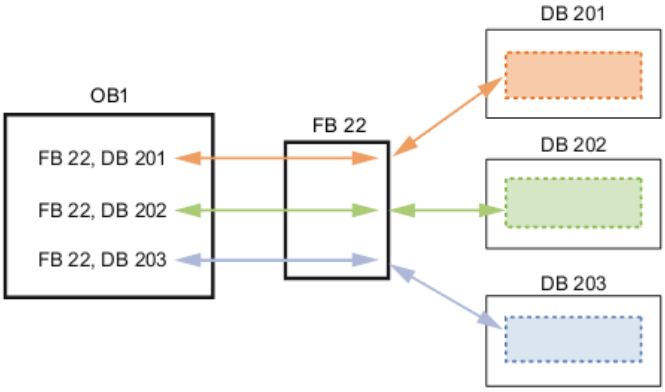



## 2.10. Khối tổ chức OB – Oganization Blocks

1. Oganization blocks (Obs): là giao diện giữa hoạt động hệ thống và chương trình người dùng. Chúng được gọi ra bởi hệ thống hoạt động và điều khiển theo quá trình:
   - Xử lý chương trình theo chu kỳ.
   - Báo động – kiểm soát xử lý chương trình.
   - Xử lý lỗi.
2. Tùy chọn khác nhau để sử dụng khối OB trong chương trình:
   - Startup OB, Cycle OB, Timing Error OB and Diagnosis OB.
   - Process Alarm OB and Time Interrupt OB.
   - Time Delay Interrupt OB.
3. Hàm chức năng – Function
4. Functions (FCs) là các khối mã không cần bộ nhớ. Dữ liệu của các biến tạm thời bị mất sau khi FC được xử lý. Các khối dữ liệu toàn cầu có thể được sử dụng để lưu trữ dữ liệu FC.

5. Functions có thể được sử dụng với mục đích:
   - Trả lại giá trị cho hàm chức năng được gọi.
   - Thực hiện công nghệ chức năng, ví dụ: điều khiển riêng với các hoạt động nhị phân.
   - Ngoài ra, FC có thể được gọi nhiều lần tại các thời điểm khác nhau trong một chương trình. Điều này tạo điều kiện cho lập trình chức năng lặp đi lặp lại phức tạp.

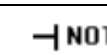
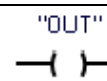


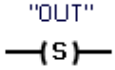
# CHƯƠNG 3: TẬP LỆNH CƠ BẢN CỦA PLC S7-1200

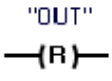
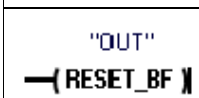
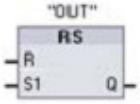
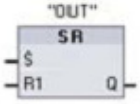
## 3.1. Các tập lệnh cơ bản

### 3.1.1. Công tắc và cuộn coil

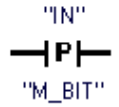
| Ký hiệu | Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|---|
| Công tắc | | | |
|  | IN | BOOL | Công tắc thường đóng hay thường mở. Các vùng nhớ có thể sử dụng la I, Q, M, D. Để có thể đọc ngay lạp tức ngõ vào có thể sử ngõ vào vật lý thay vì biến quá trình. |
|  | | | |
| Lệnh logic NOT | | | |
|  | IN/OUT | BOOL | Đảo trạng thái ngõ vào/ra |
| Cuộn coil | | | |
|  | OUT | BOOL | Trạng thái ngõ ra là kết quả xử lý của phép toán logic |
|  | OUT | BOOL | Đảo kết quả ngõ ra của phép toán logic. |

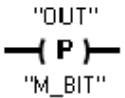
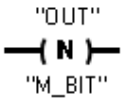
### 3.1.2. Lệnh Set và Reset

| Ký hiệu | Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|---|
| Lệnh Set và Reset 1 bit | | | |
|  | OUT | BOOL | Khi lệnh Set được tác động thì địa chỉ ngõ ra sẽ được đặt lên 1. |

| | | | |
|---|---|---|---|
|  | OUT | BOOL | Khi lệnh Reset được tác động thì địa chỉ ngõ ra sẽ được trở về 0. |
| Lệnh Set và Reset nhiều bit | | | |
|  | OUT | BOOL | Khi lệnh SET_BF được tác động, một chuỗi gồm "n" bit sẽ được đặt lên 1 bắt đầu tại địa chỉ OUT. |
|  | OUT | BOOL | Khi lệnh RESET_BF được tác động, một chuỗi gồm "n" bit sẽ được trở về 1 bắt đầu tại địa chỉ OUT. |
| Lệnh SR và RS fliplop | | | |
|  | S1, R | BOOL | Mạch chốt RS ưu tiên Set |
| | OUT | BOOL | |
|  | R,S1 | BOOL | Mạch chốt SR ưu tiên Reset |
| | OUT | BOOL | |

### 3.1.3. Lệnh nhận biết xung cạnh lên P và xung cạnh xuống N

| Ký hiệu | Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|---|
| Nhận biết xung cạnh lên và cạnh xuống | | | |
|  | IN | BOOL | Phát hiện sự thay đổi trạng thái của 1 tín hiệu vận hành (IN) từ 0->1 (Thay đổi trạng thái tín hiệu phía trước không ảnh hưởng gì đến IN). Khi đó ngõ ra mức 1, tất cả trường hợp còn lại đều mức 0. trạng thái của IN sẽ được lưu trữ trong "M_BIT" |

| | | | |
|---|---|---|---|
|  | IN | BOOL | Phát hiện sự thay đổi trạng thái của 1 tín hiệu vận hành (IN) từ 1->0 (Thay đổi trạng thái tín hiệu phía trước không ảnh hưởng gì đến IN). Khi đó ngõ ra mức 1, tất cả trường hợp còn lại đều mức 0. trạng thái của IN sẽ được lưu trữ trong “M_BIT“ |
|  | OUT | BOOL | Nếu có sự thay đổi tại RLO từ 0->1 thì biến nhớ OUT sẽ được set lên 1 cho 1 chu kỳ chương trình các trường hợp còn lại OUT đều bằng 0, M_BIT lưu lại trạng thái của OUT |
|  | OUT | BOOL | Nếu có sự thay đổi tại RLO từ 1->0 thì biến nhớ OUT sẽ được set lên 1 cho 1 chu kỳ chương trình các trường hợp còn lại OUT đều bằng 0, M_BIT lưu lại trạng thái của OUT |
| Lệnh P_TRIG và N_TRIG | | | |
|  | | BOOL | Khi ngõ vào clk có sự thay đổi trạng thái logic từ 0->1 sẽ phát ra 1 xung đồng thời trạng thái của tín hiệu lúc này sẽ được lưu lại vào “M_BIT“ |
|  | | BOOL | Khi ngõ vào clk có sự thay đổi trạng thái logic từ 1->0 sẽ phát ra 1 xung đồng thời trạng thái của tín hiệu lúc này sẽ được lưu lại vào “M_BIT“ |

### 3.2. LỆNH TIMER

Sử dụng lệnh Timer để tạo 1 chương trình trễ định thời. Số lượng của Timer phụ thuộc vào người sủ dụng và số lượng vùng nhớ của CPU. Mỗi timer sử dụng 16 byte IEC_Timer dữ liệu kiểu cấu trúc DB. Setp 7 tự động tạo khối DB khi lấy khối Timer.

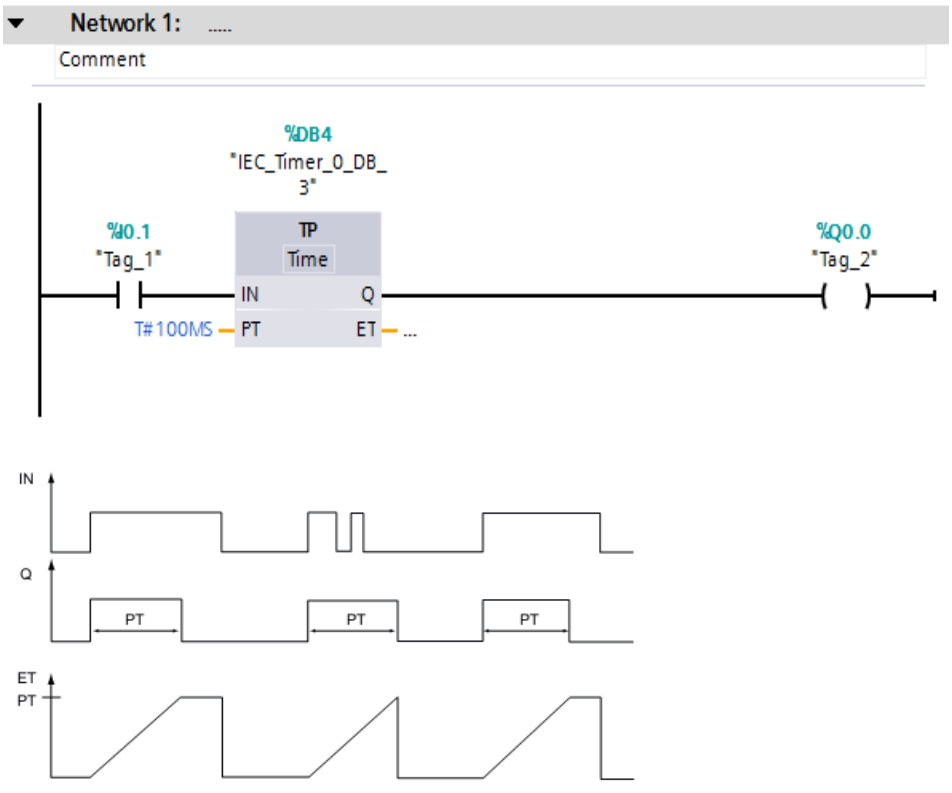
Kích thước và tầm của kiểu dữ liệu Time 32bit, lưu trữ là dữ liệu Dint.

| Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|
| IN | Bool | Ngõ vào cho phép timer hoạt đông |
| R | Bool | Reset timer |
| PT | Time | Thời gian đặt trước |
| Q | Bool | Ngõ ra |
| ET | Time | Thời gian thực hiện |

### 3.2.1.Timer TP-Timer tạo xung

Timer TP tạo một chuỗi xung với độ rộng xung đặt trước. Thay đổi PT, IN không ảnh hưởng khi timer đang chạy.

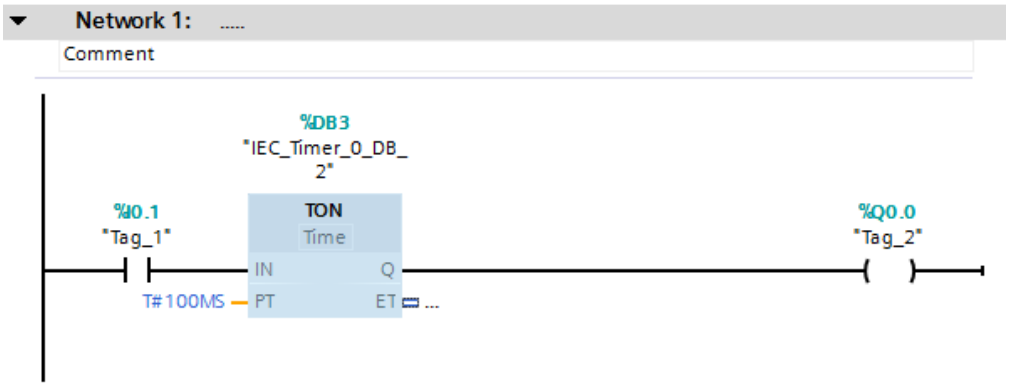
Khi đầu vào IN được tác động vào Timer sẽ tạo ra 1 xung có độ rộng bằng thời gian đặt PT.

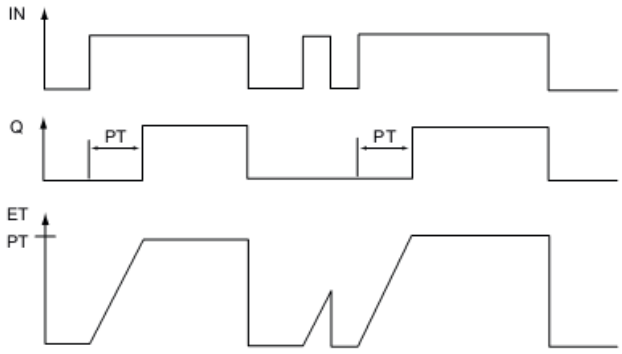


### 3.2.2.Timer TON – Timer trễ sườn lên có nhớ.

Khi ngõ vào IN được tác động và duy trì trạng thái liên tục với thời gian hơn thời gian đặt thì ngõ ra Q sẽ chuyển lên mức 1. Khi ngõ vào ngừng tác động thì reset và dừng hoạt động Timer.

Thay đổi PT khi Timer đang chạy không ảnh hưởng tới Timer

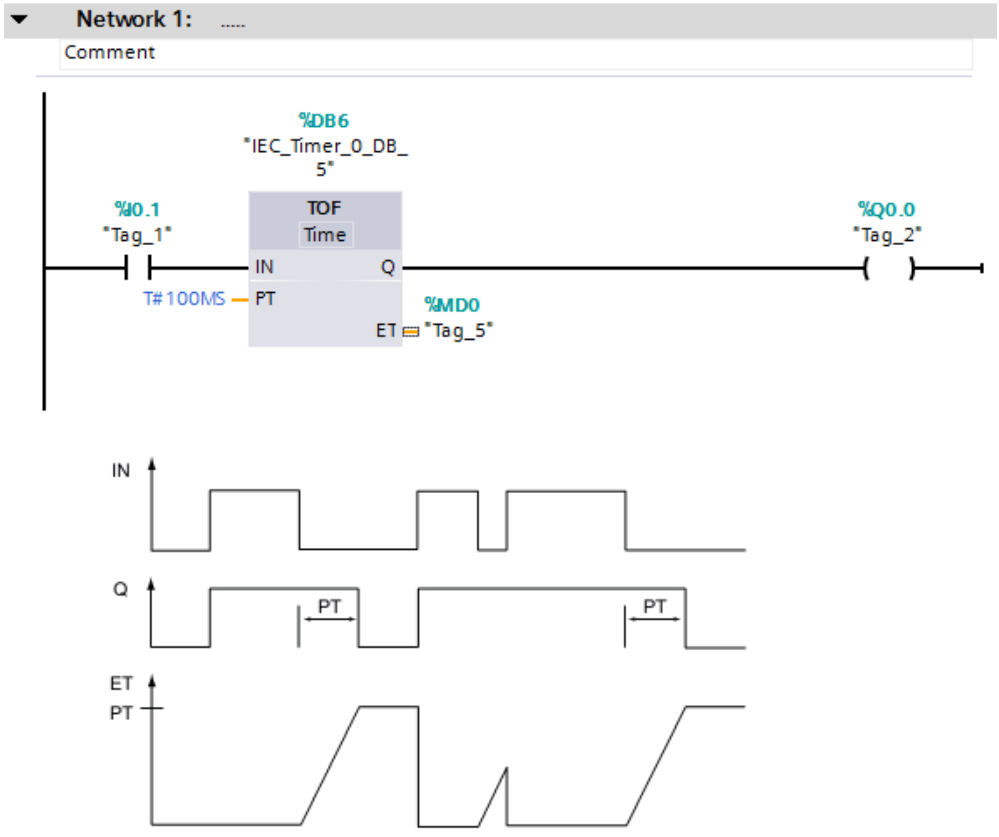



| Tham số | Khai báo | kiểu dữ liệu | Vùng nhớ | Mô tả |
|---|---|---|---|---|
| IN | Input | BOOL | I, Q, M, D, L | Ngõ vào |
| PT | Input | TIME | I, Q, M, D, L or constant | Giá trị của tham số PT phải là tích cực |
| Q | Output | BOOL | I, Q, M, D, L | Đầu ra được thiết lập khi thời gian PT hết. |
| ET | Output | TIME | I, Q, M, D, L | Giá trị thời gian hiện tại |

### 3.2.3.Timer TOF – Timer trễ sườn xuống

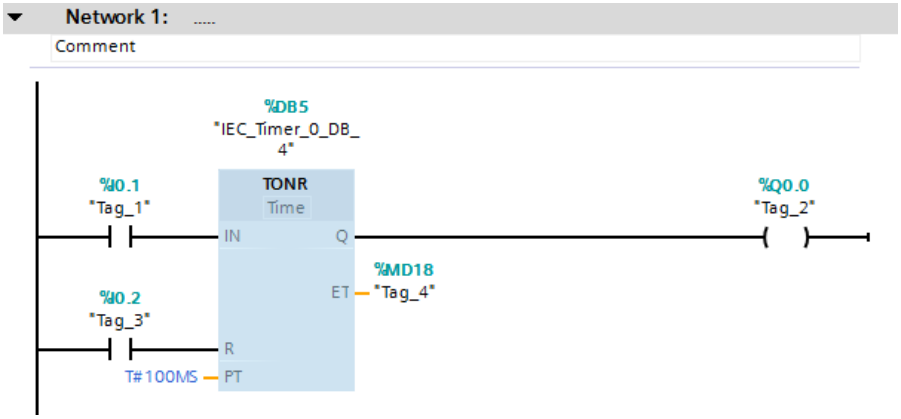
Khi ngõ vào tác động thì timer sẽ tác động và tiếp điểm thường hở của timer sẽ chuyển trạng thái lên 1. Khi ngõ vào ngừng tác động thì sau khoảng thời gian PT thì timer sẽ ngừng tác động




### 3.2.4 Timer TONR – Timer trễ sườn lên có nhớ

Khi tổng thể tác động của ngõ vào lớn hơn hay bằng thời gian đặt PT thì Timer sẽ được tác động và tiếp điểm thường mở của Timer sẽ chuyển lên mức 1. Và khi trạng thái Reset của Timer bị tác động thì Timer ngừng hoạt động và bị Reset lại.

Ví dụ:




### 3.3 Counter

Lệnh được dùng để điếm các sự kiện ở ngoài hay các sự kiện quá trình ở trong PLC. Mỗi Counter sử dụng cấu trúc lưu trữ của khối dữ liệu – DB – để làm dữ liệu của Counter. Step 7 tự động tạo DB khi lấy lệnh.

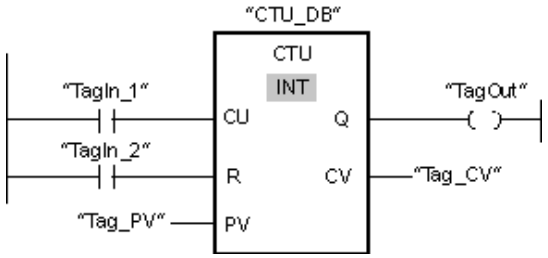
| Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|
| CU, CD | Bool | Đếm lên hay đếm xuống |
| R | Bool | Reset giá trị đếm về 0 |
| LOAD (CTD, CTUD) | Bool | Load giá trị đặt trước |
| PV | SInt, Int, DInt, ÚInt, UInt, UDInt | Giá trị đếm đặt trước |
| Q, QU | Bool | Mức 1 nếu CV>=PV |
| QD | Bool | Mức 1 nếu CV<=0 |
| CV | SInt, Int, DInt, ÚInt, UInt, UDInt | Giá trị đếm hiện hành |

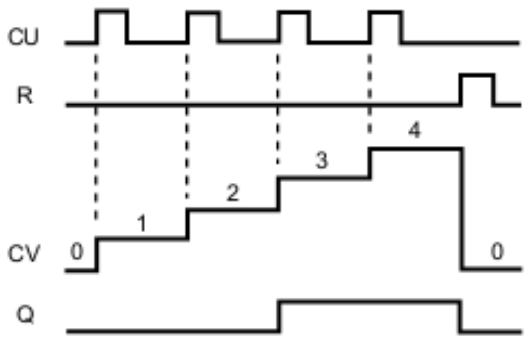
.

#### 3.3.1 Counter đếm lên – CTU

Giá trị bộ đếm CV tăng lên 1 khi tính hiệu ngõ vao CU chuyển từ 0->1. Ngõ ra Q tác động lên 1 khi CV>=PV. Nếu trạng thái R = reset được tác động thì bộ đếm CV=0

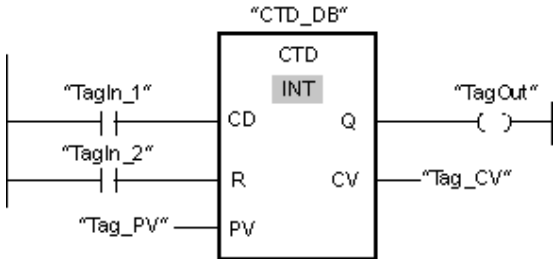
Ví dụ :

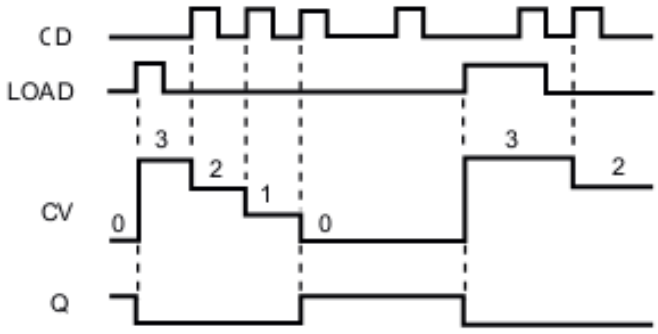



### 3.3.2 Counter đếm xuống – CTD

Giá trị bộ đếm CV được giảm 1 khi tính hiệu ngõ vào CD chuyển từ 0->1. Ngõ ra Q tác động lên 1 khi CV<=0. Nếu trạng thái Load được tác động thì CV = PV.

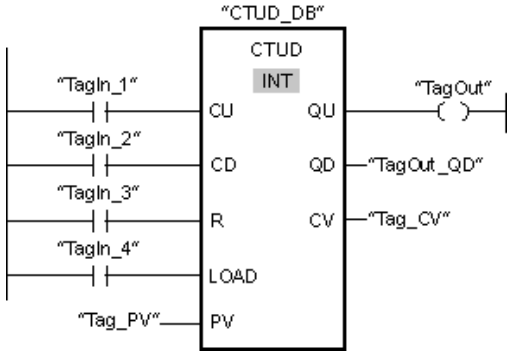


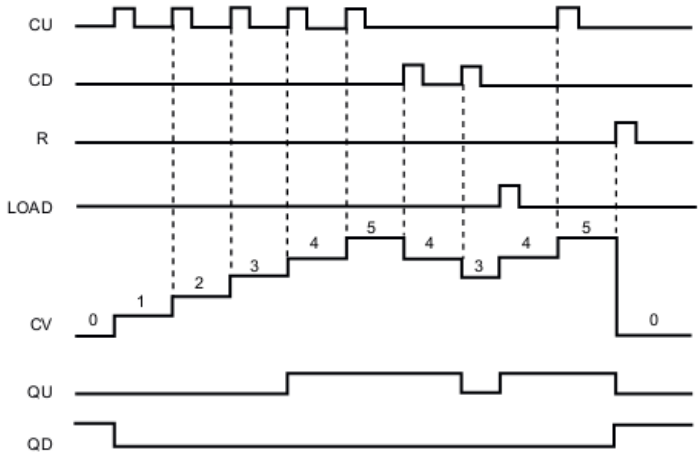



### 3.3.3 Counter đếm lên xuống – CTUD

Giá trị bộ đếm CV được tăng lên 1 khi tín hiệu ngõ vào CU chuyển từ 0->1. Ngõ vào QU được tác động lên 1 khi CV>=PV. Nếu trạng thái R = Reset được tác động thì bộ đếm CV = 0.

Giá trị bộ đếm CV được giảm 1 khi tín hiệu ngõ vào CD chuyển từ 0->1. Ngõ ra QD được tác động lên 1 khi CV<=0. Nếu trạng thái Load được tác động thì CV = PV.



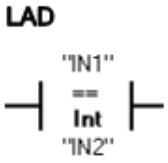




## 3.4 SO SÁNH

### 3.4.1. Lệnh CMP

So sánh 2 kiêu dữ liệu giống nhau, nếu lệnh so sánh thỏa thì ngõ ra sẽ là mức 1 = TRUE.

Kiểu dữ liệu so sánh là: Sint, Int, Dint, USInt, UDInt, Real, Lreal, String, Char, Time, DTL, Constant.

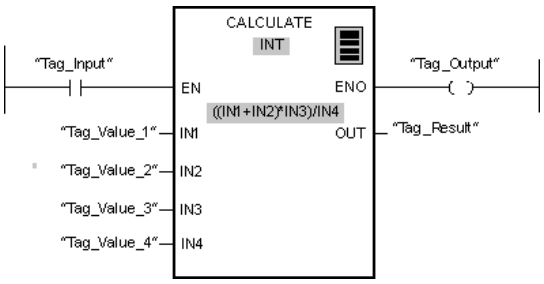



Các kiểu so sánh:

| | |
|---|---|
| = = | IN1 = IN2 |
| <> | IN1 ≠ IN2 |
| >= | IN1 >= IN2 |
| <= | IN1 <= IN2 |
| > | IN1 > IN2 |
| < | IN1 < IN2 |

### 3.4.2 Toán học

### 3.4.1 Lệnh toán học




Công dụng: thực hiện phép toán từ các ngõ vào IN1, IN2, IN(n) theo công thức OUT=…(do mình nhập vào bằng cách nhấp vào ô chính giữa khối) rồi xuất kết quả ra ngõ ra OUT. Các thông số ngõ vào dung trong khối phải chung dịnh dạng.

### 3.4.2 Lệnh Cộng, Trừ, Nhân, Chia

Cấu trúc tập lệnh Cộng, Trừ, Nhân, Chia:

- Lệnh Cộng ADD :OUT = INT1 + IN2
- Lệnh Trừ SUB : OUT = INT1 - IN2

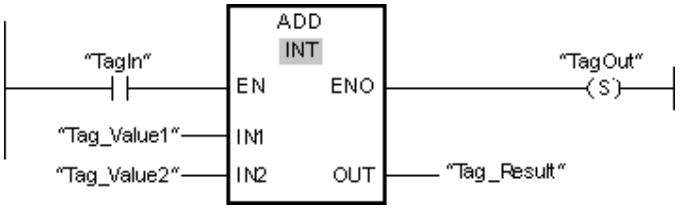
- Lệnh Nhân MUL : OUT = INT1 * IN2
- Lệnh Chia DIC : OUT = INT1 / IN2

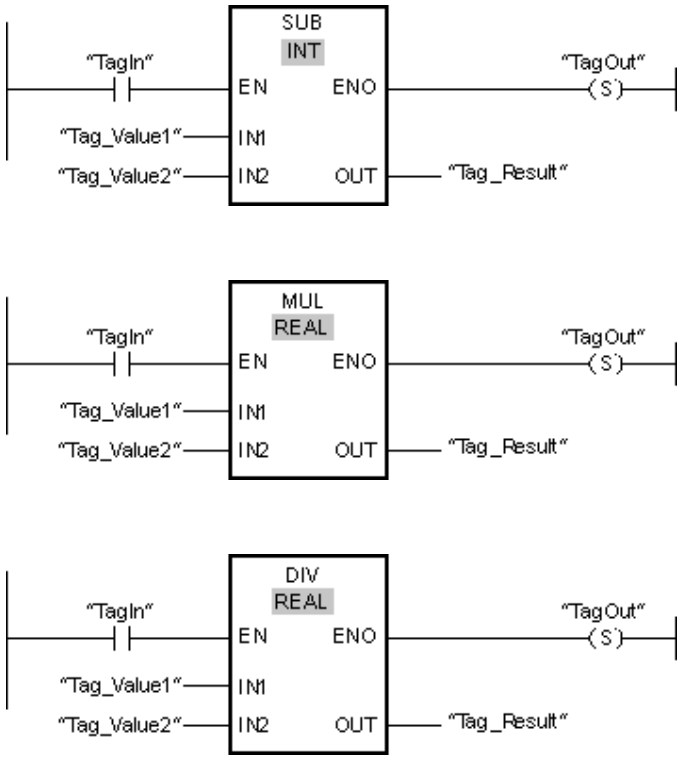
Tham số IN1, IN2 phải cùng kiểu dữ liệu ( SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, constant).

Tham số OUT có kiểu dữ liệu là SInt, Int, DInt, USInt, UInt, UDInt, Real.

Tham số ENO = 1 nếu không có lỗi xẩy ra trong quá trình thực thi. ENO = 0 nếu có lỗi xẩy ra.

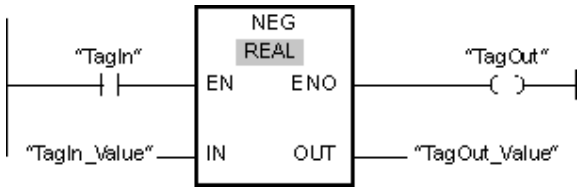
| Trạng thái | Miêu tả |
|---|---|
| 1 | Không có lỗi |
| 0 | Kếu quả toán học nằm ngoài phạm vi của kiểu dữ liệu. |
| 0 | Chia cho 0. (IN2 = 0) |
| 0 | Real/LReal: nếu một trong những giá trị đầu vào là NaN (not a number – không phải là số) sau đó được trả về NaN. |
| 0 | ADD Real/lReal: nếu cả hai giá trị IN và INF có dấu khác nhau, đây là một khai báo không hợp lệ và được trả về NaN. |
| 0 | SUB Real/LReal: nếu cả hai trị IN và INF cùng dấu, đây là một khai báo không hợp lệ và được trả về NaN |
| 0 | MUL Real/LReal: nếu một trong 2 giá trị là 0 hoặc là INF, đây là khai báo không hợp lệ vá được trả về NaN. |
| 0 | DIV Real/LReal: nếu cả hai giá trị IN bằng không INF, đây là khai báo không hợp lệ và được trả về NaN. |






### 3.4.3 Lệnh phủ định

Lệnh NEG đảo ngược dấu hiệu số học của giá trị ở trong tham số và lưu trữ các kết quả trong tham số OUT.



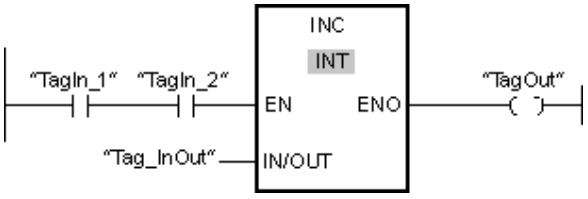

| Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|
| EN | Bool | Cho phép ngõ vào |
| ENO | Bool | Cho phép ngõ ra |
| IN | SInt, Int, DInt, Real, LReal, constant. | Toán tử đầu vào |

| | | |
|---|---|---|
| OUT | SInt, Int, DInt, Real, LReal. | Toán tử đầu ra |

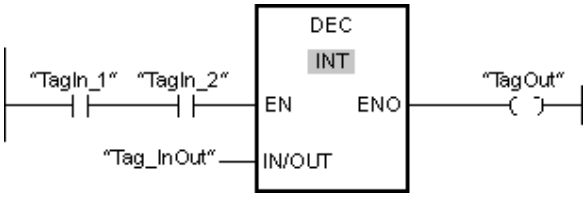
Trạng thái của ENO

- Nếu ENO = 1 : không có lỗi
- Nếu ENO = 0 : kết quả giá trị nằm ngoài tầm giá trị của kiểu dữ liệu.

### 3.4.4 Lệnh tăng, giảm




Tăng giá trị kiểu số interger lên một đơn vị. IN_OUT + 1 = IN_OUT

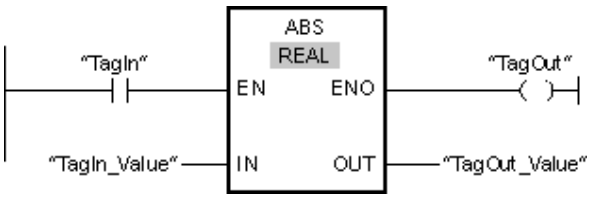



Giảm giá trị kiểu số interger xuống một đơn vị. IN_OUT - 1 = IN_OUT

| Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|
| EN | I, Q, M, L, D | Cho phép ngõ vào |
| IN/OUT | SInt, Int, DInt, USInt, UDInt | Toán tử ngõ vào và ra |
| ENO | I, Q, M, L, D | Cho phép ngõ ra |
| | 1 | Không có lỗi |
| | 0 | Kết quả giá trị nằm ngoài tầm giá trị của kiểu dữ liệu |

### 3.4.5. Lệnh giá trị tuyệt đối

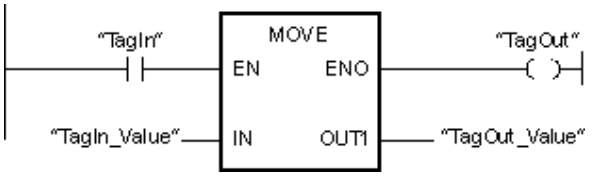
Tính giá trị tuyệt đối của một số nguyên hoặc số thực của tham số IN và lưu trữ
kết quả vào tham số OUT




| Kí hiệu | Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|---|
| | EN | I, Q, M, L, D | Cho phép ngõ vào |
| | IN | SInt, Int, DInt, Real, LReal | Toán tử ngõ vào |
| | OUT | SInt, Int, DInt, Real, LReal | Toán tử ngõ ra |
| | ENO | I, Q, M, L, D | Cho phép ngõ ra |

## 3.5. Di chuyển Move

### 4.5.1. Lệnh di chuyển Move

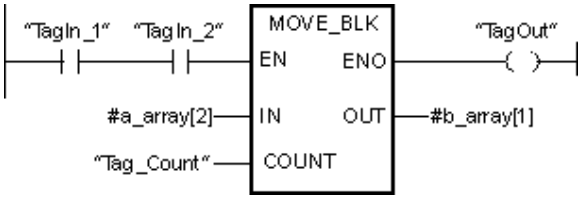



| | | |
|---|---|---|
| EN | I, Q, M, D, L | Cho phép ngõ vào |
| ENO | I, Q, M, D, L | Cho phép ngõ ra |

| | | |
|---|---|---|
| IN | I, Q, M, D, L or constant | Nguồn giá trị |
| OUT1 | I, Q, M, D, L | Nơi chuyển đến |

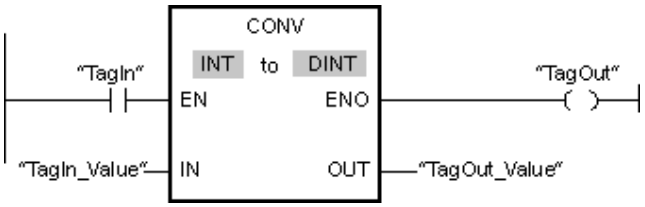
Bạn có thể sử dụng lệnh Move để chuyển các nội dung của một toán hạng IN đầu vào cho hoạt động của OUT1 đầu ra.

### 3.5. 2Lệnh Block Move:




Bạn có thể sử dụng lệnh MOVE_BLK hoạt động đẻ sao chép các nội dung của một vùng nhớ (vùng nhớ nguồn) đến một khu vực bộ nhớ xác định khác. Số lượng các giá trị để đươc sao chép vào khu vực mục tiêu được quy định trong tham số COUNT.Tham số IN quy định giá trị bắt đầu sao chép của vùng nhớ nguồn. các hoạt động sao chép được thực hiện theo hướng tăng dần các địa chỉ.

### 4.6. Lệnh CONV



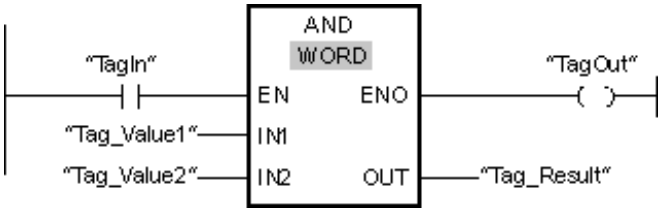

Chuyển đổi từ kiểu dữ liệu này sang kiểu dữ liệu khác.

| Tham số | Kiểu dữ liệu | Miêu tả |
|---|---|---|
| IN | Byte, Word, DWord, SInt, USInt, Int, DInt, UDInt, Real, LReal, Bcd16, Bcd23 | Giá trị ngõ vào |

| | | |
|---|---|---|
| OUT | Byte, Word, DWord, SInt, USInt, Int, DInt, UDInt, Real, LReal, Bcd16, Bcd23 | Giá trị sau chuyển đổi |

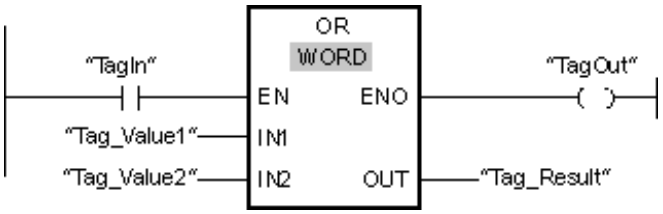
### 3.6 Lệnh toán tử word logic

#### 3.6.1 . Lệnh AND:




Công dụng:lệnh AND để kết hợp các giá trị ở ngõ vào IN2 theo các bit tương ứng với nhau bởi AND logic và xuất kết quả tại OUT.

#### 3.6.2 Lệnh OR:

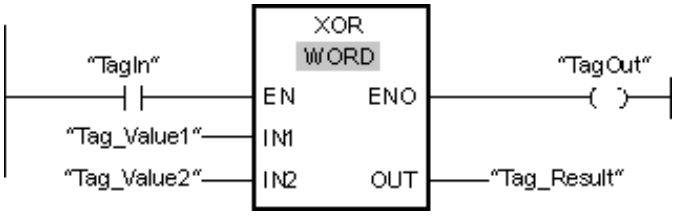



Công dụng: lệnh OR để kết hợp các giá trị ở ngõ vào IN1 và giá trị ngõ vào IN2 theo các bit tương ứng với nhau bởi OR logic và xuất kết quả tại OUT.

Bảng trạng thái của logic OR:

| X1 | X2 | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

### 3.6.2 Lệnh XOR:



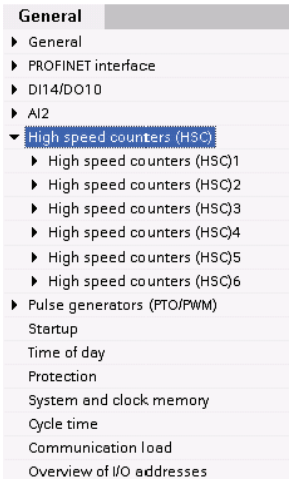

Công dụng: lệnh XOR để kết hợp các giá trị ở ngõ vào IN1 và giá trị ngõ vào IN2 theo các bit tương ứng với nhau bởi XOR logic và xuất kết quả tại OUT.

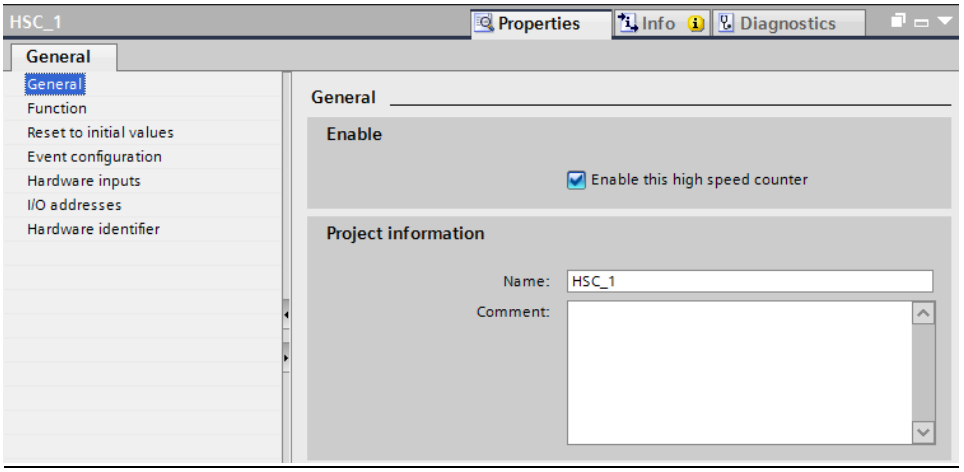
## 3.7 Bộ đếm tốc độ cao (High Speed Counter)

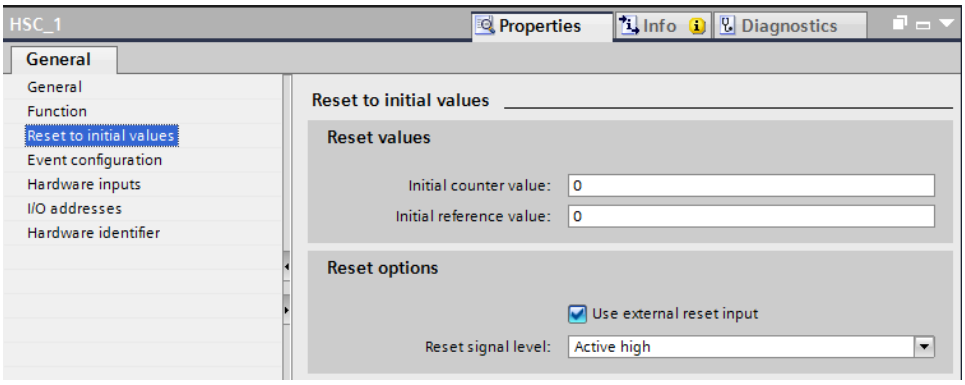
CPU cho phép bạn cấu hình lên đến 6 bộ đếm tốc độ cao

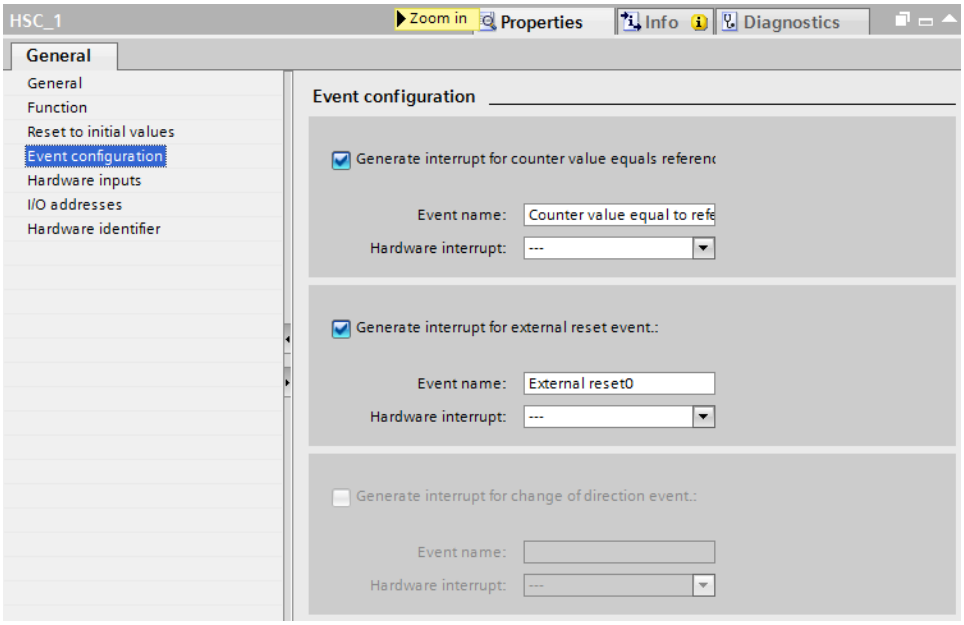
| Description | | | Default Input Assignment | | | Function |
|---|---|---|---|---|---|---|
| HSC | HSC1 | Built In<br>or Signal Board<br>or monitor PTO 0[1] | I0.0<br>I4.0<br>PTO 0 Pulse | I0.1<br>I4.1<br>PTO 0 Direction | I0.3<br>I4.3<br>- | |
| | HSC: | Built In<br>or signal board<br>or monitor PTO 1[1] | I0.2<br>I4.2<br>PTO 1 Pulse | I0.3<br>I4.3<br>PTO 1 Direction | I0.1<br>I4.1<br>- | |
| | HSC3[2] | Built In | I0.4 | I0.5 | I0.7 | |
| | HSC4[3] | Built In | I0.6 | I0.7 | I0.5 | |
| | HSC5[4] | Built In<br>or Signal Board | I1.0<br>I4.0 | I1.1<br>I4.1 | I1.2<br>I4.3 | |
| | HSC6[4] | Built In<br>or signal board | I1.3<br>I4.2 | I1.4<br>I4.3 | I1.5<br>I4.1 | |
| Mode | Single-phase counter with internal direction control | | Clock | - | - | Count or Frequency |
| | | | | | Reset | Count |
| | Single-phase counter with external direction control | | Clock | Direction | - | Count or Frequency |
| | | | | | Reset | Count |
| | Two-phase counter with 2 clock inputs | | Clock up | Clock down | - | Count or Frequency |
| | | | | | Reset | Count |
| | A/B-phase quadrature counter | | Phase A | Phase B | - | Count or Frequency |
| | | | | | Phase Z | Count |
| | Monitor pulse train outputs (PTO)[1] | | Clock | Direction | - | Count |

| High-speed counter | Data type | Default address |
|---|---|---|
| HSC1 | DInt | ID1000 |
| HSC2 | DInt | ID1004 |
| HSC3 | DInt | ID1008 |
| HSC4 | DInt | ID1012 |
| HSC5 | DInt | ID1016 |
| HSC6 | DInt | ID1020 |


General
General
PROFINET interface
DI14/DO10
AI2
High speed counters (HSC)
High speed counters (HSC)1
High speed counters (HSC)2
High speed counters (HSC)3
High speed counters (HSC)4
High speed counters (HSC)5
High speed counters (HSC)6
Pulse generators (PTO/PWM)
Startup
Time of day
Protection
System and clock memory
Cycle time
Communication load
Overview of I/O addresses

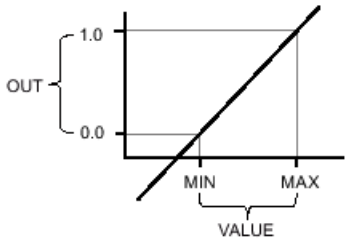

HSC_1
Properties
Info
Diagnostics
General
General
Function
Reset to initial values
Event configuration
Hardware inputs
I/O addresses
Hardware identifier
General
Enable
Enable this high speed counter
Project information
Name:
HSC_1
Comment:

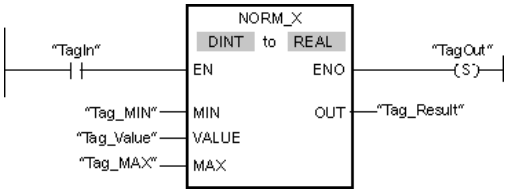

HSC_1
Properties
Info
Diagnostics
General
General
Function
Reset to initial values
Event configuration
Hardware inputs
I/O addresses
Hardware identifier
Reset to initial values
Reset values
Initial counter value:
0
Initial reference value:
0
Reset options
Use external reset input
Reset signal level:
Active high




## 3. 8. Đọc tín hiệu Analog.

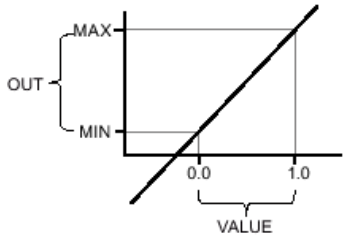
### 3.10.1. NORM_X: Normalize

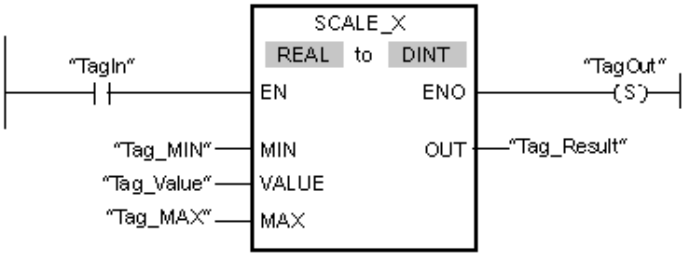


| Parameters | Operand | Value |
|---|---|---|
| MIN | Tag_MIN | 10 |
| VALUE | Tag_Value | 20 |
| MAX | Tag_MAX | 30 |
| OUT | Tag_Result | 0.5 |

## 3SCALE_X: Scale




The following table shows how the instruction works using specific operand values:

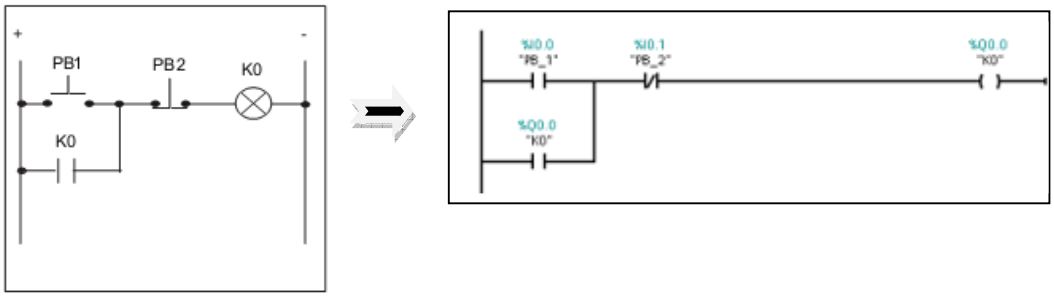
| Parameters | Operand | Value |
|---|---|---|
| MIN | Tag_MIN | 10 |
| VALUE | Tag_Value | 0.5 |
| MAX | Tag_MAX | 30 |
| OUT | Tag_Result | 20 |

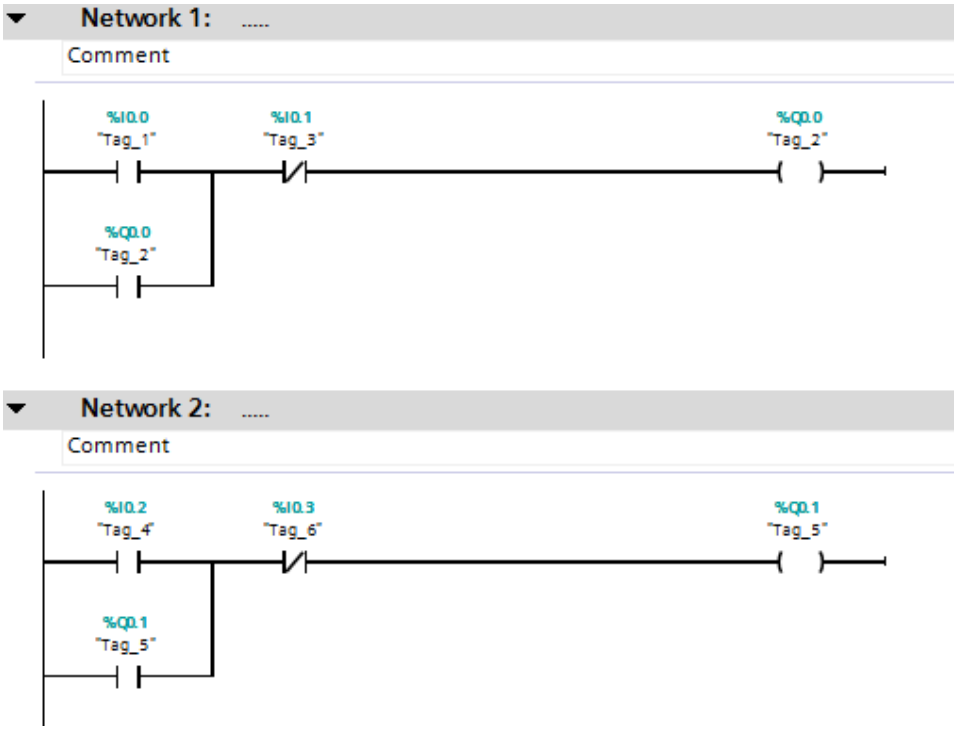
# CHƯƠNG 4: LẬP TRÌNH PLC VÀ LƯU ĐỒ GRAFCET

## 4.1 Giới thiệu lập trình lader

**4.1.1 Ladder Logic (LAD)** : là phương pháp lập trình hình thang, thích hợp trong ngành điện công nghiệp.

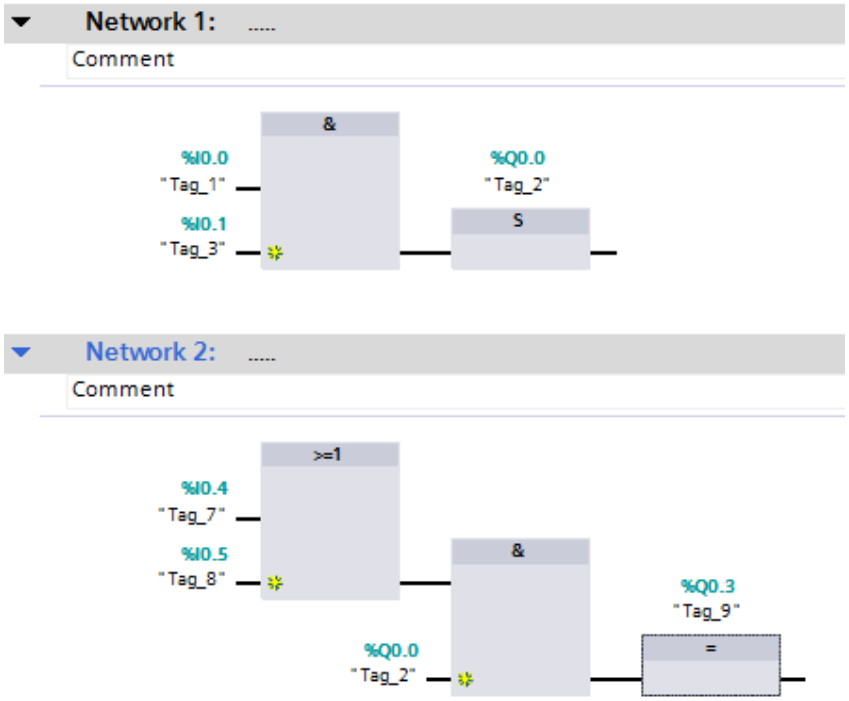
LAD



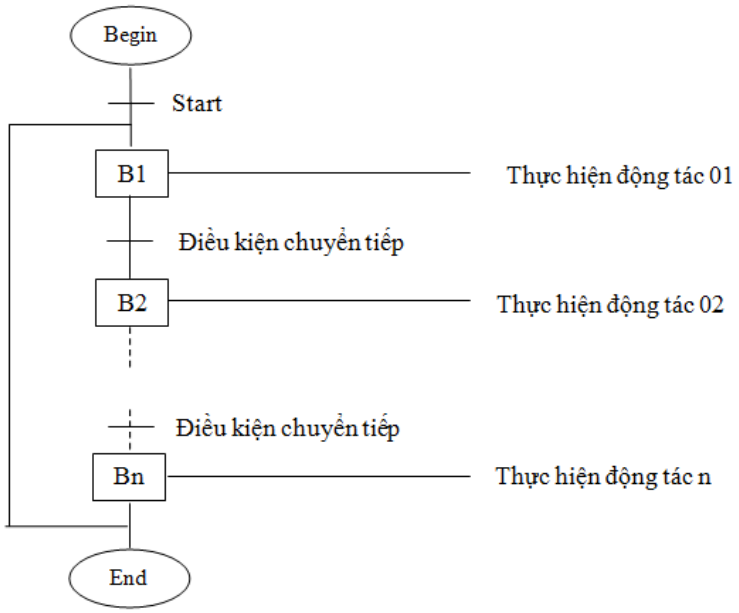



### 4.1.2 Giới thiệu lập trình FBD (Flowchart Block Diagram)

Là phương pháp lập trình theo sơ dồ khối, thích hợp cho ngành điện tử số.



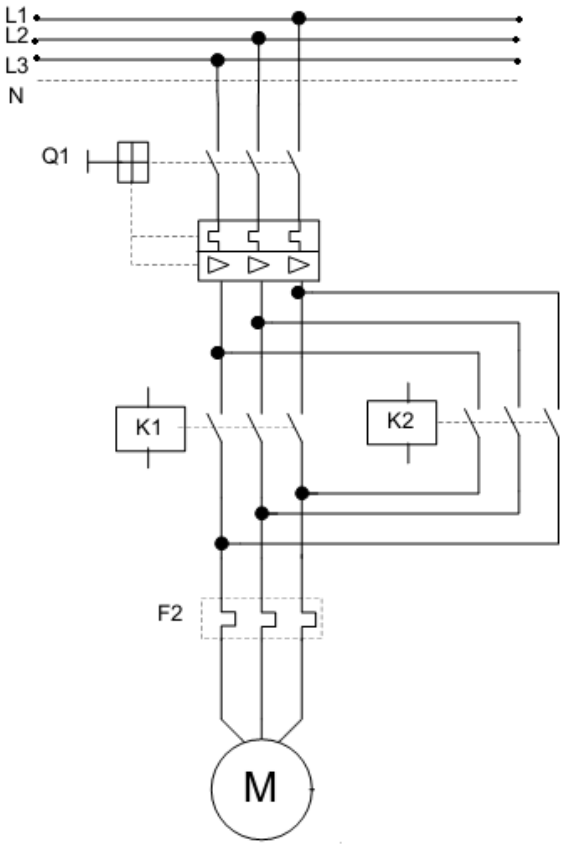

## 4.2 Lưu đồ thuật giải



# CHƯƠNG 6: MỘT SỐ ỨNG DỤNG ĐƠN GIẢN

## 5.1. Điều khiển mở máy động cơ không đồng bộ 3 pha roto lồng sóc.

### *a. Sơ đồ động lực:*




### *b. Thiết bị sử dụng trong mạch điều khiển gồm có*

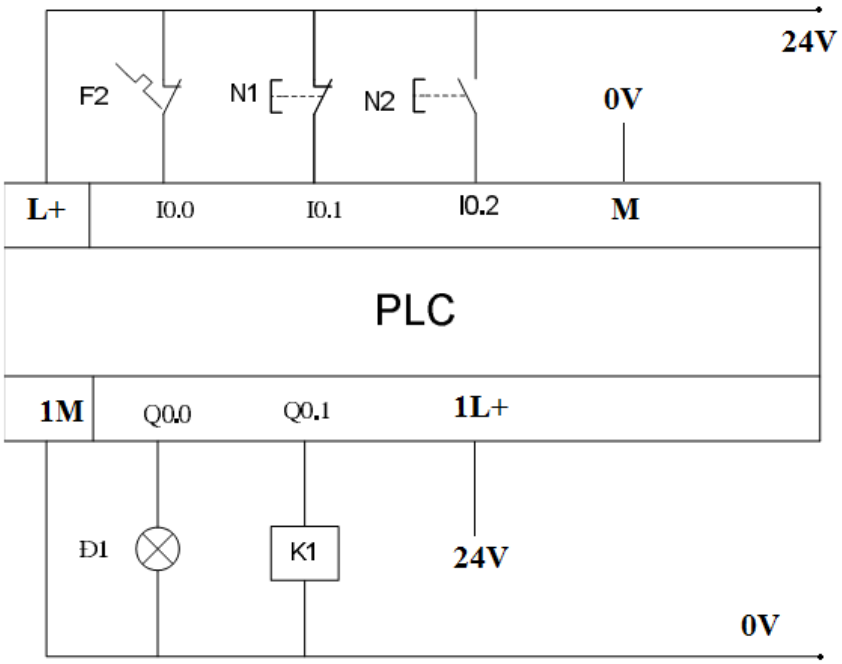
- 1 Áptomát
- 1 Nút mở máy
- 1 nút dừng
- 1 rơle nhiệt

### c. *Bảng quy định các địa chỉ:*

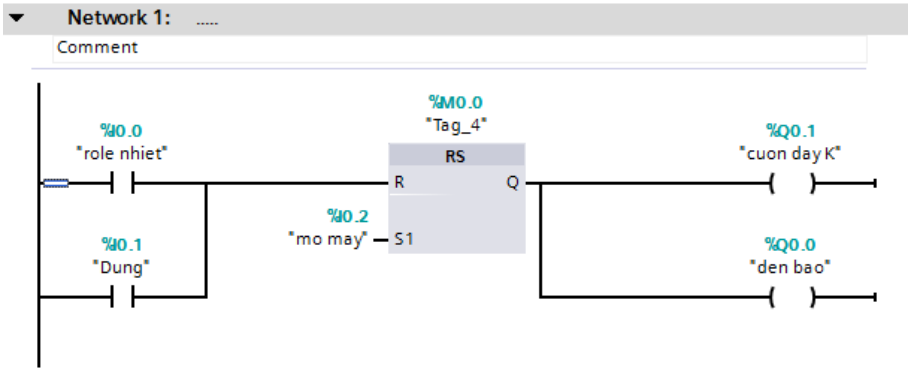
Default tag table

| | Name | Data type | Address | Retain | Visible.. | Acces... |
|---|---|---|---|---|---|---|
| 1 | role nhiet | Bool | %I0.0 | | ☑ | ☑ |
| 2 | Dung | Bool | %I0.1 | | ☑ | ☑ |
| 3 | mo may | Bool | %I0.2 | | ☑ | ☑ |
| 4 | Tag_4 | Bool | %M0.0 | | ☑ | ☑ |
| 5 | cuon day K | Bool | %Q0.1 | | ☑ | ☑ |
| 6 | den bao | Bool | %Q0.0 | | ☑ | ☑ |
| 7 | <Add new> | | | | ☑ | ☑ |

### d. *Sơ đồ kết nối PLC:*

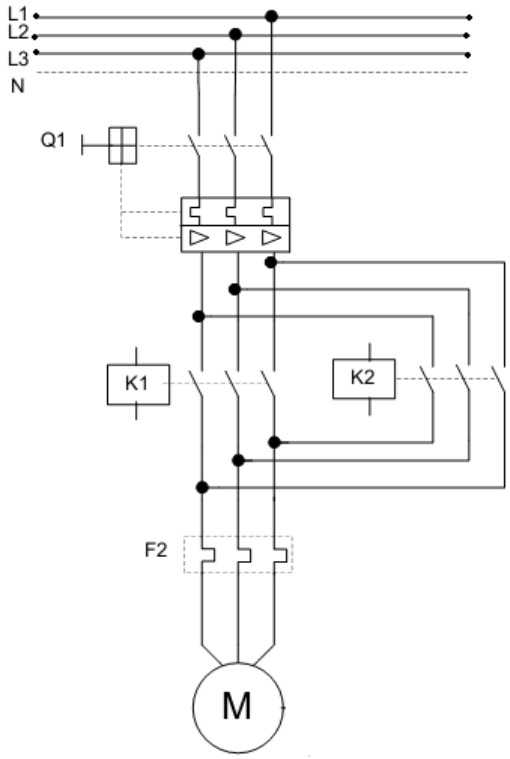



*Chương trình điều khiển:*



## 5.2. Đảo chiều trực tiếp động cơ 3 pha roto roto lồng sóc

### *a. Sơ đồ động lực*




### *b. Thiết bị sử dụng*

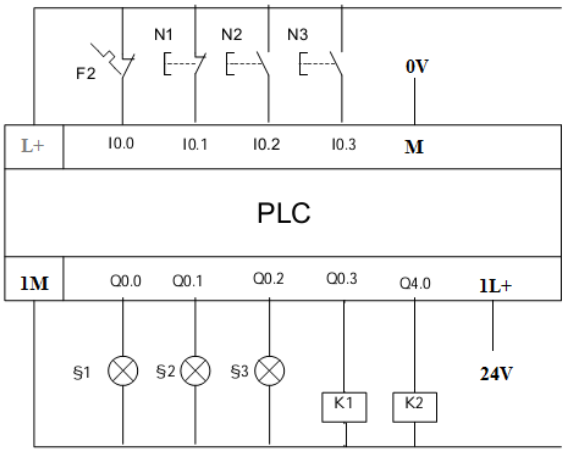
- 1 Áptomát
- 1 Nút mở máy chiều thuận
- 1 Nút mở máy chiều ngược
- 1 nút dừng
- 1 rơle nhiệt

***Bảng quy định các địa chỉ Vào/Ra***

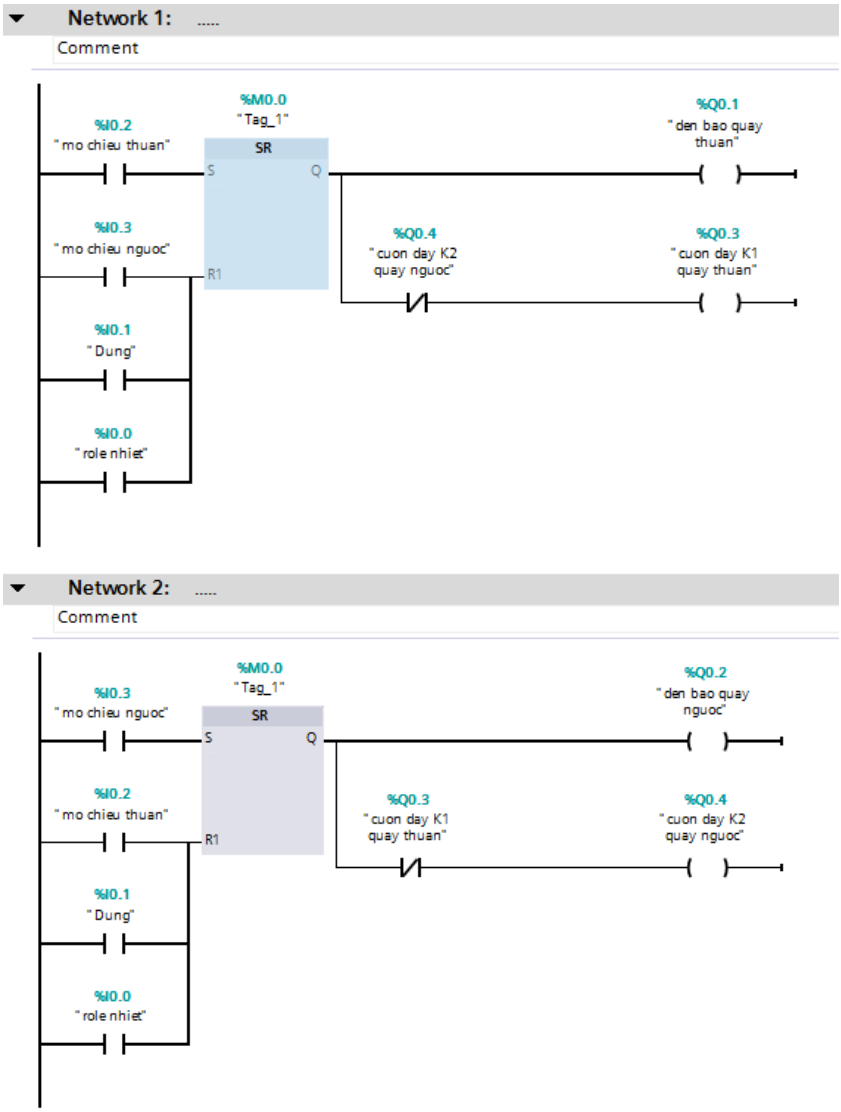
Default tag table

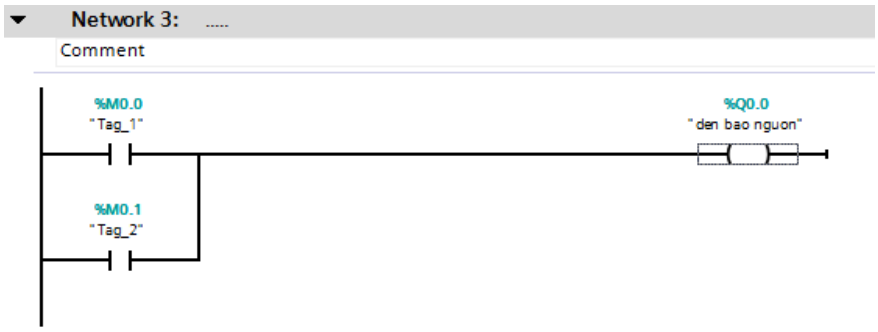
| | Name | Data type | Address | Retain | Visible.. | Acces... |
|---|---|---|---|---|---|---|
| 1 | role nhiet | Bool | %I0.0 | ☐ | ☑ | ☑ |
| 2 | Dung | Bool | %I0.1 | ☐ | ☑ | ☑ |
| 3 | mo chieu thuan | Bool | %I0.2 | ☐ | ☑ | ☑ |
| 4 | mo chieu nguoc | Bool | %I0.3 | ☐ | ☑ | ☑ |
| 5 | den bao nguon | Bool | %Q0.0 | ☐ | ☑ | ☑ |
| 6 | den bao quay thuan | Bool | %Q0.1 | ☐ | ☑ | ☑ |
| 7 | den bao quay nguoc | Bool | %Q0.2 | ☐ | ☑ | ☑ |
| 8 | cuon day K1 quay thuan | Bool | %Q0.3 | ☐ | ☑ | ☑ |
| 9 | cuon day K2 quay nguoc | Bool | %Q0.4 | ☐ | ☑ | ☑ |
| 10 | <Add new> | | | ☐ | ☑ | ☑ |

### c. *Sơ đồ kết nối PLC:*



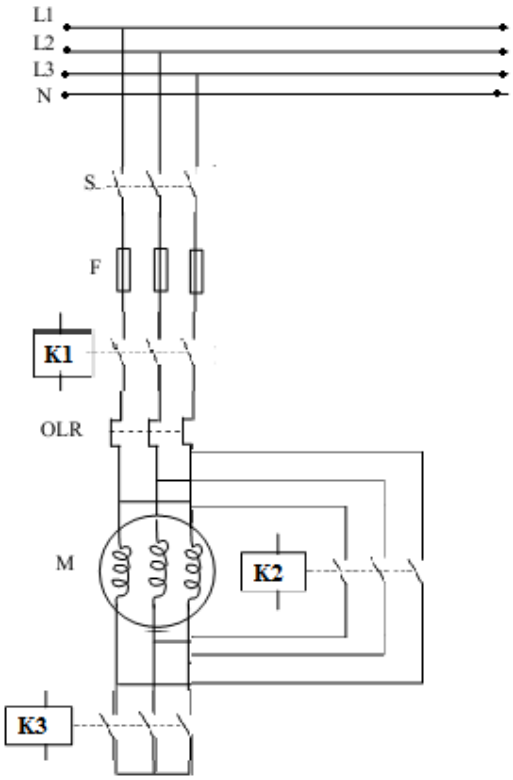

### d. *Chương trình điều khiển*



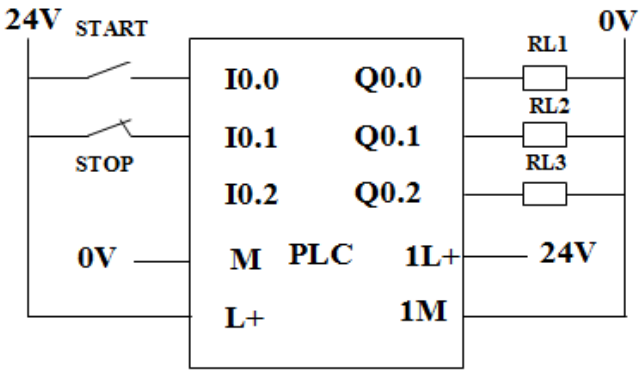



## 5.3. Khởi động sao-tam giác

### *a. Mạch động lực*



## b. Sơ đồ đấu nối PLC

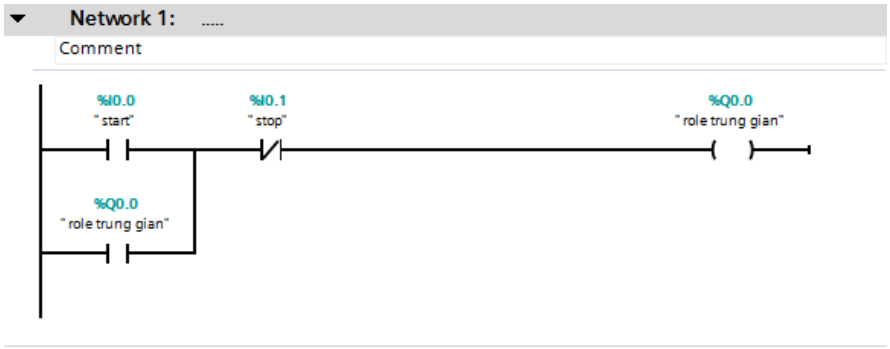


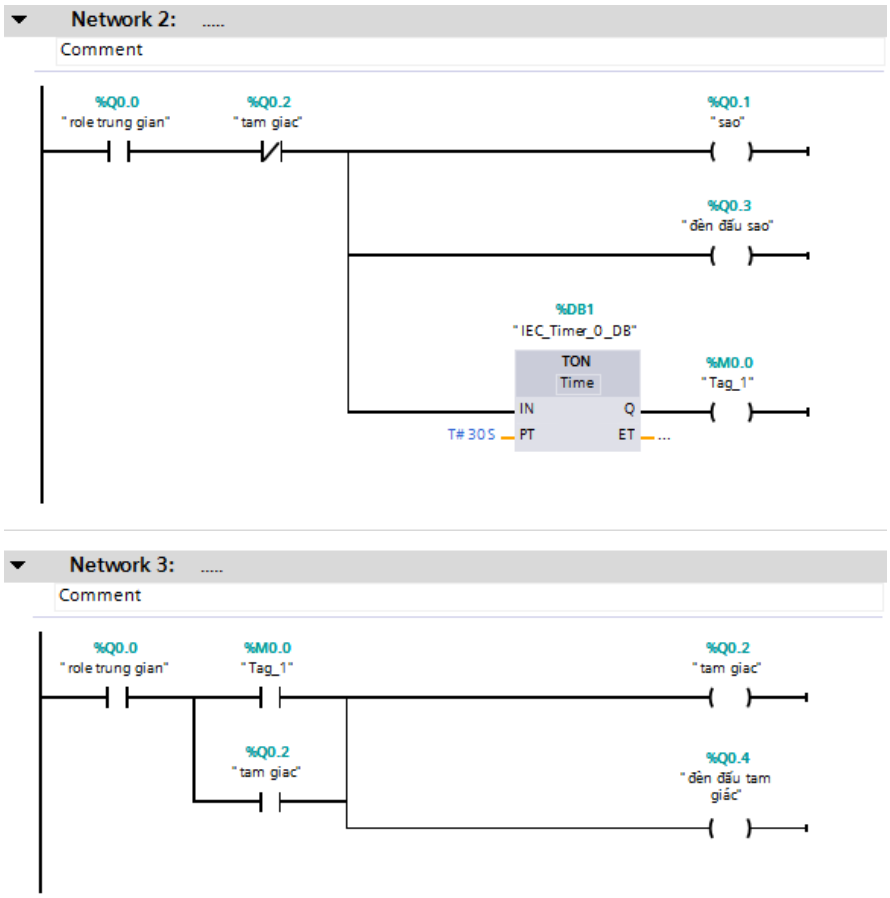

## c. *Bảng quy định các địa chỉ vào/ra*

Default tag table

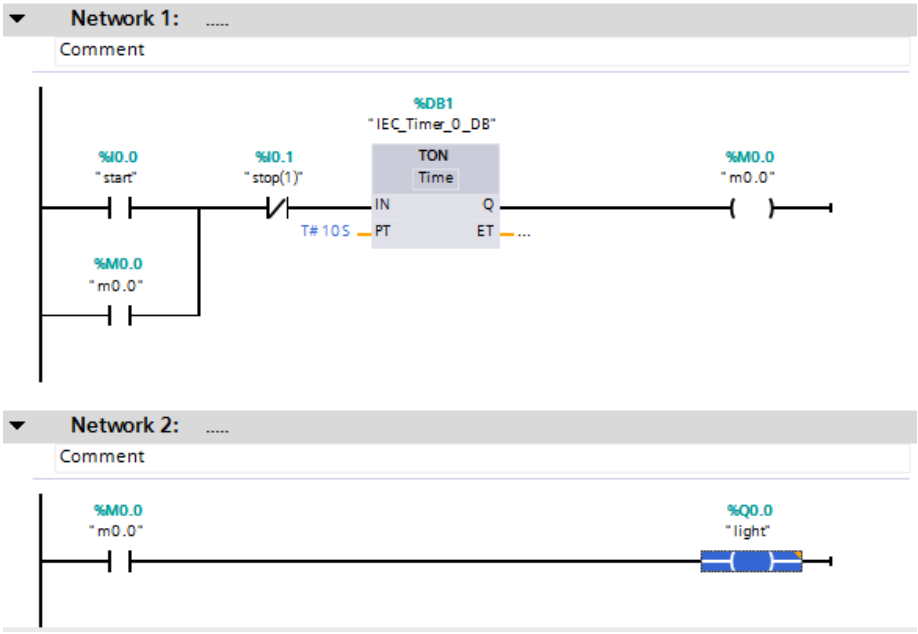
| | Name | Data type | Address | Retain | Visible.. | Acces... |
|---|---|---|---|---|---|---|
| 1 | start | Bool | %I0.0 | ☐ | ☑ | ☑ |
| 2 | stop | Bool | %I0.1 | ☐ | ☑ | ☑ |
| 3 | sao | Bool | %Q0.1 | ☐ | ☑ | ☑ |
| 4 | tam giac | Bool | %Q0.2 | ☐ | ☑ | ☑ |
| 5 | đèn đấu sao | Bool | %Q0.3 | ☐ | ☑ | ☑ |
| 6 | đèn đấu tam giác | Bool | %Q0.4 | ☐ | ☑ | ☑ |
| 7 | role trung gian | Bool | %Q0.0 | ☐ | ☑ | ☑ |
| 8 | <Add new> | | | ☐ | ☑ | ☑ |

## d. *Chương trình điều khiển*

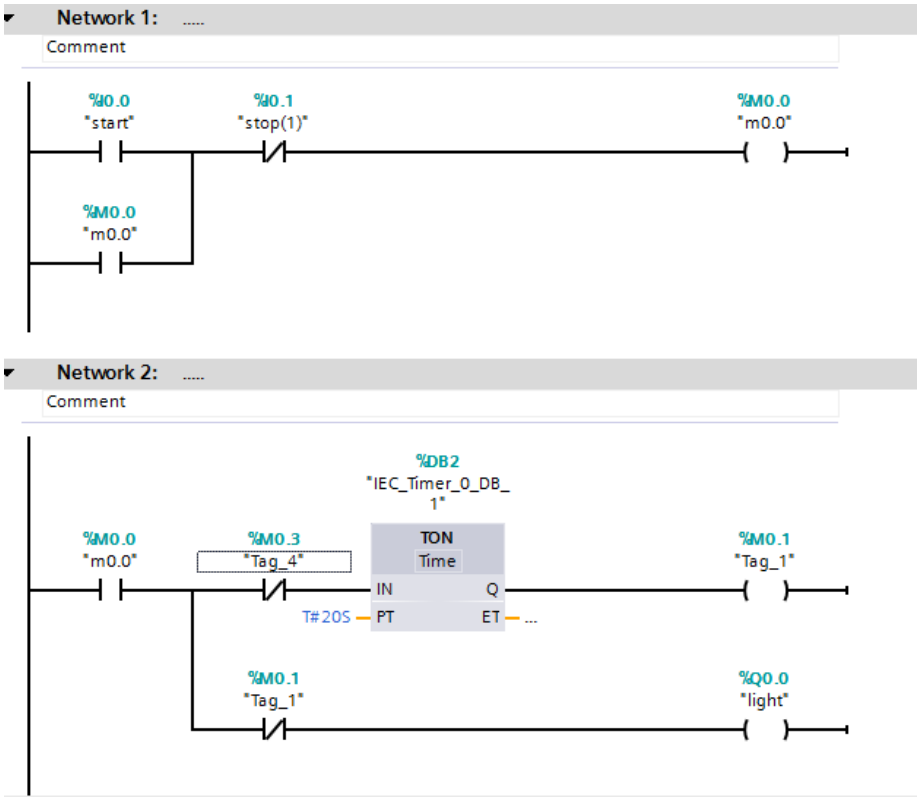


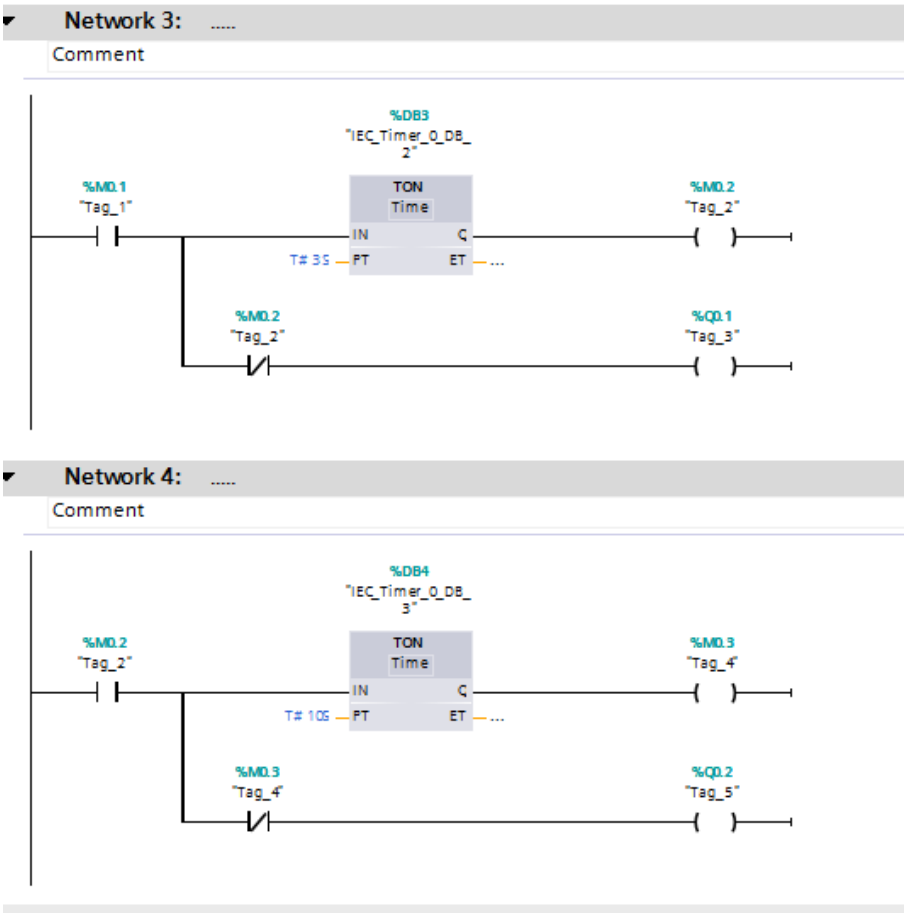



## 5.4. Viết chương trình thực hiện bật đèn Q0. 0 sau khi công tắc I0. 0 bật sau khoảng thời gian T0=10s.

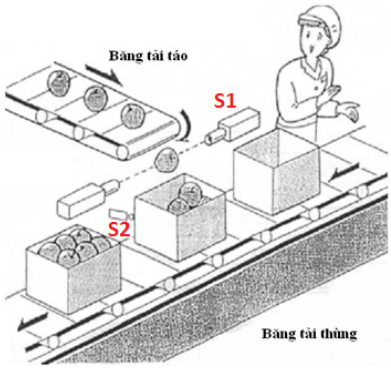


## 5.5. Viết chương trình đèn giao thông với đèn xanh 20s, vàng 3s, đèn đỏ 10s.






## 5.6. Lập trình cho băng tải

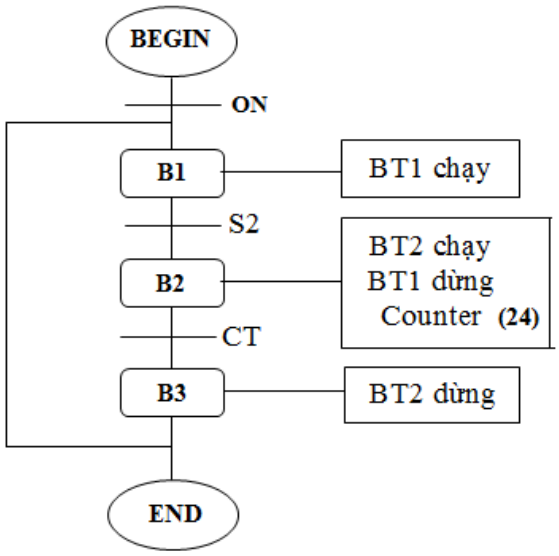



***e. Mô tả hoạt động của hệ thống băng tải***

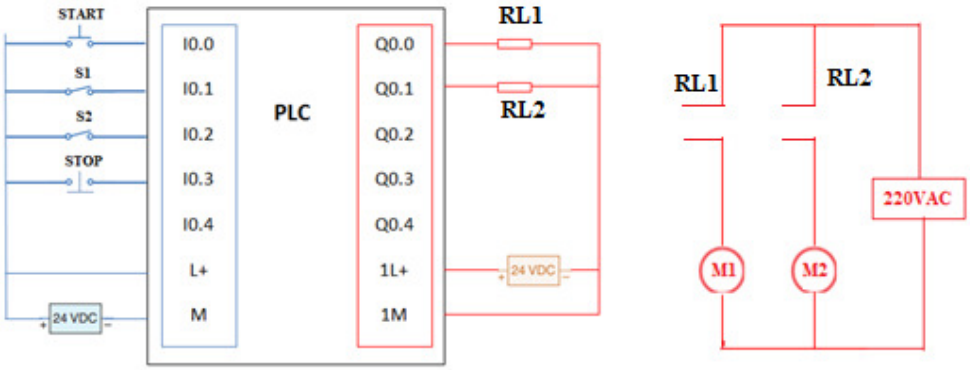
Ấn ON BT1 chạy đưa thùng vào. Khi thùng đến S2 thì BT1 dừng. BT2 chạy đưa táo rơi vào thùng. Táo được đếm bởi một cảm biến hồng ngoại S1. Khi số táo đưa vào thùng đủ 24 quả thì băng tải 2 dừng. Tiếp tục băng tải 1 chạy lại để đưa

thùng táo thành phẩm ra ngoài và đóngthùng táo mới. Hệ thống tự động hoạt động như trên cho đến khi ấn OFF thì dừng.

### *f. Lưu đồ thuật giải*



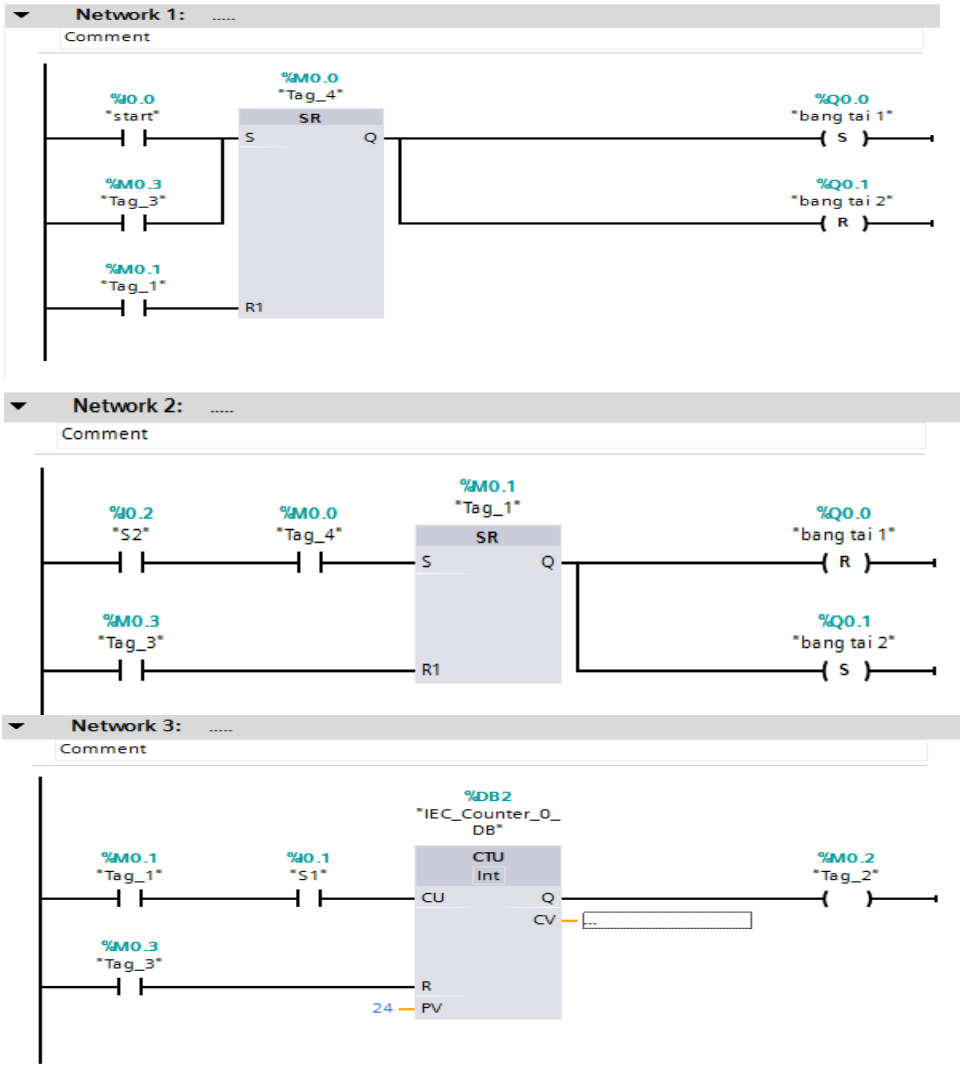

### *g. sơ đồ đấu nối plc*

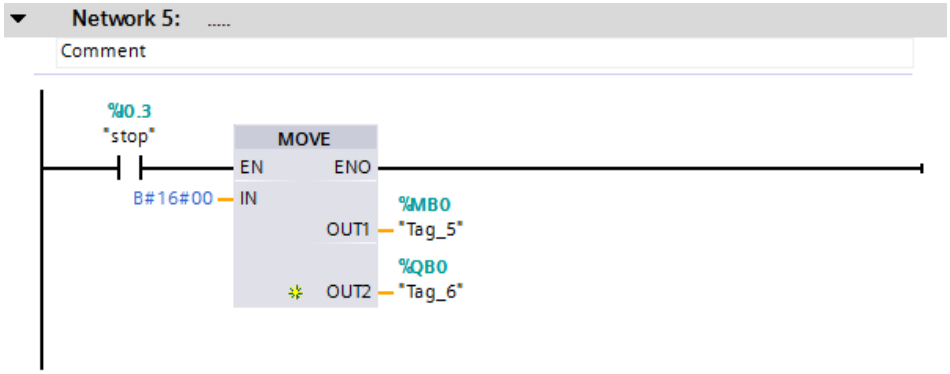


## h. Bảng tags

| | Name | Data type | Address | Retain | Visible.. | Acces... |
|---|---|---|---|---|---|---|
| 1 | start | Bool | %I0.0 | | ☑ | ☑ |
| 2 | S1 | Bool | %I0.1 | | ☑ | ☑ |
| 3 | S2 | Bool | %I0.2 | | ☑ | ☑ |
| 4 | bang tai 1 | Bool | %Q0.0 | | ☑ | ☑ |
| 5 | bang tai 2 | Bool | %Q0.1 | | ☑ | ☑ |
| 6 | stop | Bool | %I0.3 | | ☑ | ☑ |
| 7 | Tag_1 | Bool | %M0.1 | | ☑ | ☑ |
| 8 | Tag_2 | Bool | %M0.2 | | ☑ | ☑ |
| 9 | Tag_3 | Bool | %M0.3 | | ☑ | ☑ |
| 10 | Tag_4 | Bool | %M0.0 | | ☑ | ☑ |
| 11 | Tag_5 | Byte | %MB0 | | ☑ | ☑ |
| 12 | Tag_6 | Byte | %QB0 | | ☑ | ☑ |

## i. Chương trình






## *57. Bài tập điều khiển đèn giao thông.*

Giả sử cần điều khiển đèn giao thông tại ngã tư giao lộ bằng 1 công tắc gạt I0.0. Trong đó

đèn X1 sáng 4 giây, V1 sáng 2 giây, X2 sáng 5 giây và V2 sáng 2 giây.

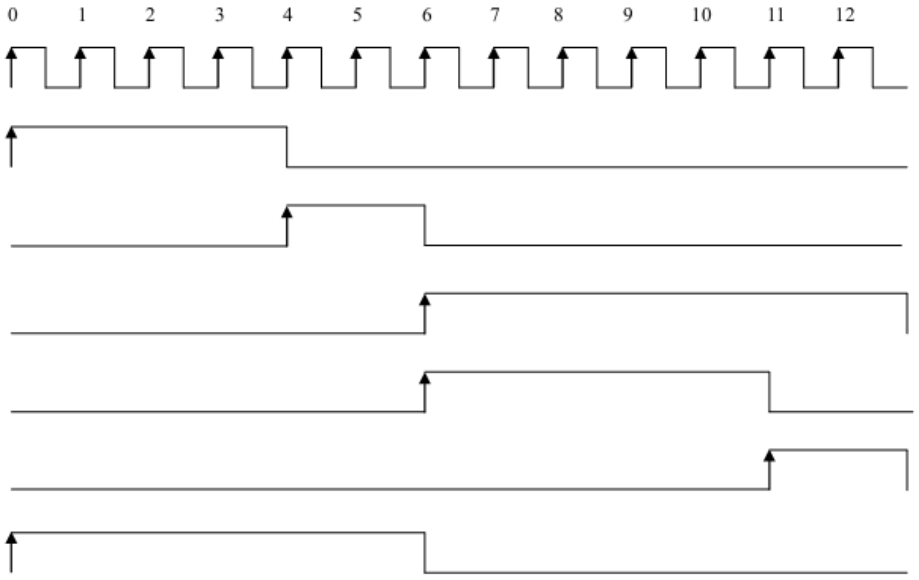
Quy tắc chung:

Đ1 sáng (giây) = X2 sáng + V2 sáng = 7 (giây)

Đ2 sáng (giây) = X1 sáng + V1 sáng = 6 (giây)

Có những trường hợp khác do yêu cầu thực tế của từng ngã tư.

**Cho giản đồ xung, hãy nhận xét và ghi tên các đèn tương ứng lên giản đồ xung**




Chương trình chỉ hoạt động khi gạt SW1 lên mức 1.

Viết chương trình điều khiển các đèn trên chạy theo giản đồ, dùng các lệnh so sánh. Có thể

thay đổi thời gian hoạt động các đèn và thực hiện lại chương trình.

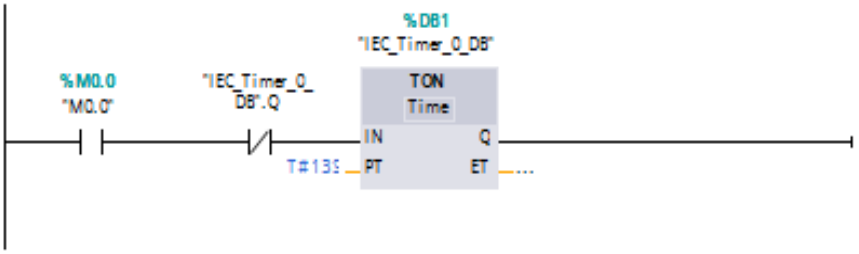
Code :

**Network 1:** .....

Comment


**Network 2:** .....

Comment

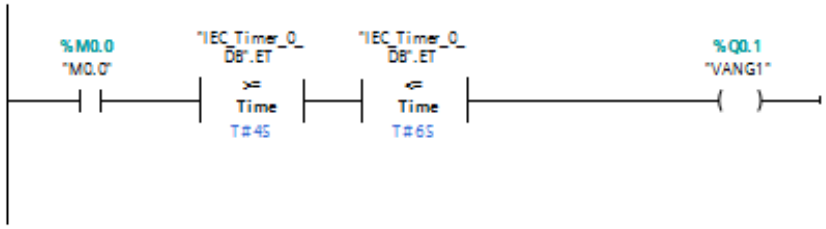


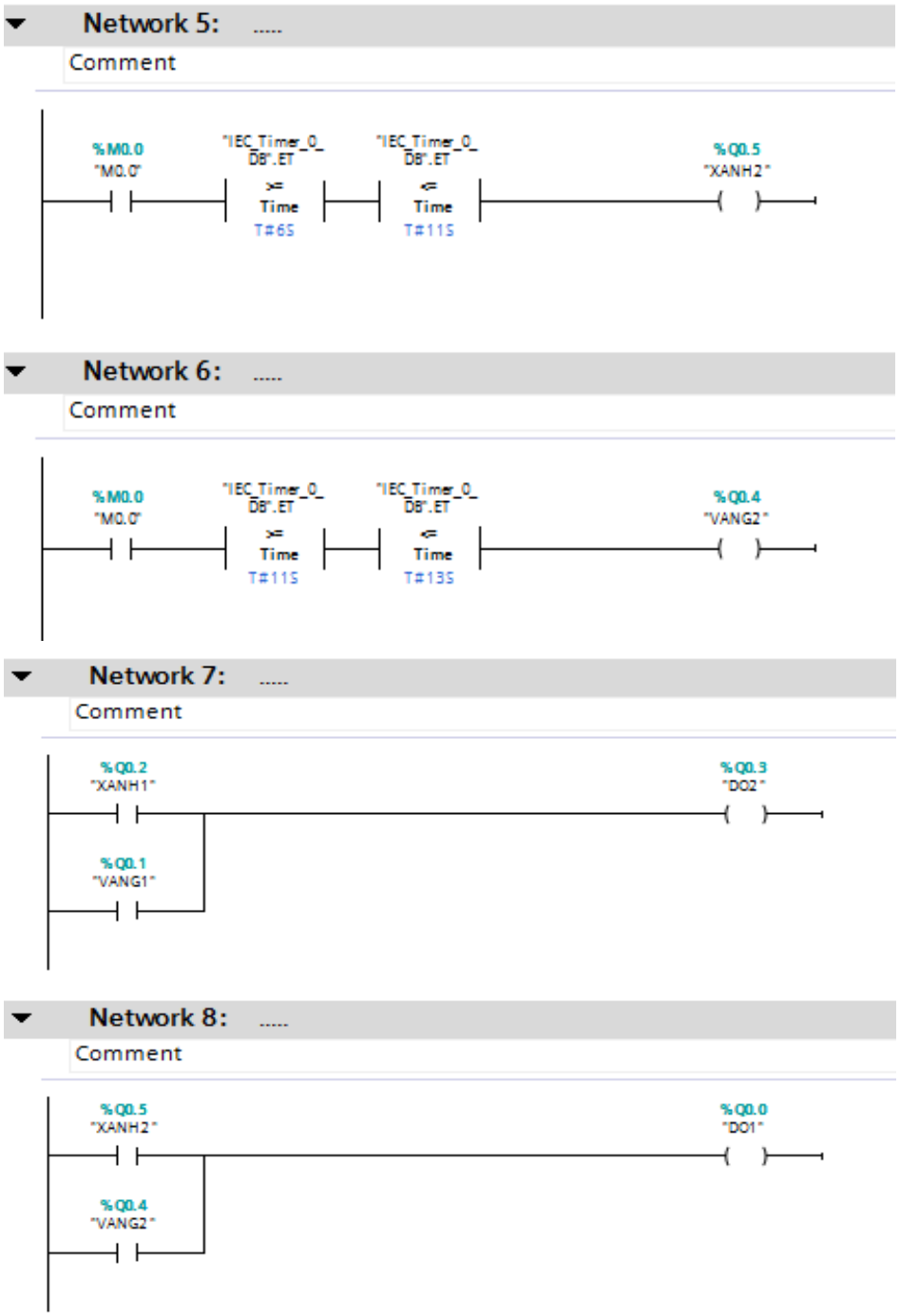

**Network 3:** .....

Comment


**Network 4:** .....

Comment






### *5.8. Đếm sản phẩm từ I0.0 và báo số lượng sản phẩm theo yêu cầu sau:*

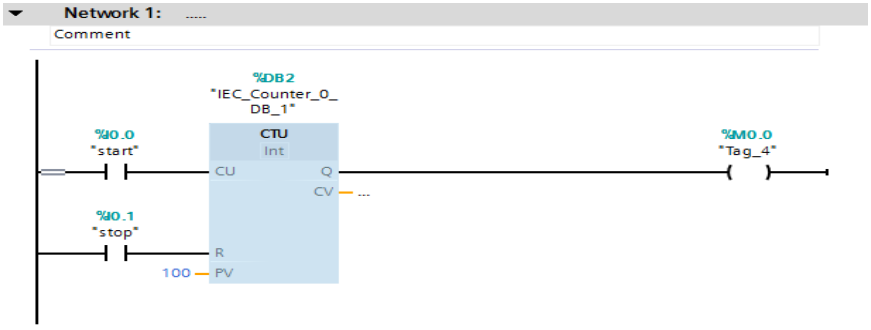
1. Không có sản phẩm đèn A sáng.
2. Từ 1 đến10 sản phẩm, đèn B sáng.
3. Từ 11 đến 20 sản phẩm, đèn C sáng.
4. Từ 21 sản phẩm trở lên đèn D sáng.

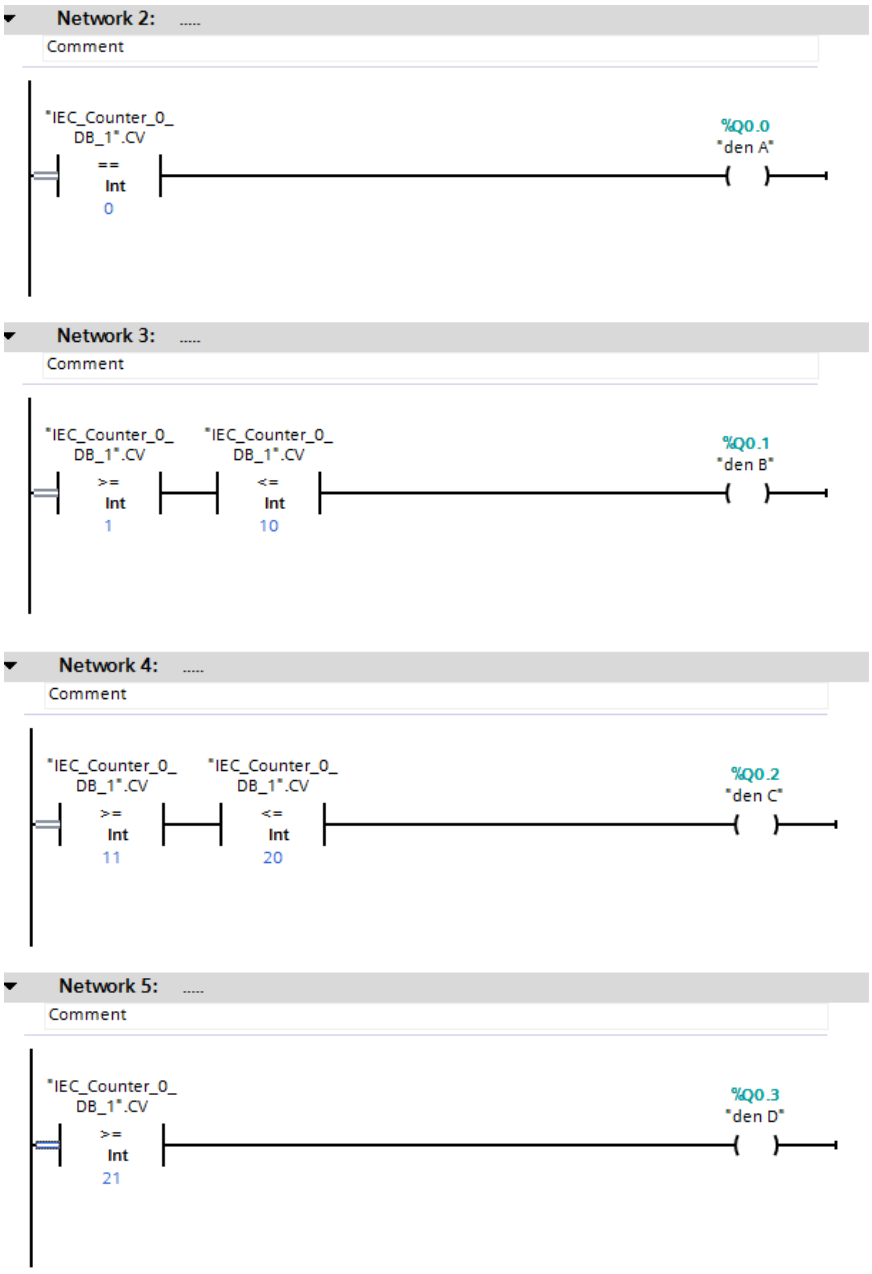
## Bảng tags

Default tag table

| | Name | Data type | Address | Retain | Visible.. | Acces... |
|---|---|---|---|---|---|---|
| 1 | start | Bool | %I0.0 | ☐ | ☑ | ☑ |
| 2 | stop | Bool | %I0.1 | ☐ | ☑ | ☑ |
| 3 | den A | Bool | %Q0.0 | ☐ | ☑ | ☑ |
| 4 | Tag_4 | Bool | %M0.0 | ☐ | ☑ | ☑ |
| 5 | den B | Bool | %Q0.1 | ☐ | ☑ | ☑ |
| 6 | den C | Bool | %Q0.2 | ☐ | ☑ | ☑ |
| 7 | den D | Bool | %Q0.3 | ☐ | ☑ | ☑ |
| 8 | <Add new> | | | ☐ | ☑ | ☑ |

## Chương trình

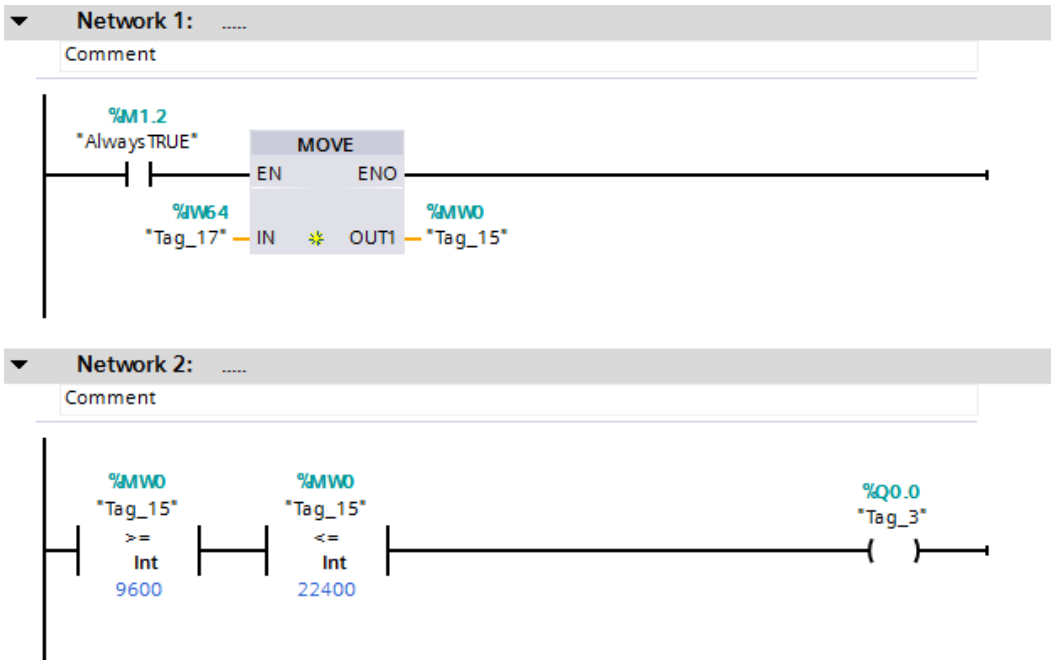





## 6.9. So sánh điều khiển đèn A(Q0.0) sáng từ điện áp 3VDC đến 7VDC 0-10VDC qui đổi thành 0-32000

3VDC =>9600

7VDC=>22400




## 5.10. Mô tả hoạt động của hệ thống trộn sơn

- Sơ đồ nguyên lý được mô tả trên bản vẽ 04

- Ấn Start → tác động mở Valve 1 và Valve 2 cho phép 2 chất lỏng bắt đầu đổ vào bình chứa.

- Khi bình chứa được đổ đầy, công tắc dò mức di chuyển lên chạm S1, làm ngắt 2 Valve 1 và 2, và khởi động Motor hoạt động để trộn lẫn 2 chất lỏng.

- Motor hoạt động như sau: Chạy thuận 5 giây, chạy ngược 5 giây; chạy 5 chu kỳ thuận ngược như vậy rồi tự động dừng.

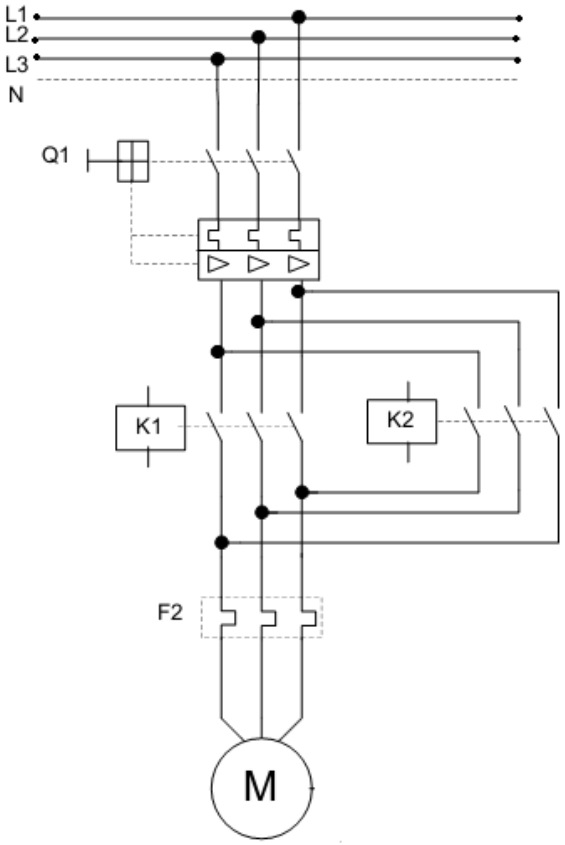
- Sau khi trộn xong thì Valve X mở để xả chất lõngđã trộn ra ngoài.

- Khi bình chứa đã xả hết thì công tắc dò mức di chuyển xuống chạm S2, tác động đóng Valve X.

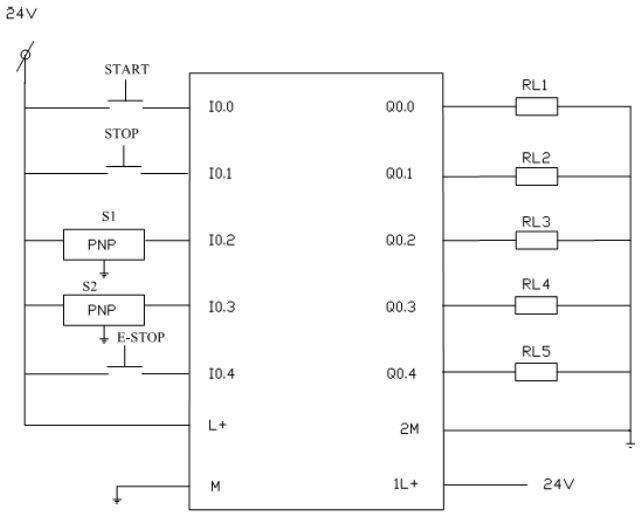
- Hệ thống tự động hoạt động lại từ đầu cho đến hết 3 mẽ trộn thì tự động dừng. Nếu thực hiện lại ta phải ấn nút Reset.

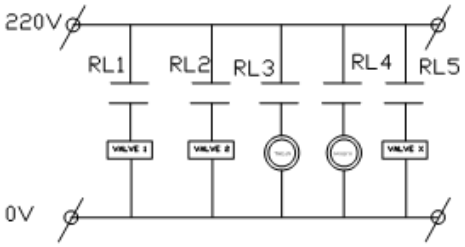
- Người ta có thể dừng hệ thống bất kỳ lúc nào bằng nút Stop.

## a. Mạch động lực.



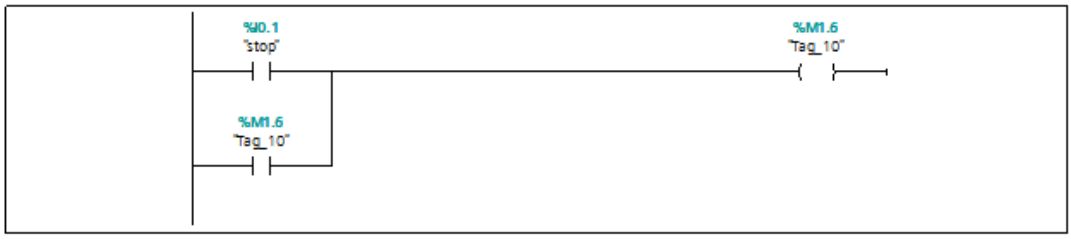

## b. Mạch điều khiển





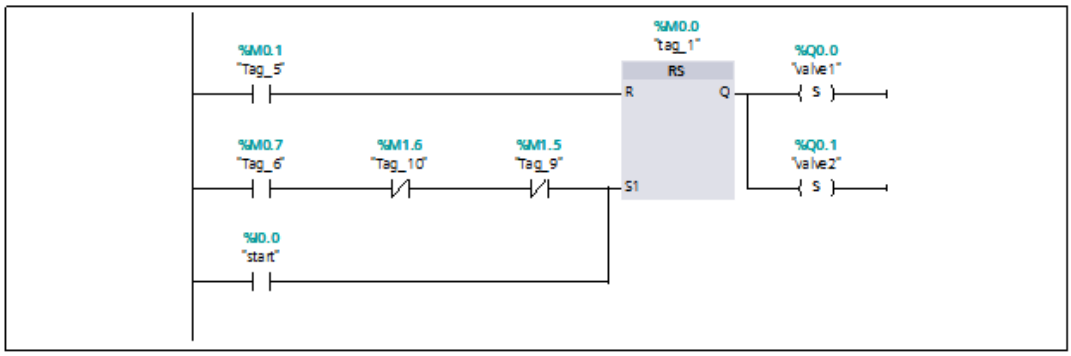

**c. chương trình.**

**Network 1:**

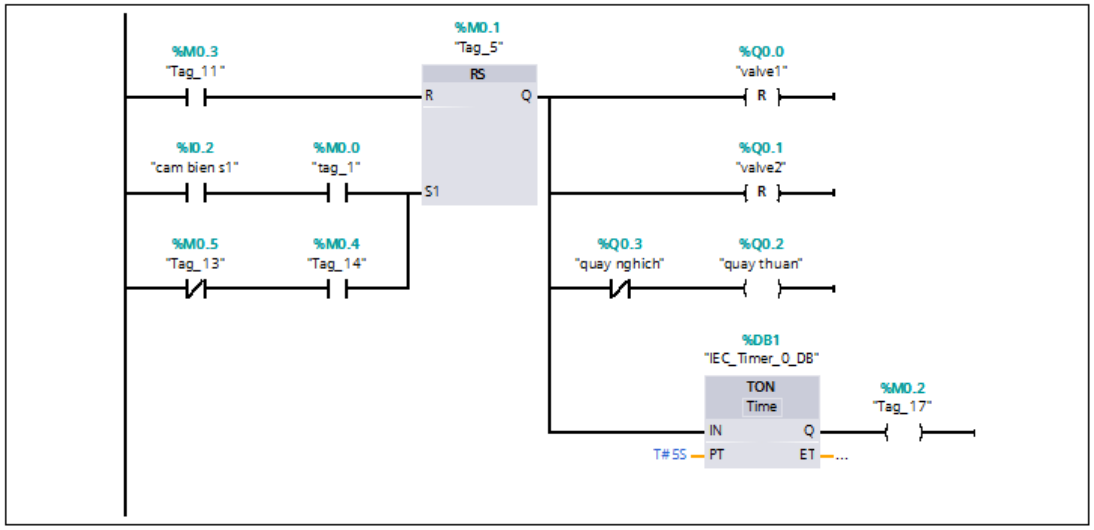



**. Network 2:**

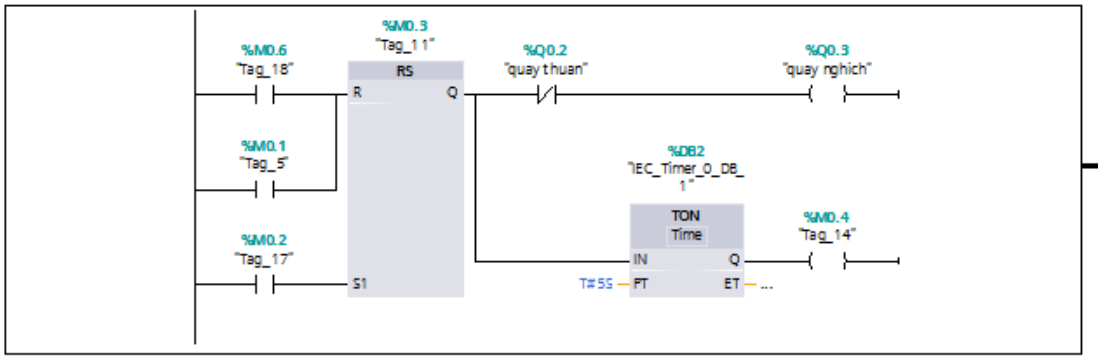
**Network 2:**



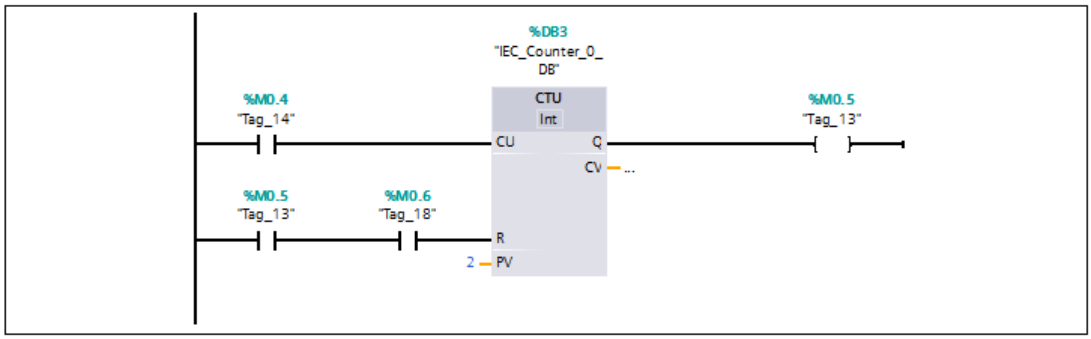

**Network 3:**



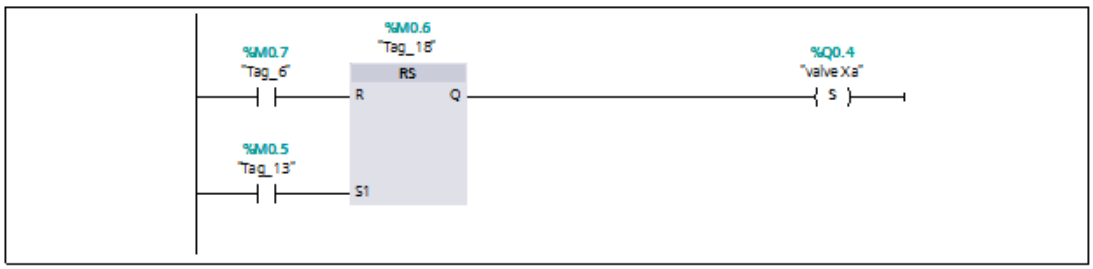
**Network 4:**



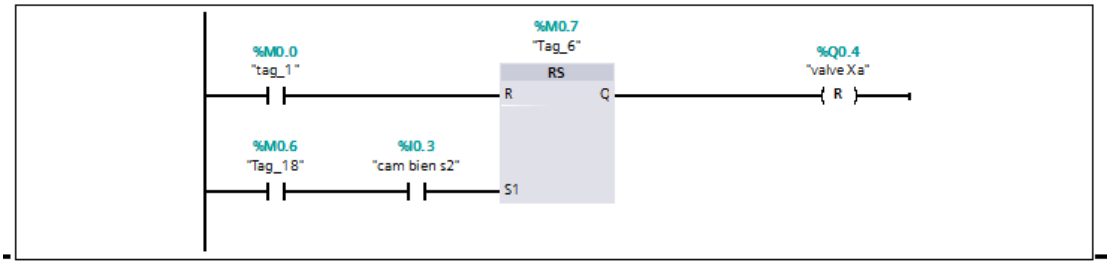

**Network 5:**




**Network 6:**



**Network 7:**




**Network 8:**

**Network 8:**