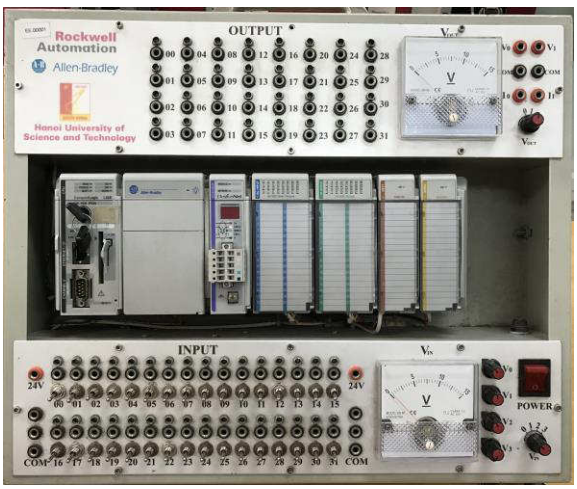
VIỆN ĐIỆN - TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

BM TỰ ĐỘNG HÓA CÔNG NGHIỆP

LẬP TRÌNH CƠ BẢN

PLC ROCKWELL AUTOMATION

(Lưu hành nội bộ)



Soạn thảo: Nhóm ĐKLG & PLC

Version: 1.0

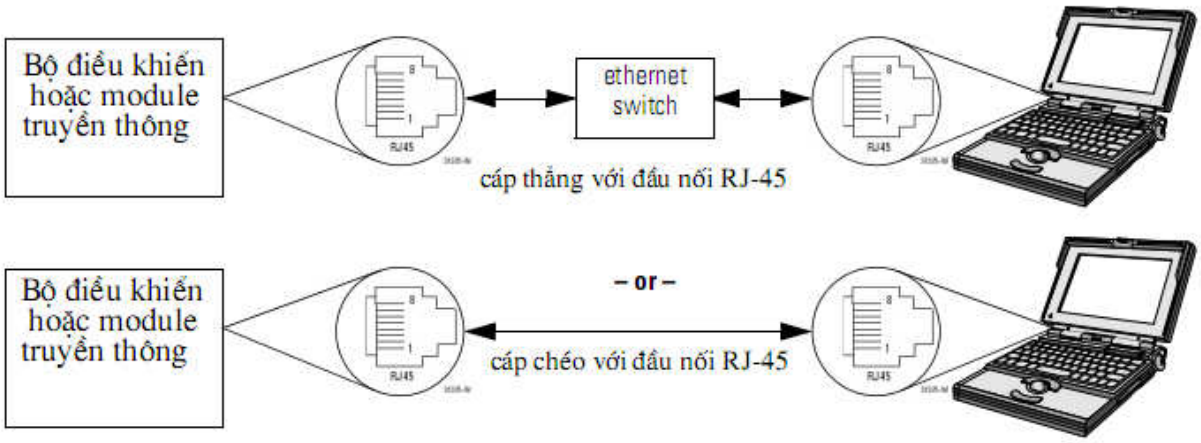
LẬP TRÌNH PLC ROCKWELL AUTOMATION

- Yêu cầu phần mềm:
    - RSLinx Classic.
    - RSLogix 5000 Enterprise Series.
    - RSLogix Emulate 5000 Chassis Monitor.
- Yêu cầu phần cứng:
    - Laptop / PC
    - PLC CompactLogix PLC 1769-L32E / PLC ControlLogix

MỤC LỤC

1. Kết nối máy tính với PLC

Việc kết nối máy tính với PLC (CompactLogix/ControlLogix) giới thiệu ở đây được thực hiện thông qua cổng Ethernet IP. Trong tài liệu này mô tả cách kết nối với CompactLogix 1769-L32E. Cách kết nối với ControlLogix gần như tương tự.

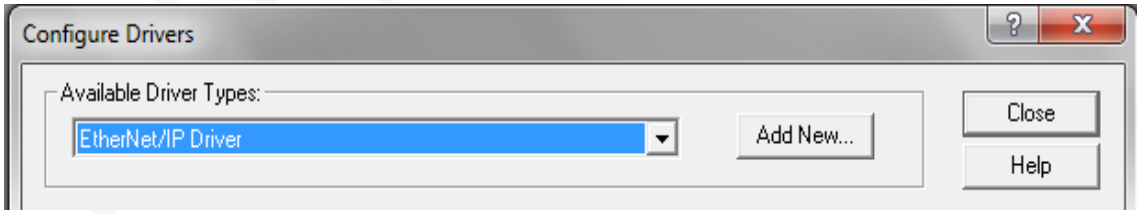


!*Lưu ý*: nếu kết nối trực tiếp máy tính với PLC không dùng switch cần đặt IP tĩnh cho máy tính. (Ví dụ: nếu IP của PLC là 192.168.1.10 thì đặt IP cho máy tính là 192.168.1.x với x khác 10).

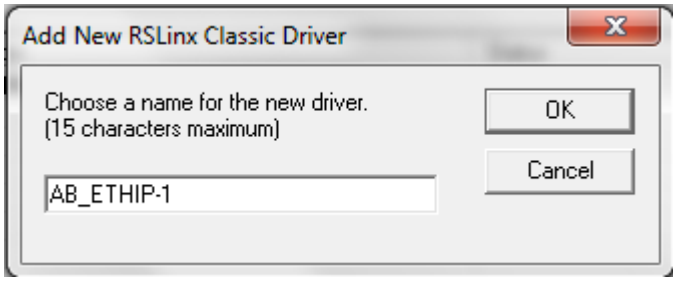
Sử dụng phần mềm **RSLinx Classic** để cấu hình kết nối.

- Bước 1: Khởi động phần mềm **RSLinx Classic**.

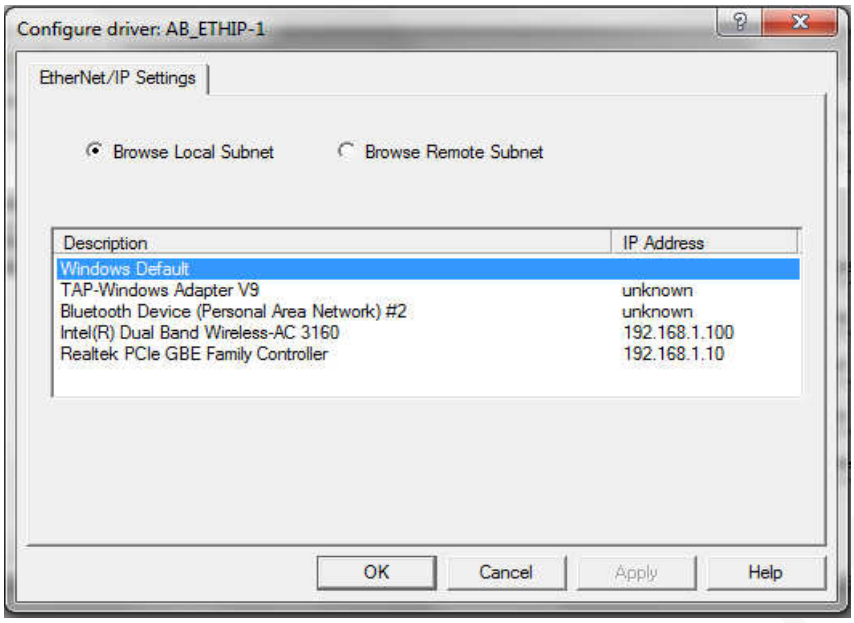
- Bước 2: Ấn vào  .
- Bước 3: Ở mục **Available Driver Types** chọn **Ethernet/IP devices** và ấn Add New...



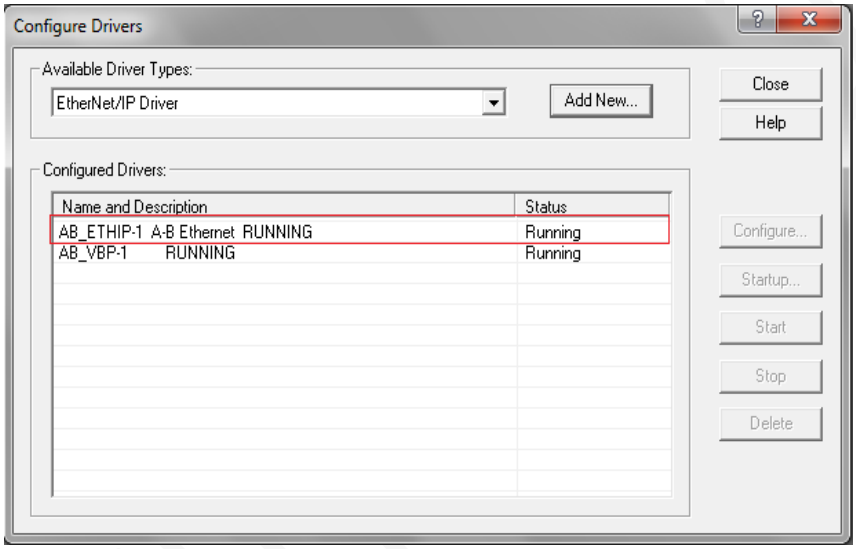
- Bước 4: Chấp nhận tên mặc định (có thể thay đổi).



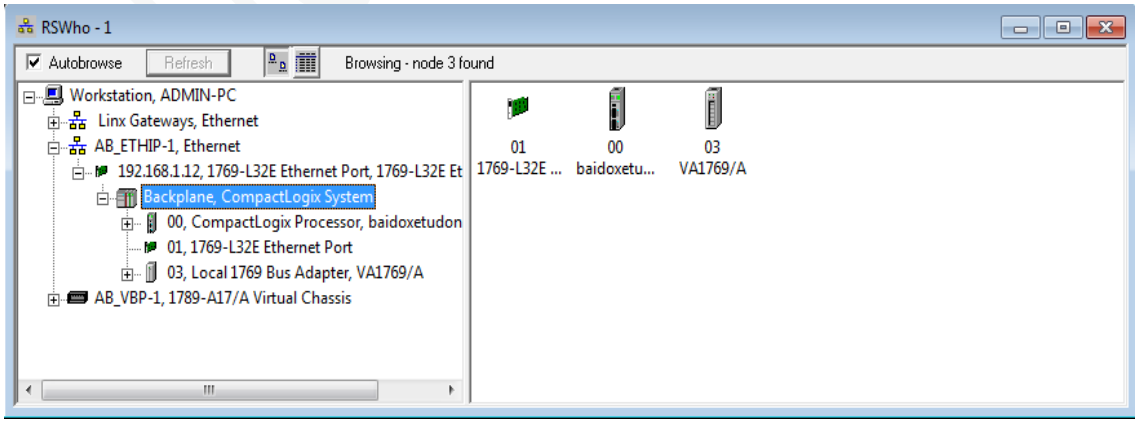
- Bước 5: Lựa chọn phần cứng (card mạng kết nối), có thể chọn mặc định theo Windows.



- Bước 6 : Ấn OK. Driver đã được cấu hình và chạy:



Ấn Close và mở mục  (RSWho). Chúng ta có thể nhìn các thiết bị tương ứng với địa chỉ IP của chúng.



Nếu mở RSWho mà không tìm thấy CPU thì có một vài nguyên nhân như sau:

1. Máy tính và CPU của PLC trùng địa chỉ IP. Cách khắc phục: thay đổi địa chỉ IP tĩnh của máy bằng cách truy cập kết nối LAN trên máy tính và thay đổi địa chỉ IP tại mục

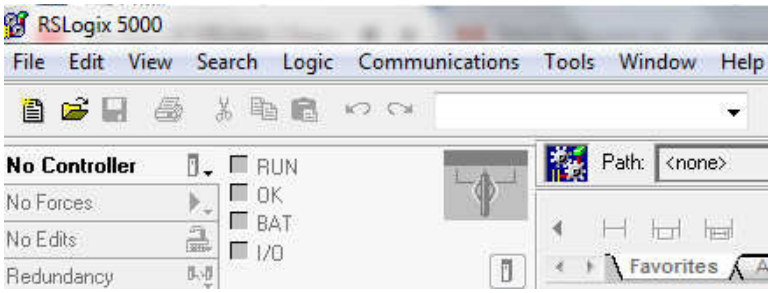
Properties/TCP-IPv4/ Properties/ Use the following IP address và thay đổi địa chỉ IP của máy tính sao cho nhóm 3 chữ số cuối khác với IP của CPU 1769-L32E.

2. CPU của PLC bị mất địa chỉ IP. Cách khắc phục: cấp lại địa chỉ IP bằng cách sử dụng phần mềm Booth-DHCP Server.
3. Mất kết nối. Cách khắc phục: kiểm tra lại cáp kết nối.

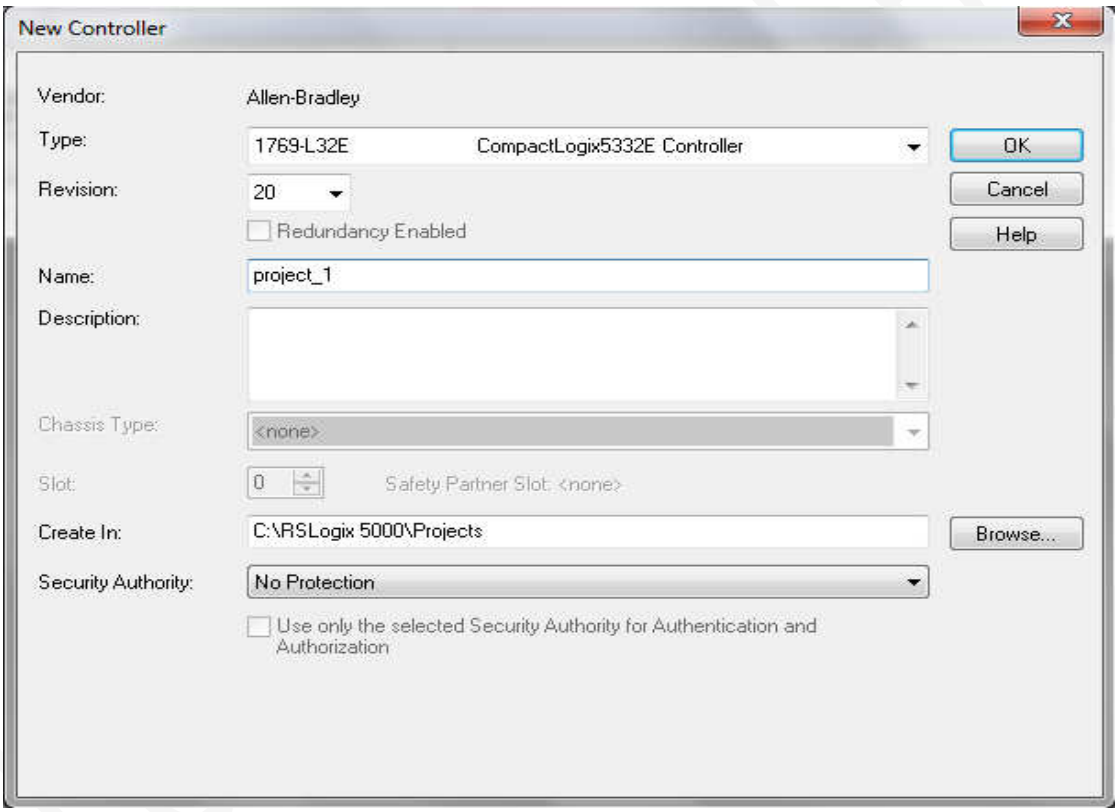
2. Tạo và lập trình một chương trình đơn giản trong RSlogix 5000

2.1 Tạo một dự án (project)

- Bước 1: Khởi động phần mềm **RSLogix 5000**
- Bước 2: Click chuột vào **File → New,** sẽ hiện ra cửa sổ **New Controller**



- Bước 3: Khai báo cấu hình cho bộ điều khiển sử dụng.

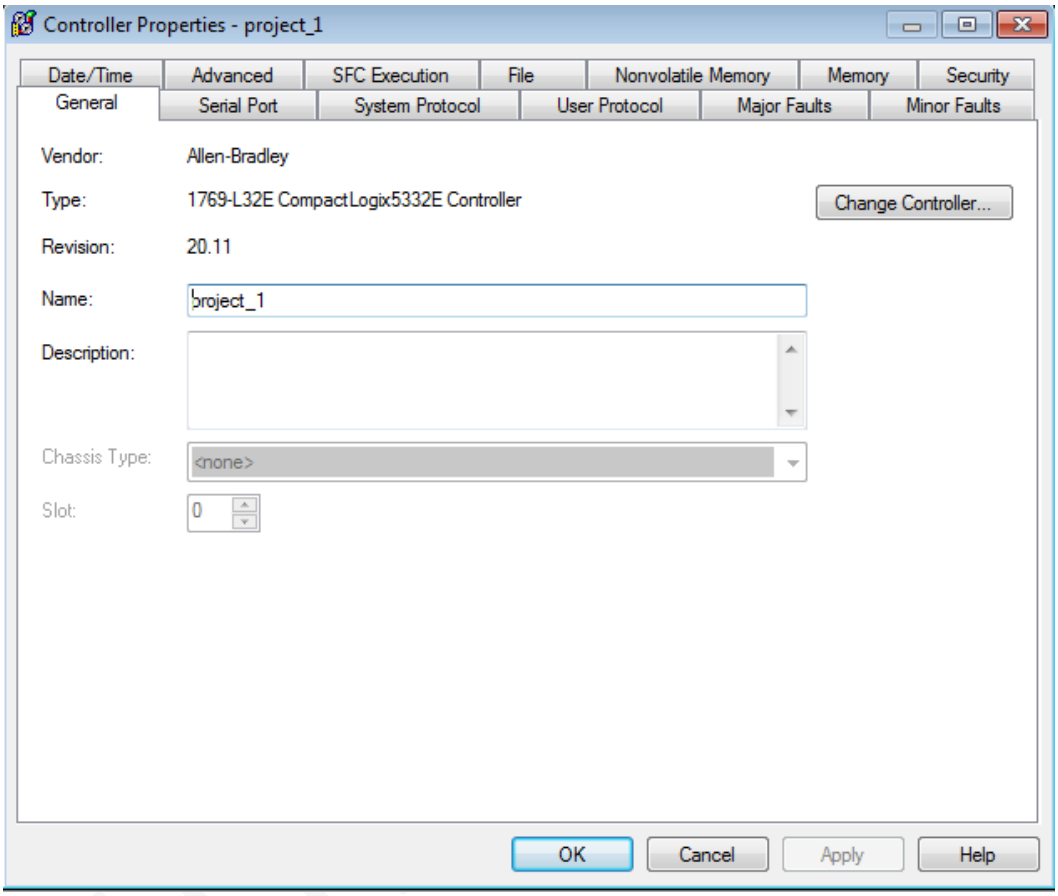


- **Type**: Loại CPU, chọn đúng loại sử dụng.
- **Revision**: Chọn Phiên bản phần mềm RSLogix 5000 đang sử dụng, cần tương thích firmware của phần cứng PLC.
- **Name**: Đặt tên cho project và tên này sẽ hiển thị bên cạnh CPU trong RSLinx để biết CPU nào là của chương trình nào.
- **Chassis Type**: Chọn đúng loại khung sử dụng cho bộ điều khiển.
- **Slot**: Thứ tự khe cắm của CPU trên khung.
- **Create In**: chọn thư mục lưu file.

- Bước 4: Chọn ***OK.***

!*Lưu ý*: Trong một dự án Logix 5000, bạn có thể đặt tên cho các phần tử của dự án như là bộ điều khiển, địa chỉ dữ liệu (tags), module vào ra I/O…. Tên chỉ được phép chứa các chữ cái, chữ số và dấu gạch dưới, phải mở đầu bằng chữ cái hay gạch dưới, có tối đa 40 ký tự và không có các gạch dưới liên tục hay kế tiếp nhau, không trùng với các tên nhạy cảm.

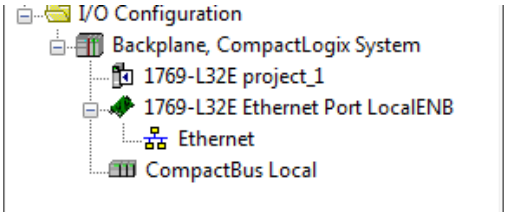
Để cấu hình lại, vào **Edit → Controller Properties**, sẽ hiện ra cửa sổ để thay đổi các thông tin mong muốn. Trong một số trường hợp có báo lỗi trên CPU, cần vào tab **Major Faults** hoặc **Minor Faults** (ở chế độ Offline) để Clear Faults.



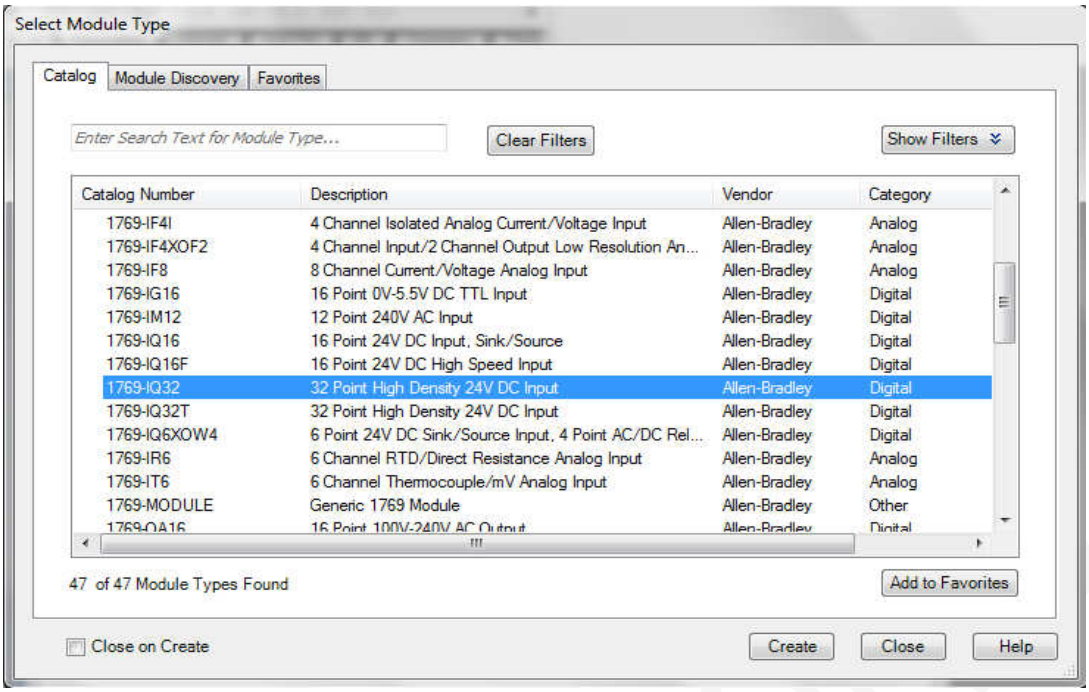
2.2 Thêm module vào ra I/O

Để giao tiếp với module vào ra trong hệ thống, chúng ta thêm các module vào thư mục I/O configuration của bộ điều khiển. Đặc tính mà chúng ta chọn sẽ xác định hành vi của module.

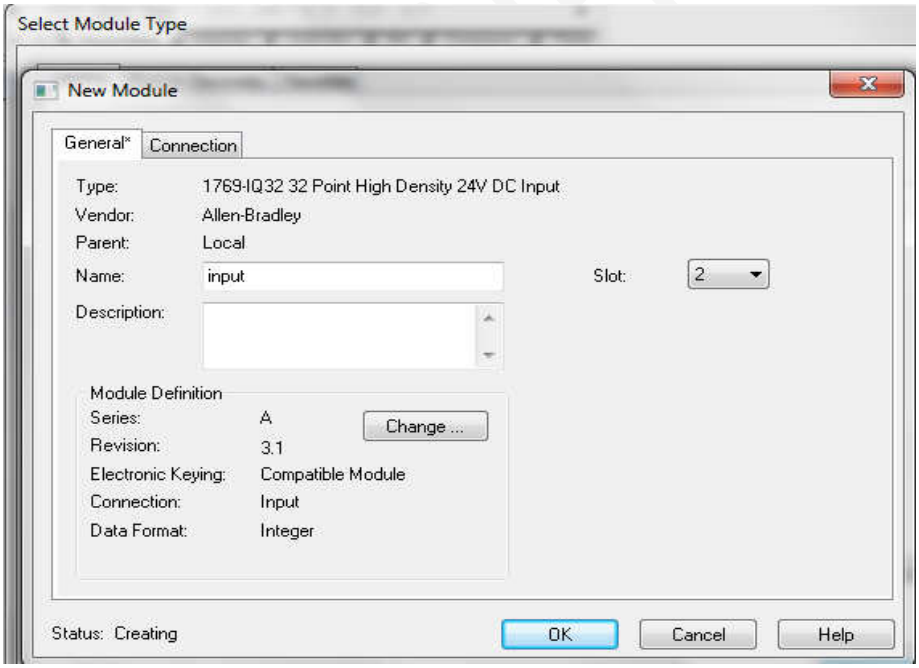
- Bước 1: Mở thư mục **I/O Configuration** và click chuột phải vào **CompactBus Local**, chọn **New Module.**



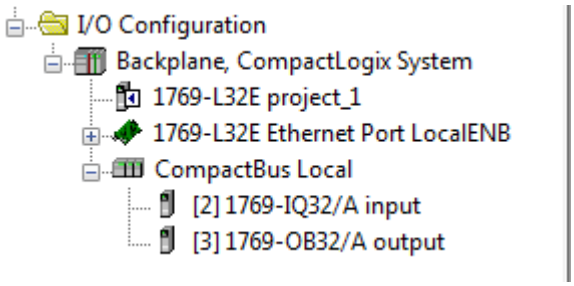
- Bước 2: Chọn loại module vào ra I/O ứng với loại sử dụng (kiểm tra mã trên module thực tế). Sau đó nhấn **Create.**



- Bước 3: Đặt tên cho module rồi khai báo khe (slot) đặt module trên khung thực tế. Đối với CompactLogix 1769 L32E, số thứ tự của slot được tính từ CPU (slot 0) và bỏ qua khối nguồn. Click **OK.**



Sau khi hoàn tất ta được:

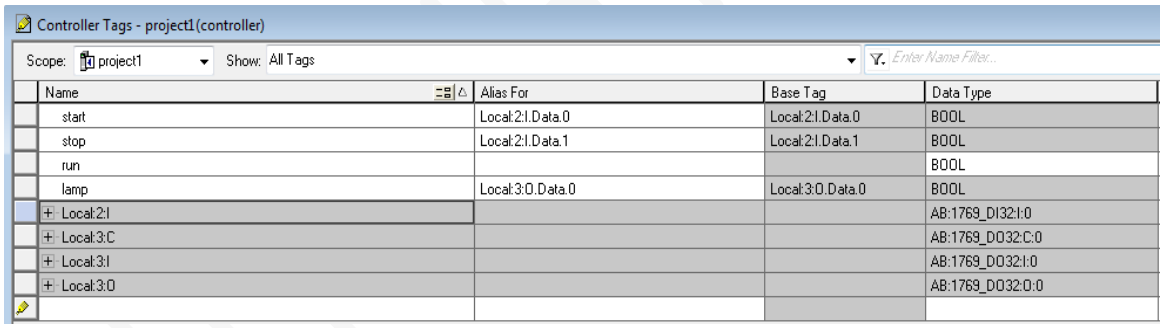


2.3. Lập trình một chương trình đơn giản

Ví dụ: Điều khiển Start-Stop.

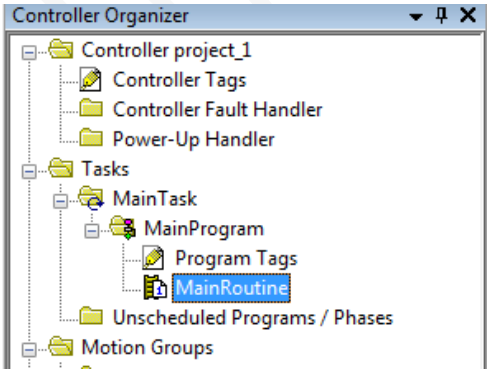
- Bước 1: Gán địa chỉ.

  - Tại mục **Controller Tags** hoặc **Program Tags**: chọn **Edit Tags** ta thiết lập các biến cần dùng cho chương trình lập trình. Theo yêu cầu điều khiển thì sẽ có 2 đầu vào: nút start và stop; 1 đầu ra: lamp; 1 biến trung gian: run
  - Tại mục **Alias for**
    - Đối với đầu vào (start) ta chọn Local:2:I, bấm vào dấu + và chọn Local:2:I:Data và chọn điều khiển đầu vào là 0 (công tắc 00), tương tự ta chọn nút stop với đầu vào là 1 (công tắc 01).
    - Đối với tín hiệu đầu ra (lamp) ta chọn Local:3:O, bấm vào dấu + và chọn Local:3:O:Data và chọn 0 (đèn 00).
    - Tín hiệu run là tín hiệu ra logic trung gian.

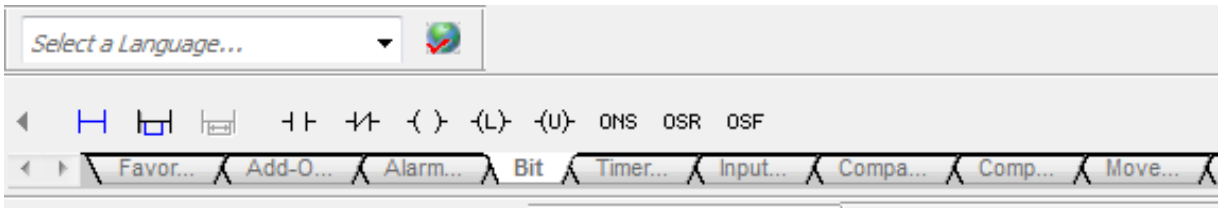


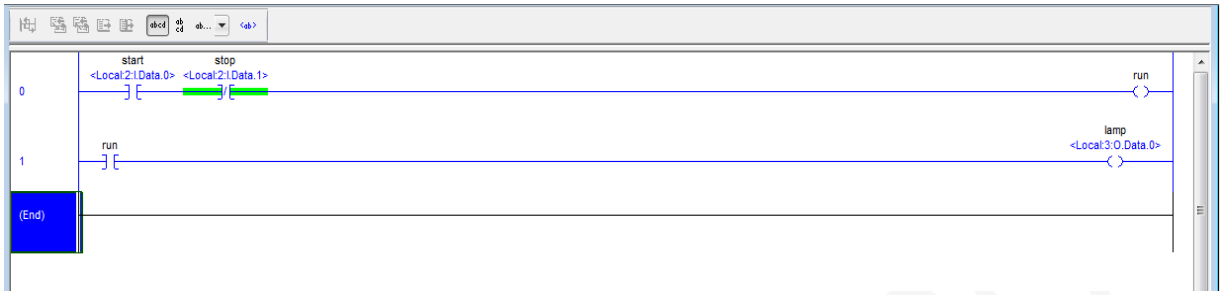
- Bước 2: Viết chương trình.

Từ cây thư mục chọn **Tasks / MainTask/ MainProgram/ MainRoutine**

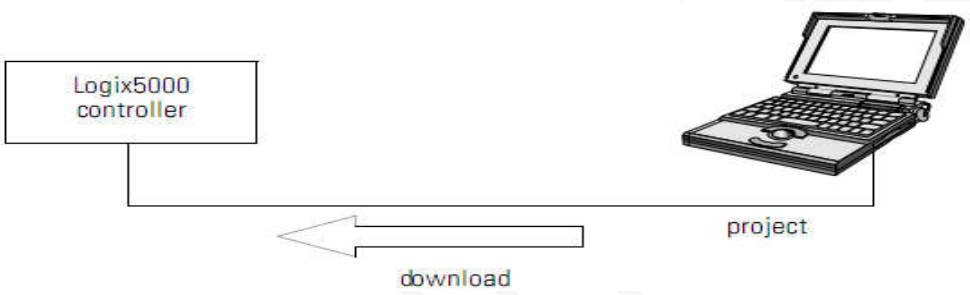


Lựa chọn mục các lệnh về "Bit" và kéo thả các biểu tượng. Với biến *start* ta sử dụng tiếp điểm vào thường mở, *stop* sử dụng tiếp điểm vào thường đóng. *run* và *lamp* sử dụng tiếp điểm đầu ra. Nhấp đúp hoặc nhấp chuột phải vào từng biểu tượng để khai báo tên biến ở bước 1.

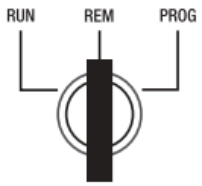




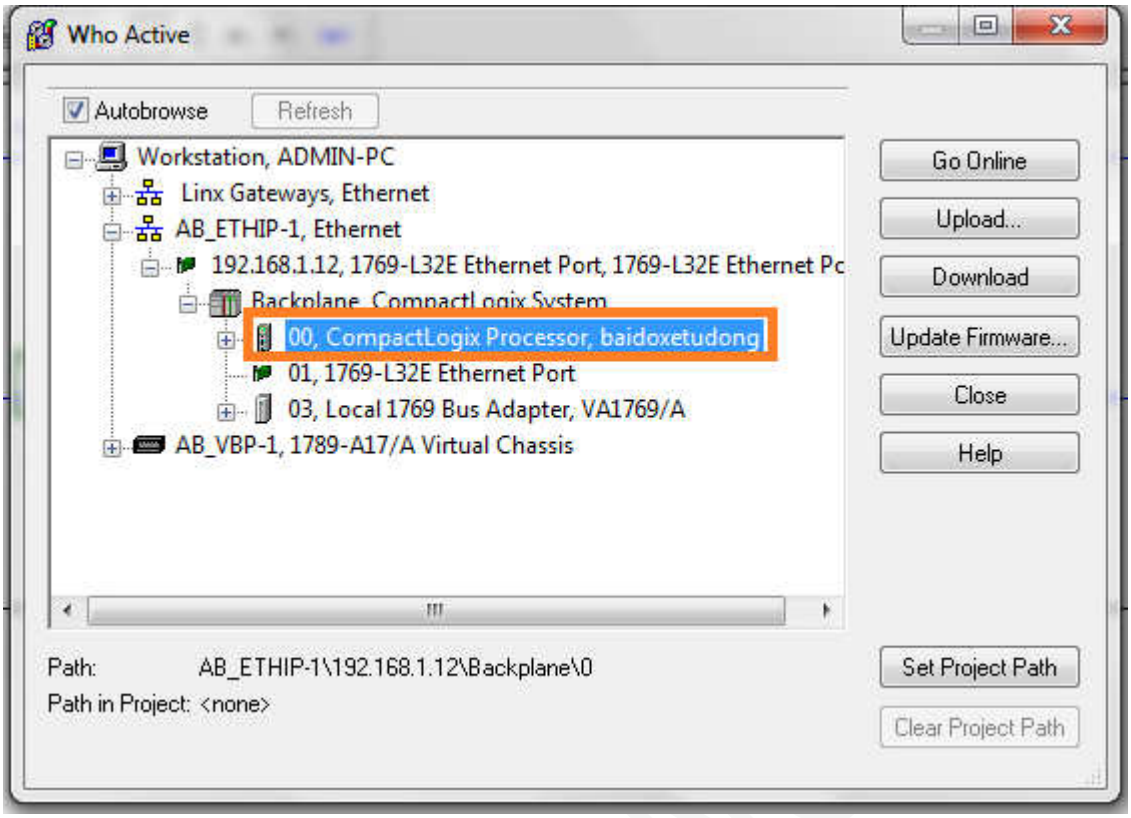
2.4. Download chương trình xuống bộ điều khiển



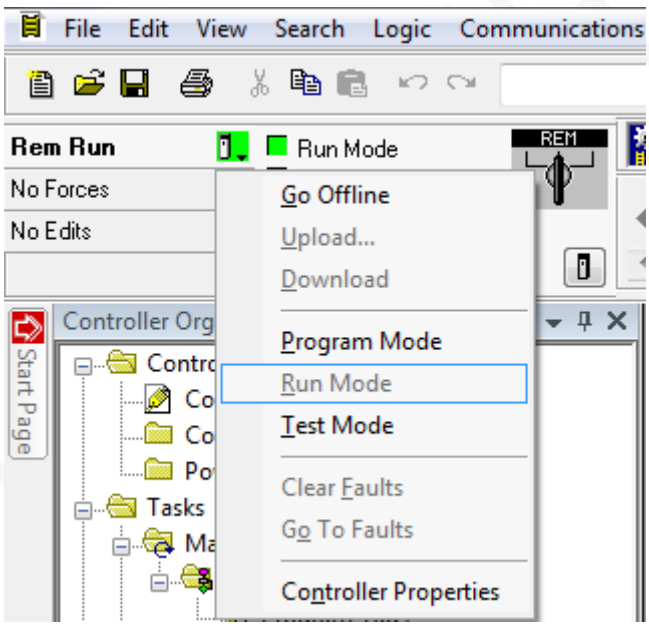
- Chuyển khóa công tắc của bộ điều khiển về trạng thái Remote (REM):



- Ở **MainRoutine** đang mở, chọn 
- Chọn **CompactLogix Processor** và ấn **Download**



- Chọn chế độ làm việc: click vào  và chọn Runmode



- Kiểm tra ví dụ trước bằng cách gạt công tác input 0 và input 1 trên kit thí nghiệm.

!Lưu ý: Khi tải xuống một chương trình mới thì chương trình và dữ liệu hiện thời trong bộ điều khiển sẽ mất. Nếu phiên bản firmware của bộ điều khiển và chương trình không tương thích, cần phải cập nhật lại firmware cho bộ điều khiển. Để cập nhật firmware cho bộ điều khiển thì trước tiên phải cài đặt và sử dụng FLASH Programming Tool.

3. Hướng dẫn sử dụng phần mềm giả lập RSLogix Emulate

3.1. Tổng quan về RSLogix Emulate 5000

RSLogix Emulate là phần mềm giả lập PLC họ Logix5000 của hãng Rockwell trên PC, có vai trò như một PLC thực, có thể nạp các chương trình đã được lập trình bằng phần mềm RSLogix, sau đó xử lý và điều khiển. RSLogix Emulate giúp người dùng có thể viết chương trình và kiểm nghiệm ở mọi lúc mà không cần có bộ PLC thực. RSLogix Emulate 5000 kết nối với RSLogix 5000 thông qua phần mềm RSLinx.

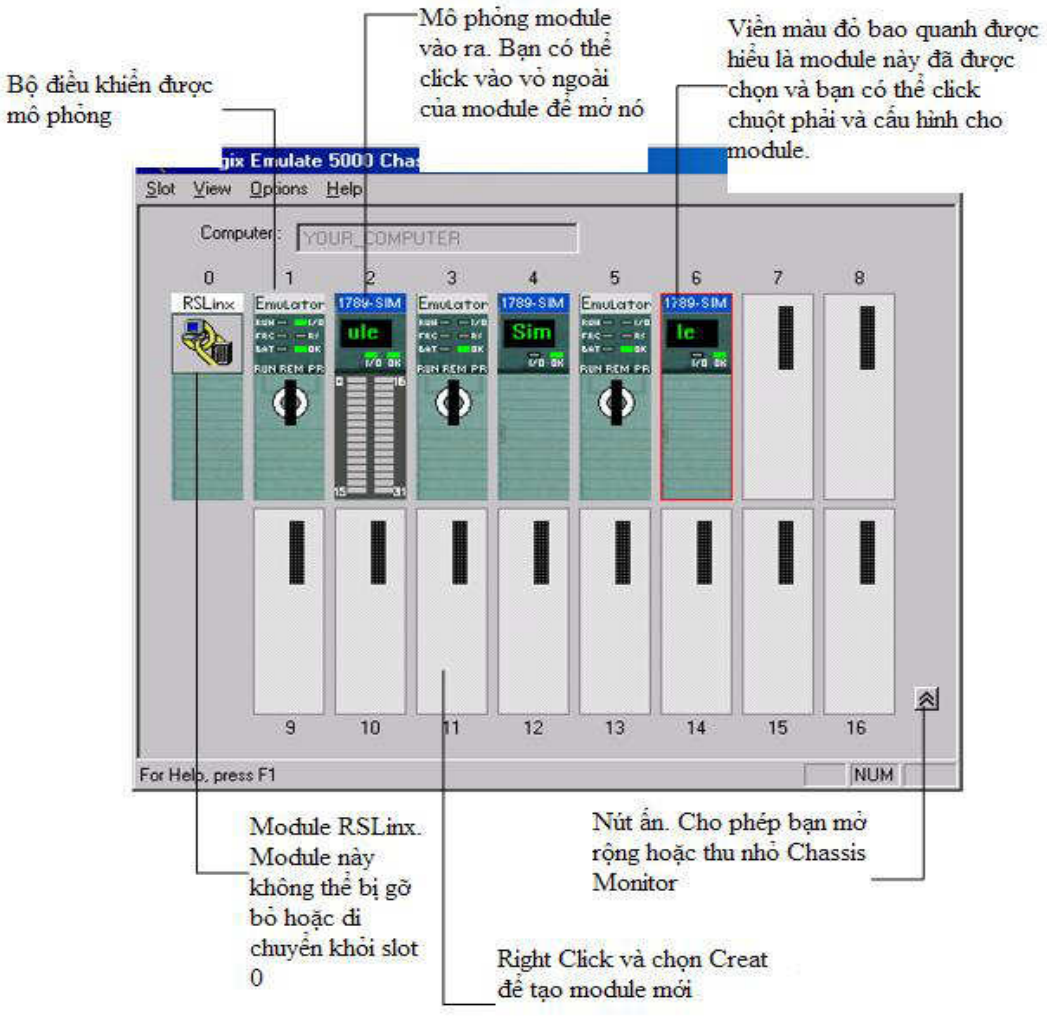
RSLogix Emulate gồm 2 thành phần chính:

- *Chassis Monitor*: là một ứng dụng giả lập khung chứa của PLC, cho phép ta cấu hình, thêm các module vào khung.
- *Emulation modules*: Gồm những module đã được mô phỏng tương tự các module xử lý Logix5000 hoặc các module vào ra.

Chú ý: Thời gian thực hiện chương trình ở trong RSLogix Emulate 5000 sẽ không thể bằng thời gian thực hiện ở trong bộ xử lý họ Logix 5000, vì thời gian thực hiện chương trình trong RSLogix Emulate 5000 còn phụ thuộc vào những ứng dụng khác đang chạy đồng thời trên PC.

3.2. Tìm hiểu giao diện Chassis Monitor

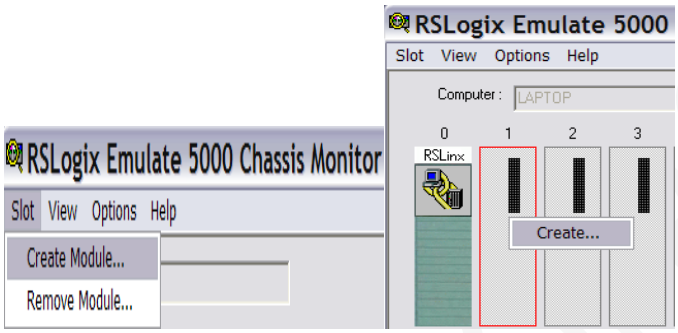
Dưới đây là cửa sổ Chassis Monitor gồm 3 bộ xử lý và 3 module I/O được mô phỏng:



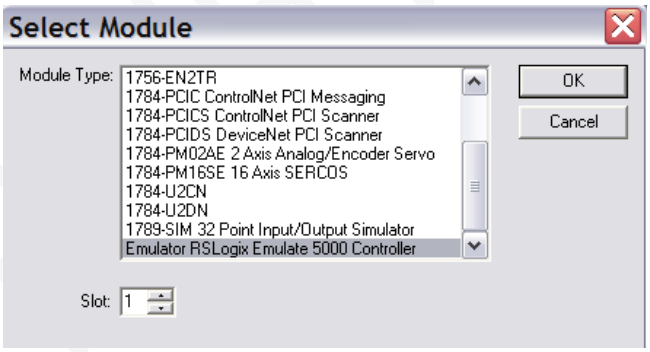
Chassis Monitor là nơi bạn có thể tạo và cấu hình các module điều khiển và giả lập những module vào ra. Những module này được đặt vào các khe của Chassis Monitor như ở những Chassis vật lý thông thường. Module RSLinx luôn ở vị trí khe cắm đầu tiên tức Slot 0, chúng ta không thể gỡ bỏ hay chuyển sang Slot khác.

3.3. Các bước giả lập bộ điều khiển và module vào ra

- Bước 1. Chọn menu *Slot* → *New Module* hoặc click chuột phải vào một slot bất kỳ chọn *Creat*.

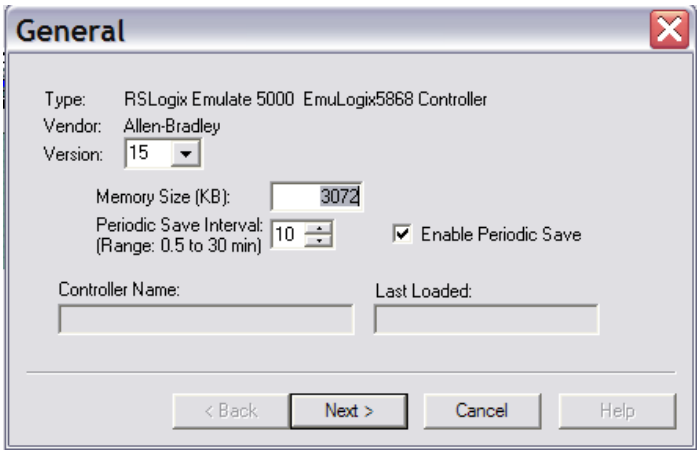


- Bước 2**.** Từ khung *Module Type* chọn loại module mà bạn muốn khởi tạo. Nếu bạn muốn giả lập bộ xử lý thì chọn *Emulator RSLogix Emulate 5000 Controller* hoặc giả lập module vào ra I/O thì chọn *1789-SIM 32 Point Input/Output Simulator*.

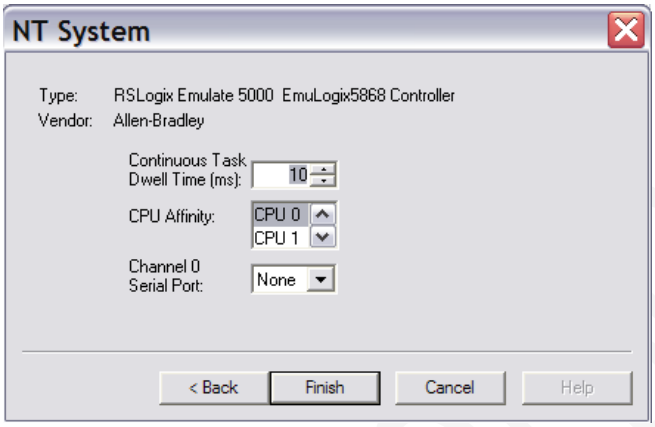


- Bước 3**.** Chọn Slot mà bạn muốn đặt vào. Bắt đầu từ Slot 1(trừ Slot 0). Nhấn *OK*
- Bước 4. Cấu hình cho module vừa tạo.

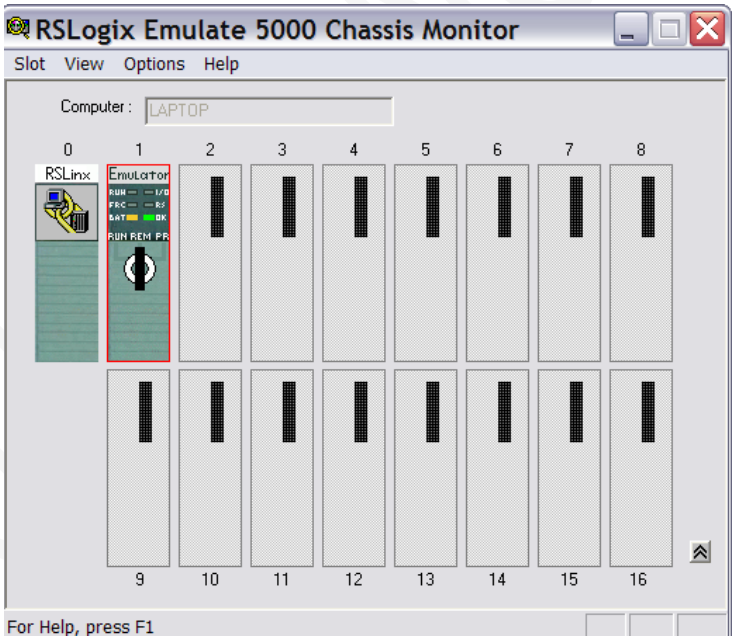
Trường hợp giả lập Module điều khiển



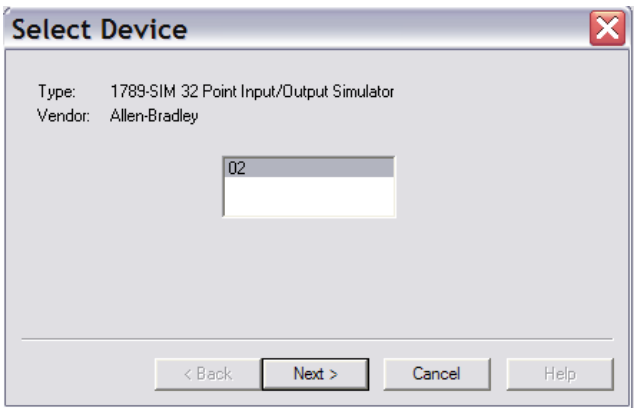
- **Version**: Chọn phiên bản RSLogix Emulate. Ví dụ ở đây là version 15.
- **Memory Size**: Dung lượng bộ nhớ bạn muốn. Đơn vị là KB.
- **Periodic Save Interval**: Khoảng thời gian lưu lại. Từ 0.5 đến 30 phút.
- Sau đó ấn **Next.**



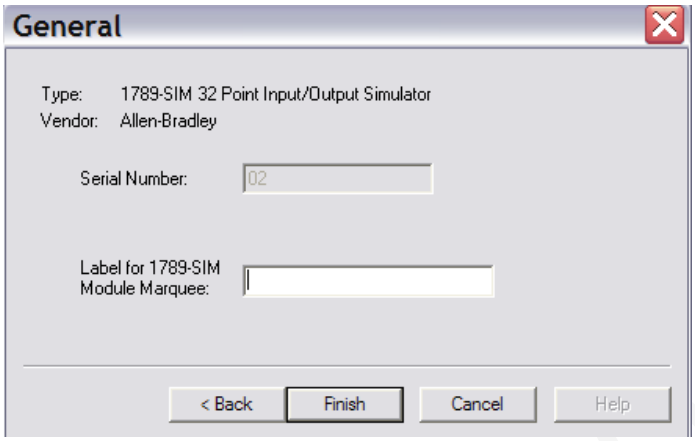
- Chọn **Finish** ta được:



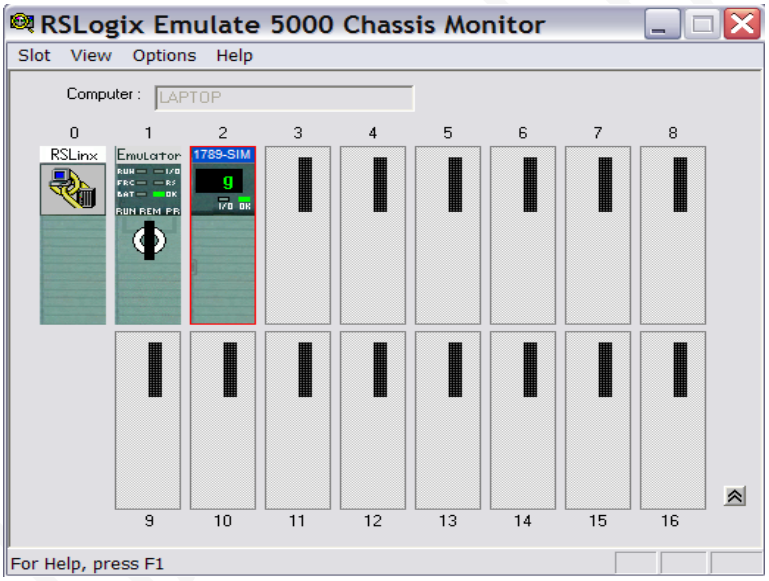
Trường hợp giả lập Module vào ra I/O



- Click **Next:**

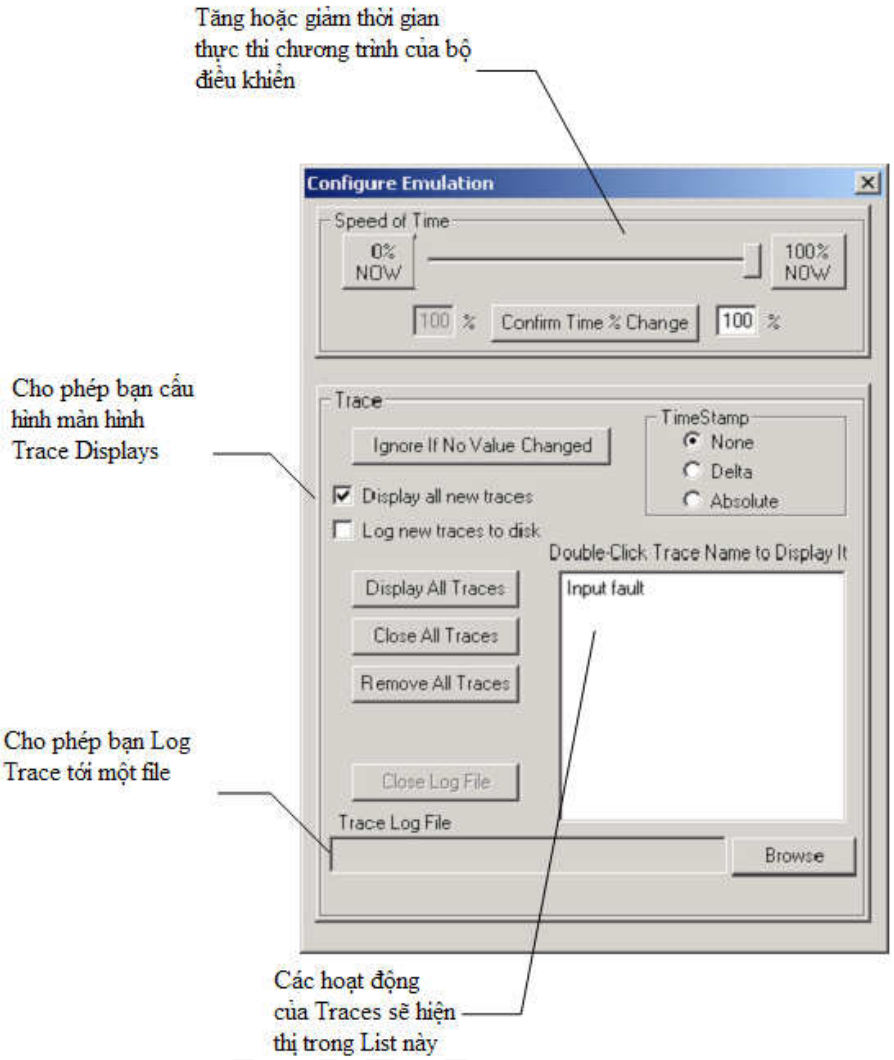


- **Label for 1789 – SIM Module Marquee** : Đặt tên cho module vào ra.
- Sau đó chọn **Finish**. Như vậy là ta đã thiết lập xong giả lập module vào ra I/O.



3.4. Cấu hình giả lập bộ điều khiển

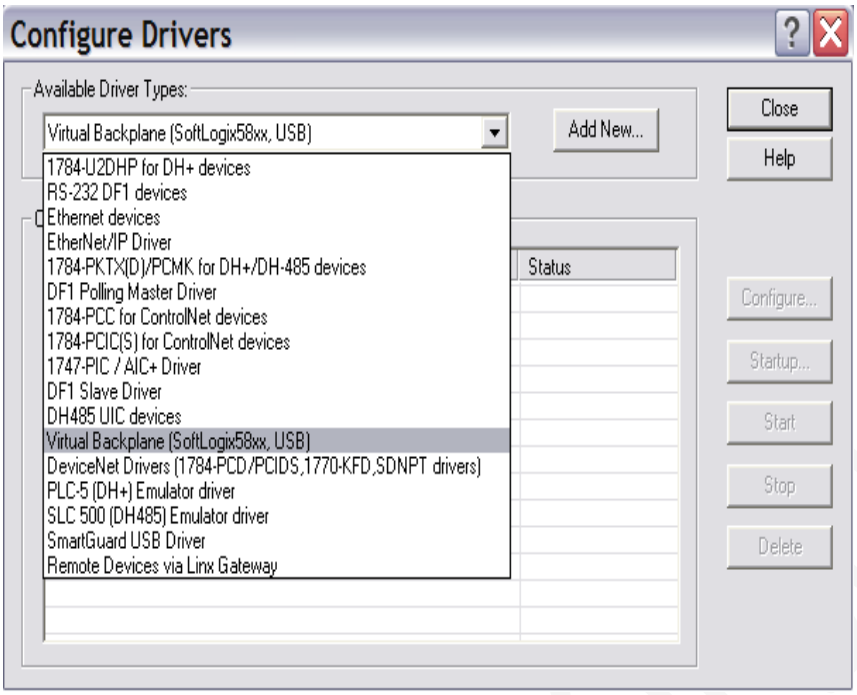
Muốn cấu hình bộ điều khiển click chuột phải vào bộ điều khiển và chọn **Configure Emulation.**



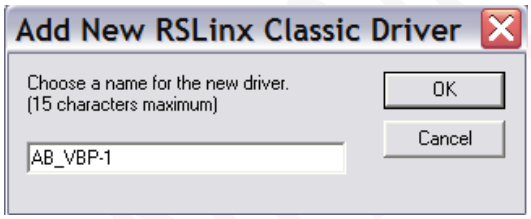
Speed of Time cho phép tăng hoặc giảm thời gian Emulator thực thi chương trình điều khiển. Khi giảm thời gian này, các timer cũng chậm lại, thời gian của 1 vòng quét cũng giảm, cho phép ta quan sát được cách thức chương trình điều khiển được thực thi.

3.7. Cấu hình Driver cho RSLogix Emulate 5000 trong RSLinx

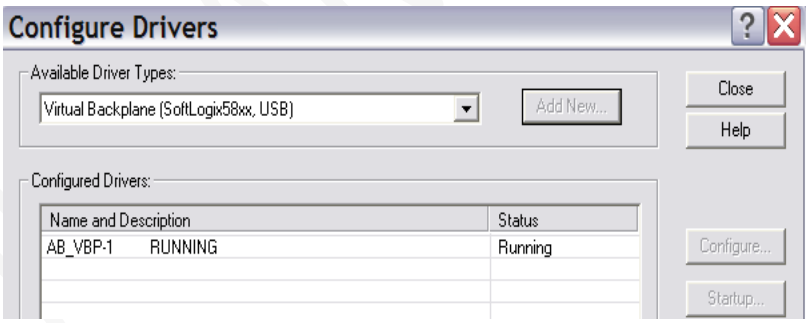
- Bước 1. Trong cửa sổ RSLinx chọn **Communications → Configure Drivers**. Một cửa sổ xuất hiện
- Bước 2. Chọn **Virtual Backplane (SoftLogix 58xx)** từ **Available Driver Types**



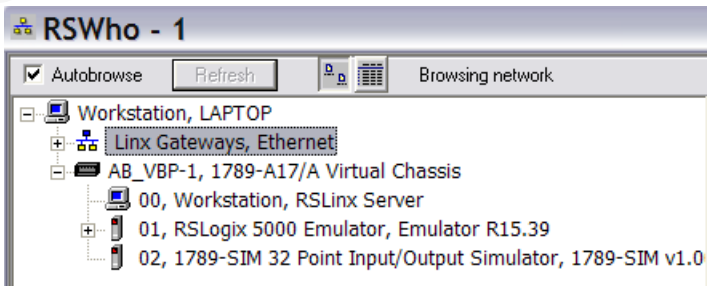
- Bước 3. Click **Add New**



- Bước 4. Click **OK**

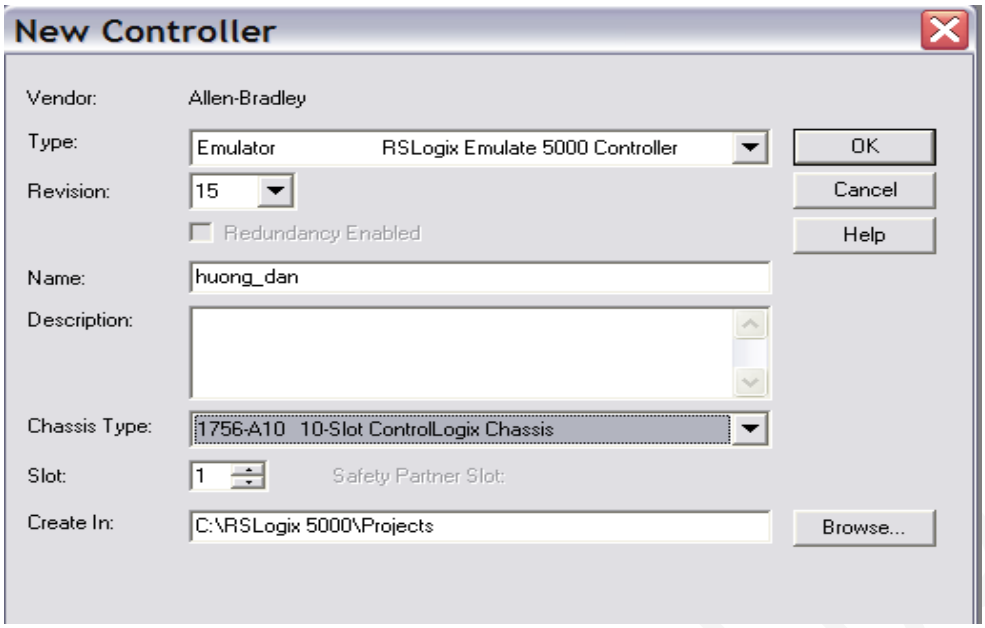


- Bước 5. Chọn Close. Sau đó chọn **Communication → RSWho** để kiểm tra:

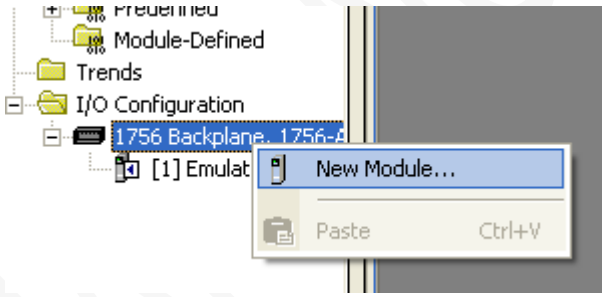


3.5. Kết nối với RSLogix 5000

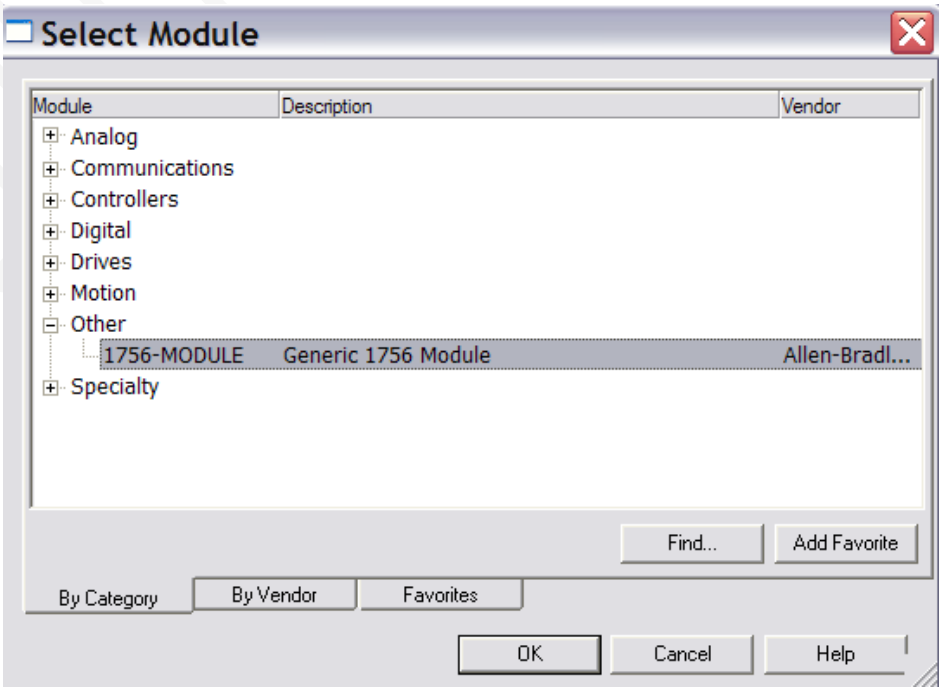
- Bước 1. Trong cửa sổ RSLogix chọn **File → New**



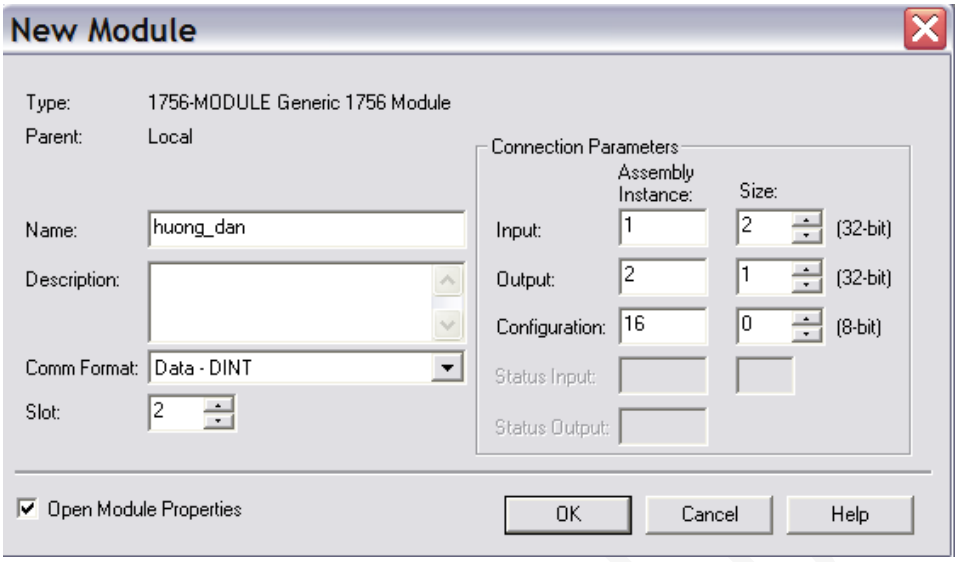
- Bước 2. Trong mục **Type** chọn bộ điều khiển là **Emulator RSLogix Emulate 5000 Controller.** Revision là 15. Đặt tên cho project và nhấn **OK.**
- Bước 3. Đưa thêm module. Ấn chuột phải vào bộ điều khiển chọn **New Module**.



- Bước 4. Chọn loại module là **1756-MODULE**. Sau đó nhấn **OK**.

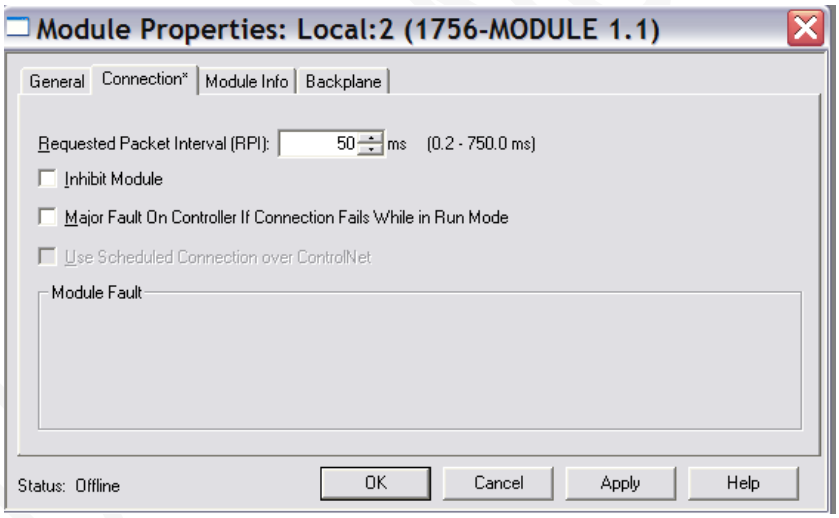


- Bước 5. Một cửa sổ mới sẽ hiện ra. Ta chỉnh định như hình dưới:



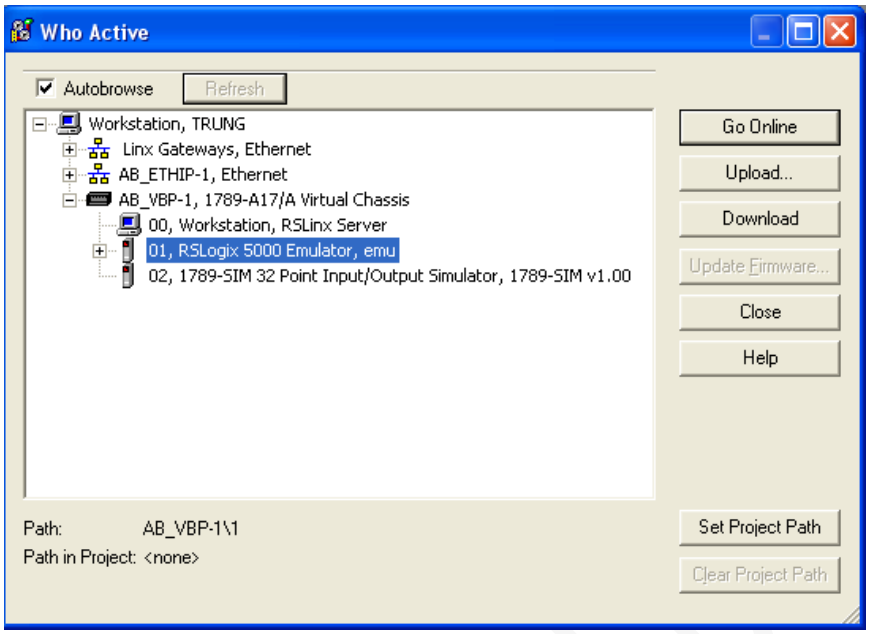
Chú ý: Mục Slot chọn Slot chứa bộ điều khiển trong Chassis của RSEmulate 5000. Ví dụ ở đây ta chọn Slot 1.

- Bước 6. Chọn **OK.** Cửa sổ mới hiện ra, đặt RPI tối thiểu là 50.



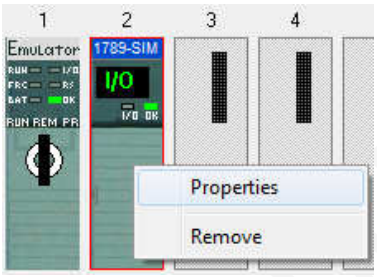
- Bước 7. Tạo đường dẫn tới bộ điều khiển. Chọn **Who Active**

- Bước 8. Ấn vào Emulator mà bạn muốn sử dụng cho project, rồi chọn **Set Project Path**. Ấn Close.

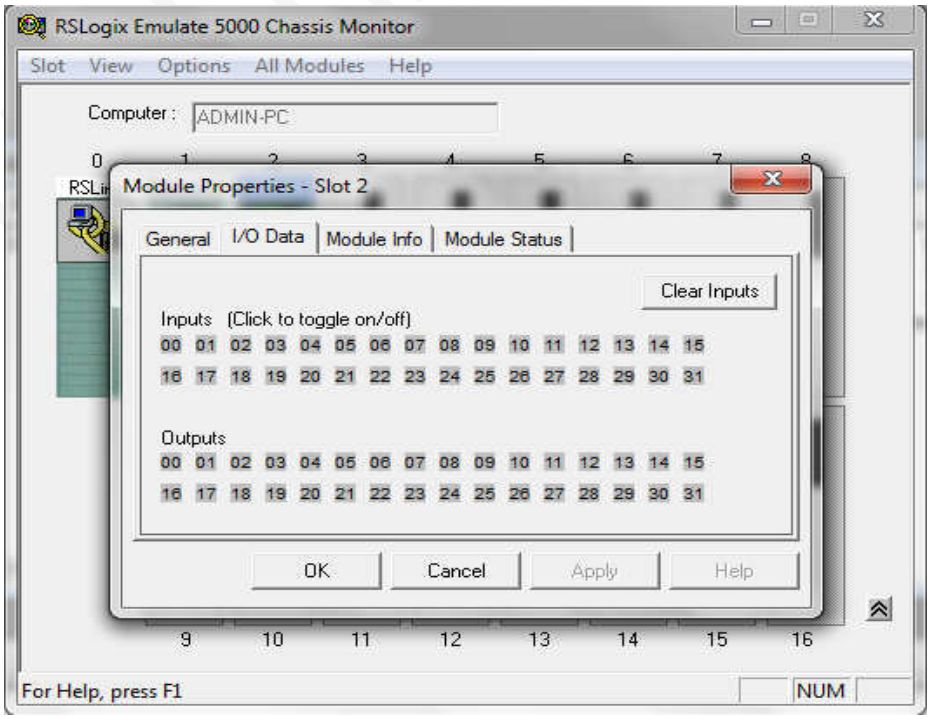


Bây giờ ta có thể lập trình, dowload và chạy thử chương trình.

Để xem trạng thái các ngõ vào ra số. Nhấp phải chuột vào module ngõ vào ra số và chọn **Properties.**



Hộp thoại **Module Properties** xuất hiện. Chọn tab **I/O Data** để hiển thị trạng thái các ngõ vào ra số.



4. Nhóm các lệnh xử lí bit, Timer và Counter

Để tìm hiểu thêm về chức năng của một lệnh trong RSLogix5000, chọn lệnh và ấn F1.

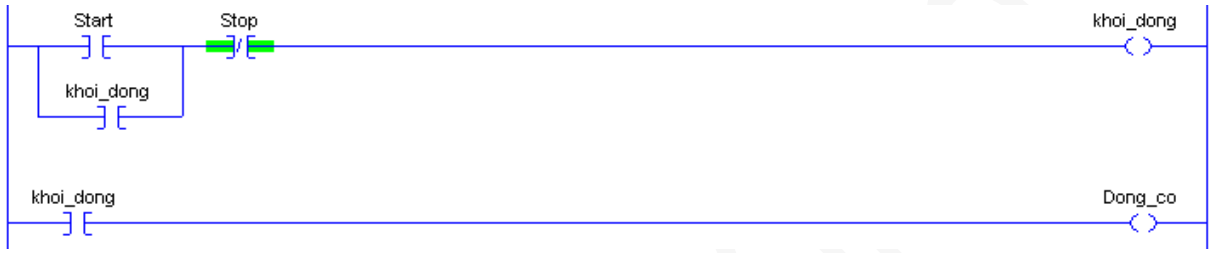
4.1. Nhóm các lệnh xử lí bit

Examine on: Tiếp điểm thường mở, khi có tác động sẽ đóng.

Examine off: Tiếp điểm thường đóng, khi có tác động sẽ mở.

Output Energize: Đầu ra, bật On khi điều kiện đầu vào là đúng.

Ví dụ:

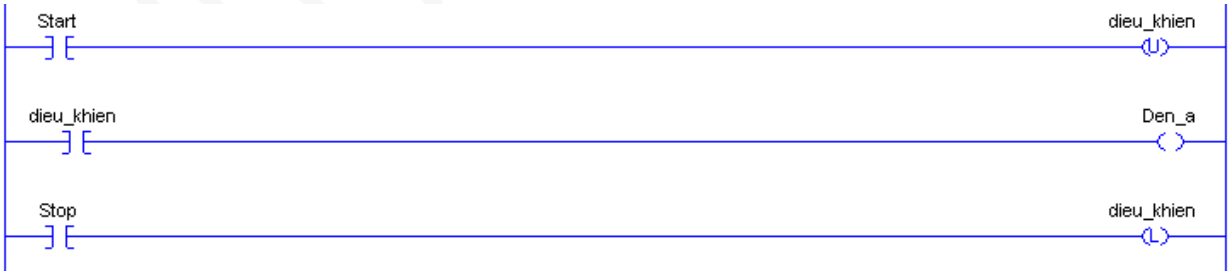


Khi *Start* được kích hoạt thì rơle *khoi_dong* có điện, đóng các tiếp điểm *khoi_dong*. Tiếp điểm *khoi_dong* thứ nhất đóng vai trò tiếp điểm tự duy trì, cấp điện cho rơle *khoi_dong* ngay cả khi *Start* ngừng được kích hoạt. Tiếp điểm *khoi_dong* thứ hai cấp nguồn cho rơle *Dong_co* để đóng điện cho động cơ. Khi muốn dừng động cơ ta kích hoạt *Stop* ngắt điện của rơle *khoi_dong*, làm các tiếp điểm *khoi_dong* mở, ngắt điện của động cơ. Chú ý đây là mạch Start-Stop rất hay dùng trong lập trình bằng PLC.

Để mô phỏng chuyển sang chế độ Online/Runmode và ấn chuột phải vào Start, chọn Toggle Bit

Output Latch (L), Output Unlatch (U) : là các lệnh đầu ra duy trì, (L) dùng để bật 1 bit còn (U) dùng để tắt 1 bit, các lệnh này thường dùng theo 1 cặp với cùng 1 địa chỉ.

Ví dụ:



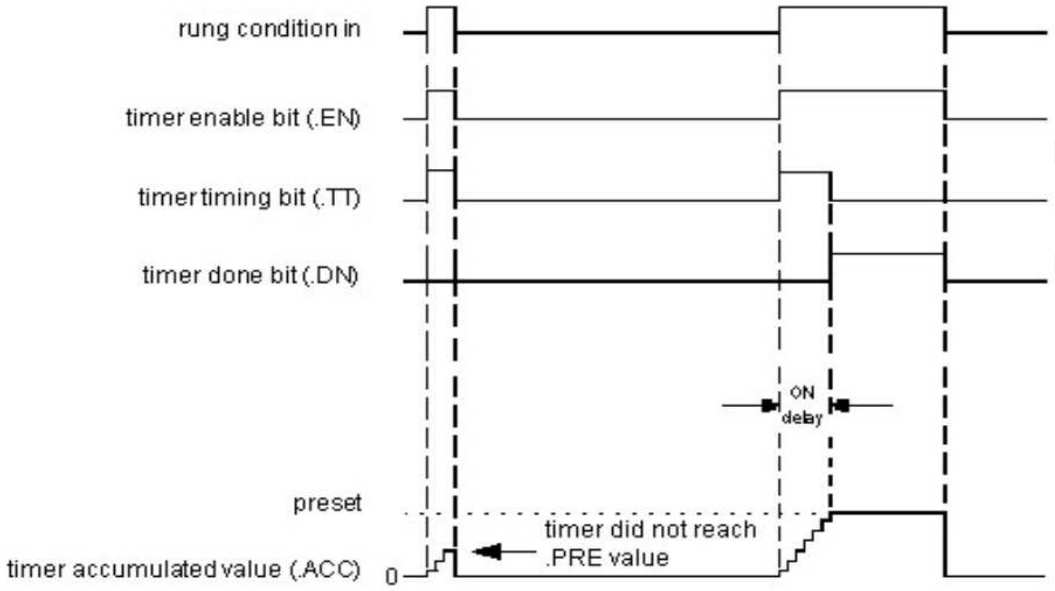
Khi *Start* được kích hoạt, đầu ra *dieu_khien (U)* có điện (và sẽ tự duy trì kể cả khi *Start* ngừng kích hoạt) sẽ đóng tiếp điểm *dieu_khien*, *Den_a* bật. Khi ta ấn *Stop* đầu ra *dieu_khien (L)* có điện nó sẽ mở tiếp điểm *dieu_khien*, *Den_a* tắt.

4.2. Timer và Counter (TON, TOF, RTO, CTU, CTD, RES)

Các lệnh Timer/Counter không tác động lên các cờ trạng thái toán học.

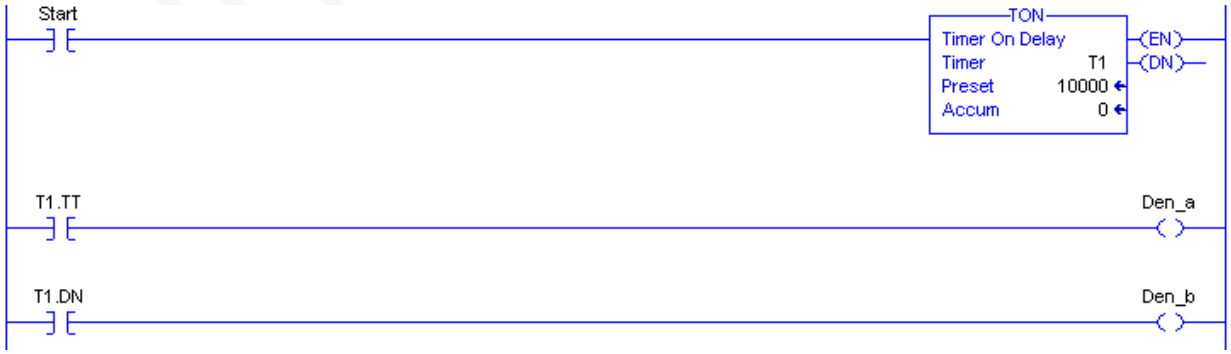
4.2.1 Timer On Delay TON

Dùng để bật hoặc tắt 1 đầu ra sau khi timer (bộ định thời) được kích hoạt và đếm trong một khoảng thời gian định trước. Mỗi khi đầu vào bị ngắt thì lệnh TON sẽ được reset lại (giá trị trong Accum về 0).



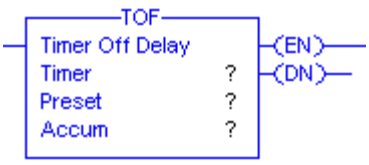
Tên	Kiểu dữ liệu	Mô tả
Timer	Timer	Tên bộ định thời
.EN	Bool	On khi đầu vào on
.TT	Bool	On trong khi bộ định thời đang đếm.
.DN	Bool	On khi điều kiện thời gian thỏa mãn .ACC >= .PRE
.PRE	Dint	Khoảng thời gian đặt trước (Preset) (ms).
.ACC	Dint	Giá trị đếm (Accum).

Ví dụ:

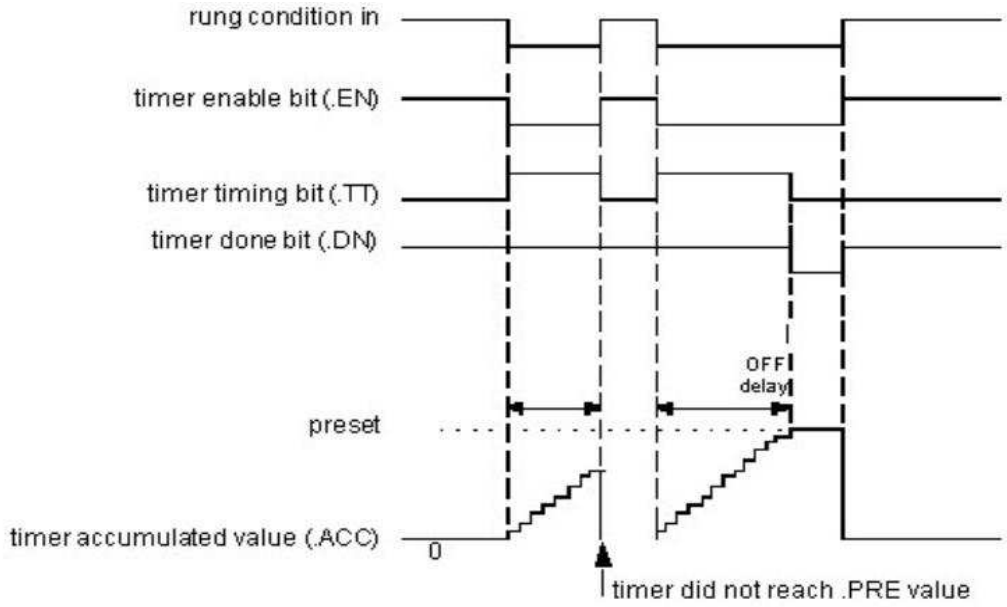


Khi *Start* được kích hoạt, bộ định thời *T1* có điện và bắt đầu đếm thời gian từ 0-10s (quan sát giá trị của Accum). Trong khoảng thời gian 10s này tiếp điểm *T1.TT* on, Den_a bật và *T1.DN* off, *Den_b* tắt. Sau khi hết 10s tiếp điểm *T1.TT* off, *Den_a* tắt, T1.DN on, *Den_b* sáng. Khi *Start* ngừng được kích hoạt thì bộ định thời *T1* sẽ được reset lại (giá trị trong Accum trở về 0).

4.2.2. Timer Off Delay TOF

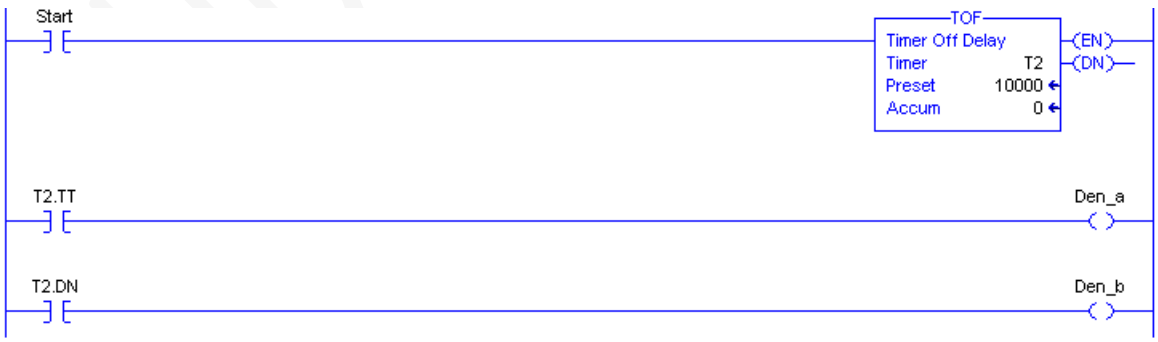


Dùng để bật hoặc tắt 1 đầu ra sau khi đầu vào của bộ định thời tắt một khoảng thời gian định trước. Mỗi khi đầu vào bộ định thời được kích hoạt, lệnh TOF sẽ được reset lại (giá trị trong Accum trở về 0).



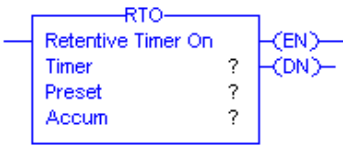
Tên	Kiểu dữ liệu	Mô tả
Timer	Timer	Tên bộ định thời
.EN	Bool	On khi đầu vào on
.TT	Bool	On trong khi bộ định thời đang đếm.
.DN	Bool	On khi đầu vào ON. Off khi điều kiện thời gian là đúng .ACC >= .PRE
.PRE	Dint	Khoảng thời gian đặt trước (Preset) (ms).
.ACC	Dint	Giá trị đếm (Accum).

Ví dụ:



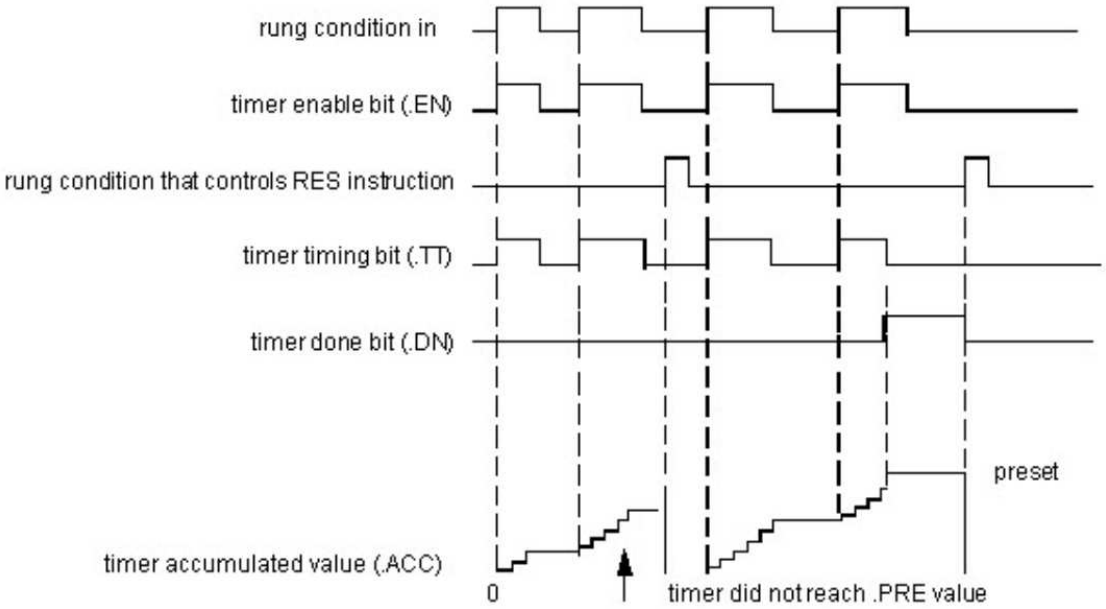
Khi *Start* được kích hoạt, bộ định thời *T2* có điện *Den_a* tắt, *Den_b* bật. Khi ngừng kích hoạt *Start* sẽ ngắt điện của *T2*, khi đó *T2* sẽ đếm từ 0-10s (xem trong Accum), trong 10s này tiếp điểm *T1.TT* on *Den_a* bật (lúc này *Den-b* vẫn bật). Sau khi đếm xong 10s, các tiếp điểm *T2.TT* và *T2.DN* đều off, cả *Den_a* và *Den_b* đều tắt. Lúc này giá trị trong Accum vẫn giữ 10s và khi đầu vào *T2* on trở lại thì nó được reset (giá trị trong Accum về 0).

4.2.3. Retentive Timer On RTO

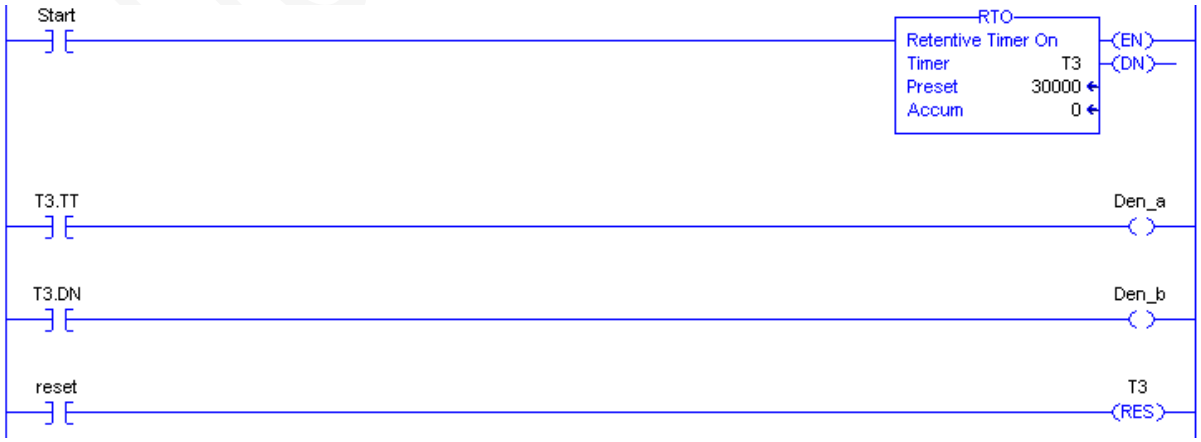


Hàm thời gian có nhớ dùng để bật hoặc tắt một đầu ra khi điều kiện đầu vào là đúng. Khi mất nguồn hàm RTO tiếp tục duy trì tại giá trị tại lúc dừng (.ACC vẫn giữ nguyên giá trị). Muốn reset lệnh RTO ta phải dùng lệnh RES để xóa giá trị trong .ACC.

Tên	Kiểu dữ liệu	Mô tả
Timer	Timer	Tên bộ định thời
.EN	Bool	On khi đầu vào on
.TT	Bool	On trong khi bộ định thời đang đếm.
.DN	Bool	On khi điều kiện thời gian là đúng .ACC >=.PRE
.PRE	Dint	Khoảng thời gian đặt trước (Preset) (ms).
.ACC	Dint	Giá trị đếm (Accum).



Ví dụ:



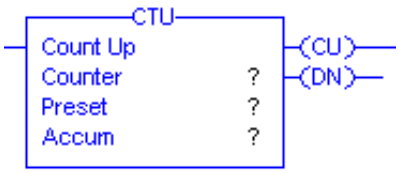
Khi *Start* được kích hoạt, bộ định thời *T3* có điện và sẽ bắt đầu đếm thời gian từ 0-30s, trong khoảng thời gian khi *T3* đang đếm thì *Den_a* bật, *Den_b* tắt. Giả sử khi đang đếm đến 10s thì *Start* off, khi đó giá trị trong *Accum* vẫn giữ nguyên ở 10s (10000), cả *Den_a* và *Den_b*

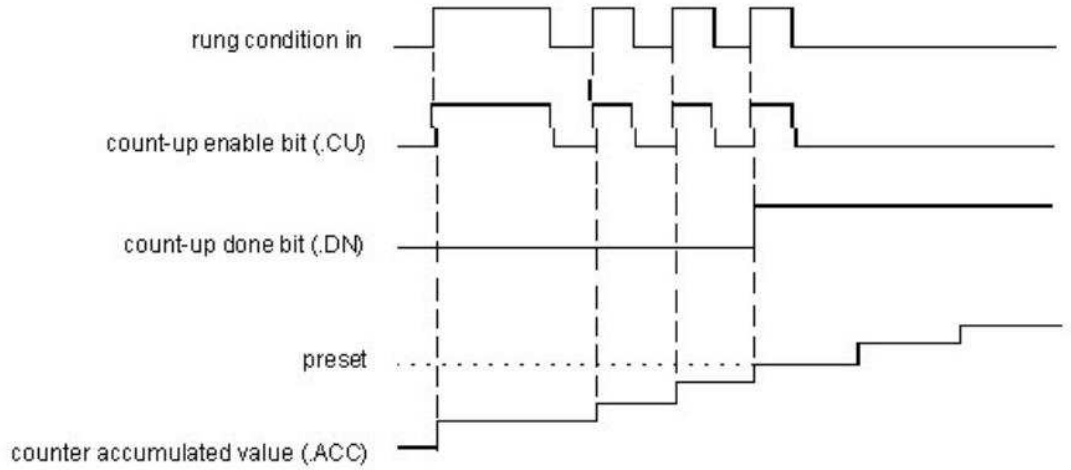
đều tắt. Khi *Start* on trở lại *T3* lại tiếp tục đếm thời gian từ giá trị 10s, *Den_a* bật. Sau khi đếm xong 30s thì *Den_a* tắt, *Den_b* bật. Khi này dù *Start* có off thì do *T3* không được reset lại nên *Den_b* vẫn sáng.

Khi tiếp điểm *reset* on, lệnh RES sẽ reset lại bộ định thời *T3* (xóa giá trị trong Accum về 0), *Den_b* tắt.

4.2.4. Count Up CTU

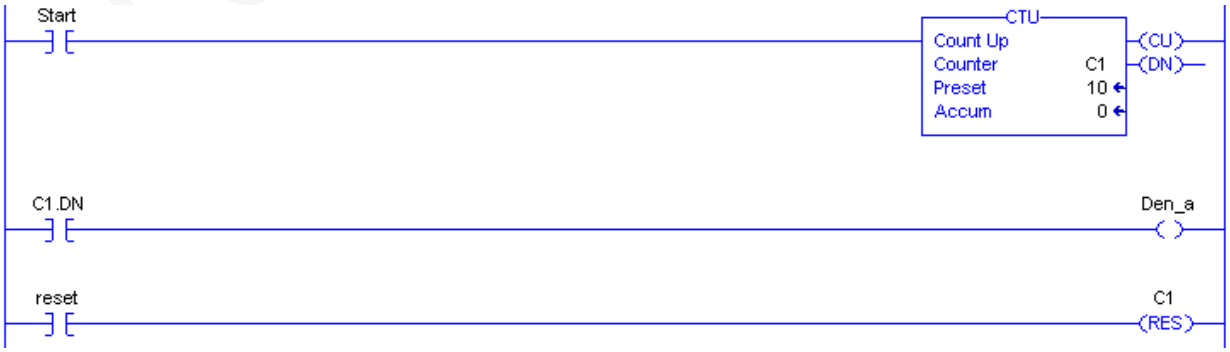
Dùng để đếm số lần chuyển từ sai sang đúng của điều kiện đầu vào. Khi đầu vào chuyển từ off lên on thì giá trị *Accum* của bộ đếm sẽ tăng thêm 1, đến khi bằng giá trị đặt (*Preset*) thì đầu ra .DN sẽ được kích hoạt. Muốn reset lại bộ đếm ta dùng lệnh RES để xóa giá trị trong .ACC.





Tên	Kiểu dữ lệu	Mô tả
Counter	Count	Tên bộ đếm
.CU	Bool	On khi đầu vào on
.DN	Bool	On khi điều kiện .ACC >= .PRE là đúng.
.PRE	Dint	Giá trị đặt trước (Preset).
.ACC	Dint	Giá trị đếm (Accum).

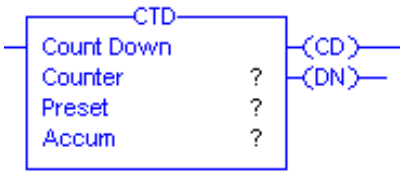
Ví dụ:

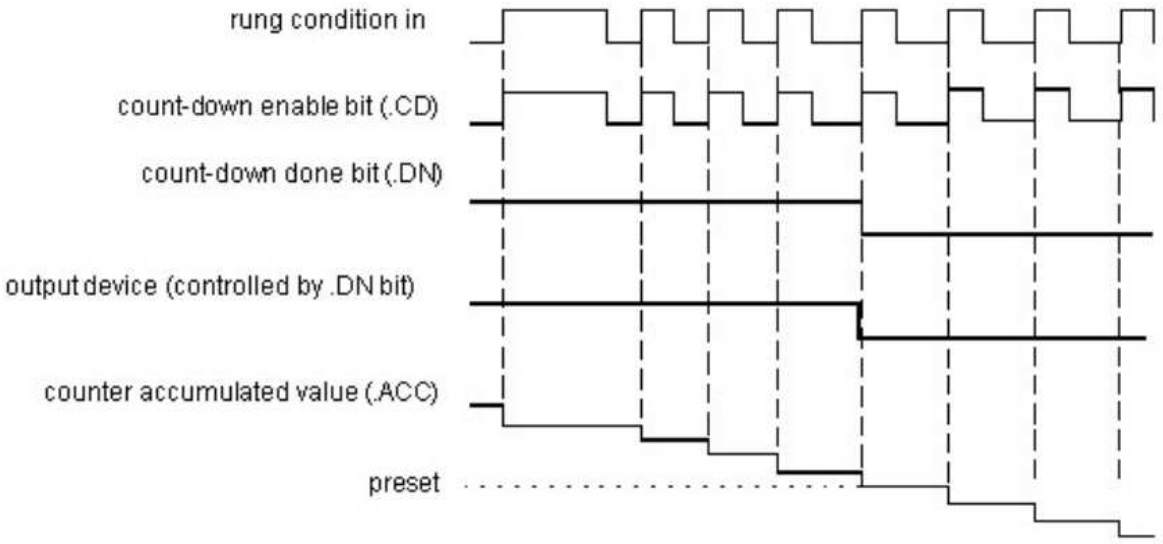


Mỗi khi *Start* chuyển từ off sang on thì giá trị trong *Accum* tăng thêm 1 đơn vị, khi *Accum* ≥ 10 thì tiếp điểm *C1.DN* sẽ tác động *Den_a* bật. Khi tiếp điểm *reset* on, lệnh RES sẽ reset lại bộ đếm *C1* (xóa giá trị trong *Accum* trở về 0) và *Den_a* tắt.

4.2.5. Counter Down CTD

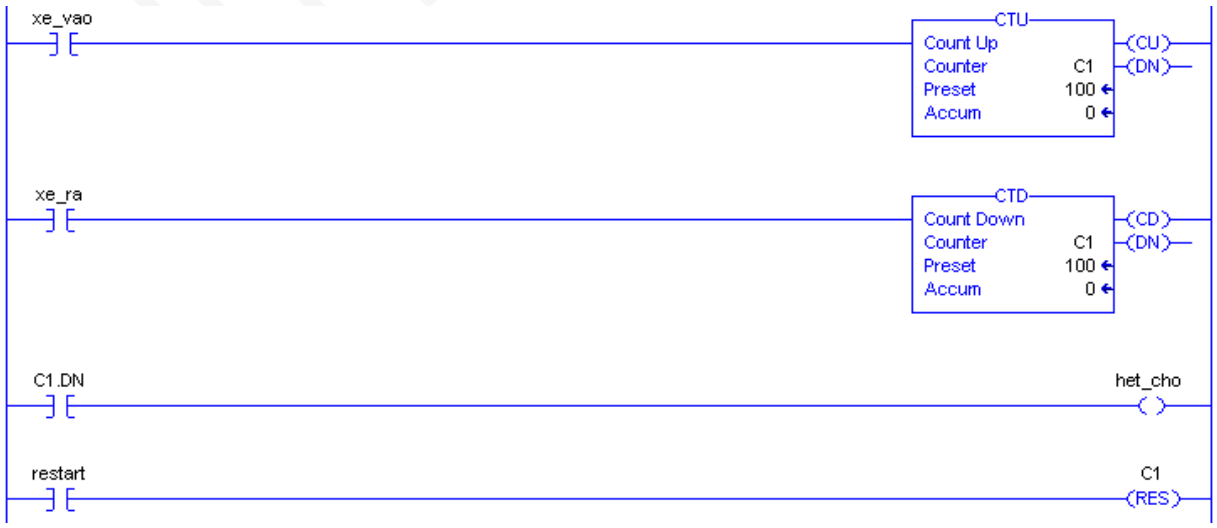
Dùng để đếm số lần chuyển từ sai sang đúng của điều kiện đầu vào. Khi đầu vào chuyển từ off sang on giá trị *Accum* của bộ đếm sẽ bị trừ đi 1. Nếu Accum ≥ Preset thì đầu ra .DN on. Muốn reset lại bộ đếm ta dùng lệnh RES để xóa giá trị trong .ACC. CTD thường được sử dụng với CTU có cùng cấu trúc.





Tên	Kiểu dữ lệu	Mô tả
Counter	Count	Tên bộ đếm
.CU	Bool	On khi đầu vào on
.DN	Bool	On khi điều kiện .ACC >= .PRE là đúng.
.PRE	Dint	Giá trị đặt trước.
.ACC	Dint	Giá trị đếm.

Ví dụ: Điều khiển bãi đỗ xe tự động



Mỗi lần *xe_vao* chuyển từ off sang on thì bộ đếm C1 lại tăng thêm 1 đơn vị. Mỗi lần *xe_ra* chuyển từ off sang on bộ đếm C1 lại trừ đi 1 đơn vị. Khi giá trị *Accum* >=100 thì tiếp

điểm C1.DN tác động đèn báo *het_cho* sáng. Khi tiếp điểm *restart* on lệnh RES sẽ reset lại bộ đếm C1 (Giá trị trong *Accum* về 0).

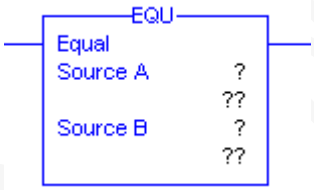
4.2.6. Reset RES

Dùng để xóa bộ định thời, bộ đếm hoặc cấu trúc điều khiển (Control structure). Khi lệnh RES được thực hiện nó sẽ xóa các bộ đếm (Counter), bộ định thời (Timer) hoặc cấu trúc điều khiển có cùng địa chỉ.

Sử dụng lệnh RES cho:	Giá trị bị xóa
Timer	Giá trị trong .ACC
Counter	Giá trị trong .ACC
Control	Giá trị trong .POS

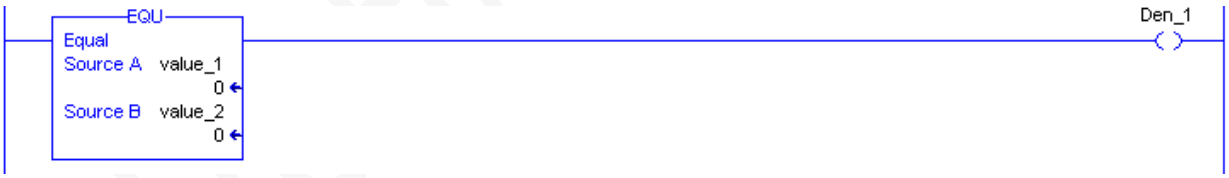
4.3. Một số lệnh so sánh

4.3.1. EQN (Equal to) EQU



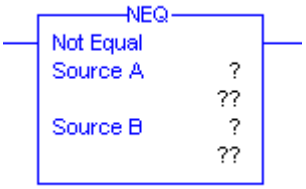
Lệnh so sánh bằng. Khi đầu vào on, EQU sẽ tiến hành so sánh Source A với Source B, nếu chúng bằng nhau thì đầu ra sẽ on, ngược lại đầu ra off.

Ví dụ:



Khi đầu vào on lệnh EQN sẽ so sánh *value_1* với *value_2,* nếu *value_1 = value_2* thì *Den_1* bật.

4.3.2. NEQ (Not Equal to) NEQ

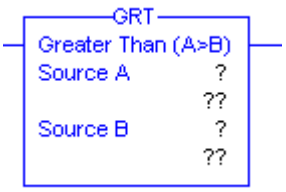


Lệnh so sánh không bằng. Khi đầu vào on, NEQ sẽ tiến hành so sánh Source A với Source B, nếu chúng không bằng nhau thì đầu ra sẽ on, ngược lại đầu ra off.

Ví dụ:

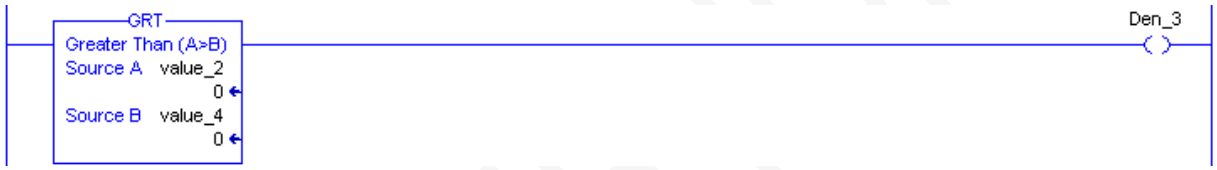
Khi đầu vào on, nếu *value_3* khác *value_4* thì *Den_2* sáng và ngược lại.

4.3.3. GRT(Greater Than) GRT



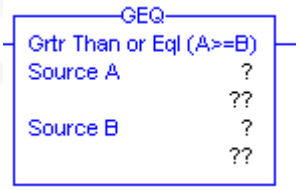
Lệnh so sánh lớn hơn. Khi đầu vào on, GRT sẽ tiến hành so sánh Source A với Source B, nếu Source A lớn hơn Source B thì đầu ra sẽ on, ngược lại đầu ra off.

Ví dụ:



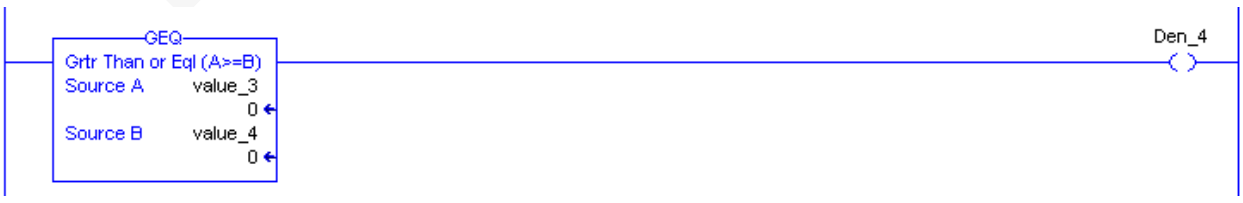
Khi đầu vào on, lệnh GRT sẽ so sánh *value_2* với *value_4* nếu *value_2* > *value_4* thì *Den_3* sáng, ngược lại *Den_3* không sáng.

4.3.4. GEQ (Greater Than or Equal) GEQ



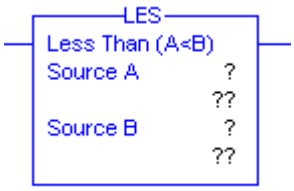
Lệnh so sánh lớn hơn hoặc bằng. Khi đầu vào on, GEQ sẽ tiến hành so sánh Source A với Source B, nếu SourceA lớn hơn hoặc bằng Source B thì đầu ra on, ngược lại đầu ra off.

Ví dụ:



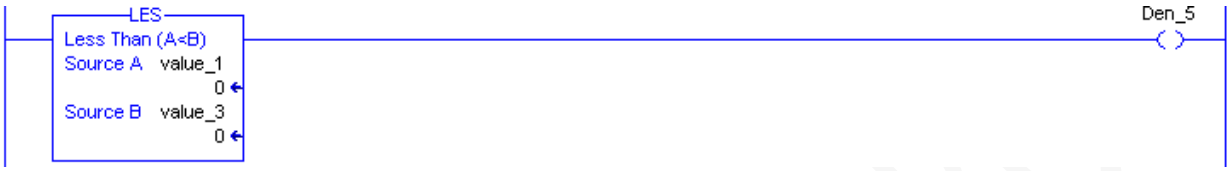
Khi đầu vào on, nếu *value_3* >= *value_4* là đúng thì *Den_4* sẽ sáng và ngược lại.

4.3.5. LES (Less Than) LES



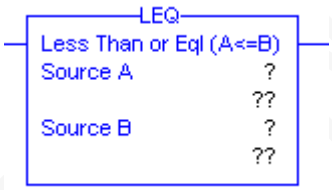
Lệnh so sánh nhỏ hơn. Khi đầu vào on, LES sẽ tiến hành so sánh Source A với Source B, nếu Source A nhỏ hơn Source B thì đầu ra sẽ on, ngược lại đầu ra off.

Ví dụ:



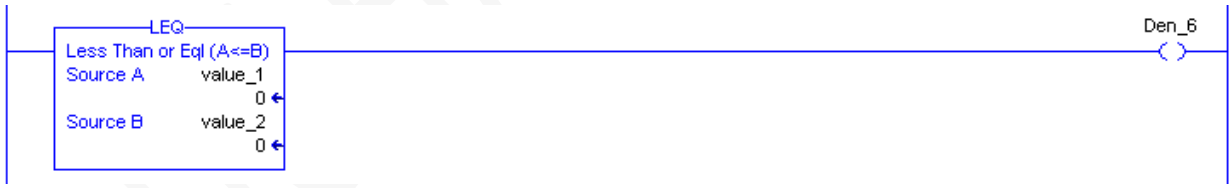
Nếu *value_1* < *value_3* là đúng thì *Den_5* sẽ sáng, ngược lại nếu *value_1* >= *value_3* thì *Den_5* sẽ không sáng.

4.3.6. LEQ (Less Than or Equal) LEQ



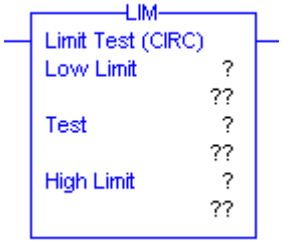
Lệnh so sánh nhỏ hơn hoặc bằng. Khi đầu vào on, LEQ sẽ tiến hành so sánh Source A với Source B, nếu Source A nhỏ hơn hoặc bằng Source B thì đầu ra sẽ on, ngược lại đầu ra off.

Ví dụ:



Nếu *value_1* <= *value_2* là đúng thì *Den_6* sẽ sáng, ngược lại nếu *value_1* > *value_2* thì *Den_6* sẽ không sáng.

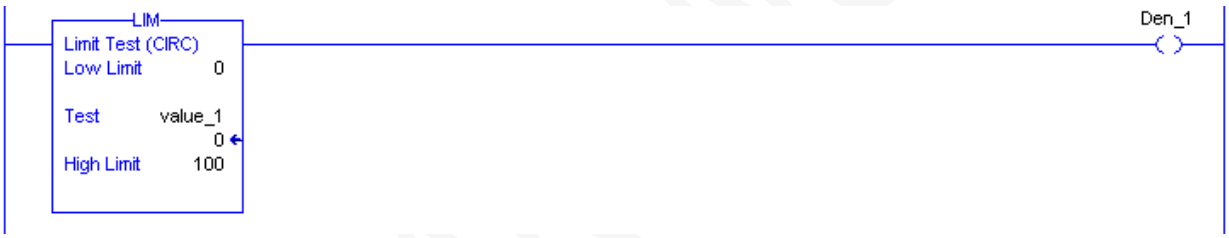
4.3.7. Limit LIM



Khi đầu vào ON, LIM sẽ tiến hành so sánh giá trị *Test* với *Low Limit* và *High Limit*. Khi (*Low Limit <= High Limit*), nếu (*Low Limit <= Test <= High Limit*) là đúng thì đầu ra LIM sẽ on, ngược lại đầu ra off. Khi (*Low Limit > High Limit*), nếu (*Test >= Low Limit*) hoặc (*Test <= High Limit*) là đúng thì đầu ra sẽ on, ngược lại đầu ra off.

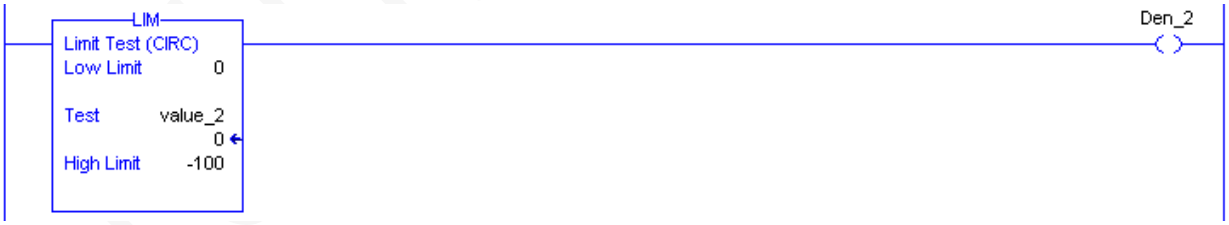
Toán hạng	Kiểu dữ liệu	Hình thức	Mô tả
Low Limit	Sint Int, Dint, Real	Trực tiếp Tag	Giá trị giới hạn dưới
Test	Sint Int, Dint, Real	Trực tiếp Tag	Giá trị kiểm tra
High Limit	Sint Int, Dint, Real	Trực tiếp Tag	Giá trị giới hạn trên
Tag Sint, Int sẽ chuyển thành giá trị Dint bởi sign-extension.			

Ví dụ:



Do *Low limit=0 < High Limit = 100* nên nếu *value_1* nằm trong đoạn [0,100] thì *Den_1* sẽ sáng, ngược lại nếu *value_1 < 0* hoặc *value_1 > 100* thì *Den_1* sẽ tắt.

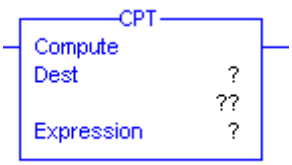
Ví dụ:



Do *Low Limit = 0 > High Limit = -100* nên nếu *value_2 >= 0* hoặc *value_2 <= -100* thì *Den_2* sẽ sáng, ngược lại nếu *-100 < value_2 < 0* thì *Den_2* sẽ tắt .

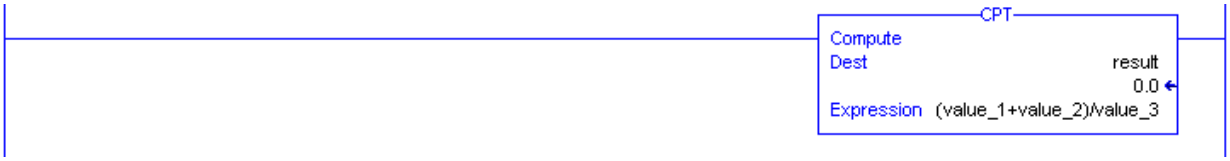
4.4. Một số lệnh toán học

4.4.1 Compute CPT



Khi đầu vào on, lệnh CPT sẽ tiến hành tính toán các biểu thức trong *Expression* kết quả được lưu trong *Dest*. Lệnh CPT hoạt động chậm và sử dụng nhiều bộ nhớ hơn so với các lệnh tính toán khác nhưng nó có ưu điểm là bạn có thể nhập vào những biểu thức phức tạp trong một lệnh (không giới hạn chiều dài của một biểu thức).

Ví dụ:

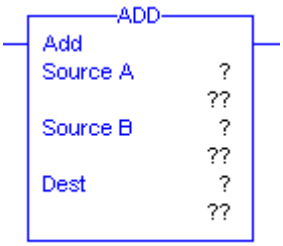


Khi đầu vào on, lệnh CPT sẽ thực hiện phep toán lấy *value_1* cộng với *value_2* sau đó chia cho *value_3*, kết quả được lưu trong tag *result*.

Thứ tự thực hiện các phép toán:

Thứ tự	Phép toán
1	()
2	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN.
3	**
4	- (dấu âm), NOT
5	*, / , MOD
6	- (trừ), +
7	AND
8	XOR
9	OR

4.4.2. Add 

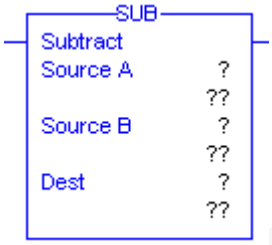


Lệnh cộng, khi đầu vào on lệnh ADD sẽ tiến hành cộng *Source A* với *Source B*, kết quả được lưu trong *Dest*.

Ví dụ:

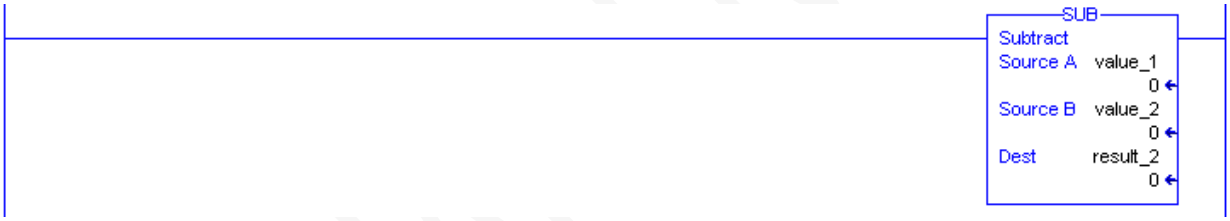
Khi đầu vào on, lệnh ADD sẽ lấy *value_3* cộng với *value_4* kết quả của phép cộng này được lưu trong tag *result.*

4.4.3. Subtract SUB



Lệnh trừ, khi đầu vào on lệnh SUB sẽ lấy *Source A* trừ đi *Source B* , kết quả được lưu trong *Dest*.

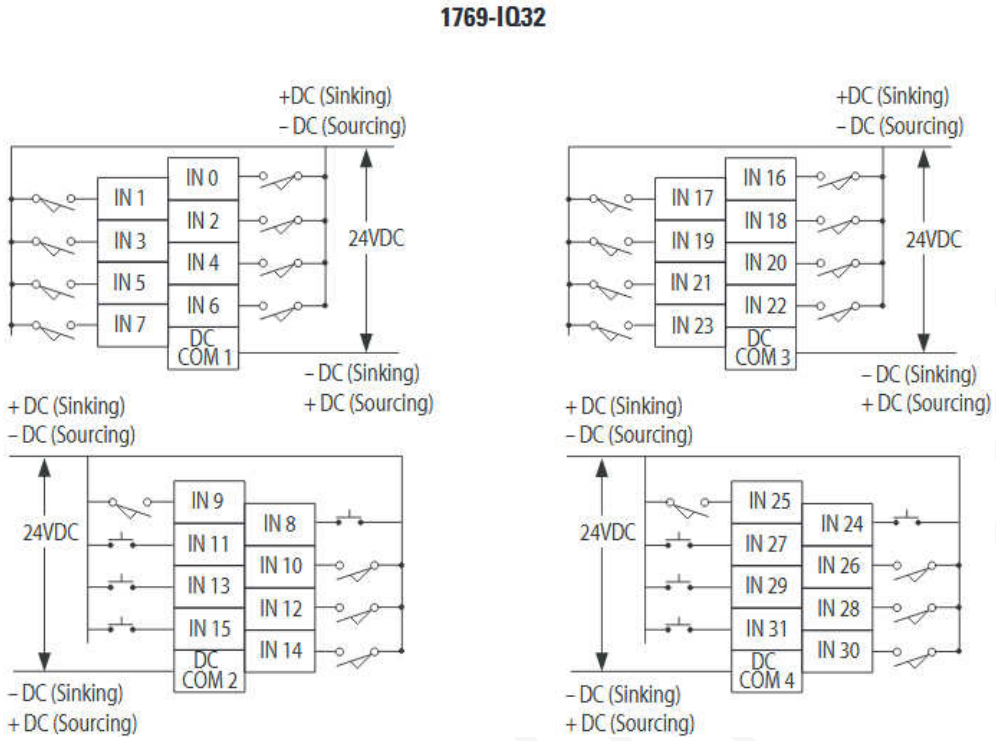
Ví dụ:



Khi đầu vào on, lệnh ADD lấy *value_1* trừ đi *value_2* kết quả được lưu trong tag *result_2*

5. Sơ đồ đấu dây

1769-IQ32: Compact 24VDC sink/source input module



1769-OB32: Compact solid-state 24VDC source output module