Bộ điều khiển Hệ thống Servo

Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Điều khiển định vị RD78G(H))

Khóa đào tạo này dành cho những người xây dựng hệ thống điều khiển chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R lần đầu tiên.

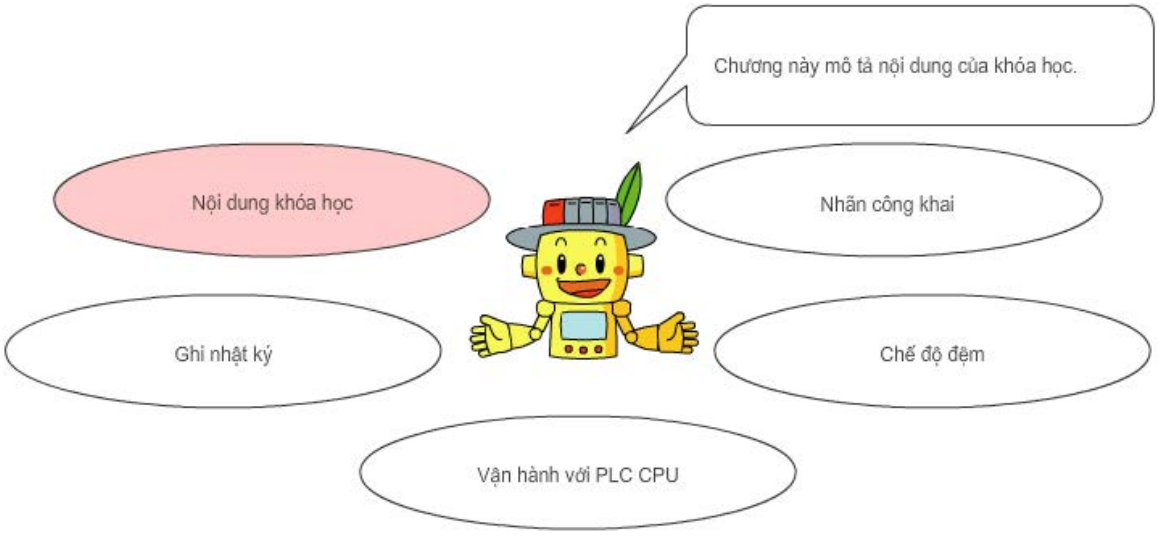
Nhấp vào nút Forward ở góc trên bên phải của màn hình để chuyển sang trang tiếp theo.

L(CTS)00767VIE

Giới thiệu

Mục đích của khóa học

Khóa học này được thiết kế nhằm mục đích phát triển kiến thức và hiểu biết về hoạt động điều khiển định vị của hệ thống điều khiển Chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R.

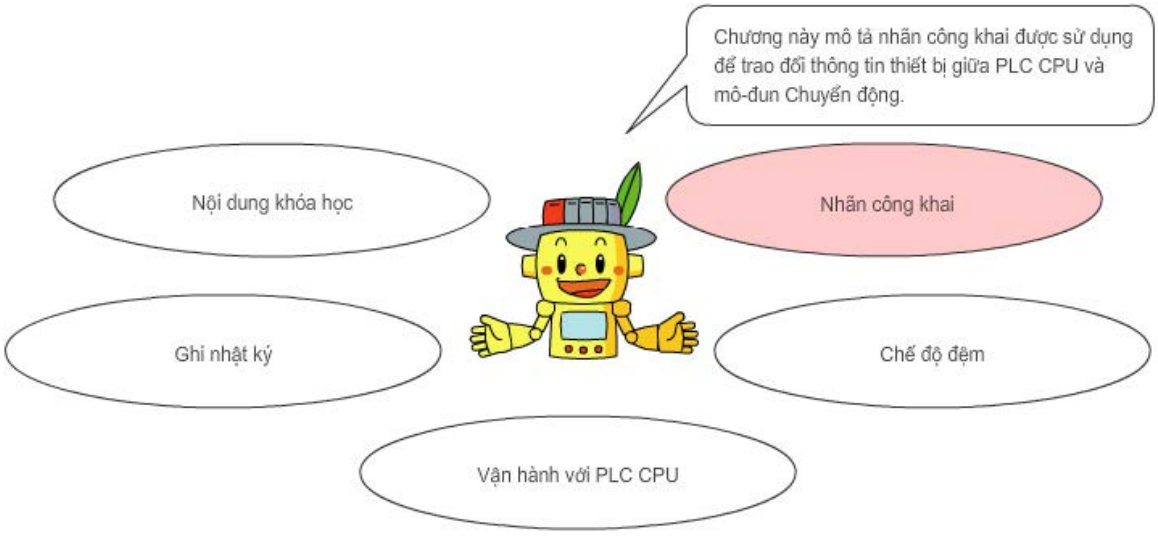


Khóa học này là phần tiếp nối của khóa học Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Khởi động RD78G(H)).
Đảm bảo rằng bạn đã hoàn thành khóa học Khởi động trước khi tham gia khóa học này.

Giới thiệu

Mục đích của khóa học

Khóa học này được thiết kế nhằm mục đích phát triển kiến thức và hiểu biết về hoạt động điều khiển định vị của hệ thống điều khiển Chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R.

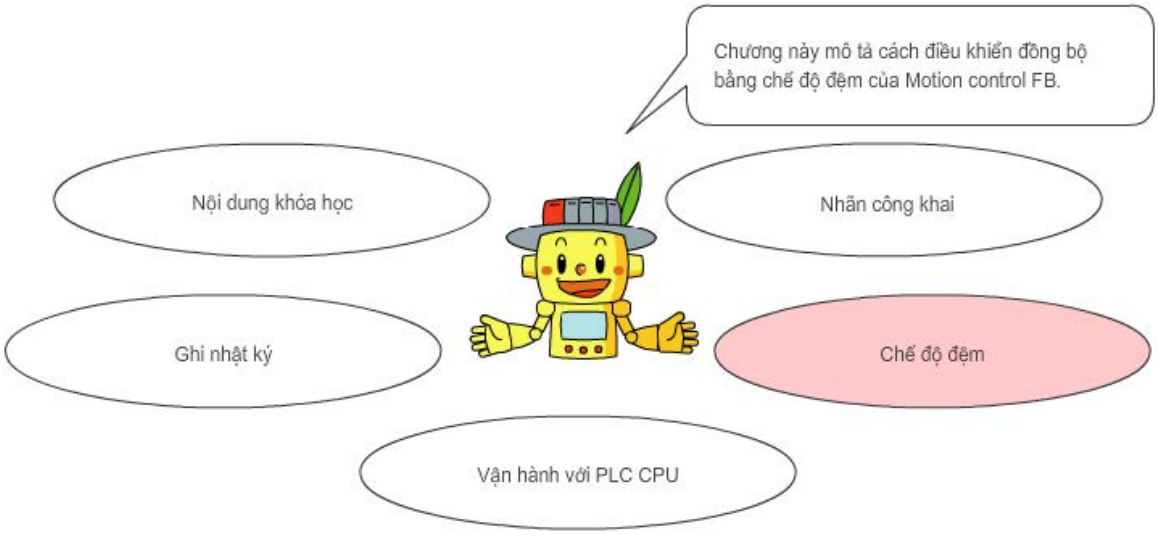


Khóa học này là phần tiếp nối của khóa học Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Khởi động RD78G(H)).
Đảm bảo rằng bạn đã hoàn thành khóa học Khởi động trước khi tham gia khóa học này.

Giới thiệu

Mục đích của khóa học

Khóa học này được thiết kế nhằm mục đích phát triển kiến thức và hiểu biết về hoạt động điều khiển định vị của hệ thống điều khiển Chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R.

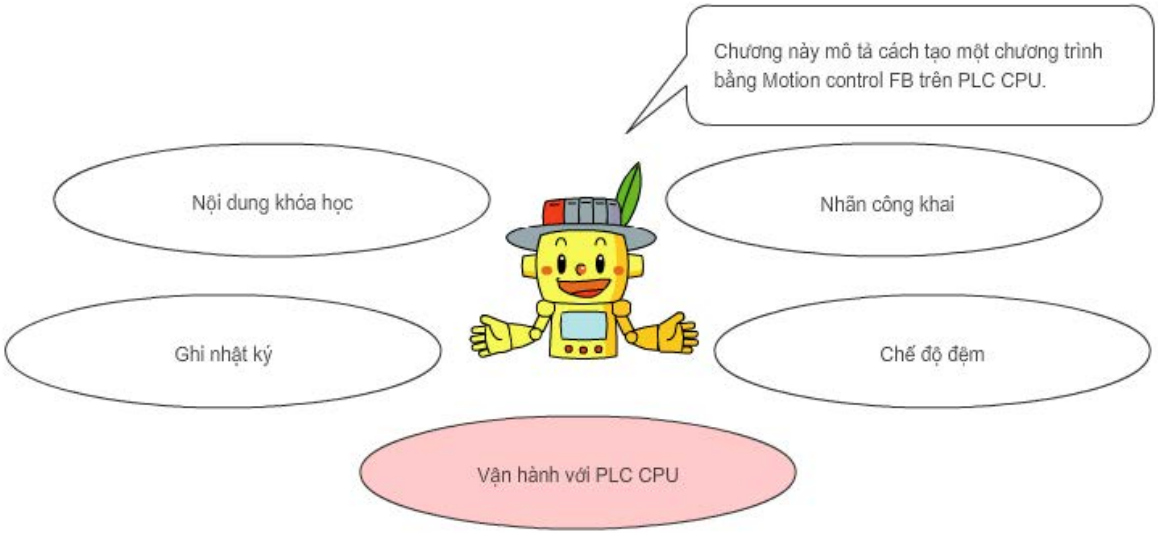


Khóa học này là phần tiếp nối của khóa học Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Khởi động RD78G(H)).
Đảm bảo rằng bạn đã hoàn thành khóa học Khởi động trước khi tham gia khóa học này.

Giới thiệu

Mục đích của khóa học

Khóa học này được thiết kế nhằm mục đích phát triển kiến thức và hiểu biết về hoạt động điều khiển định vị của hệ thống điều khiển Chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R.

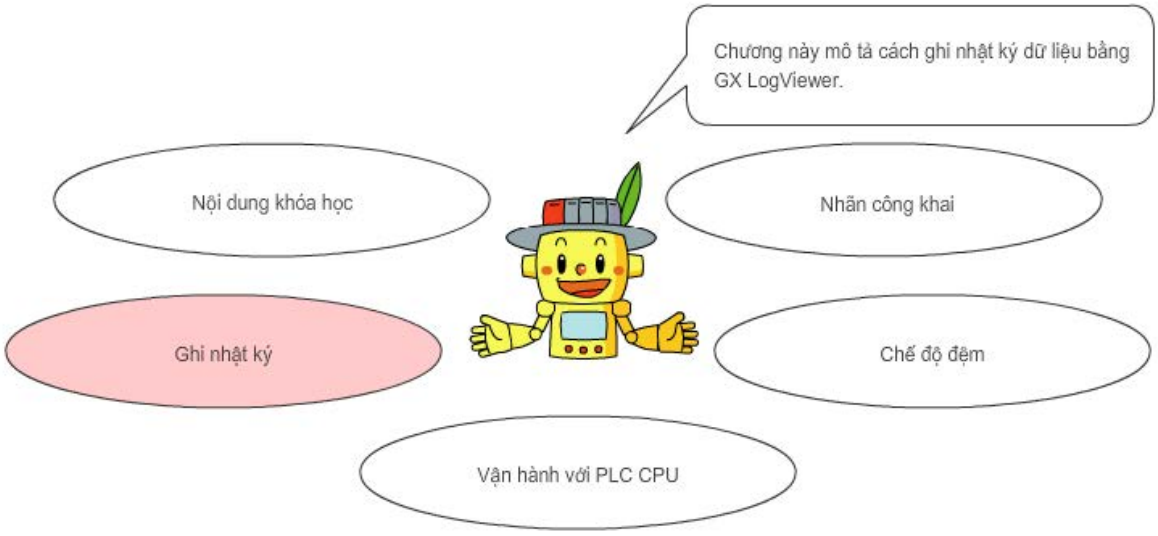


Khóa học này là phần tiếp nối của khóa học Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Khởi động RD78G(H)).
Đảm bảo rằng bạn đã hoàn thành khóa học Khởi động trước khi tham gia khóa học này.

Giới thiệu

Mục đích của khóa học

Khóa học này được thiết kế nhằm mục đích phát triển kiến thức và hiểu biết về hoạt động điều khiển định vị của hệ thống điều khiển Chuyển động bằng mô-đun Chuyển động dòng MELSEC iQ-R.



Khóa học này là phần tiếp nối của khóa học Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Khởi động RD78G(H)).
Đảm bảo rằng bạn đã hoàn thành khóa học Khởi động trước khi tham gia khóa học này.

Giới thiệu

Cấu trúc của khóa học

Dưới đây là mục lục của khóa học này.
Chúng tôi khuyên bạn nên bắt đầu từ Chương 1.

Chương 1 - Nội dung khóa học

Chương này mô tả nội dung của khóa học.

Chương 2 - Nhãn công khai

Chương này mô tả nhãn công khai được sử dụng để trao đổi thông tin thiết bị giữa PLC CPU và mô-đun Chuyển động.

Chương 3 - Chế độ đệm

Chương này mô tả cách điều khiển đồng bộ bằng chế độ đệm của Motion control FB.

Chương 4 - Vận hành với PLC CPU

Chương này mô tả cách tạo một chương trình bằng Motion control FB trên PLC CPU.

Chương 5 - Ghi nhật ký

Chương này mô tả cách ghi nhật ký dữ liệu bằng GX LogViewer.

Bài kiểm tra cuối khóa

Tổng cộng 4 phần (7 câu hỏi)

Đến trang tiếp theo	>	Đến trang tiếp theo.
Trở lại trang trước	<	Trở lại trang trước.
Di chuyển đến trang mong muốn	TOC	"Mục lục" sẽ được hiển thị, cho phép bạn điều hướng đến trang mong muốn.
Thoát khỏi bài học	X	Trở lại trang trước. Cửa sổ chẳng hạn như màn hình "Nội dung" và bài học sẽ được đóng lại.

Thận trọng khi sử dụng

■Lưu ý về an toàn

Khi sử dụng sản phẩm thực tế cho mục đích học tập, vui lòng đọc kỹ phần "Biện pháp phòng ngừa an toàn" được mô tả trong hướng dẫn sử dụng của sản phẩm sẽ sử dụng và chú ý đến sự an toàn và sử dụng đúng cách.

■Lưu ý trong khóa học này

Hình ảnh màn hình được hiển thị trong khóa học có thể khác với phần mềm thực tế tùy vào phiên bản. Khóa học sử dụng phiên bản phần mềm sau đây.
Phiên bản mới nhất của mỗi phần mềm, hãy truy cập website Mitsubishi Electric FA để kiểm tra.

MELSOFT GX Works3	Ver.1.066U	Motion Control Setting function Ver.1.012N
GX LogViewer	Ver.1.106K	
MELSOFT MR Configurator2	Ver.1.110Q trở lên	

PLC CPU phải có firmware phiên bản 44 trở lên (46 trở lên cho RD78GH).
Mô-đun chuyển động phải có firmware phiên bản 10 trở lên.
Vui lòng tham khảo Trang web MITSUBISHI ELECTRIC FA hoặc hướng dẫn cấu hình mô-đun để biết cách cập nhật phiên bản firmware.

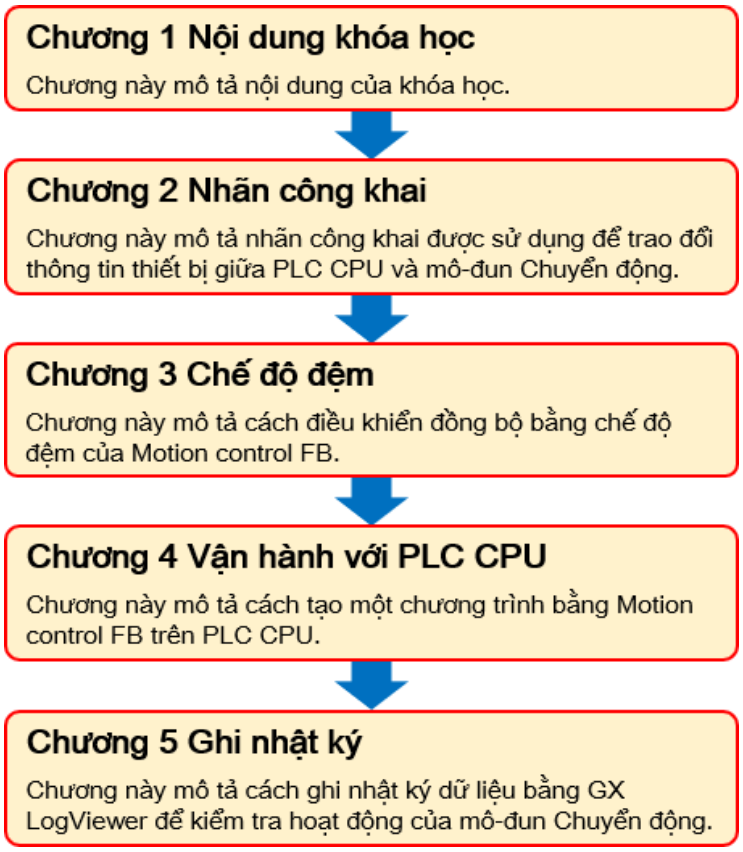
Biểu tượng  là sách hướng dẫn tham khảo.
Những nội dung của các sách hướng dẫn được mô tả trong khóa học này là của những phiên bản sau đây.
Nếu khác phiên bản thì vị trí phần mô tả và các nội dung có thể hơi khác biệt.

Tên sách hướng dẫn	Mã sách hướng dẫn	Phiên bản
MELSEC iQ-R Motion Module User's Manual (Startup)	IB-0300406	E
MELSEC iQ-R Motion Module User's Manual (Application)	IB-0300411	E
MELSEC iQ-R Motion Module User's Manual (Network)	IB-0300426	E
MELSEC iQ-R Programming Manual (Motion Module Instructions, Standard Functions/Function Blocks)	IB-0300431	E
MELSEC iQ-R Programming Manual (Motion Control Function Blocks)	IB-0300533	C
MELSEC iQ-R Structured Text (ST) Programming Guide Book	SH-081483	F
MELSEC iQ-R Programming Manual (CPU Module Instructions, Standard Functions/Function Blocks)	SH-081266	Z
MELSEC iQ-R CPU Module User's Manual (Application)	SH-081264	AK

Chương 1 Nội dung khóa học

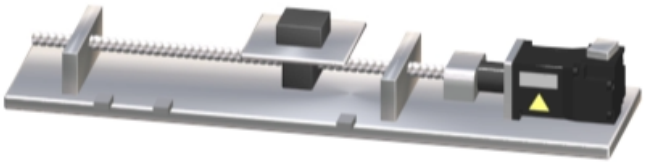
1.1 Tổng quan về khóa học

Phần sau đây cho thấy nội dung tổng quan về khóa học.



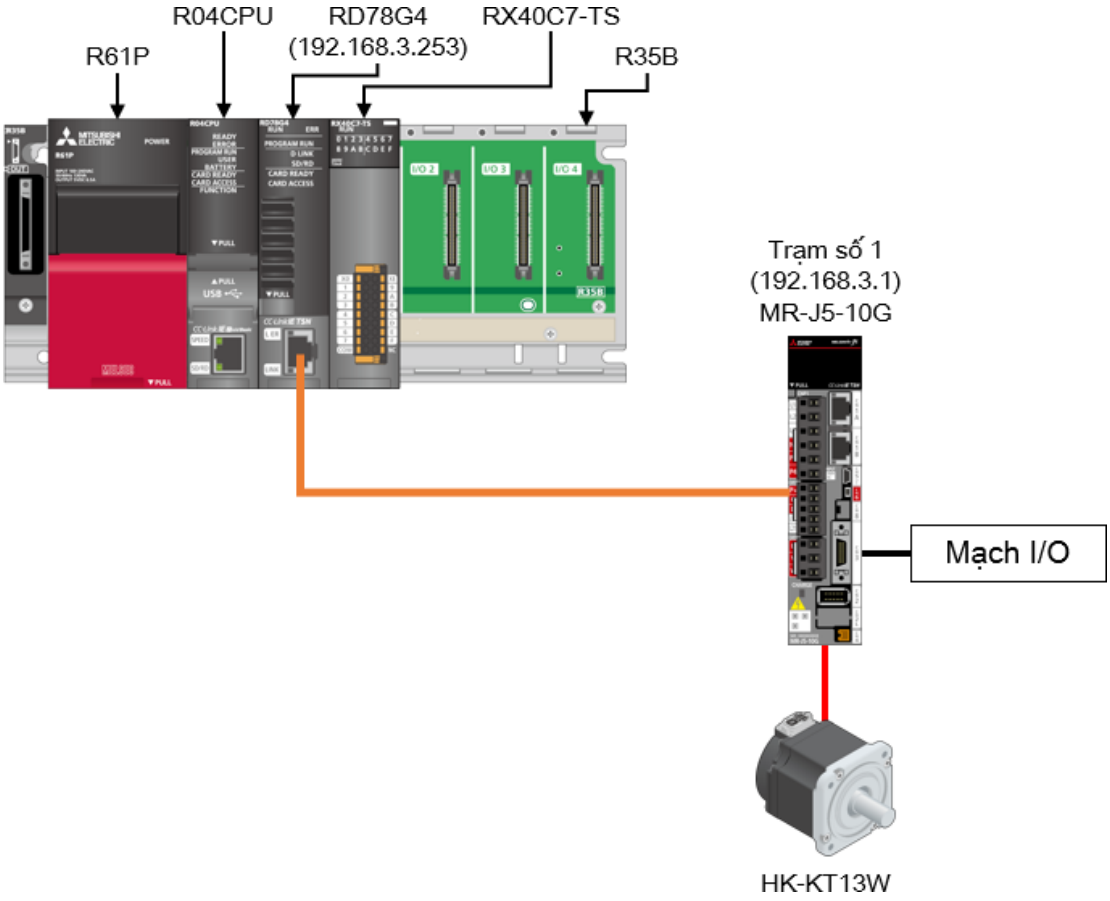
1.2 Cấu hình máy mục tiêu

Khóa học này sử dụng cơ chế vitme bi một trục tương tự như cơ chế được sử dụng trong khóa học Khởi động.



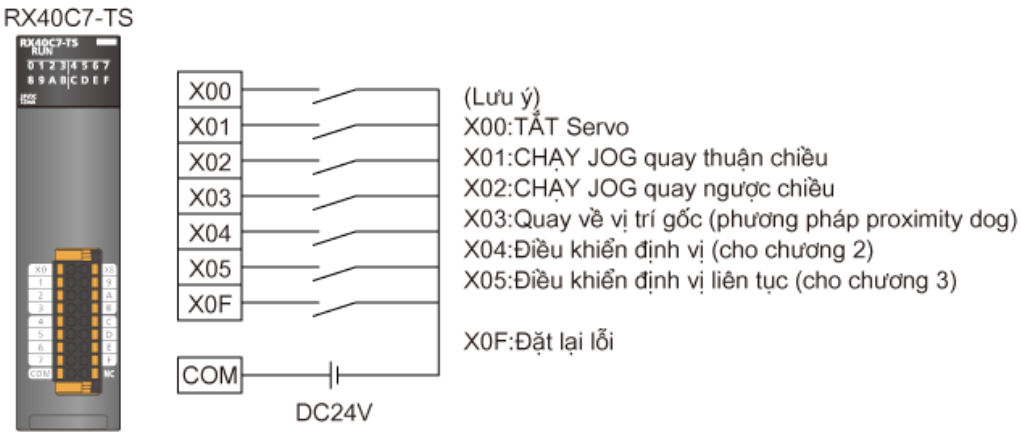
1.3 Cấu hình hệ thống mục tiêu

Cấu hình của hệ thống mục tiêu như sau.
Tháo mô-đun đầu vào từ xa khỏi hệ thống được sử dụng trong khóa học Khởi động và lắp mô-đun đầu vào RX40C7-TS vào khe 1 của khối cơ sở của bộ điều khiển khả trình.
Số trạm của bộ khuếch đại servo MR-J5-10G đã được thay đổi thành 1 và địa chỉ IP đã được thay đổi thành 192.168.3.1.



1.4 Đi dây mạch ngoài

Phương thức đi dây nguồn điện cho bộ điều khiển khả trình và bộ khuếch đại servo cũng như phương thức kết nối của mô-tơ servo giống với phương thức mô tả trong khóa học Khởi động.
Phần sau đây cho thấy cách đi dây mạch ngoài của mô-đun đầu vào.



(Lưu ý) Vì Số I/O của RX40C7-TS là 0020H, X20 đến X25 và X2F được sử dụng trong chương trình.

Chương 2 Nhãn công khai

Khi mô-đun đầu vào của bộ điều khiển khả trình điều khiển mô-đun Chuyển động như trong hệ thống được sử dụng trong khóa học này và được mô tả ở chương 1, PLC CPU và mô-đun Chuyển động phải trao đổi thông tin thiết bị.

Có hai phương pháp sau đây.

1. Sử dụng nhãn công khai.
2. Sử dụng bộ nhớ đệm của mô-đun Chuyển động.

Chương này mô tả cách trao đổi dữ liệu bằng nhãn công khai.

Tải xuống chương trình mẫu sẽ được sử dụng trong chương này và chương 3 bằng cách nhấp vào liên kết bên dưới.

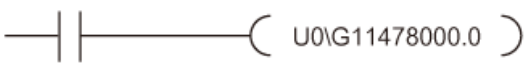
RD78GBasic2_sample1.zip (1.34MB)

[Điểm]
Khi sử dụng bộ nhớ đệm, hãy sao chép dữ liệu sẽ được trao đổi đến khu vực người dùng (U□\G11478000 to G11997999).

(Ví dụ về chương trình)

<PLC CPU>

Bắt đầu định vị

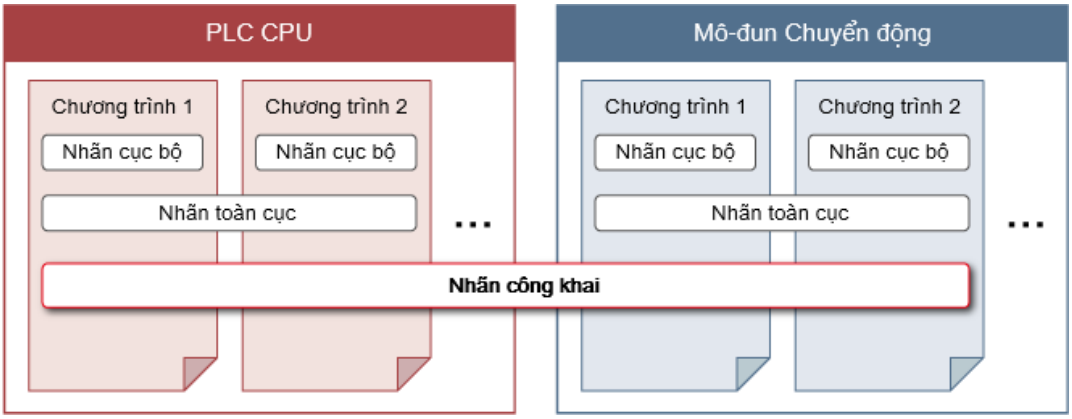


<Mô-đun Chuyển động>

```
MC_MoveAboslute_1(
      Execute:= G11478000.0 ,
        :
        :
);
```

2.1 Nhãn công khai là gì?

Nhãn công khai là nhãn dùng chung có thể sử dụng trong cả mô-đun Chuyển động và PLC CPU.
Phần sau đây cho thấy khu vực áp dụng của nhãn cục bộ, nhãn toàn cục và nhãn công khai.



2.2 Cài đặt nhãn công khai

(1) Cách đăng ký nhãn công khai

Đăng ký nhãn công khai từ nhãn toàn cục của mô-đun Chuyển động.

Đảm bảo rằng cột "Public Label" hiển thị trong trình biên tập nhãn toàn cục trên màn hình Motion Module Setting Function.

Đặt "Enabled" cho nhãn sẽ được đăng ký thành nhãn công khai.

Thao tác này sẽ kích hoạt cột "Motion Control Attribute".

Chọn xem mỗi nhãn có được đọc hoặc ghi từ/đến PLC CPU không.

	Label Name	Data Type	Class	Initial	Constant	Japanese	English(Display Target)	Chinese	Remark	Public Label	Motion Control Attribute
1	G_bSVONCMD	Bit	VAR_GLOBAL				Servo ON			Enabled	WRITE (=> Motion)
2	G_leJogVelocity	FLOAT [Double Precision]	VAR_GLOBAL				JOG Velocity			Enabled	WRITE (=> Motion)
3	G_bJogFwd	Bit	VAR_GLOBAL				JOG Forward			Enabled	WRITE (=> Motion)
4	G_bJogBwd	Bit	VAR_GLOBAL				JOG Backward			Enabled	WRITE (=> Motion)
5	G_bJogBusy	Bit	VAR_GLOBAL				JOG Busy			Enabled	READ (Motion =>)
6	G_lePosition0	FLOAT [Double Precision]	VAR_GLOBAL				Position0 Address			Disabled	-
7	G_bHomingCMD	Bit	VAR_GLOBAL				Homing Command			Enabled	WRITE (=> Motion)
8	G_bHomingDone	Bit	VAR_GLOBAL				Homing Done			Enabled	READ (Motion =>)
9	G_bHomingReq	Bit	VAR_GLOBAL				Homing Request			Enabled	READ (Motion =>)
10	G_bPosCMD	Bit	VAR_GLOBAL				Positioning Command			Enabled	WRITE (=> Motion)
11	G_bPosDone	Bit	VAR_GLOBAL				Positioning Done			Enabled	READ (Motion =>)
12	G_bPosReq	Bit	VAR_GLOBAL				Positioning Start Request			Enabled	READ (Motion =>)
13	G_bErrorReset	Bit	VAR_GLOBAL				Error Reset			Enabled	WRITE (=> Motion)
14	G_bContPosCMD	Bit	VAR_GLOBAL				Continuous Positioning Command			Enabled	WRITE (=> Motion)
15	G_bContPosReq	Bit	VAR_GLOBAL				Continuous Positioning Start Request			Enabled	WRITE (=> Motion)
16	G_bContPosDone	Bit	VAR_GLOBAL				Continuous Positioning Done			Enabled	READ (Motion =>)
17											

[Điểm]
Nếu cột nhãn công khai không hiển thị, hãy cuộn bảng sang bên phải.

2.2 Cài đặt nhãn công khai

(2) Loại dữ liệu có thể đăng ký là nhãn công khai
Bảng sau đây cho thấy loại dữ liệu có thể đăng ký thành nhãn công khai.

<table>
<tr><th>Loại biến</th><th>Loại</th><th>Lựa chọn mảng</th><th>Cài đặt nhãn công khai</th><th>Chú ý</th></tr>
<tr><td rowspan="6">Nhãn toàn cục</td><td rowspan="2">Loại đơn giản</td><td>Không</td><td>○</td><td rowspan="11">Không thể cài đặt cho nhãn và lớp sau.
■Nhãn
• Nhãn loại chuỗi
• Nhãn loại bộ đếm thời gian
• Nhãn loại bộ đếm
• Nhãn loại bộ đếm dài
• Nhãn bộ đếm thời gian có lưu trữ
• Nhãn loại bộ đếm thời gian có lưu trữ dài
• Nhãn loại bộ đếm thời gian dài
■Lớp
• VAR_GLOBAL_CONSTANT lớp</td></tr>
<tr><td>Có</td><td>△(Lưu ý 1,2)</td></tr>
<tr><td rowspan="2">Loại dữ liệu có cấu trúc</td><td>Không</td><td>△(Lưu ý 3)</td></tr>
<tr><td>Có</td><td>△(Lưu ý 1,2,4,5)</td></tr>
<tr><td rowspan="2">FB
(Bao gồm Motion control FB)</td><td>Không</td><td>×</td></tr>
<tr><td>Có</td><td>×</td></tr>
<tr><td>Chương trình</td><td>-</td><td>-</td><td>×</td></tr>
<tr><td>Nhãn cục bộ khối chương trình</td><td>-</td><td>-</td><td>×</td></tr>
<tr><td>Loại dữ liệu có cấu trúc</td><td>-</td><td>-</td><td>△(Lưu ý 3,5)</td></tr>
<tr><td>Dữ liệu có cấu trúc Motion control FB</td><td>-</td><td>-</td><td>△(Lưu ý 6,7)</td></tr>
</table>

(Lưu ý)
1. Không thể cấu hình cài đặt nhãn công khai cho từng yếu tố của mảng.
2. Không thể đặt "Enabled"cho nhãn công khai khi sử dụng mảng loại bit. (Trong dữ liệu có cấu trúc, chỉ có thành phần tương ứng là không thể đặt thành "Enabled".)
3. Khi loại chuỗi được sử dụng làm thành phần của loại dữ liệu có cấu trúc, thành phần đó sẽ không thể đặt thành "Enabled".
4. Dữ liệu có cấu trúc với tối đa bốn lớp có thể được công khai.
5. Khi một mảng của dữ liệu có cấu trúc được sử dụng làm thành phần của loại dữ liệu có cấu trúc, thành phần đó sẽ không thể đặt thành "Enabled".
6. Thành phần đó có thể được sử dụng trong chương trình PLCopen Motion control FB bởi mô-đunCPU.
7. Khi loại chuỗi được sử dụng trong dữ liệu có cấu trúc Motion control FB thì không thể đặt chính loại dữ liệu có cấu trúc Motion control FB đó.

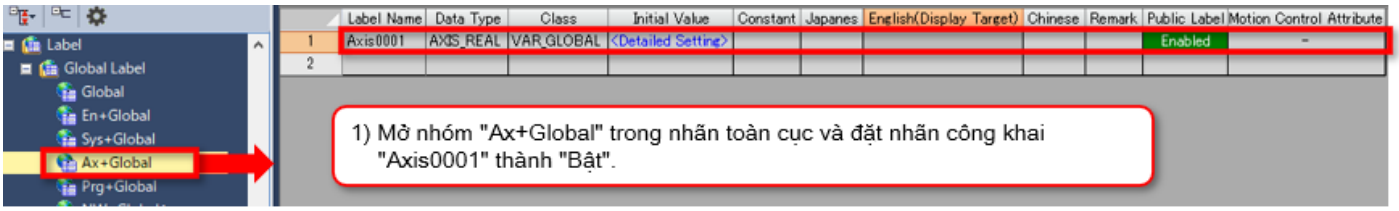
2.2 Cài đặt nhãn công khai

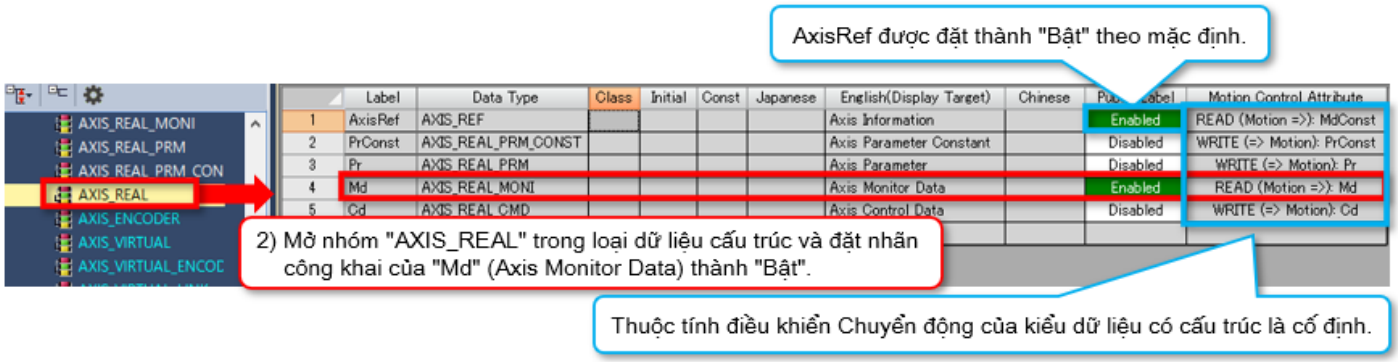
(3) Cách đăng ký dữ liệu có cấu trúc thành nhãn công khai

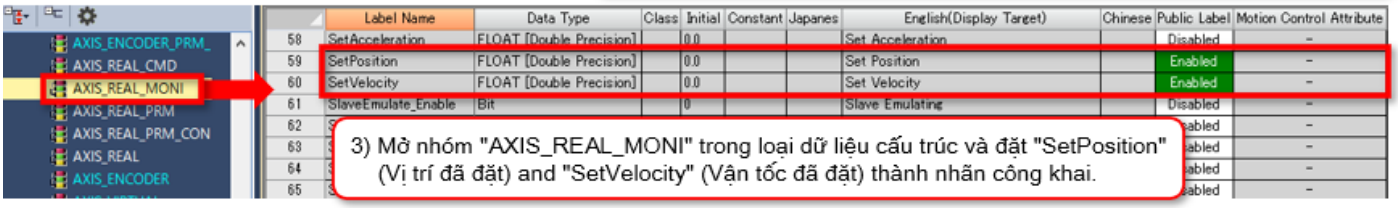
Để đặt thành phần của loại dữ liệu có cấu trúc đã chuẩn bị trong hệ thống, chẳng hạn như dữ liệu giám sát trục, thành nhãn công khai, hãy đăng ký nhãn công khai theo lớp dữ liệu có cấu trúc như hiển thị bên dưới.

Khóa học này mô tả cách đăng ký Vị trí đã đặt (SetPosition) và Vận tốc đã đặt (SetVelocity), dữ liệu giám sát (Md) của trục truyền động thực tế (Axis_Real), thành nhãn công khai.

[Cách đặt AxisName.Md.SetPosition (Điều khiển vị trí hiện tại) và AxisName.Md.SetVelocity (Điều khiển vận tốc hiện tại) thành nhãn công khai]



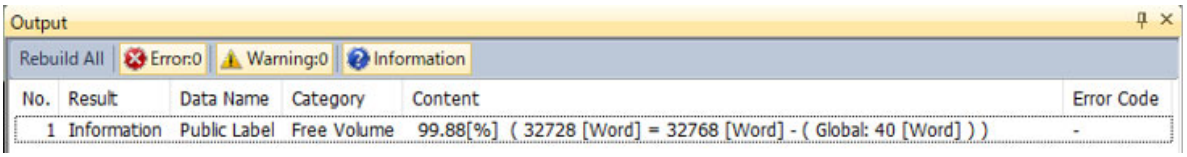




(4) Ánh xạ nhãn công khai

Chọn [Convert] → [Rebuild All] trong menu.

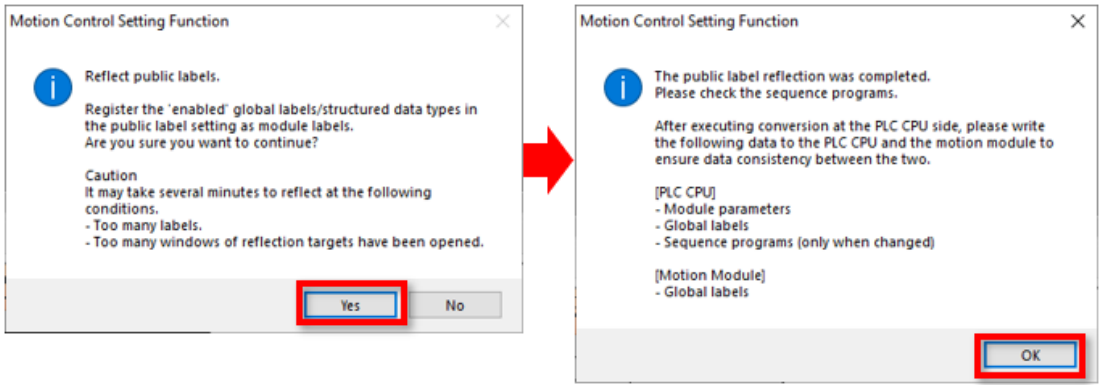
Dung lượng trống của nhãn công khai được hiển thị dưới dạng Thông tin trong cửa sổ đầu ra.



Khi quá trình xây dựng lại hoàn tất một cách thành công, chọn [Convert] → [Reflect Public Labels] trong menu.

Nhấp vào [Yes] trong cửa sổ bật lên sau đây.

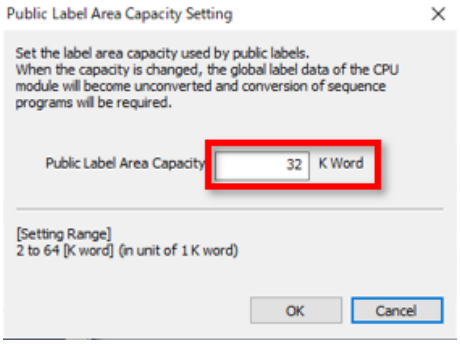
Khi xuất hiện thông báo cho biết nhãn công khai được ánh xạ thành công, hãy nhấp vào nút [OK].



(Lưu ý) Dung lượng bộ nhớ có thể được sử dụng để đăng ký nhãn công khai theo mặc định là 32K từ.

Dung lượng có thể tăng lên tối đa 64K từ.

Để thay đổi dung lượng, hãy đặt kích thước bộ nhớ từ [Convert] → [Public Label Capacity Setting] trong menu.



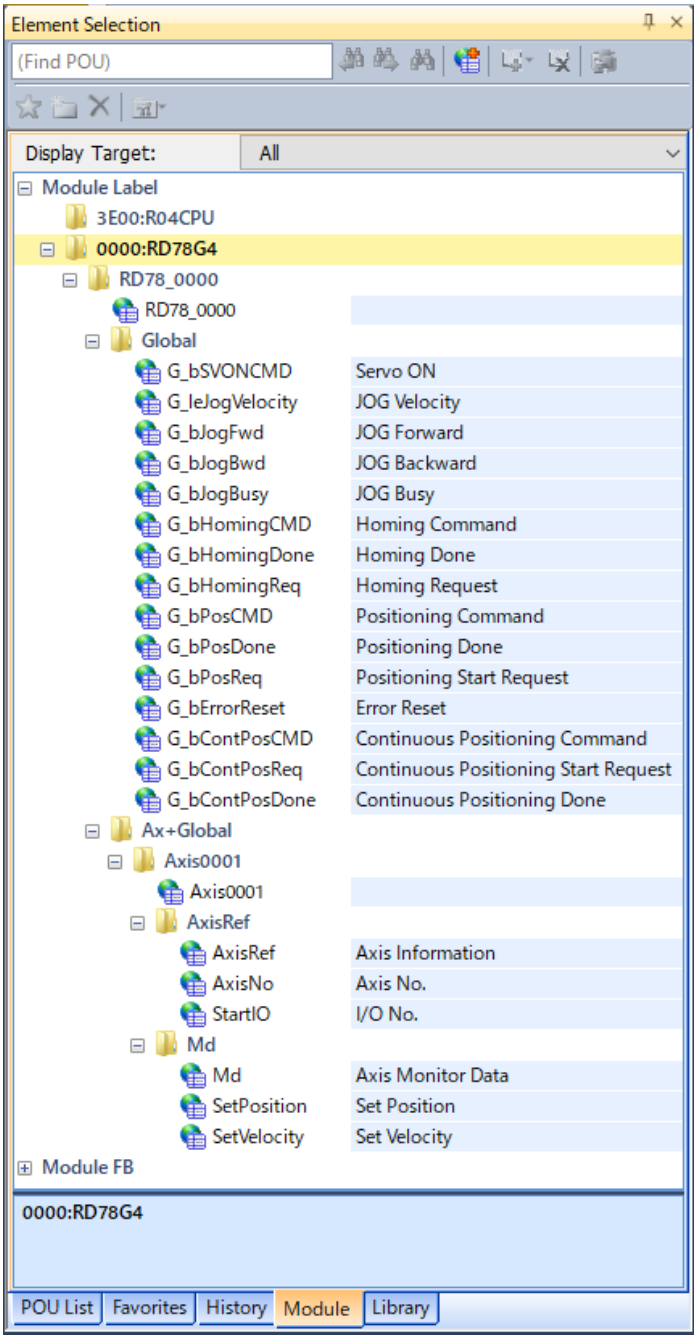
(5) Kiểm tra nhãn từ phía PLC CPU

Nhãn công khai được ánh xạ sẽ được đăng ký vào nhãn mô-đun bên phía PLC CPU.

Chọn nhãn mô-đun từ cửa sổ Element Selection của GX Works3 và kiểm tra xem nhãn công khai đã được đăng ký theo [0000:RD78G4] trong [Module Label] chưa.

Sau khi thay đổi cài đặt nhãn công khai, hãy luôn thực hiện lại "Reflect Public Labels".

Khi sử dụng nhãn công khai trong PLC CPU, hãy xây dựng lại tất cả chương trình.



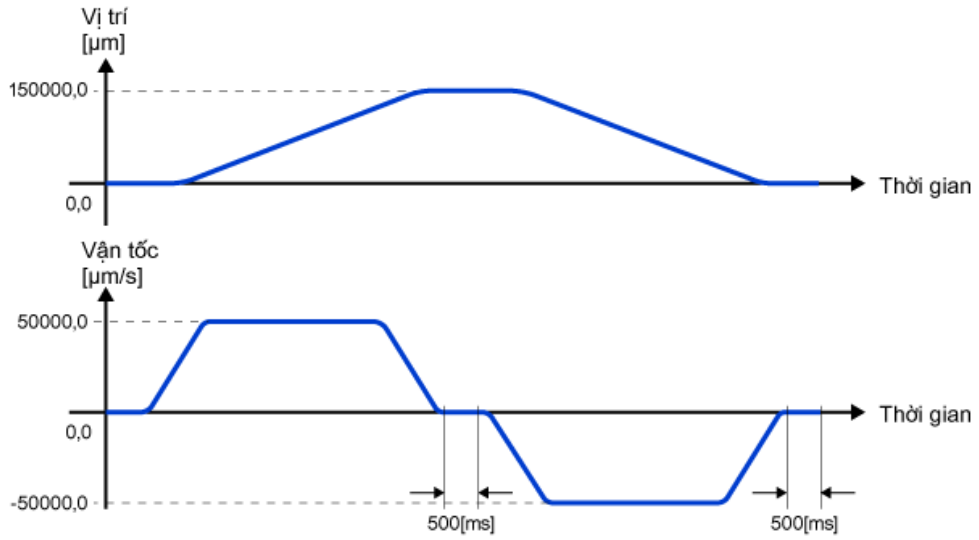
2.3 Ví dụ về chương trình

(1) Hoạt động của chương trình mẫu

Tín hiệu đầu vào của chương trình mẫu dùng trong chương này được chỉ định như sau.

Đầu vào	Hoạt động
X20	Servo tắt (Lưu ý)
X21	Hoạt động CHẠY JOG quay thuận chiều
X22	Hoạt động CHẠY JOG quay ngược chiều
X23	Quay về vị trí gốc
X24	Điều khiển định vị
X25	Điều khiển định vị liên tục (Chương 3)

Phần sau đây cho thấy mô hình hoạt động của X24: Điều khiển định vị.

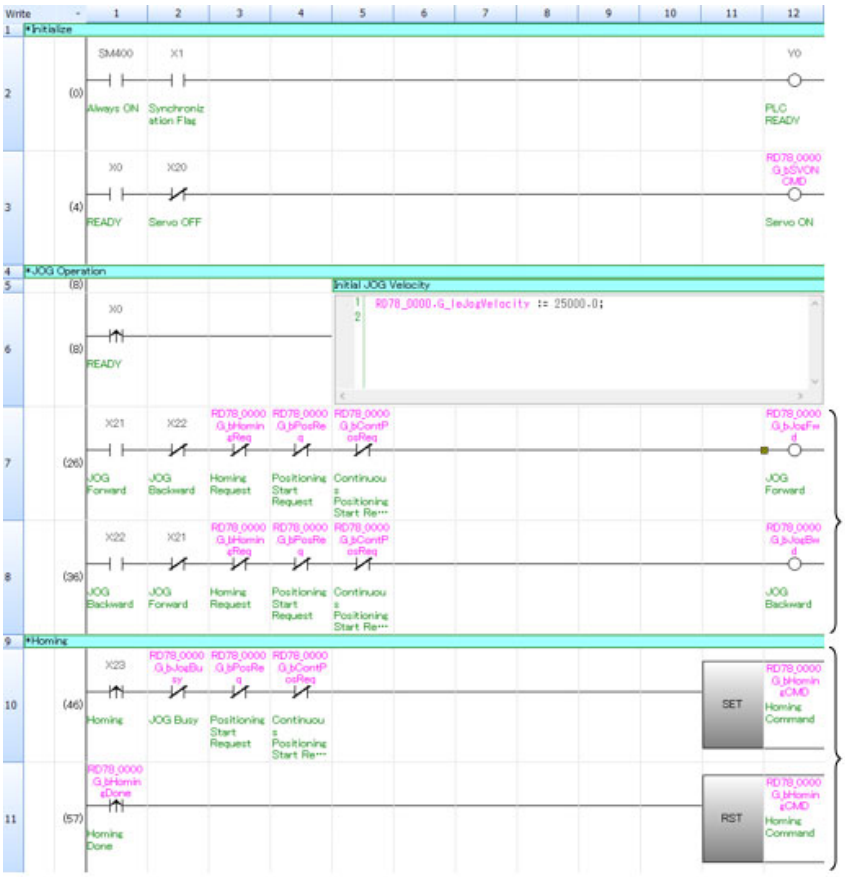


(Lưu ý) Chương trình mẫu này tự động BẬT servo khi PLC CPU được đặt thành RUN.
Khi nguồn được bật với tín hiệu khởi động BẬT, mô-tơ servo có thể được kích hoạt.

2.3 Ví dụ về chương trình

(2) Chương trình PLC CPU

1) MAIN (chương trình quét, ladder)



Trước tiên, hãy bật Y0.

Khi X0 bật, hoạt động BẬT servo sẽ được thực thi.
Bật X20 để thực thi TẮT servo.

Đặt giá trị ban đầu của vận tốc CHẠY JOG.
Chương trình này sử dụng ST nội tuyến.
Vì thuộc tính điều khiển chuyển động của nhãn toàn cục "G_JogVelocity" lưu trữ vận tốc CHẠY JOG được đặt thành "GHI (→Chuyển động)", giá trị số phải được đặt trong PLC CPU.

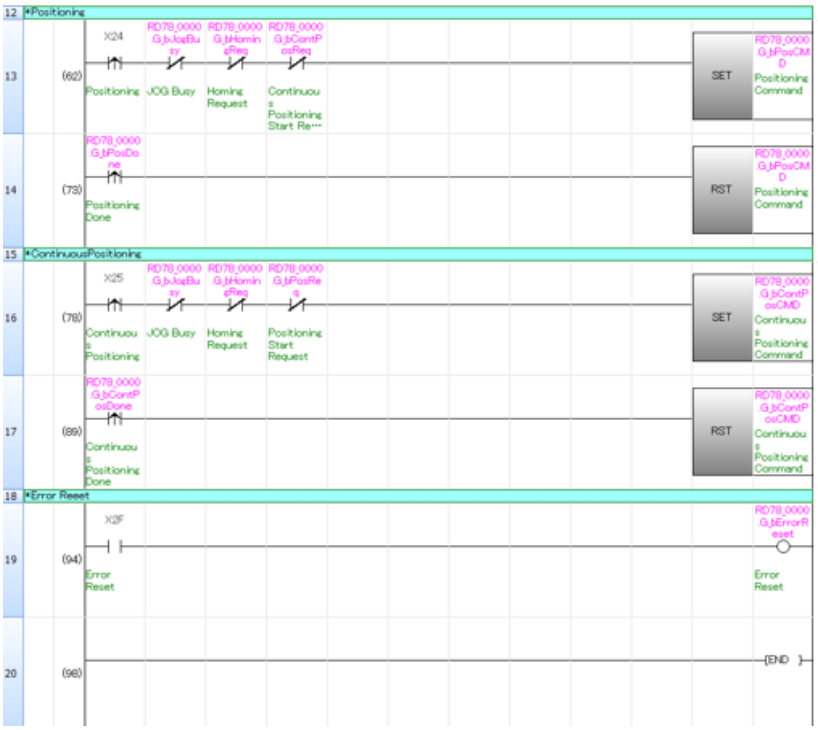
Bật tín hiệu bắt đầu của hoạt động CHẠY JOG.
Tín hiệu ngăn hoạt động quay thuận chiều và ngược chiều bắt đầu cùng một lúc.
Khóa liên động được đặt để ngăn hoạt động CHẠY JOG bắt đầu khi một chương trình khác đang được thực thi.

Quy trình bắt đầu quay về vị trí gốc (X23) được giữ lại trong nhãn công khai G_bHomingCMD và được gửi đến mô-đun Chuyển động làm điều kiện bắt đầu của quy trình quay về vị trí gốc.
Khóa liên động được đặt để ngăn hoạt động quay về vị trí gốc bắt đầu khi một chương trình khác đang được thực thi.
Sau khi nhận thấy mô-đun Chuyển động đã bật tín hiệu hoàn thành quay về vị trí gốc, G_bHomingCMD sẽ được đặt lại ở sườn lên của tín hiệu đó.

(2) Chương trình PLC CPU

1) Phần tiếp theo của MAIN (chương trình quét, ladder)

(Tiếp tục từ trang trước)



Sườn lên của hoạt động bắt đầu điều khiển định vị (X24) được giữ lại trong G_bPosCMD và được gửi đến mô-đun Chuyển động làm điều kiện bắt đầu hoạt động điều khiển định vị.
Khóa liên động được đặt để ngăn hoạt động điều khiển định vị bắt đầu khi một chương trình khác đang chạy.
Sau khi nhận thấy mô-đun Chuyển động đã bật tín hiệu hoàn thành quay về vị trí gốc, G_bPosCMD sẽ được đặt lại ở sườn lên của tín hiệu đó.

Chương trình bắt đầu hoạt động định vị liên tục được mô tả trong chương 3.
Để biết thông tin chi tiết, vui lòng tham khảo 3.4.

Lỗi được đặt lại khi X2F bật.

2.3 Ví dụ về chương trình

(2) Chương trình PLC CPU

2) MONITOR (ST, chương trình quét)

SetPosition (Vị trí đã đặt) và SetVelocity (Vận tốc đã đặt) của bộ giám sát trục được đặt làm nhãn công khai được lưu trữ trong các thiết bị từ D0 và D2.

Vì SetPosition và SetVelocity là loại số thực có độ chính xác kép nên chúng được chuyển đổi thành loại từ kép có dấu để PLC CPU có thể dễ dàng xử lý. (Lưu ý)

Mặc dù thiết bị từ này không được sử dụng trong khóa học này nhưng chúng được sử dụng để hiển thị dữ liệu trên chương trình tuần tự khác và GOT, cũng như cho các mục đích khác.

```
1  D0:D := LREAL_TO_DINT(RD78_0000.Axis0001.Md.SetPosition);
2  D2:D := LREAL_TO_DINT(RD78_0000.Axis0001.Md.SetVelocity);
3
```

Chỉ định loại từ kép có dấu với "D0:D".

(Lưu ý) Khi loại số thực có độ chính xác kép được chuyển đổi thành loại từ kép có dấu, nếu giá trị cần chuyển đổi nằm ngoài phạm vi từ -2147483648 đến 2147483647 thì sẽ xảy ra lỗi tính toán.

2.3 Ví dụ về chương trình

(3) Chương trình mô-đun Chuyển động

1) ServoON_JOG (kiểu thực thi thông thường)

```
1  //-----Servo ON-----
2  MC_Power_1(
3      Axis     := Axis0001.AxisRef,
4      Enable   := TRUE ,
5      ServoON  := G_bSVONCMD,
6      Busy     => bPowerBusy
7  );
8
9  //-----JOG Operation-----
10 //Initial Value Setting
11 IF (bPowerBusy) THEN
12     leJogAcceleration := 50000.0;
13     leJogDeceleration := 50000.0;
14     leJogJerk         := 0.0;
15 END_IF;
16
17 //JOG
18 MCv_Jog_1(
19     Axis         := Axis0001.AxisRef ,
20     JogForward   := G_bJogFwd ,
21     JogBackward  := G_bJogBwd ,
22     Velocity     := G_leJogVelocity ,
23     Acceleration:= leJogAcceleration ,
24     Deceleration:= leJogDeceleration ,
25     Jerk         := leJogJerk ,
26     Busy         => G_bJogBusy
27 );
28
```

Nhận tín hiệu BẬT servo (G_bSVONCMD) từ PLC CPU và thực thi BẬT servo.

Lưu giá trị tăng tốc CHẠY JOG, giảm tốc CHẠY JOG và giật CHẠY JOG vào nhãn khi đầu ra Bận của MC_Power_1 bật.

Nhận tín hiệu bắt đầu CHẠY JOG và vận tốc CHẠY JOG từ PLC CPU.

Trả đầu ra Bận về PLC CPU.

(Lưu ý) Trong chương trình mẫu này, tín hiệu I/O của FB không được sử dụng hoặc không được thay đổi so với giá trị ban đầu sẽ bị bỏ qua.

2.3 Ví dụ về chương trình

(3) Chương trình mô-đun Chuyển động

2) Quay về vị trí gốc (kiểu thực thi thông thường)

```
1  //-----Homing Operation-----
2  //Initial Value Setting, Operation Start Request
3  IF G_bHomingCMD THEN
4      G_TePositionO := 0.0 ;
5      G_bHomingReq  := TRUE ;
6  ELSE
7      G_bHomingReq  := FALSE ;
8  END_IF;
9
10 //Homing
11 MC_Home_1(
12     Axis             := Axis0001.AxisRef ,
13     Execute          := G_bHomingReq,
14     Position         := G_TePositionO ,
15     Done             => bHomingDone ,
16     Busy             => G_bHomingBusy ,
17     CommandAborted   => bHomingAborted ,
18     Error            => bHomingError
19 );
20
21 //Done Signal => PLC CPU
22 G_bHomingDone := bHomingDone OR bHomingAborted OR bHomingError;
23
```

Nhận tín hiệu điều khiển quay về vị trí gốc (G_bHomingCMD) từ PLC CPU.
Lưu địa chỉ định vị gốc vào nhãn và bật yêu cầu quay về vị trí gốc (G_bHomingReq).

Tắt G_bHomingReq khi G_bHomingCMD tắt.

Trả về tín hiệu thực thi hoàn tất cho PLC CPU sau khi hoàn tất thành công hoạt động quay về vị trí gốc (BẬT đầu ra Xong), quá trình thực thi bị gián đoạn (đầu ra CommandAborted BẬT) hoặc xảy ra lỗi (đầu ra Lỗi BẬT).

(3) Chương trình mô-đun Chuyển động

3) Định vị (kiểu thực thi thông thường)

```
1  //-----Positioning Operation-----
2  //Initial Value Setting, Operation Start Request
3  IF G_bPosCMD THEN
4      fePosition1       := 150000.0;
5      fePosVelocity     := 50000.0;
6      fePosAcceleration := 100000.0;
7      fePosDeceleration := 100000.0;
8      fePosJerk         := 200000.0;
9      G_bPosReq := TRUE;
10 ELSE
11     G_bPosReq := FALSE;
12 END_IF;
13
14 //Positioning1
15 MC_MoveAbsolute_1(
16     Axis             := Axis0001.AxisRef ,
17     Execute          := G_bPosReq ,
18     Position         := fePosition1 ,
19     Velocity         := fePosVelocity ,
20     Acceleration     := fePosAcceleration ,
21     Deceleration     := fePosDeceleration ,
22     Jerk             := fePosJerk ,
23     Direction        := MC_DIRECTION__mcShortestWay ,
24     BufferMode       := MC_BUFFER_MODE__mcAborting ,
25     Done             => bMoveAbs1Done ,
26     CommandAborted   => bMoveAbs1Aborted ,
27     Error            => bMoveAbs1Error
28 );
29 //Dwell
30 TON_1(IN:= bMoveAbs1Done ,PT:= T#500ms ,Q=> bDwell1_out );
31 //Positioning2
32 MC_MoveAbsolute_2(
33     Axis             := Axis0001.AxisRef ,
34     Execute          := bDwell1_out ,
35     Position         := G_fePosition0 ,
36     Velocity         := fePosVelocity ,
37     Acceleration     := fePosAcceleration ,
38     Deceleration     := fePosDeceleration ,
39     Jerk             := fePosJerk ,
40     Direction        := MC_DIRECTION__mcShortestWay ,
41     BufferMode       := MC_BUFFER_MODE__mcAborting ,
42     Done             => bMoveAbs2Done ,
43     CommandAborted   => bMoveAbs2Aborted ,
44     Error            => bMoveAbs2Error
45 );
46 //Dwell
47 TON_2(IN:= bMoveAbs2Done ,PT:= T#500ms ,Q=> bDwell2_out );
48
49 //Error Signal, Aborted Signal
50 bPosError   := bMoveAbs1Error OR bMoveAbs2Error;
51 bPosAborted := bMoveAbs1Aborted OR bMoveAbs2Aborted;
52
53 //Done Signal => PLC CPU
54 G_bPosDone := bDwell2_out OR bPosError OR bPosAborted;
55
```

Nhận tín hiệu bắt đầu điều khiển định vị (G_bPosCMD) từ PLC CPU. Dữ liệu cần để định vị được lưu vào nhãn và yêu cầu bắt đầu điều khiển định vị (G_bPositioningReq) được bật.

Tắt G_bPositioningReq sau khi PLC CPU tắt G_bPosCMD.

MC_MoveAbsolute thực thi hoạt động định vị.

Khởi động bộ đếm thời gian tác động trễ kích hoạt thời gian dừng khi đầu ra Xong của MC_MoveAbsolute_1 bật.

Sau khi hết thời gian dừng, MC_MoveAbsolute sẽ định vị lại.

Khởi động bộ đếm thời gian tác động trễ kích hoạt thời gian dừng khi đầu ra Xong của MC_MoveAbsolute_2 bật.

Trả về tín hiệu thực thi hoàn tất cho PLC CPU sau khi hết thời gian dừng hoặc đầu ra Lỗi hoặc đầu ra CommandandAborted của MC_MoveAbsolute bật.

2.3 Ví dụ về chương trình

(3) Chương trình mô-đun Chuyển động

4) ErrorReset (kiểu thực thi thông thường)

```
1  //Axis Error Reset
2  MC_Reset_1(
3      Axis    := Axis0001.AxisRef ,
4      Execute := G_bErrorReset
5  );
6
7  //System Error Reset
8  MCv_MotionErrorReset_1(
9      Execute := G_bErrorReset
10 );
11
```

Nhận tín hiệu đặt lại lỗi (G_bErrorReset) từ PLC CPU và thực hiện đặt lại lỗi trục và đặt lại lỗi hệ thống.

2.4 Ghi chương trình

Ghi chương trình và tham số vào PLC CPU và mô-đun Chuyển động.

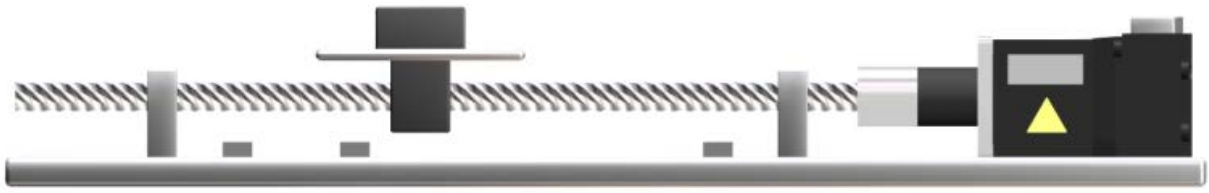
1) Sau khi tất cả chương trình trong PLC CPU được xây dựng lại, hãy chọn [Online] → [Write to PLC] trong thanh công cụ của GX Works3 để ghi tất cả dữ liệu vào PLC CPU.

2) Khi tham số được ghi vào PLC CPU, giao tiếp với mô-đun Chuyển động được bật.
Chọn [Online] → [Write to Module] trong thanh công cụ của Motion Control Setting Function để ghi tất cả dữ liệu vào mô-đun Chuyển động.

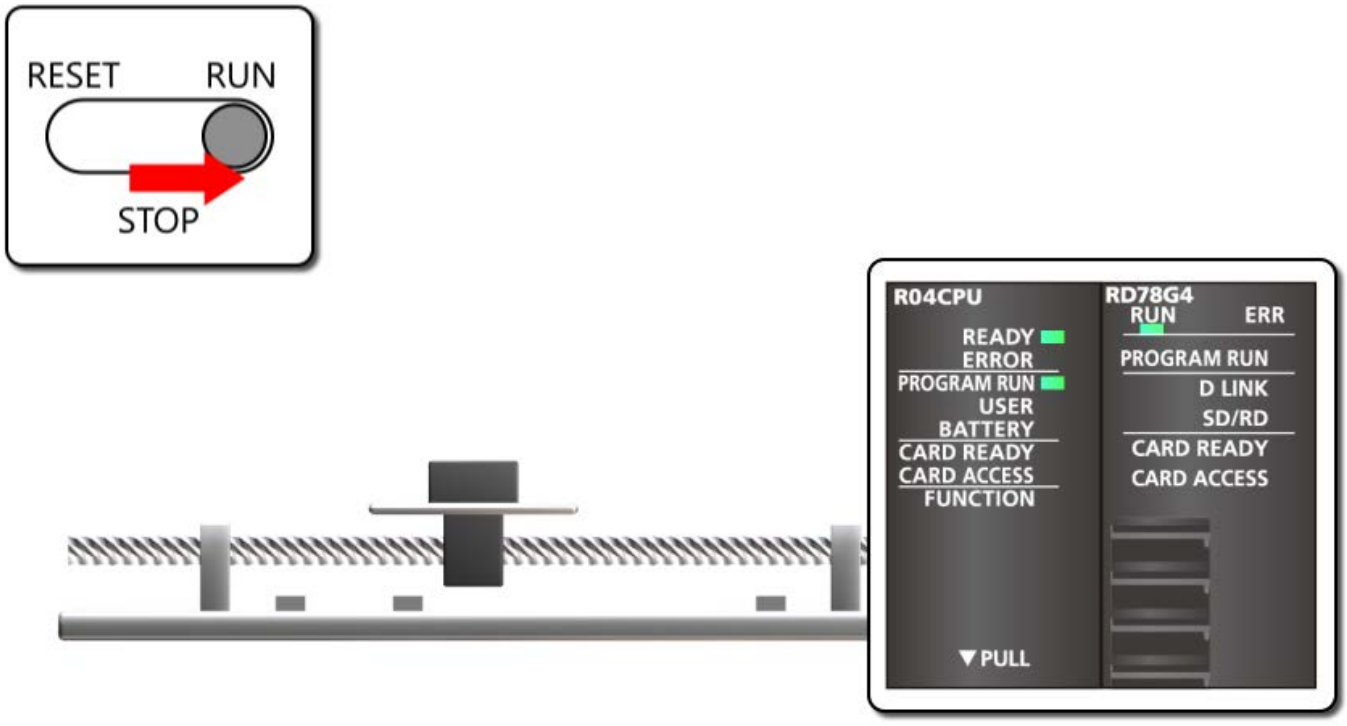
3) Đặt lại PLC CPU để hoàn tất hoạt động ghi.

2.5 Kiểm tra hoạt động

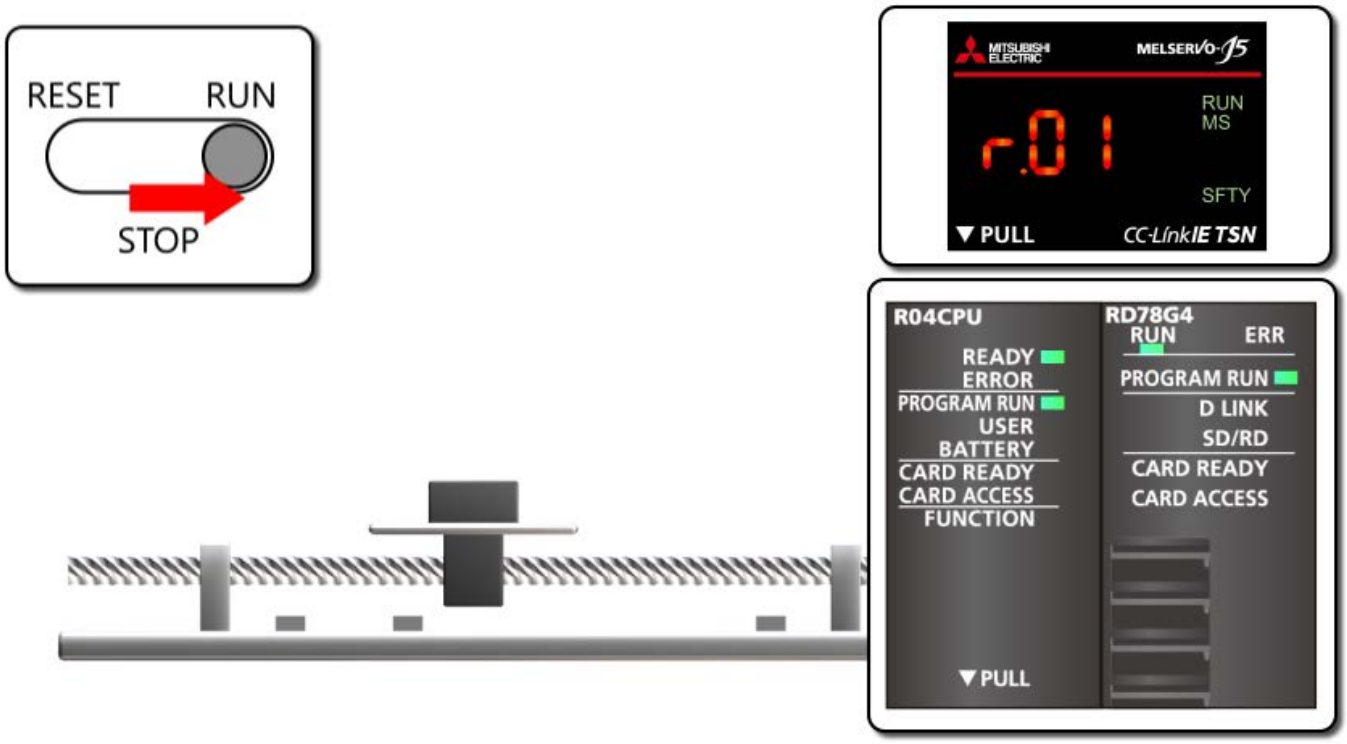
Nhấp vào nút phát ở phần dưới bên trái của cửa sổ.



Kiểm tra hoạt động của chương trình mẫu.
Trước khi bắt đầu hoạt động, hãy đảm bảo chương trình của PLC CPU và mô-đun Chuyển động đã được ghi lại.

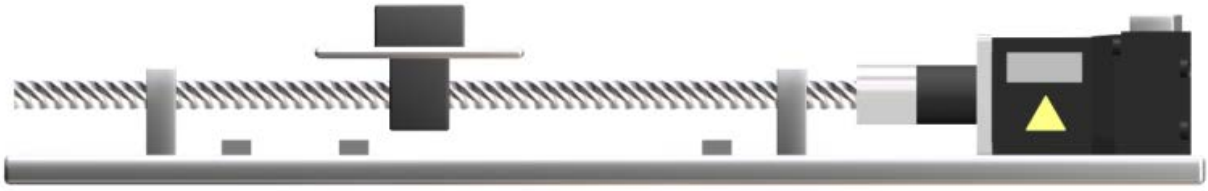


Đặt công tắc RUN/STOP/RESET của PLC CPU thành CHẠY.
Đèn READY và đèn PROGRAM RUN của PLC CPU bật.
Đèn RUN của mô-đun Chuyển động bật.



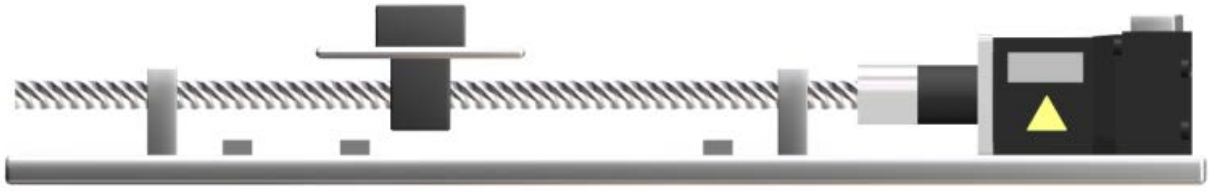
Chờ cho đến khi đèn PROGRAM RUN của mô-đun Chuyển động bật. "r.01" được hiển thị trên bộ khuếch đại servo. (Các điểm sáng lên.) Mô-tơ servo tiến vào trạng thái BẬT servo.

2.5 Kiểm tra hoạt động



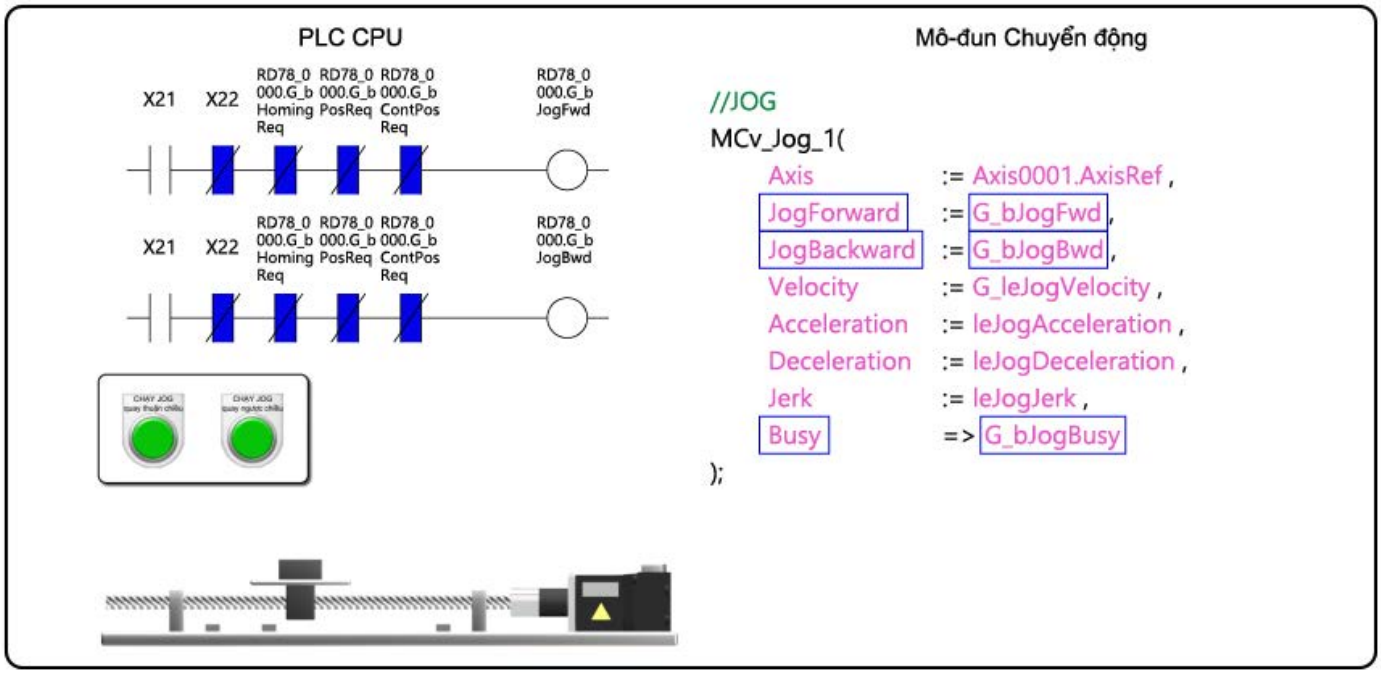
Bật X20 để thực thi TẮT servo.
"r.01" được hiển thị trên bộ khuếch đại servo. (Các điểm nhấp nháy.)
Tắt X20 để thực hiện lại hoạt động BẬT servo.

2.5 Kiểm tra hoạt động



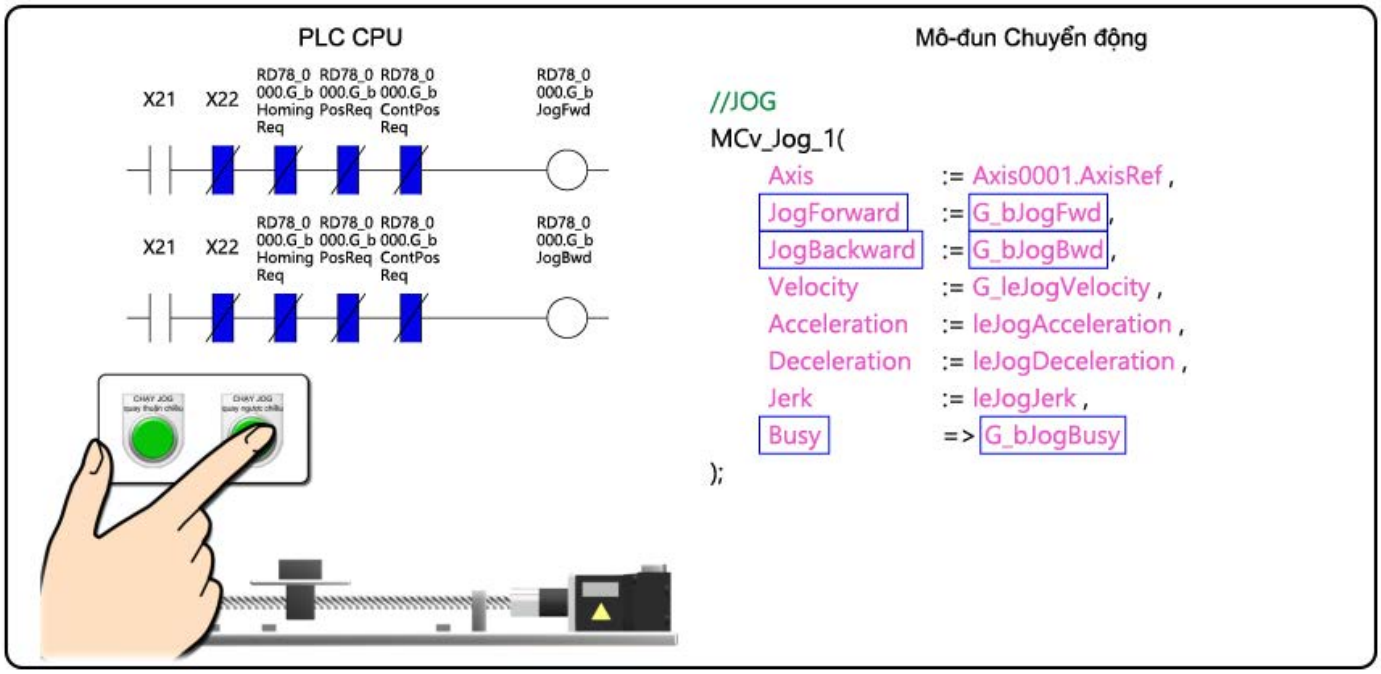
Bật CHẠY JOG quay thuận chiều (X21) để di chuyển trục theo hướng tăng địa chỉ và tắt để dừng lại.
Bật CHẠY JOG quay ngược chiều (X22) để di chuyển trục theo hướng giảm địa chỉ và tắt để dừng lại.

2.5 Kiểm tra hoạt động



Kiểm tra hoạt động giám sát chương trình.
Khi bật X21, "RD78_0000.G_bJogFwd" và "G_bJogFwd" phía mô-đun Chuyển động đều bật.
Khi đầu vào JogForward của MCv_Jog_1 bật, CHẠY JOG quay thuận chiều sẽ bắt đầu.

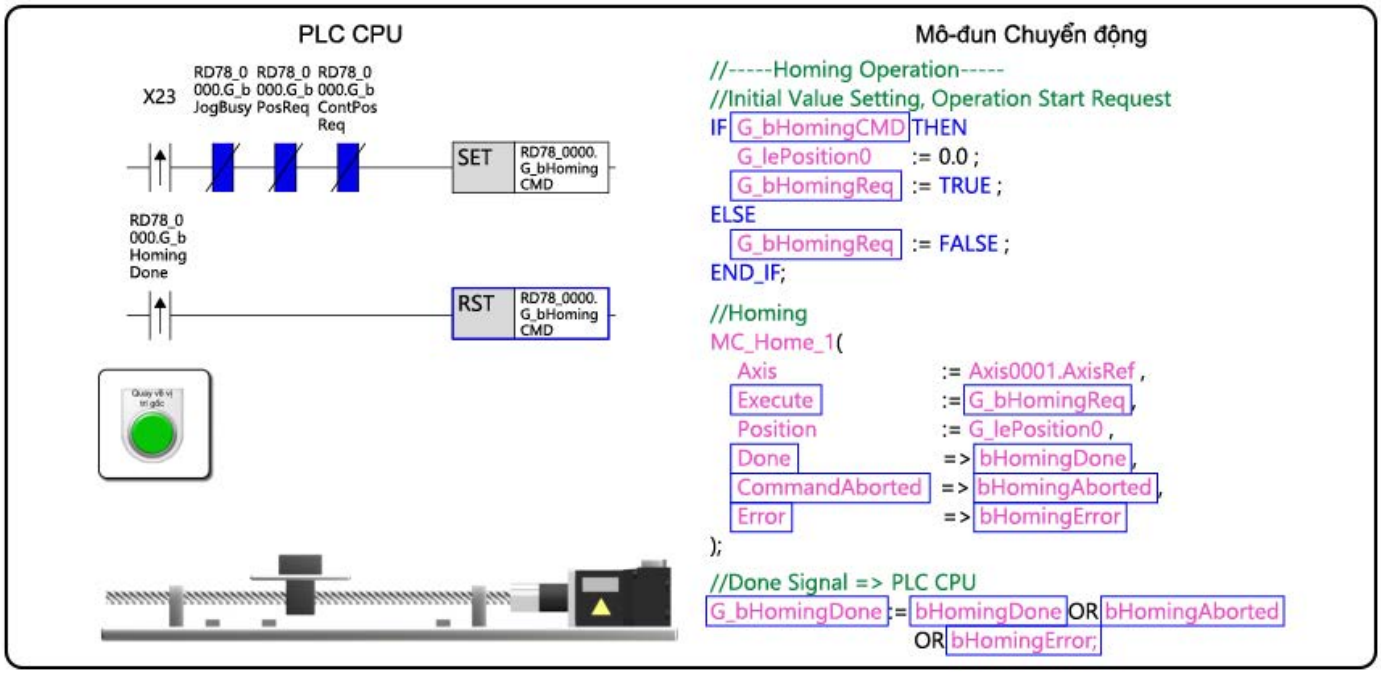
2.5 Kiểm tra hoạt động



Khi bật X22, "RD78_0000.G_bJogBwd" và "G_bJogBwd" phía mô-đun Chuyển động đều bật.
Khi đầu vào JogBackward của MCv_Jog_1 bật, CHẠY JOG quay ngược chiều sẽ bắt đầu.

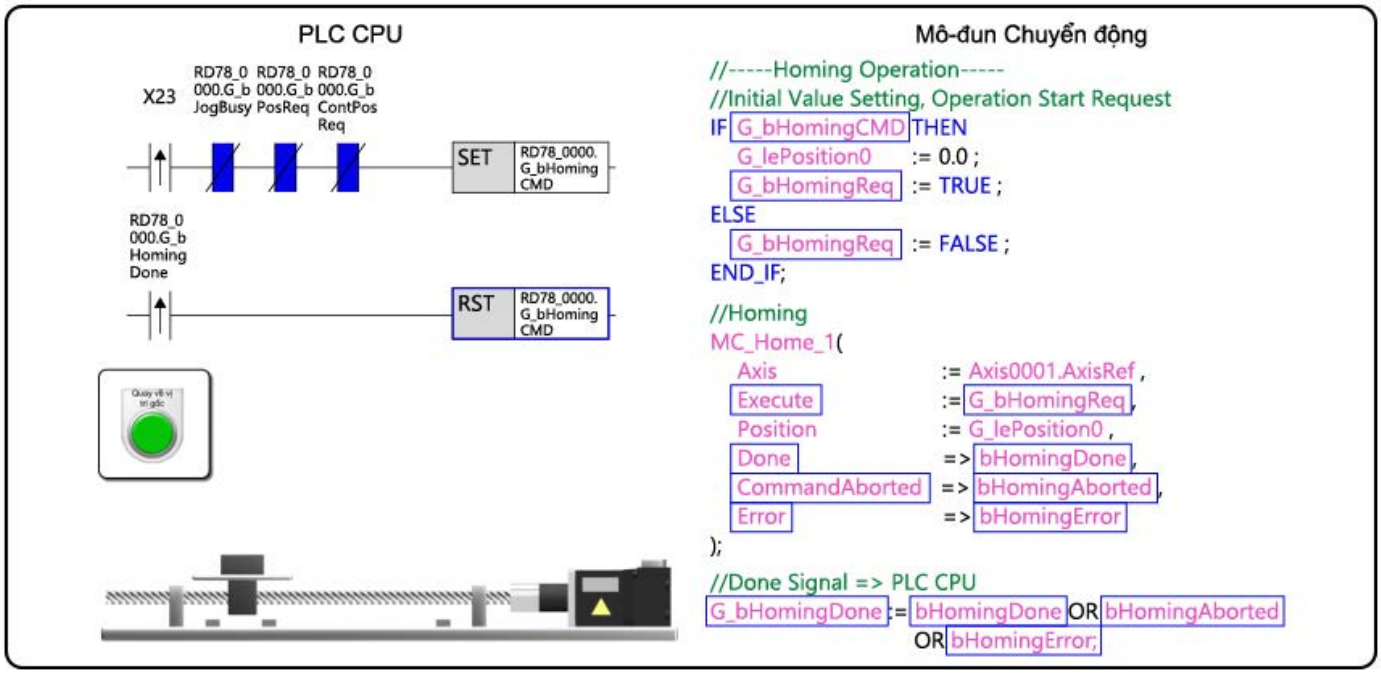
Bật quay về vị trí gốc (X23) để bắt đầu quy trình quay về vị trí gốc.
Thực thi hoạt động quay về vị trí gốc với phương pháp proximity dog
(33 được trừ từ Pr.PT45)
Trục dừng xa hơn một chút so với dog và đặt điểm đó làm vị trí gốc.

2.5 Kiểm tra hoạt động



Kiểm tra hoạt động giám sát chương trình.
"RD78_0000.G_bHomingCMD" được đặt khi X23 bật.
"G_bHomingCMD" phía mô-đun Chuyển động bật và
"G_bHomingReq", lệnh thực thi của MC_Home_1, bật.

2.5 Kiểm tra hoạt động



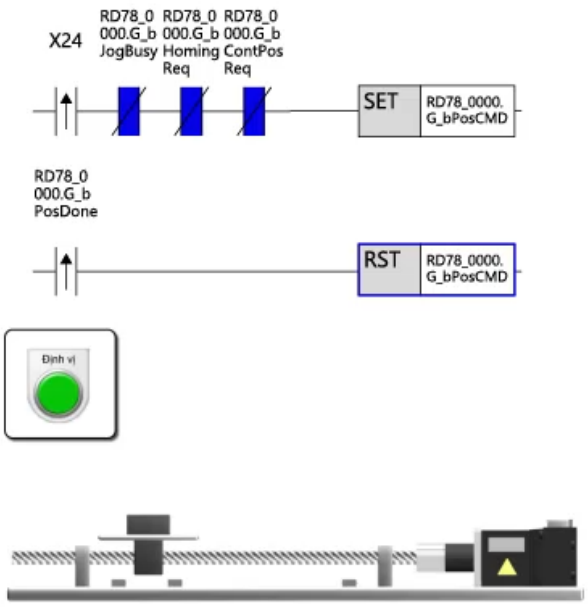
Khi quy trình quay về vị trí gốc hoàn tất, đầu ra Xong và "G_bHomingDone" sẽ bật.
"G_bHomingCMD" phía PLC CPU được đặt lại về trạng thái ban đầu.

2.5 Kiểm tra hoạt động

Bật quy trình bắt đầu định vị (X24) sẽ bắt đầu chuyển động tịnh tiến. Trục tiến lên 150 mm rồi dừng trong 0,5 giây và lùi lại 150 mm rồi dừng trong 0,5 giây.

2.5 Kiểm tra hoạt động

PLC CPU



Mô-đun Chuyển động

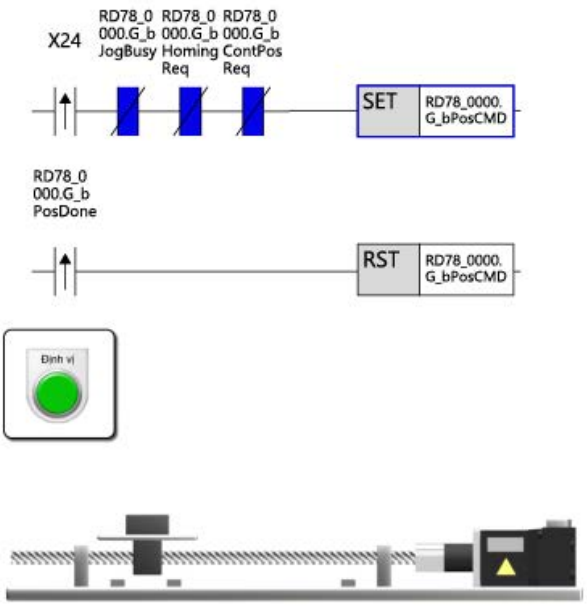
```
//-----Positioning Operation-----
//Initial Value Setting, Operation Start Request
IF G_bPosCMD THEN
   iePosition1          := 150000.0;
   iePosVelocity        := 50000.0;
   iePosAcceleration    := 100000.0;
   iePosDeceleration    := 100000.0;
   iePosJerk            := 200000.0;
   G_bPosReq  := TRUE;
ELSE
   G_bPosReq  := FALSE;
END_IF;

//Positioning1
MC_MoveAbsolute_1(
   Axis          := Axis0001.AxisRef,
   Execute       := G_bPosReq,
   Position      := iePosition1,
   Velocity      := iePosVelocity,
   Acceleration  := iePosAcceleration,
   Deceleration  := iePosDeceleration,
   Jerk          := iePosJerk,
   Direction     := MC_DIRECTION_mcShortestWay,
   BufferMode    := MC_BUFFER_MODE_mcAborting,
   Done          => bMoveAbs1Done,
   CommandAborted=> bMoveAbs1Aborted,
   Error         => bMoveAbs1Error
);
```

Kiểm tra hoạt động giám sát chương trình.
"RD78_0000.G_bPosCMD" được đặt khi X24 bật.
"G_bPosCMD" phía mô-đun Chuyển động bật và "G_bPosReq", lệnh thực thi của MC_MoveAbsolute_1, bật.

2.5 Kiểm tra hoạt động

PLC CPU



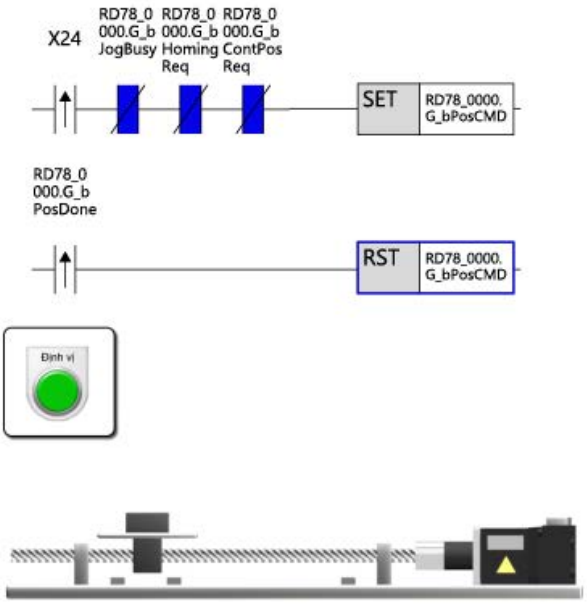
Mô-đun Chuyển động

```
//Positioning2
MC_MoveAbsolute_2(
   Axis          := Axis0001.AxisRef,
   Execute       := bDwell1_out,
   Position      := G_lePosition0,
   Velocity      := lePosVelocity,
   Acceleration  := lePosAcceleration,
   Deceleration  := lePosDeceleration,
   Jerk          := lePosJerk,
   Direction     := MC_DIRECTION_mcShortestWay,
   BufferMode    := MC_BUFFER_MODE_mcAborting,
   Done                => bMoveAbs2Done,
   CommandAborted      => bMoveAbs2Aborted,
   Error               => bMoveAbs2Error
);
//Dwell
TON_2(IN:= bMoveAbs2Done,PT:= T#500ms ,Q=> bDwell2_out);
//Error Signal, Aborted Signal
bPosError     := bMoveAbs1Error OR bMoveAbs2Error;
bPosAborted   := bMoveAbs1Aborted OR bMoveAbs2Aborted;
//Done Signal => PLC CPU
G_bPosDone    := bDwell2_out OR bPosError OR bPosAborted;
```

Sau khi chuyển động tịnh tiến hoàn tất và thời gian dừng kết thúc, "G_bPosDone" sẽ bật.
"G_bPosCMD" phía PLC CPU được đặt lại về trạng thái ban đầu.

2.5 Kiểm tra hoạt động

PLC CPU



Mô-đun Chuyển động

```
//Positioning2
MC_MoveAbsolute_2(
   Axis          := Axis0001.AxisRef,
   Execute       := bDwell1_out,
   Position      := G_lePosition0,
   Velocity      := lePosVelocity,
   Acceleration  := lePosAcceleration,
   Deceleration  := lePosDeceleration,
   Jerk          := lePosJerk,
   Direction     := MC_DIRECTION_mcShortestWay,
   BufferMode    := MC_BUFFER_MODE_mcAborting,
   Done          => bMoveAbs2Done,
   CommandAborted => bMoveAbs2Aborted,
   Error         => bMoveAbs2Error
);
//Dwell
TON_2(IN:= bMoveAbs2Done,PT:= T#500ms ,Q=> bDwell2_out);
//Error Signal, Aborted Signal
bPosError     := bMoveAbs1Error OR bMoveAbs2Error;
bPosAborted   := bMoveAbs1Aborted OR bMoveAbs2Aborted;
//Done Signal => PLC CPU
G_bPosDone    := bDwell2_out OR bPosError OR bPosAborted;
```

Thao tác này hoàn tất việc kiểm tra hoạt động.
Sang trang tiếp theo.

2.6 Tóm tắt chương này

Trong chương này, bạn đã học về:

- Nhãn công khai là gì?
- Cài đặt nhãn công khai
- Ví dụ về chương trình
- Ghi chương trình
- Kiểm tra hoạt động

Những điểm quan trọng

Nhãn công khai là gì?	• Nhãn công khai là nhãn dùng chung có thể sử dụng trong cả mô-đun Chuyển động và PLC CPU.
Cài đặt nhãn công khai	• Đăng ký nhãn công khai từ nhãn toàn cục của mô-đun Chuyển động. • Chọn xem mỗi nhãn có được đọc hoặc ghi từ/đến PLC CPU không. • Để đặt thành phần của loại dữ liệu có cấu trúc đã chuẩn bị trong hệ thống thành nhãn công khai, hãy đăng ký nhãn công khai theo lớp của loại dữ liệu có cấu trúc. • Sau khi đặt nhãn công khai trong mô-đun Chuyển động, xây dựng lại tất cả chương trình và ánh xạ nhãn công khai. • Nhãn công khai được đăng ký vào nhãn mô-đun ở phía PLC CPU.
Ví dụ về chương trình	• Chương này mô tả ví dụ chương trình sau: chương trình ladder của PLC CPU sử dụng nhãn công khai để trao đổi tín hiệu bắt đầu định vị và tín hiệu hoàn thành định vị.
Ghi chương trình	• Trước tiên, ghi dữ liệu vào PLC CPU rồi ghi vào mô-đun Chuyển động.
Kiểm tra hoạt động	• Bạn đã kiểm tra hoạt động của chương trình mẫu trong video.

Chương 3 Chế độ đệm

Chế độ đệm liên tục thực thi các hoạt động bằng cách khởi động nhiều FB hoạt động của Motion control FB.
Chế độ có thể được thiết lập bằng đầu vào BufferMode của Motion control FB.
Có thể khởi động đồng thời tối đa hai FB cho mỗi trục và nhóm trục.

(Ví dụ) MC_MoveAbsolute

```
MC_MoveAbsolute_1(
    Axis            := Axis0001.AxisRef ,
    Execute         := G_bPositioningReq ,
    ContinuousUpdate:= FALSE ,
    Position        := lePosition1 ,
    Velocity        := lePosVelocity ,
    Acceleration    := lePosAcceleration ,
    Deceleration    := lePosDeceleration ,
    Jerk            := lePosJerk ,
    Direction       := MC_DIRECTION_mcShortestWay ,
    BufferMode      := MC_BUFFER_MODE_mcAborting ,
    Options         := 0 ,//mcAccDec
    Done            => bMoveAbs1Done ,
    CommandAborted  => bMoveAbs1Aborted ,
    Error           => bMoveAbs1Error
);
```

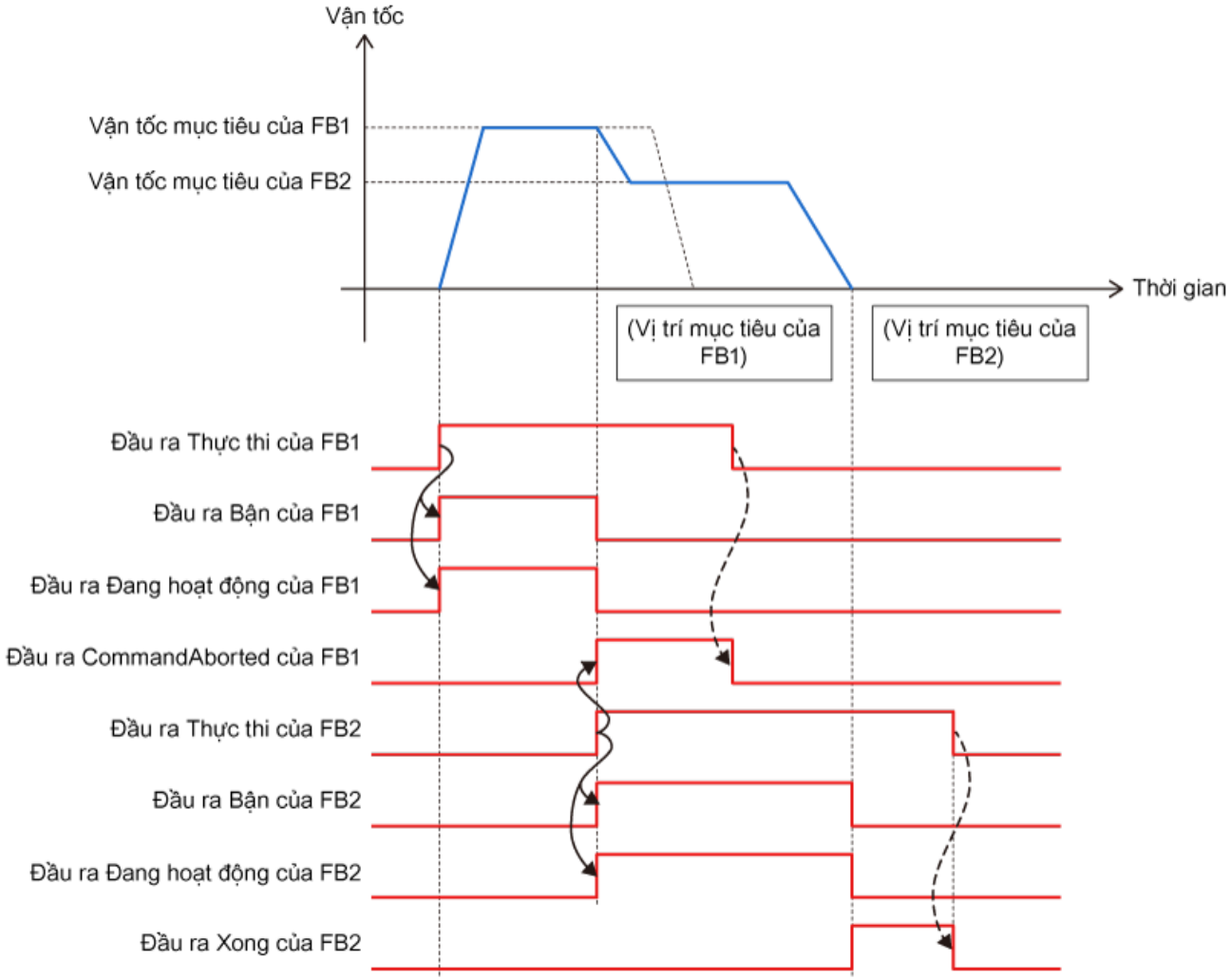
- 0 hoặc MC_BUFFER_MODE _ mcAborting ••• FB đang được thực thi sẽ bị gián đoạn và FB tiếp theo sẽ được thực thi ngay lập tức.
- 1 hoặc MC_BUFFER_MODE_mcBuffered ••• Sau khi hoạt động của FB đang được thực thi hoàn tất, FB tiếp theo sẽ được thực thi.
- 2 hoặc MC_BUFFER_MODE_mcBlendingLow ••• Vận tốc thấp hơn trong số những vận tốc mục tiêu cho FB đang được thực thi và FB sẽ được đệm được đặt thành vận tốc chuyển đổi.
- 3 hoặc MC_BUFFER_MODE_mcBlendingPrevious ••• Vận tốc mục tiêu của FB đang được thực thi được đặt thành vận tốc chuyển đổi.
- 4 hoặc MC_BUFFER_MODE_mcBlendingNext ••• Vận tốc mục tiêu của FB sẽ được đệm được đặt thành vận tốc chuyển đổi.
- 5 hoặc MC_BUFFER_MODE_mcBlendingHigh ••• Vận tốc cao hơn trong số những vận tốc mục tiêu cho FB đang được thực thi và FB sẽ được đệm được đặt thành vận tốc chuyển đổi.

[Điểm]
Đối với đầu vào Direction và BufferMode, hãy chỉ định số hoặc bộ liệt kê ENUM bắt đầu với MC_BUFFER_MODE và MC_DIRECTION.
Để biết thông tin chi tiết về bộ liệt kê ENUM, hãy tham khảo hướng dẫn sau.

MELSEC iQ-R Programming Manual (Motion Control Function Blocks)
2 VARIABLES AND MOTION CONTROL FB
2.2 List of Enumerators

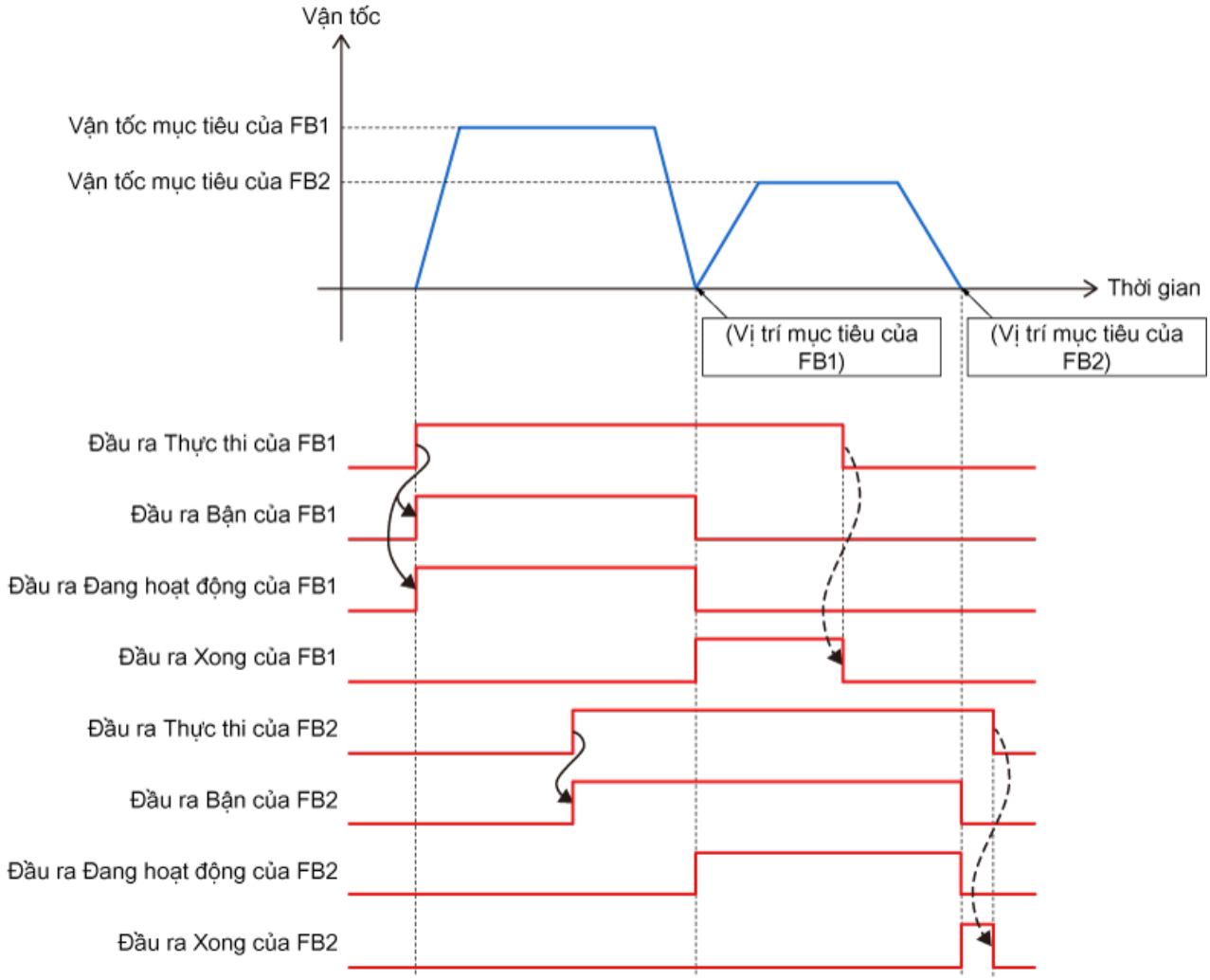
3.1 Hủy bỏ

Sơ đồ sau đây mô tả hoạt động khi BufferMode được đặt thành 0:mcAborting.
FB đang được thực thi sẽ bị gián đoạn và FB tiếp theo sẽ được thực thi ngay lập tức.



3.2 Đệm

Sơ đồ sau đây mô tả hoạt động khi BufferMode được đặt thành 1:mcBuffered.
Khi hoạt động của FB đang được thực thi hoàn tất, FB tiếp theo sẽ được thực thi.

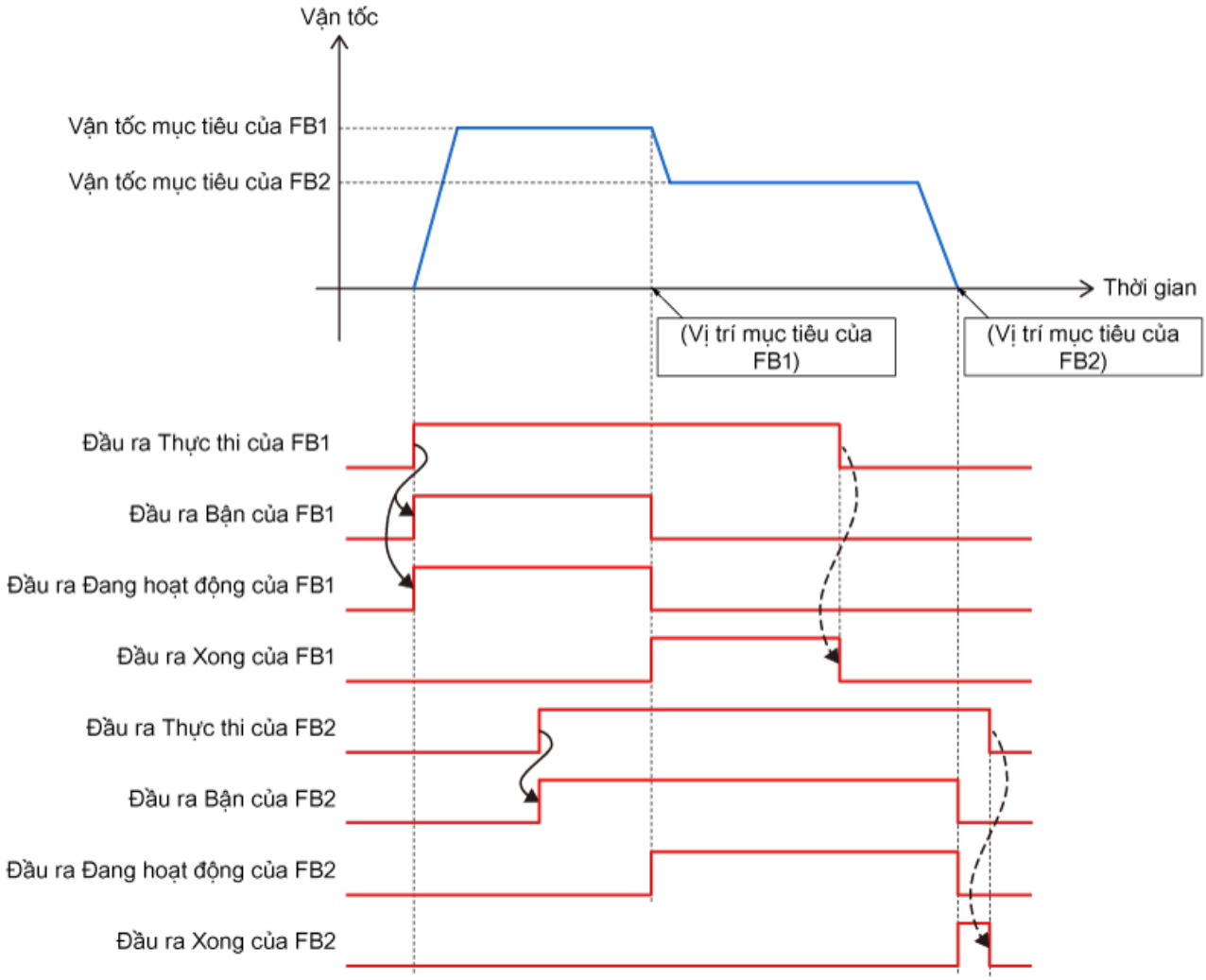


3.3 Phối hợp

Khi BufferMode được đặt thành mcBlending***, FB tiếp theo sẽ được thực thi nối tiếp ngay sau khi đến vị trí đích của FB đang được thực thi.
Trong mô tả sau, FB được thực thi đầu tiên là FB1 và FB được đệm là FB2.

(1) BlendingPrevious

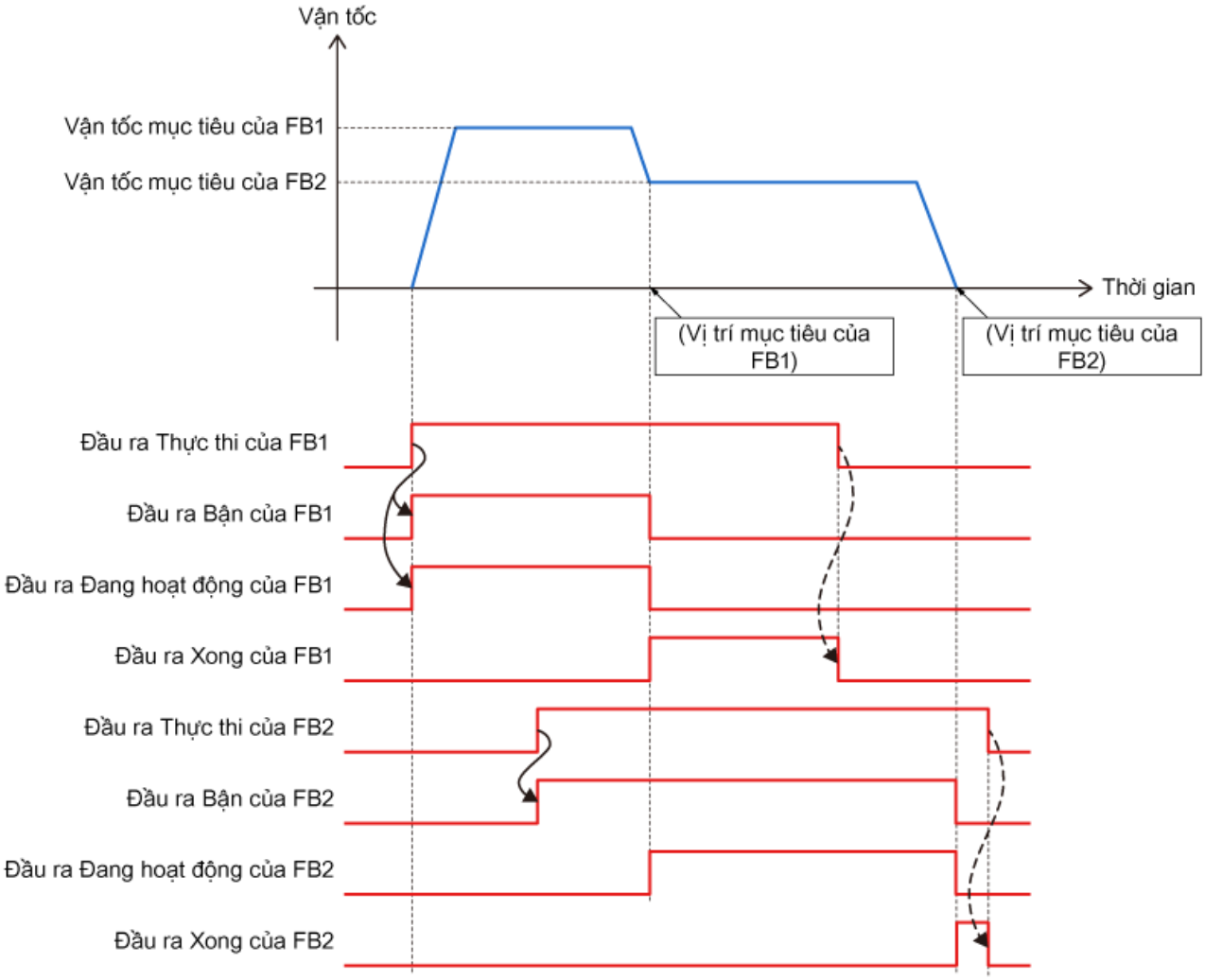
Sơ đồ sau đây mô tả hoạt động khi BufferMode được đặt thành 3: mcBlendingPrevious.
Hoạt động được thực hiện ở vận tốc mục tiêu của FB1 đến khi đạt vị trí mục tiêu của FB1.
Khi hoạt động được chuyển sang FB2, vận tốc được đổi thành vận tốc mục tiêu của FB2
và di chuyển đến vị trí mục tiêu của FB2.



(2) BlendingNext

Sơ đồ sau đây mô tả hoạt động khi BufferMode được đặt thành 4: mcBlendingNext.

Vận tốc thay đổi thành vận tốc mục tiêu của FB2 khi hoạt động đạt đến vị trí đích của FB1.



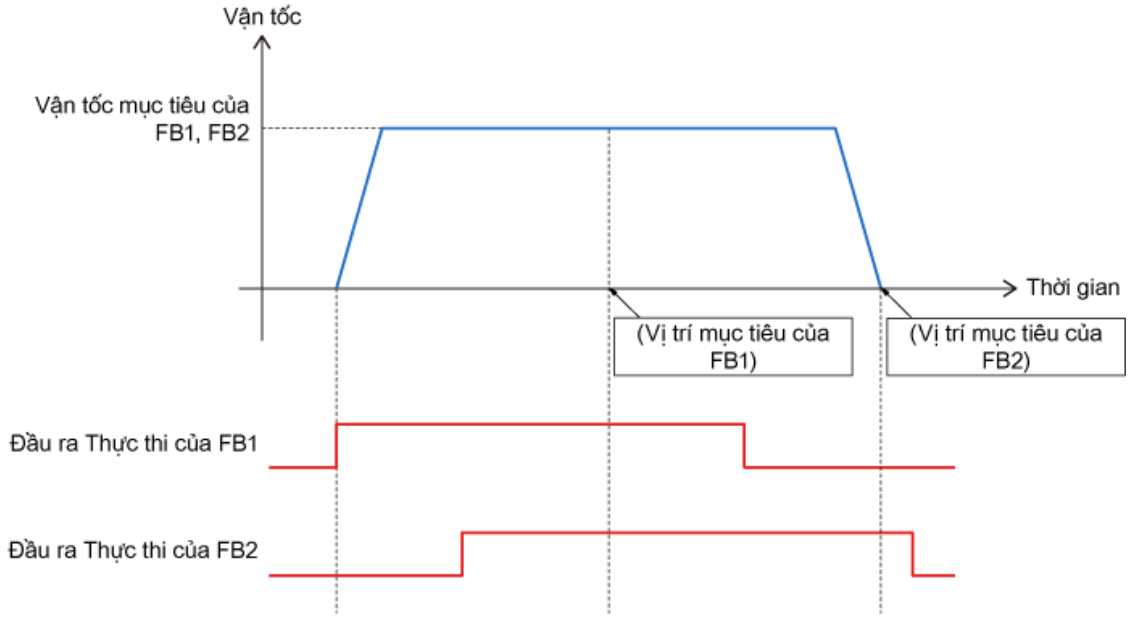
(3) BlendingLow, BlendingHigh

Hoạt động khi BufferMode được đặt thành 2: mcBlendingLow hoặc 5: mcBlendingHigh thay đổi tùy thuộc vào vận tốc mục tiêu nào lớn hơn giữa FB1 và FB2.

Giá trị cài đặt	Vận tốc mục tiêu của FB1 > Vận tốc mục tiêu của FB2	Vận tốc mục tiêu của FB1 < Vận tốc mục tiêu của MFB2
2: mcBlendingLow	Hoạt động tương tự BlendingPrevious	Hoạt động tương tự BlendingNext
5: mcBlendingHigh	Hoạt động tương tự BlendingNext	Hoạt động tương tự BlendingPrevious

[Điểm]

Sơ đồ sau đây mô tả dạng sóng vận tốc cho BlendingPrevious, BlendingNext, BlendingHigh và BlendingLow khi vận tốc mục tiêu của FB1 và FB2 bằng nhau.



3.4 Ví dụ về chương trình

(1) Hoạt động của chương trình mẫu

Chương này sử dụng chương trình mẫu được sử dụng ở Chương 2.

Kiểm tra sự khác biệt trong hoạt động của chế độ đệm trong chương trình bắt đầu bằng X25.

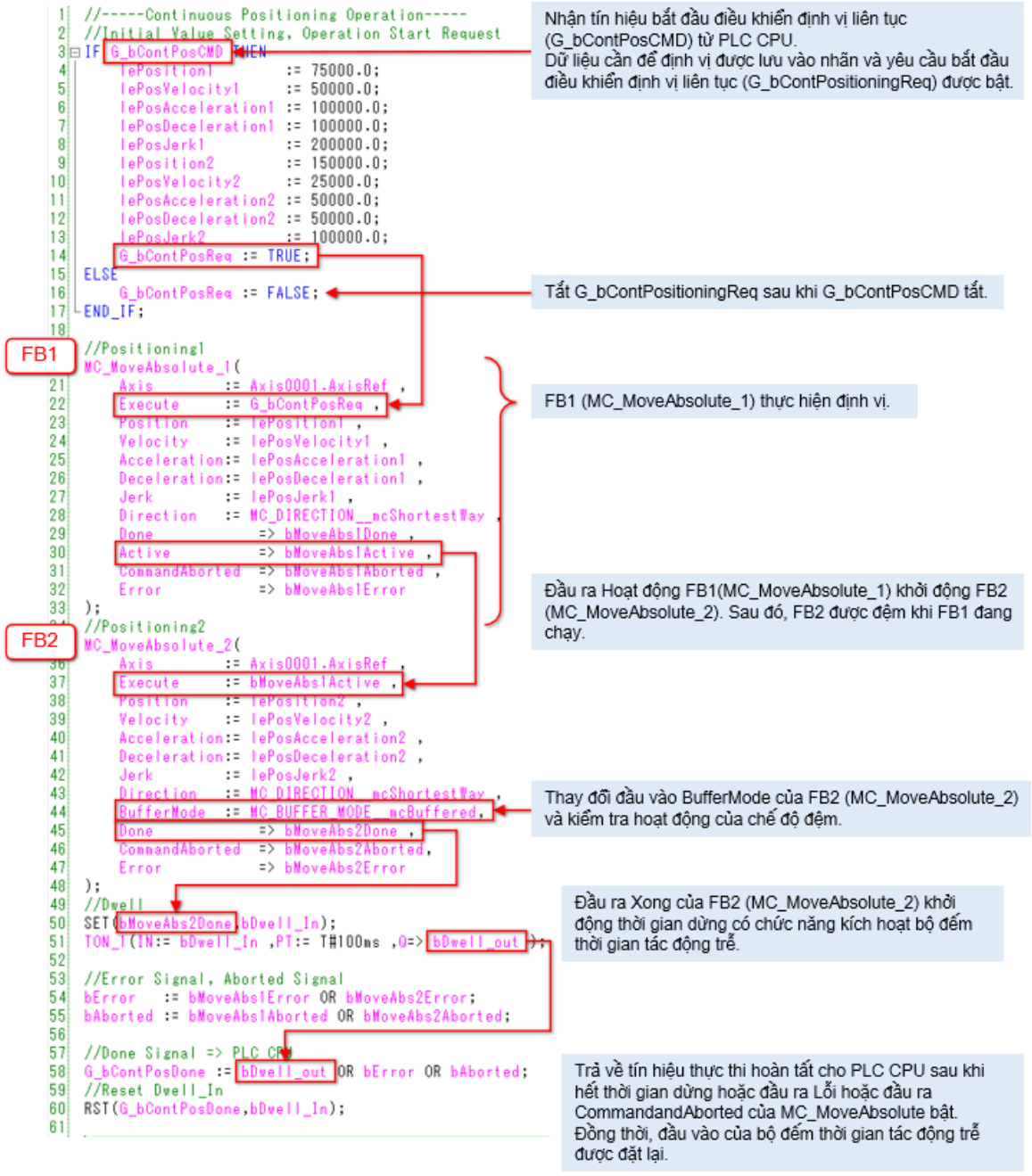
Mục	FB1 (MC_MoveAbsolute)	FB2 (MC_MoveAbsolute)
Địa chỉ định vị	75000,0 [µm]	150000,0 [µm]
Vận tốc	50000,0 [µm/s]	25000,0 [µm/s]
Tăng tốc, giảm tốc	100000,0 [µm/ s^2]	50000,0 [µm/ s^2]
Giật	200000,0 [µm/ s^3]	100000,0 [µm/ s^3]

(2) Chương trình PLC CPU

MAIN (chương trình quét, ladder)

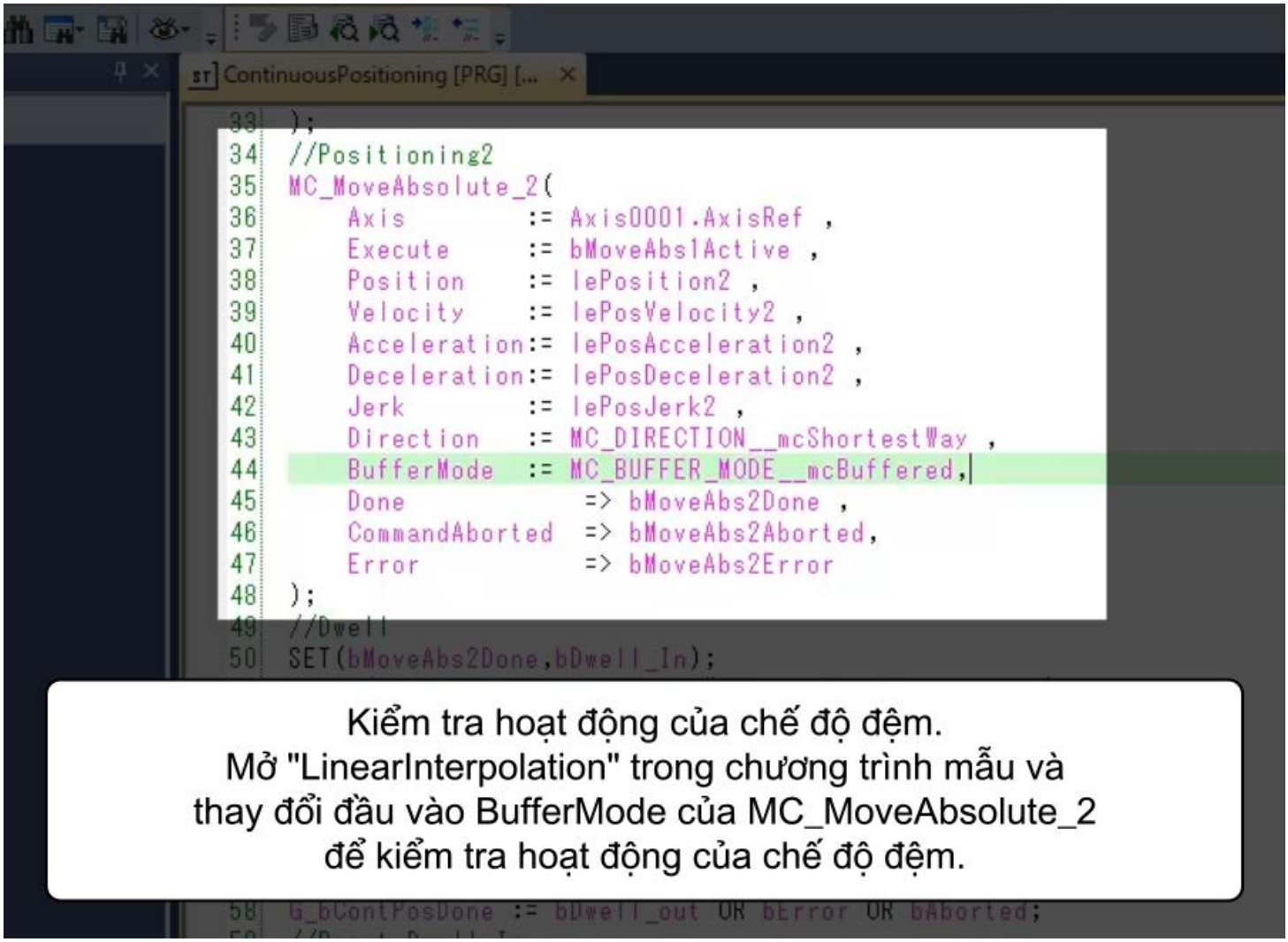
Sườn lên của bắt đầu điều khiển định vị liên tục (X25) được giữ lại trong G_bContPosCMD và được gửi đến mô-đun Chuyển động làm điều kiện bắt đầu hoạt động điều khiển định vị liên tục.
Khóa liên động được đặt để ngăn hoạt động điều khiển định vị bắt đầu khi một chương trình khác đang chạy.
Sau khi nhận thấy mô-đun Chuyển động đã bật tín hiệu hoàn thành quay về vị trí gốc liên tục, G_bContPosCMD sẽ được đặt lại ở sườn lên của tín hiệu đó.

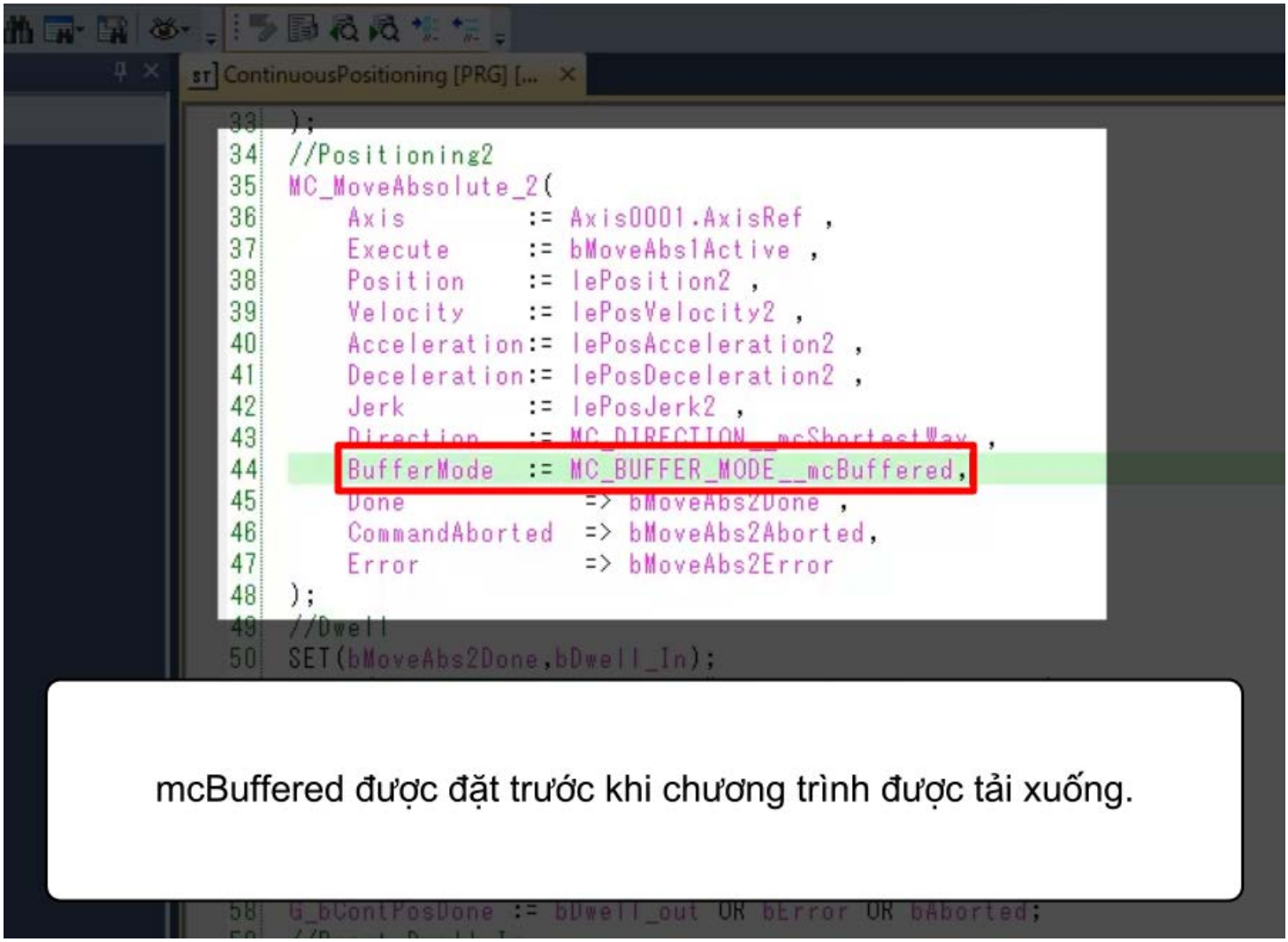
(3) Chương trình mô-đun chuyển động
ContinuousPositioning (kiểu thực thi thông thường)



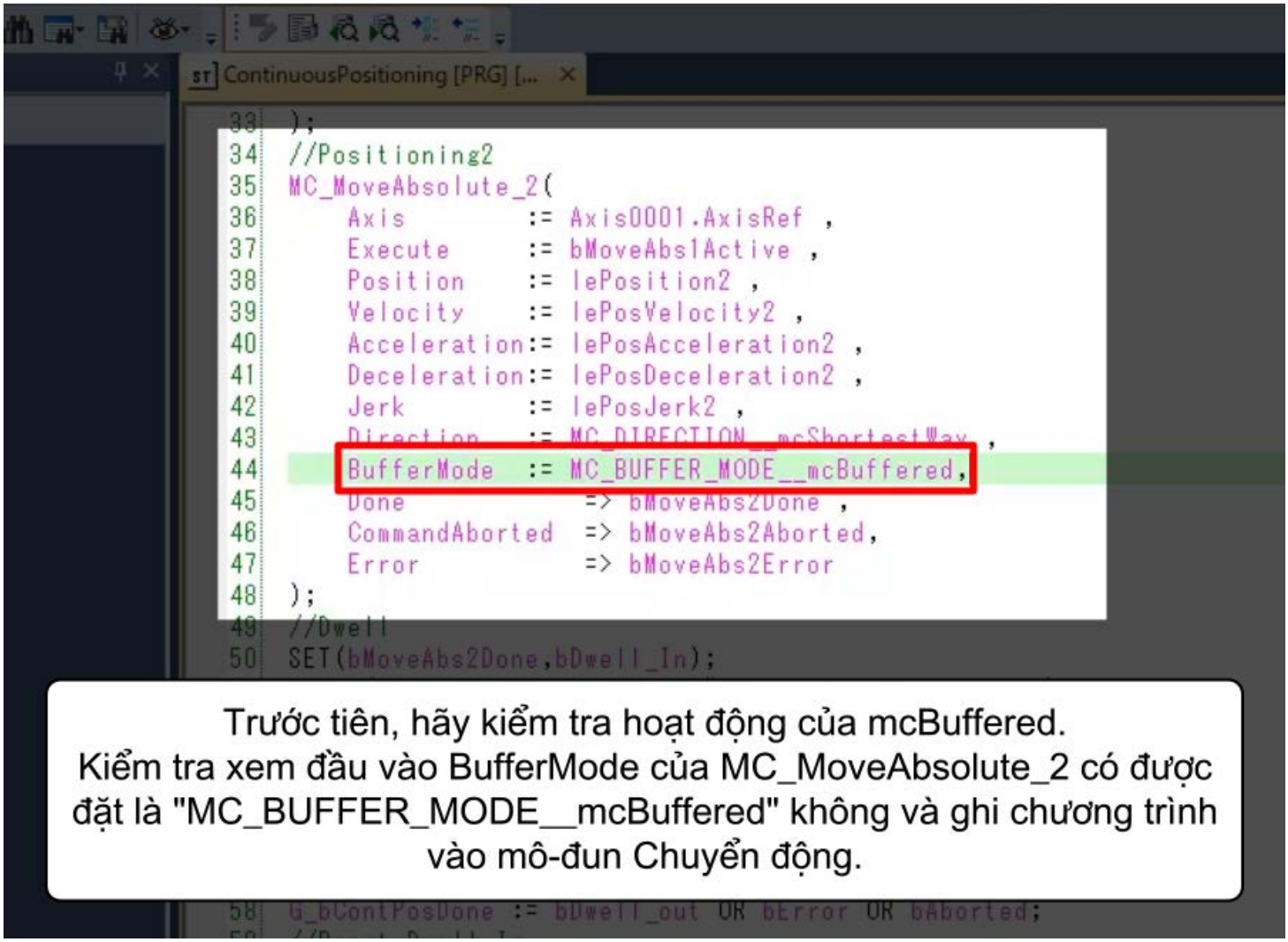
Nhấp vào nút phát ở phần dưới bên trái của cửa sổ.

3.5 Kiểm tra hoạt động

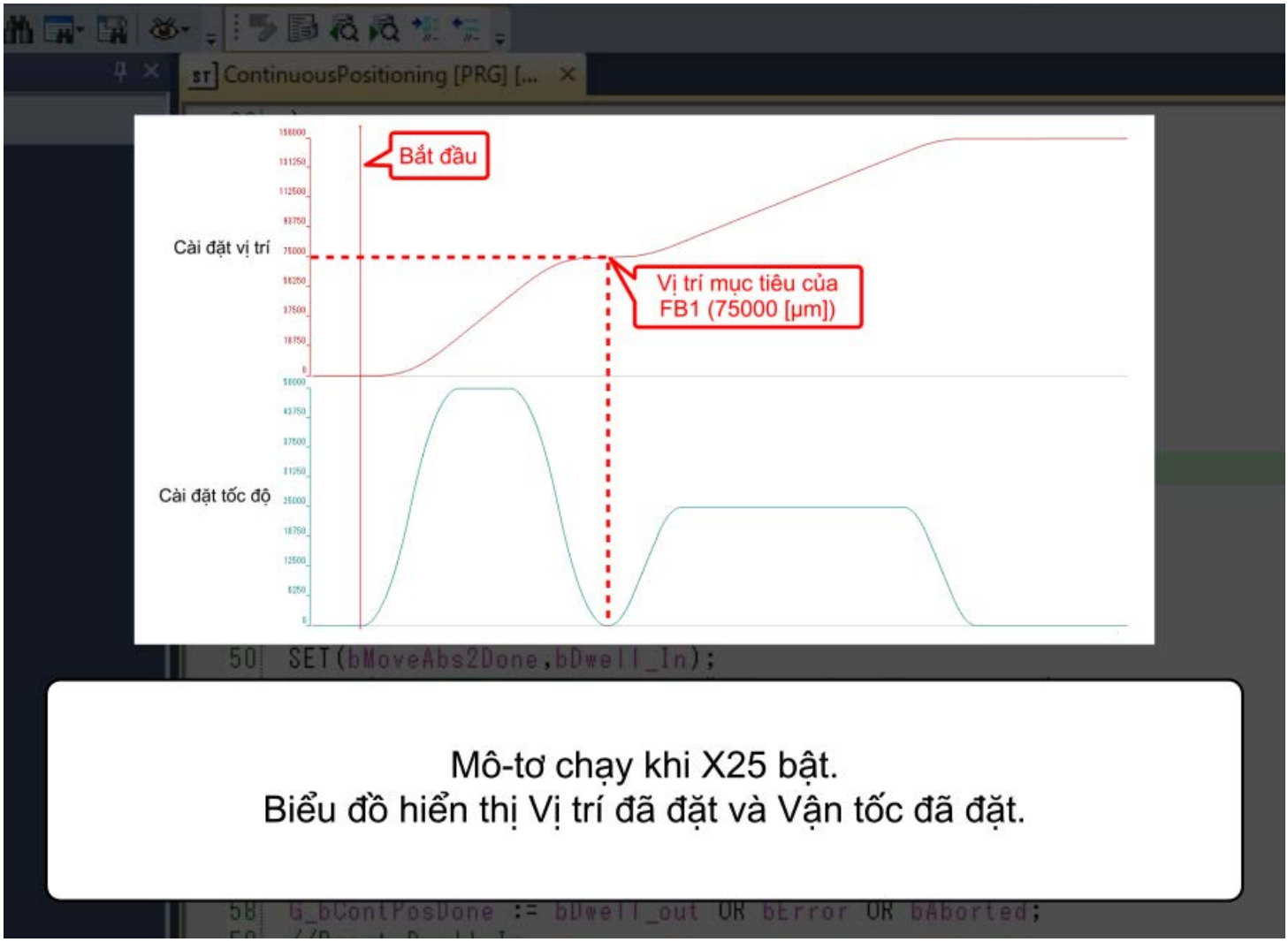



ContinuousPositioning [PRG] [...
);
//Positioning2
MC_MoveAbsolute_2(
Axis := AxisOOO1.AxisRef ,
Execute := bMoveAbs1Active ,
Position := lePosition2 ,
Velocity := lePosVelocity2 ,
Acceleration:= lePosAcceleration2 ,
Deceleration:= lePosDeceleration2 ,
Jerk := lePosJerk2 ,
BufferMode := MC_BUFFER_MODE__mcBuffered,
Done => bMoveAbs2Done ,
CommandAborted => bMoveAbs2Aborted,
Error => bMoveAbs2Error
);
//Dwell
SET(bMoveAbs2Done,bDwell_In);
mcBuffered được đặt trước khi chương trình được tải xuống.

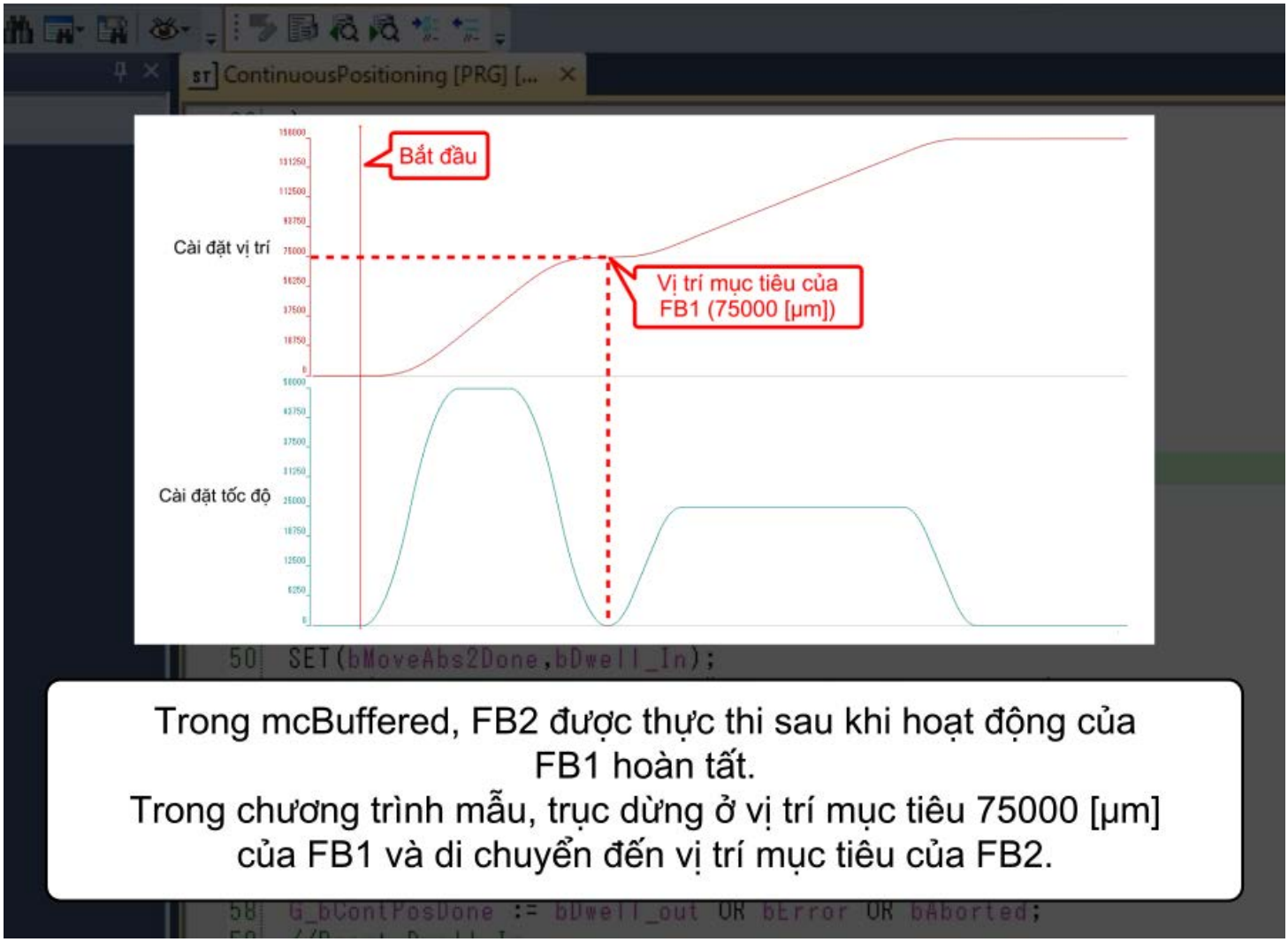
3.5 Kiểm tra hoạt động



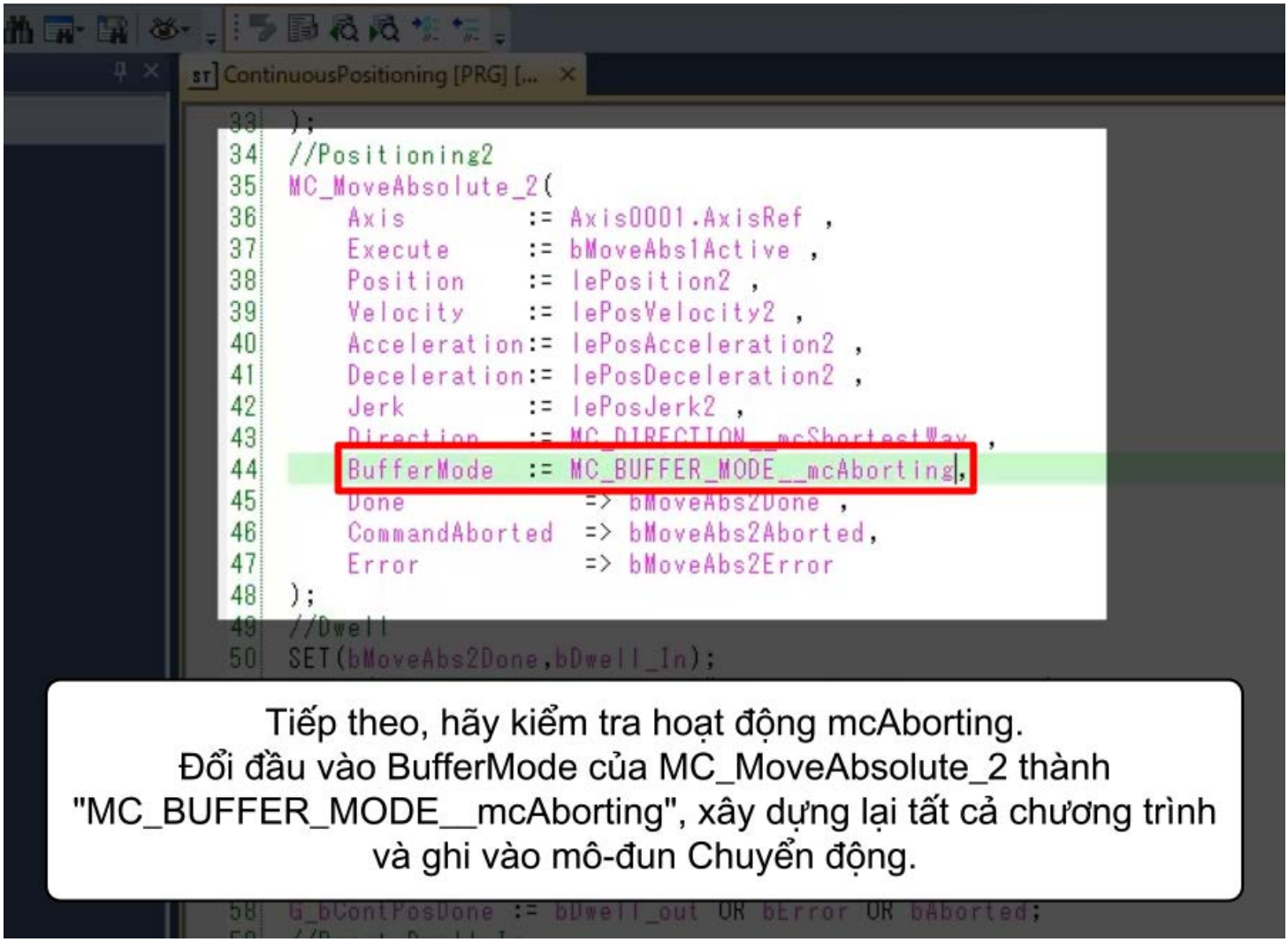
Trước tiên, hãy kiểm tra hoạt động của mcBuffered.
Kiểm tra xem đầu vào BufferMode của MC_MoveAbsolute_2 có được đặt là "MC_BUFFER_MODE__mcBuffered" không và ghi chương trình vào mô-đun Chuyển động.



Mô-tơ chạy khi X25 bật.
Biểu đồ hiển thị Vị trí đã đặt và Vận tốc đã đặt.

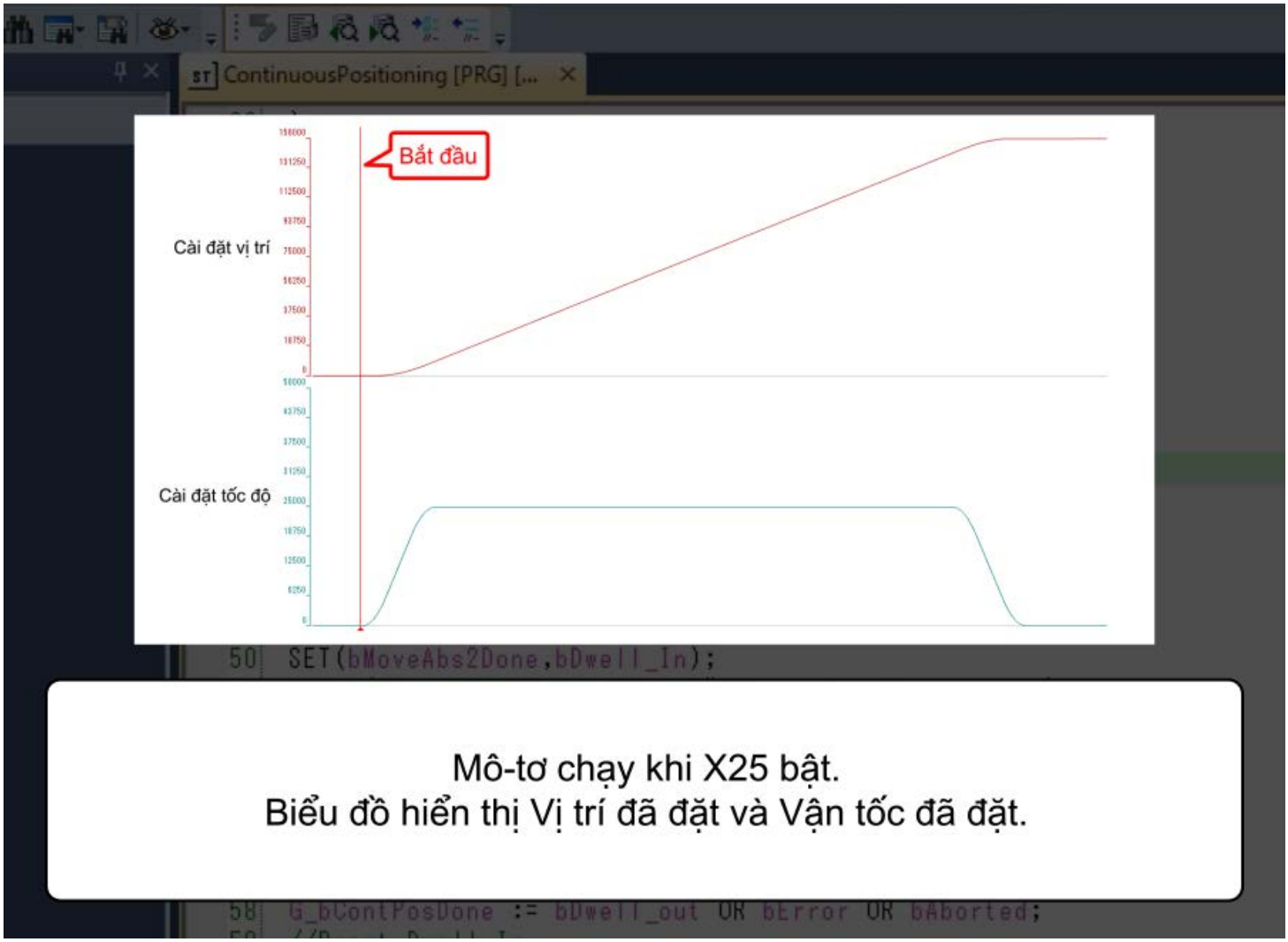


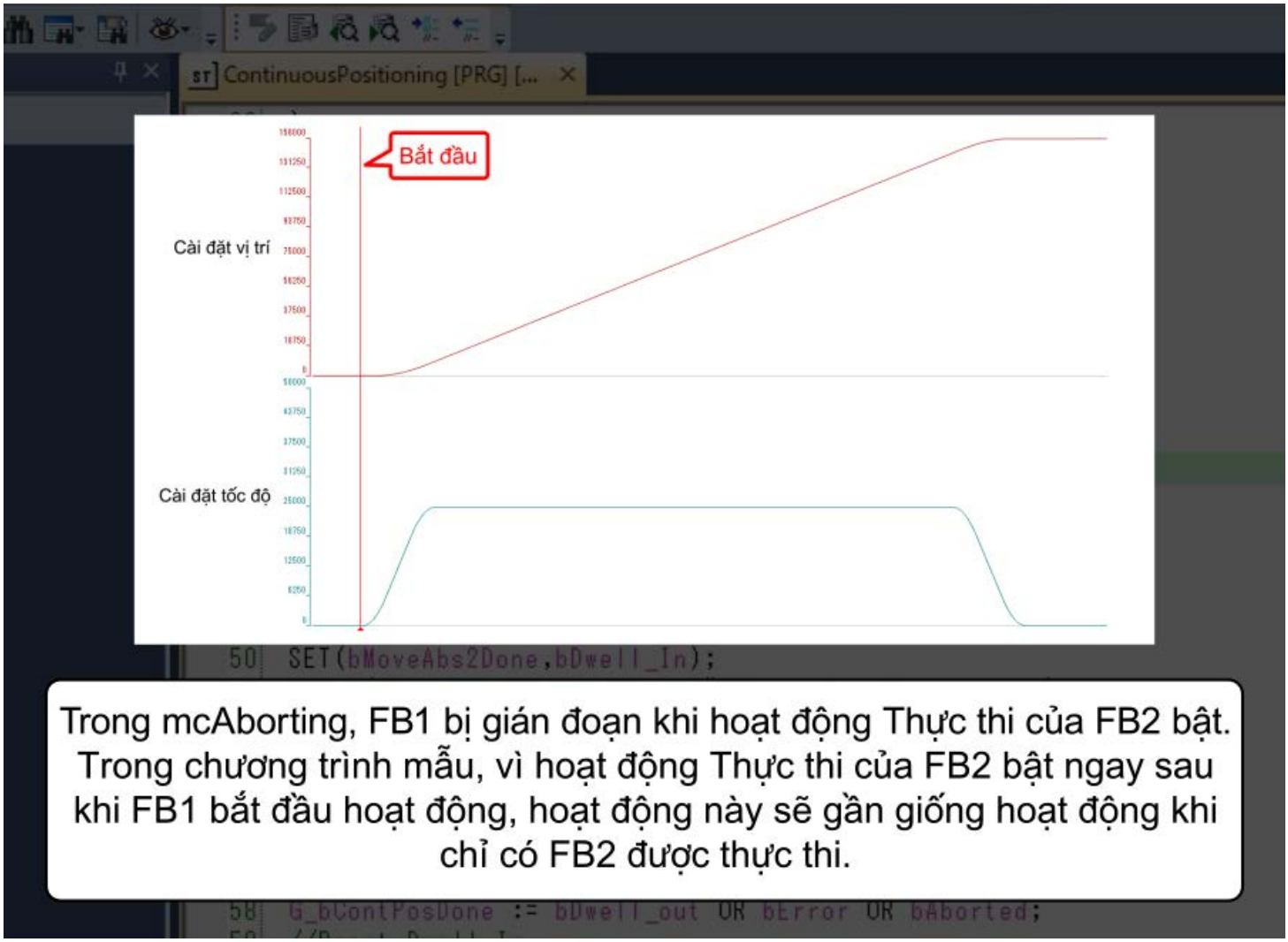
Trong mcBuffered, FB2 được thực thi sau khi hoạt động của FB1 hoàn tất.
Trong chương trình mẫu, trục dừng ở vị trí mục tiêu 75000 [µm] của FB1 và di chuyển đến vị trí mục tiêu của FB2.



Tiếp theo, hãy kiểm tra hoạt động mcAborting.
Đổi đầu vào BufferMode của MC_MoveAbsolute_2 thành "MC_BUFFER_MODE__mcAborting", xây dựng lại tất cả chương trình và ghi vào mô-đun Chuyển động.

3.5 Kiểm tra hoạt động

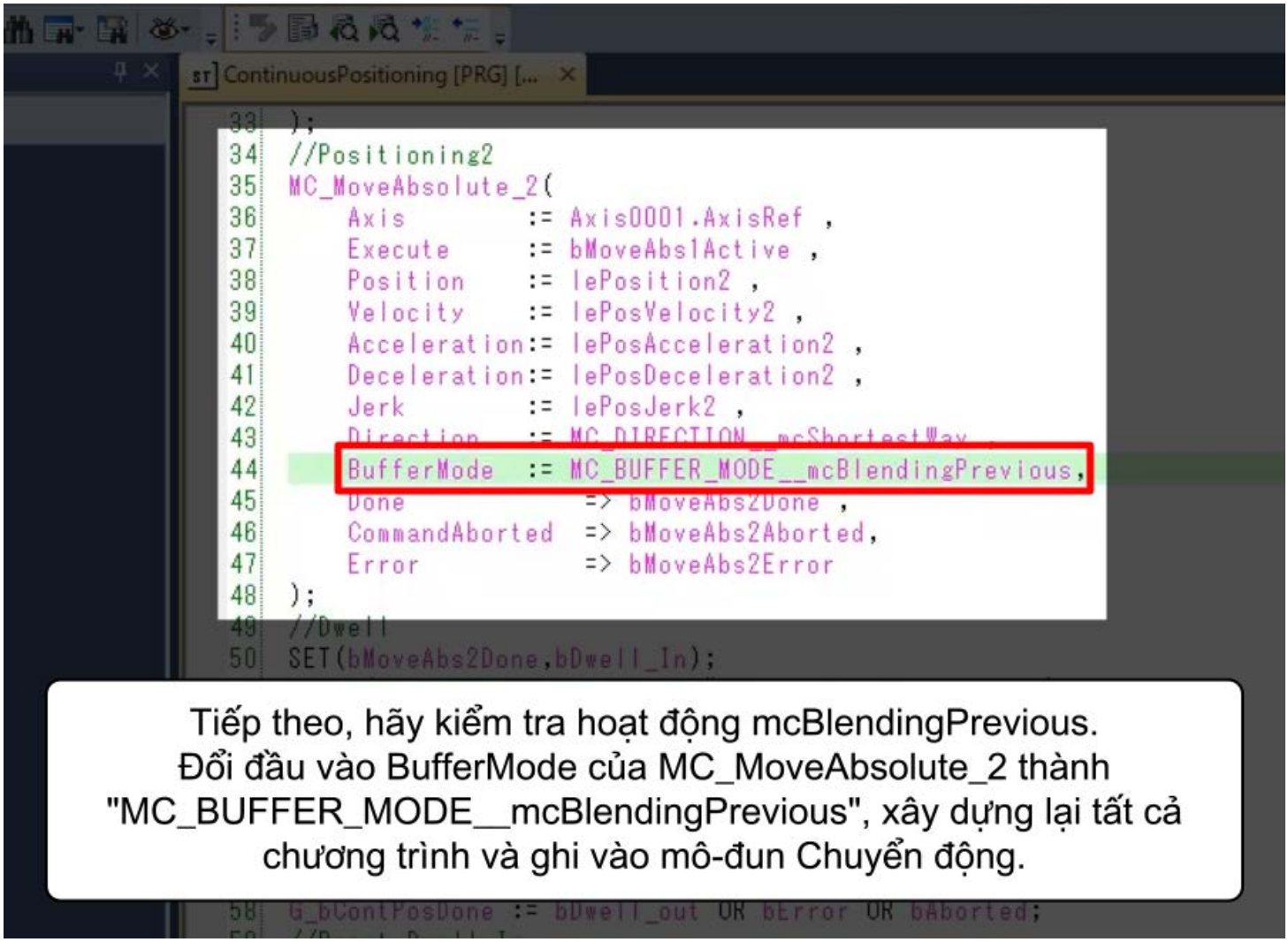




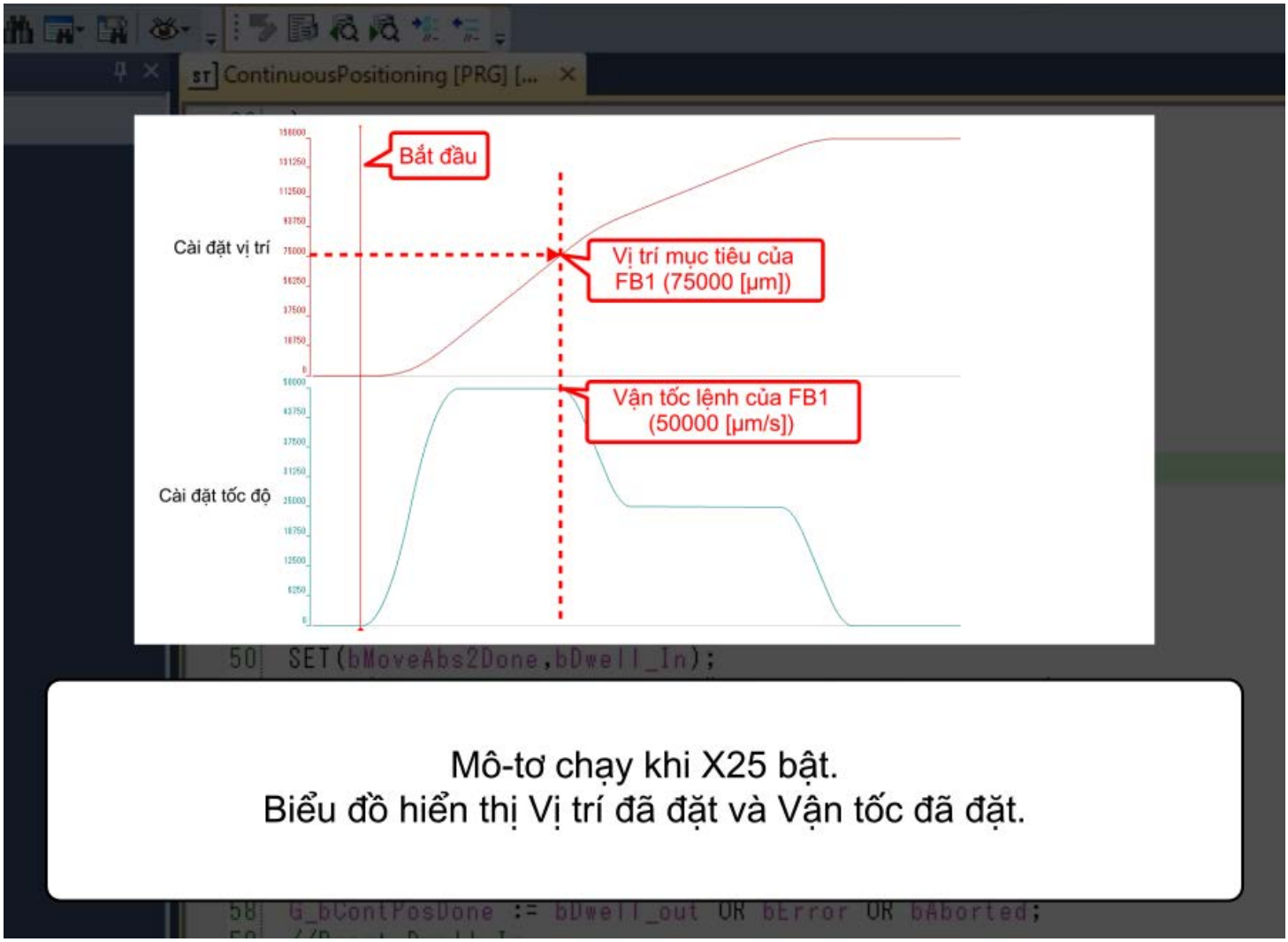
Trong mcAborting, FB1 bị gián đoạn khi hoạt động Thực thi của FB2 bật. Trong chương trình mẫu, vì hoạt động Thực thi của FB2 bật ngay sau khi FB1 bắt đầu hoạt động, hoạt động này sẽ gần giống hoạt động khi chỉ có FB2 được thực thi.

3.5 Kiểm tra hoạt động

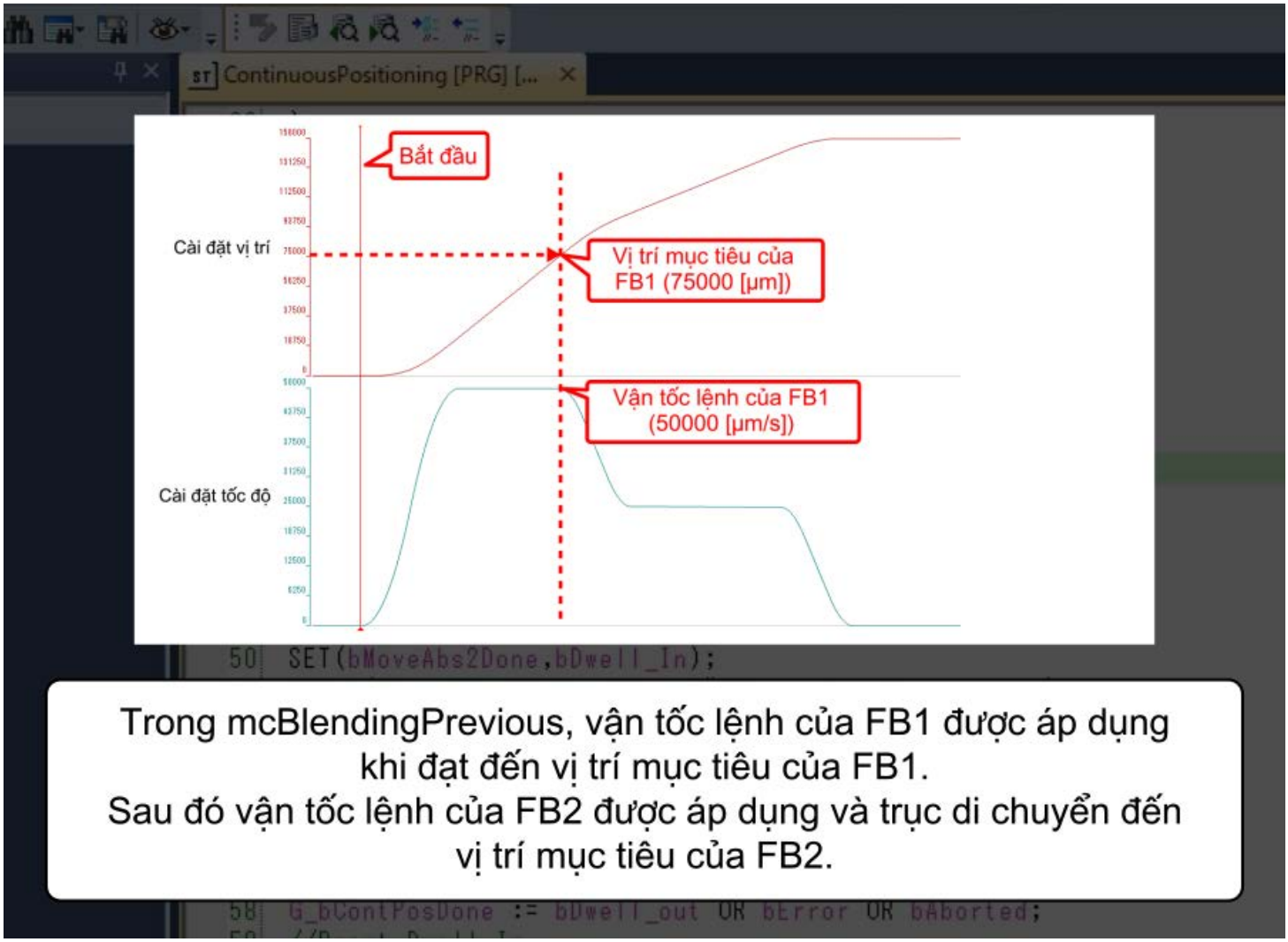
Kiểm tra tín hiệu I/O của FB1 và FB2 tại thời điểm bắt đầu.
Đầu ra CommandAborted của FB1 đã bật, biểu thị FB1 bị gián đoạn.



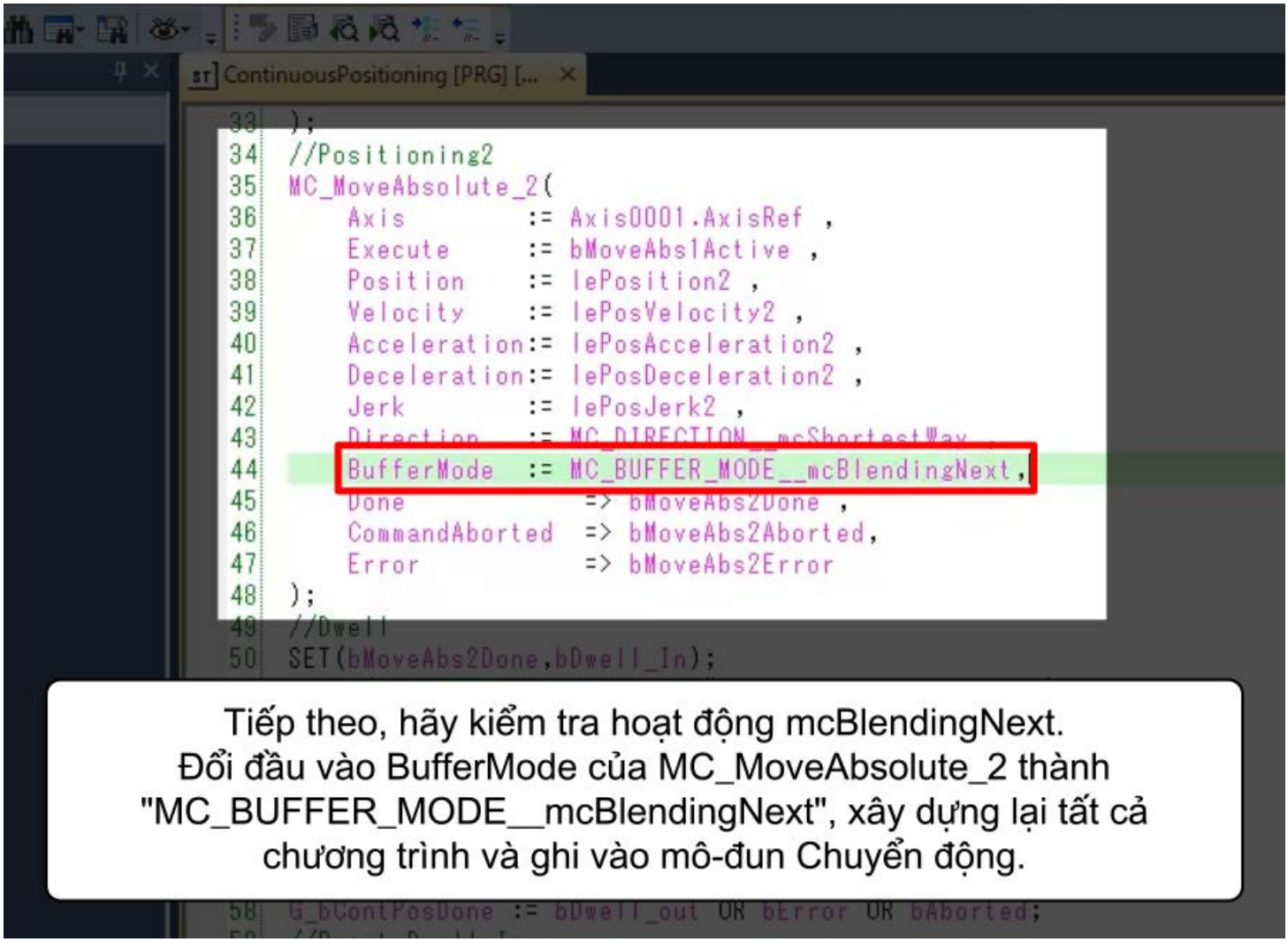
Tiếp theo, hãy kiểm tra hoạt động mcBlendingPrevious. Đổi đầu vào BufferMode của MC_MoveAbsolute_2 thành "MC_BUFFER_MODE__mcBlendingPrevious", xây dựng lại tất cả chương trình và ghi vào mô-đun Chuyển động.



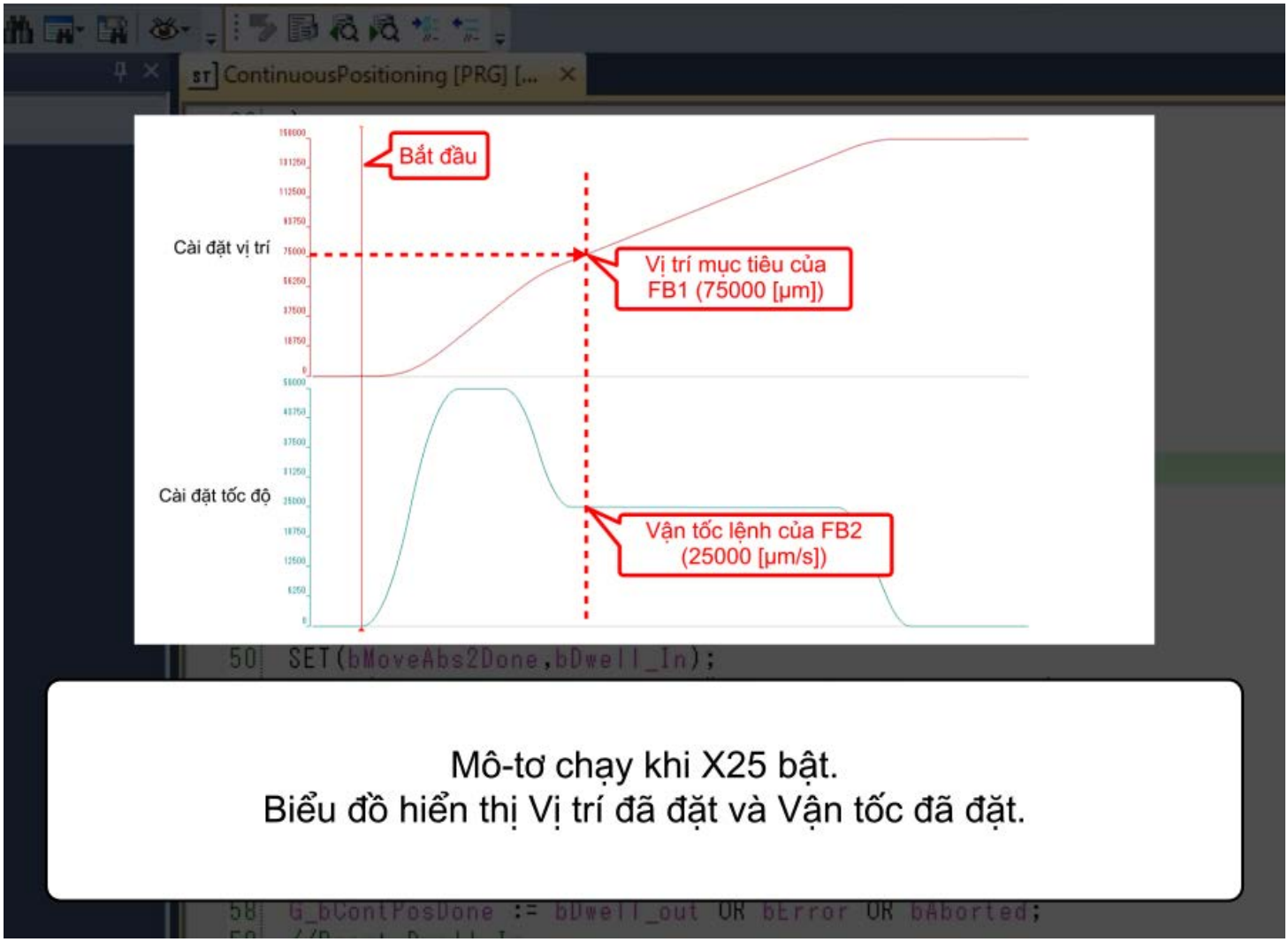
Mô-tơ chạy khi X25 bật.
Biểu đồ hiển thị Vị trí đã đặt và Vận tốc đã đặt.



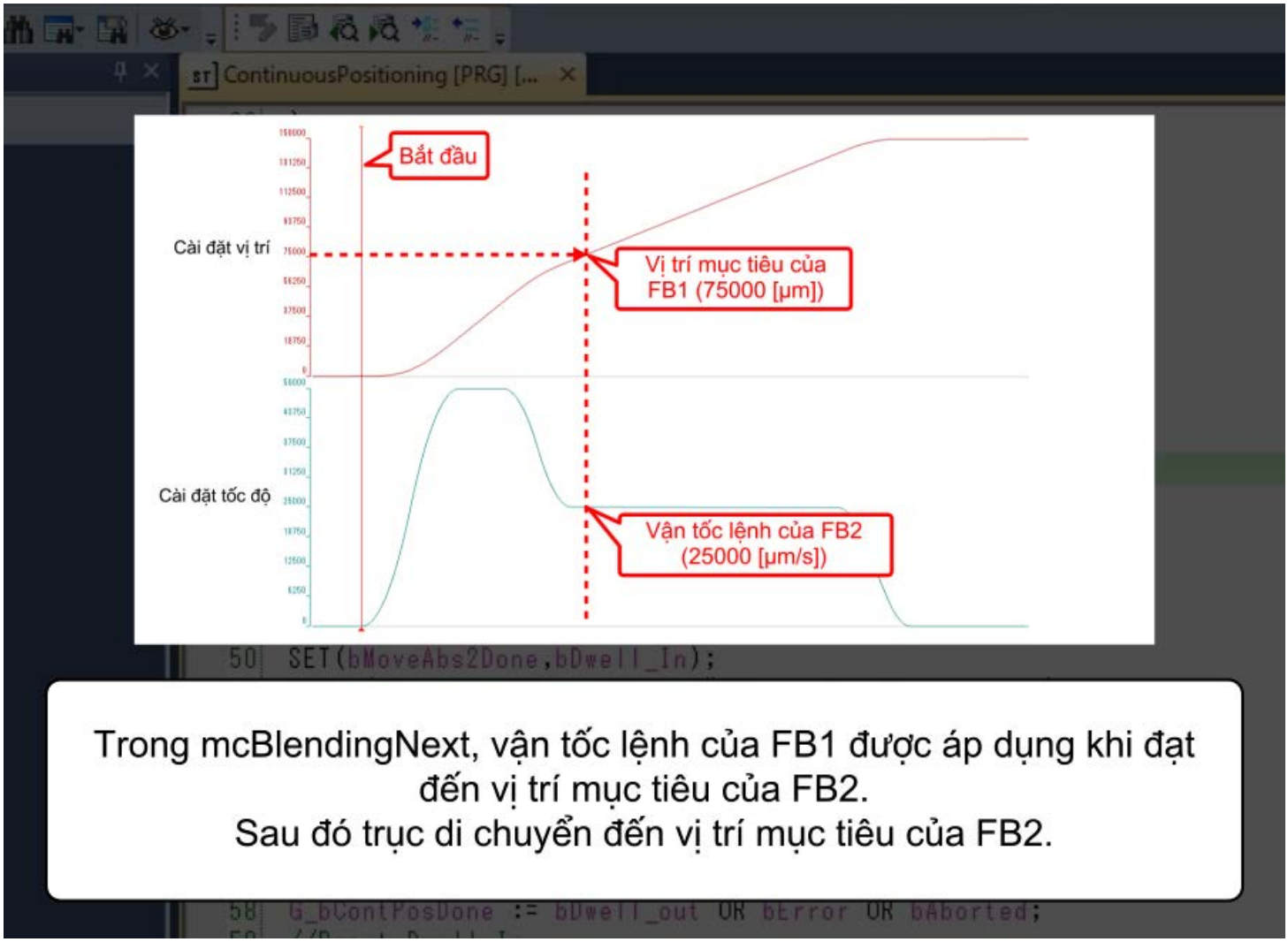
Trong mcBlendingPrevious, vận tốc lệnh của FB1 được áp dụng khi đạt đến vị trí mục tiêu của FB1.
Sau đó vận tốc lệnh của FB2 được áp dụng và trục di chuyển đến vị trí mục tiêu của FB2.



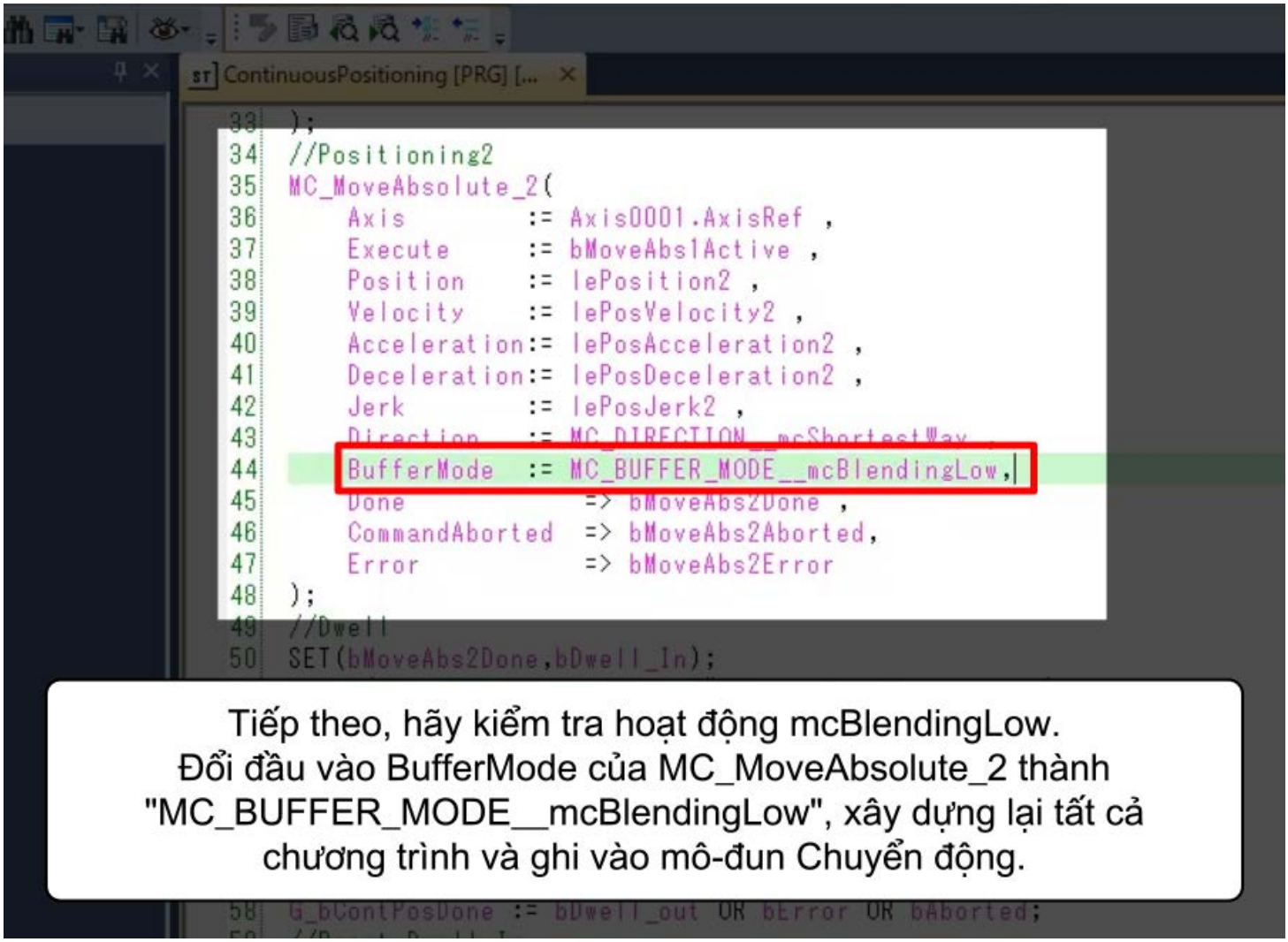
Tiếp theo, hãy kiểm tra hoạt động mcBlendingNext.
Đổi đầu vào BufferMode của MC_MoveAbsolute_2 thành "MC_BUFFER_MODE__mcBlendingNext", xây dựng lại tất cả chương trình và ghi vào mô-đun Chuyển động.



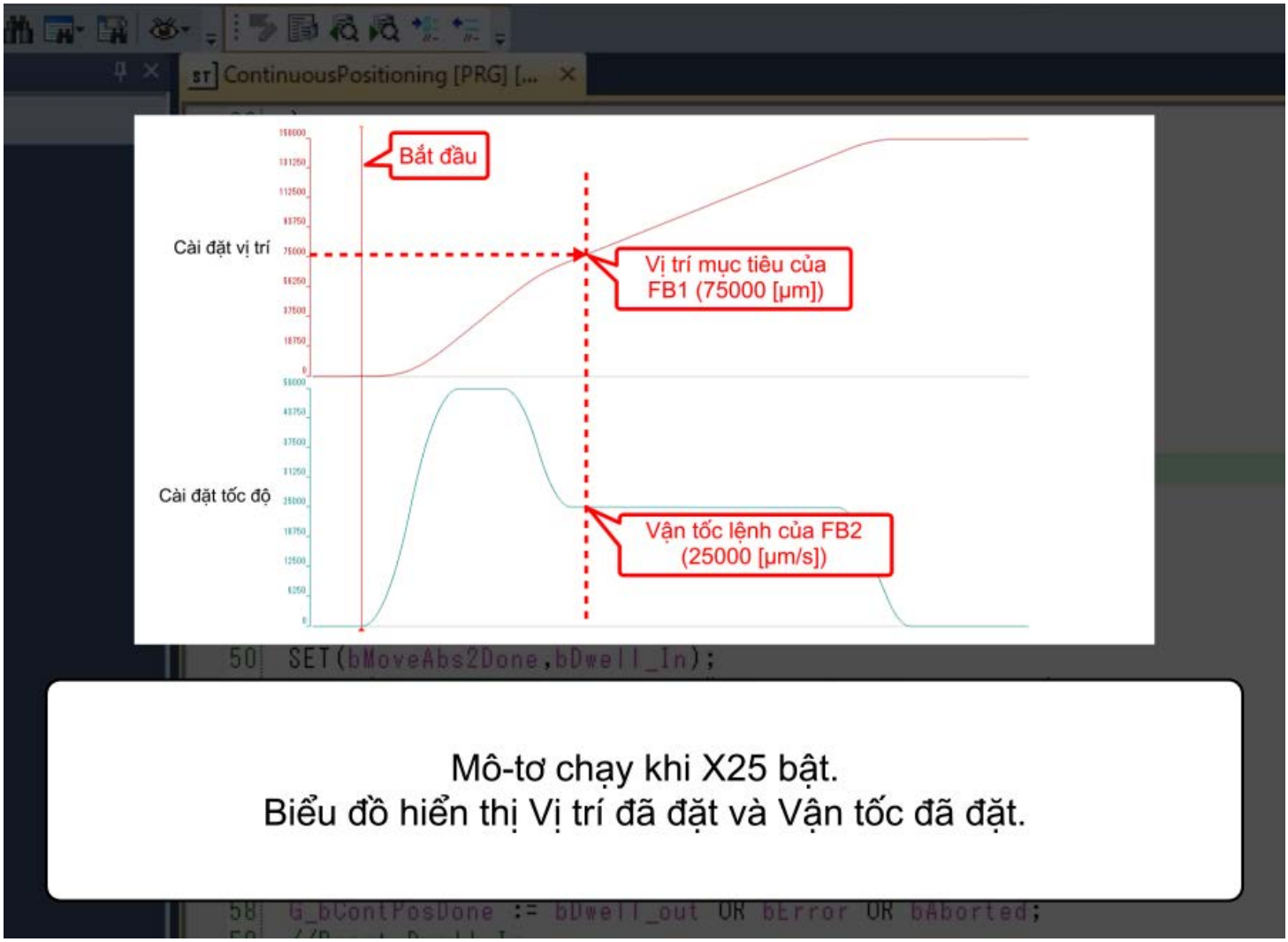
Mô-tơ chạy khi X25 bật.
Biểu đồ hiển thị Vị trí đã đặt và Vận tốc đã đặt.



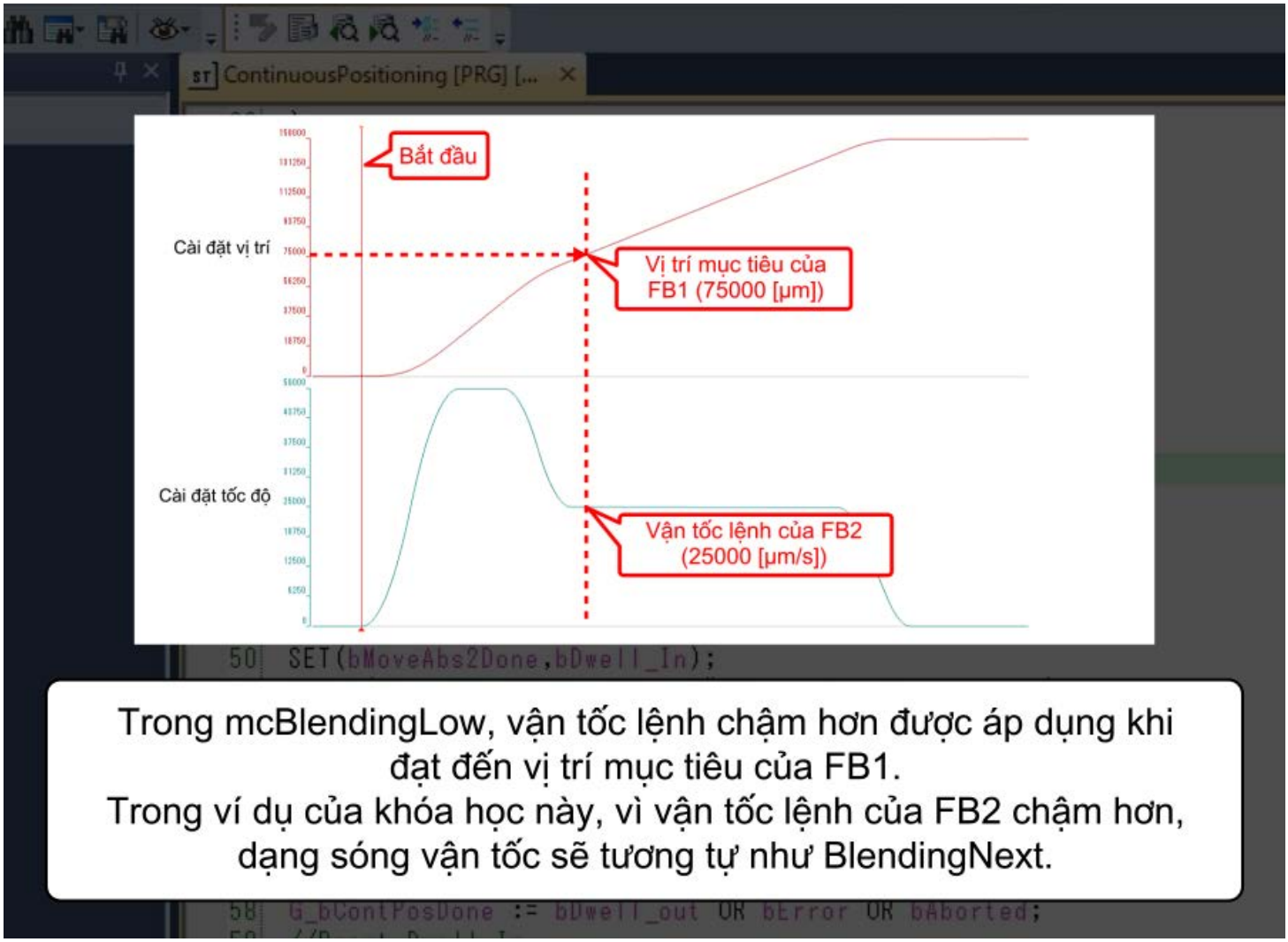
Trong mcBlendingNext, vận tốc lệnh của FB1 được áp dụng khi đạt đến vị trí mục tiêu của FB2.
Sau đó trục di chuyển đến vị trí mục tiêu của FB2.



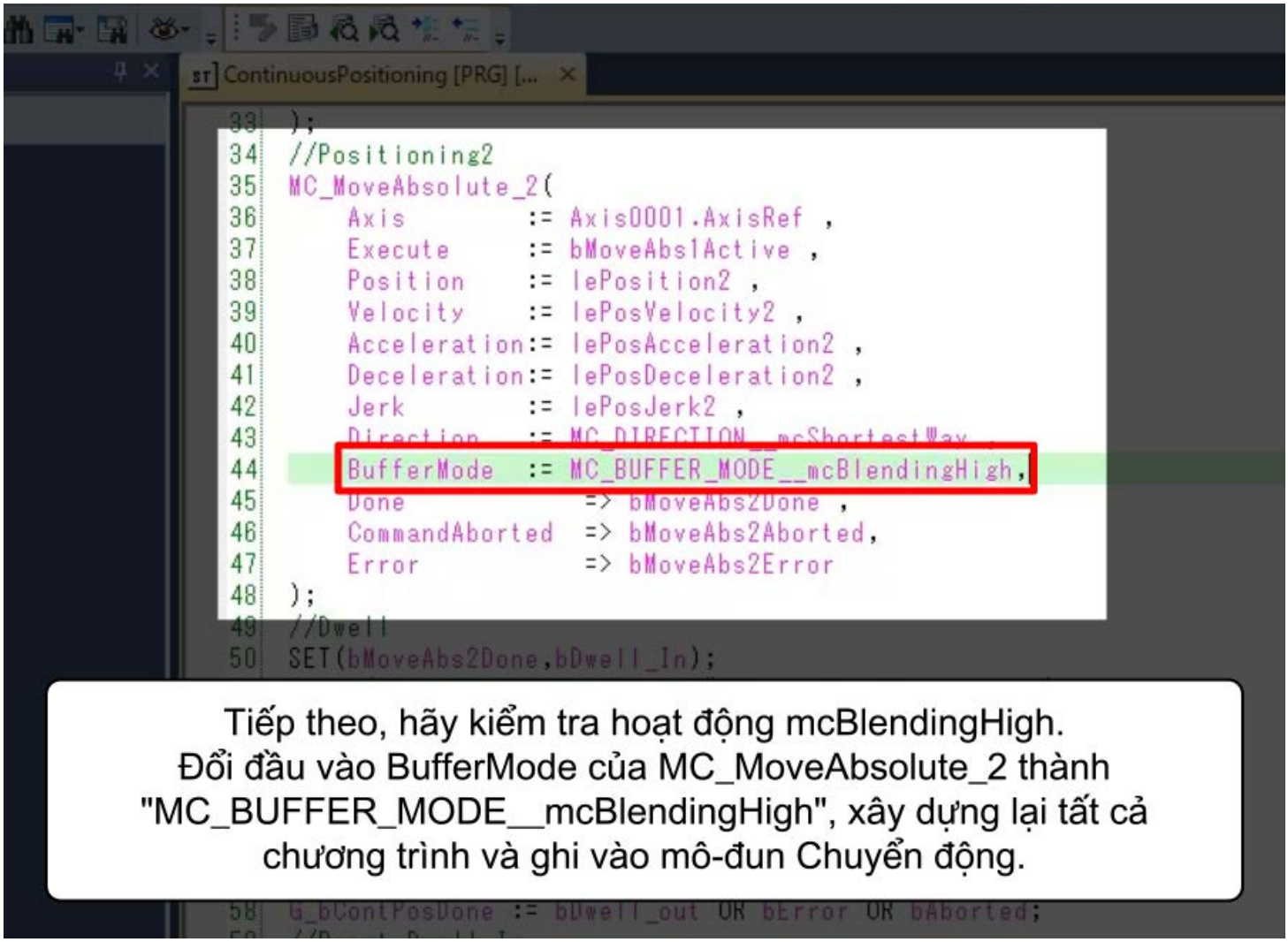
Tiếp theo, hãy kiểm tra hoạt động mcBlendingLow.
Đổi đầu vào BufferMode của MC_MoveAbsolute_2 thành "MC_BUFFER_MODE__mcBlendingLow", xây dựng lại tất cả chương trình và ghi vào mô-đun Chuyển động.



Mô-tơ chạy khi X25 bật.
Biểu đồ hiển thị Vị trí đã đặt và Vận tốc đã đặt.

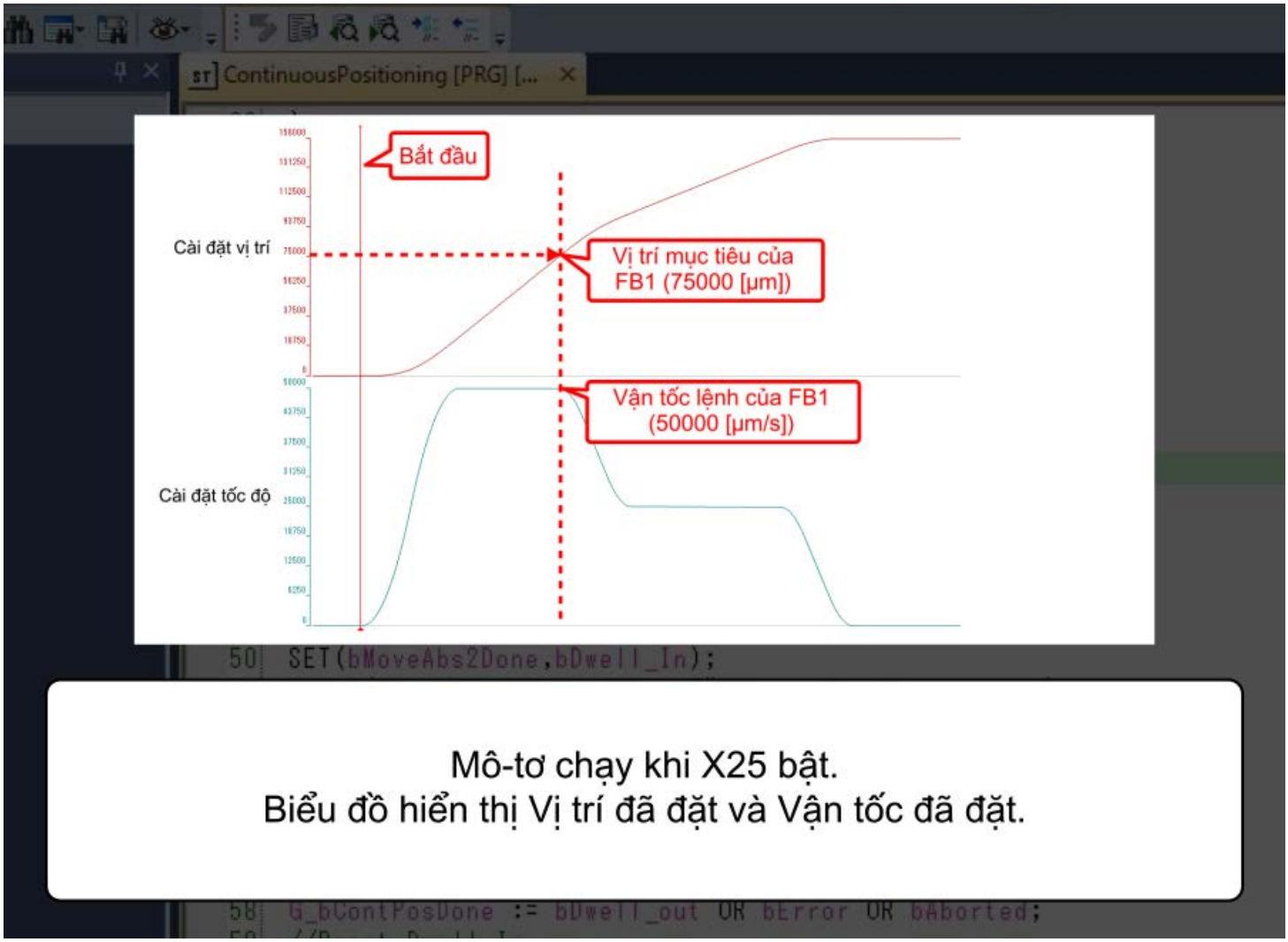


Trong mcBlendingLow, vận tốc lệnh chậm hơn được áp dụng khi đạt đến vị trí mục tiêu của FB1.
Trong ví dụ của khóa học này, vì vận tốc lệnh của FB2 chậm hơn, dạng sóng vận tốc sẽ tương tự như BlendingNext.

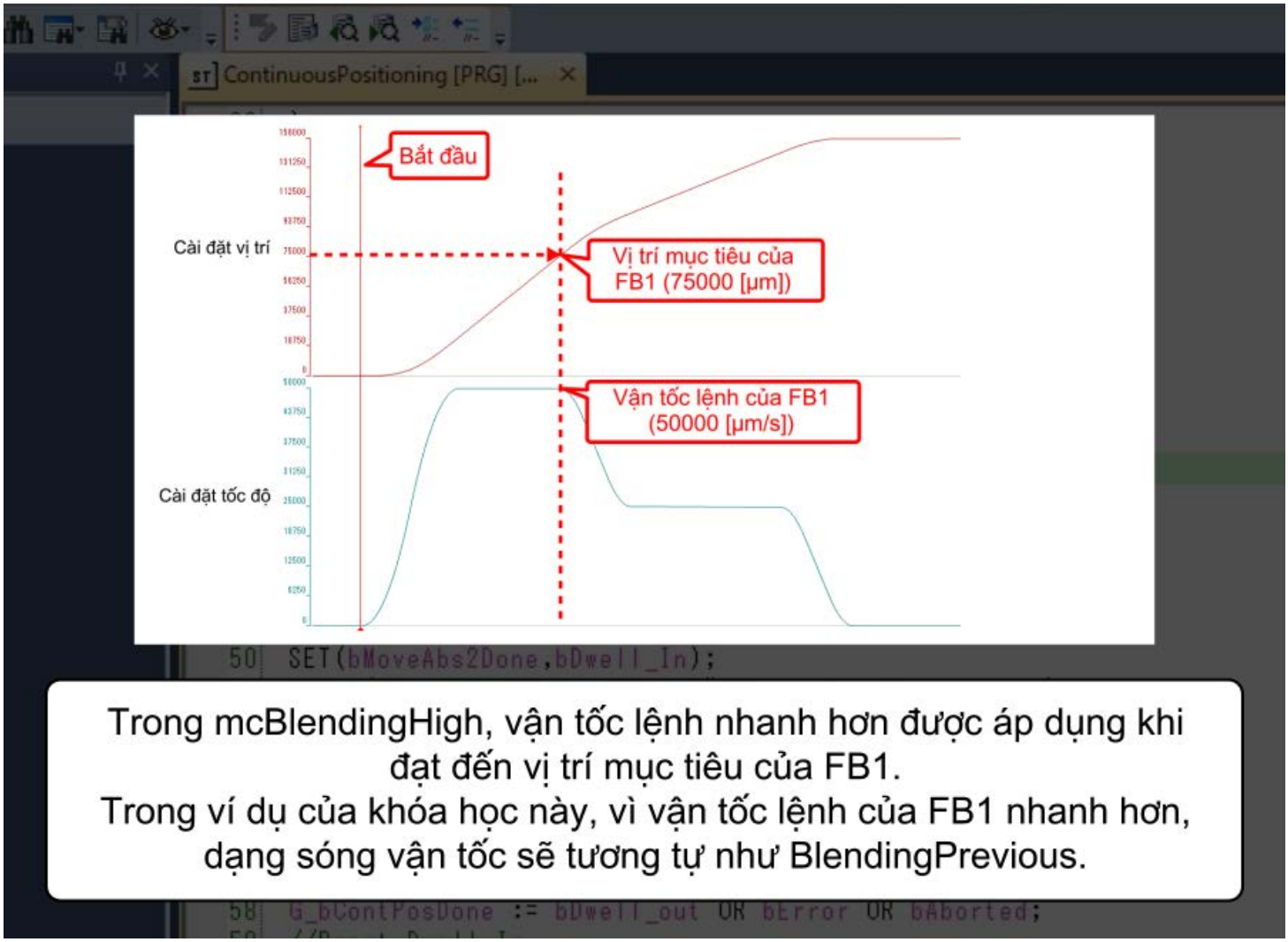


Tiếp theo, hãy kiểm tra hoạt động mcBlendingHigh.
Đổi đầu vào BufferMode của MC_MoveAbsolute_2 thành "MC_BUFFER_MODE__mcBlendingHigh", xây dựng lại tất cả chương trình và ghi vào mô-đun Chuyển động.

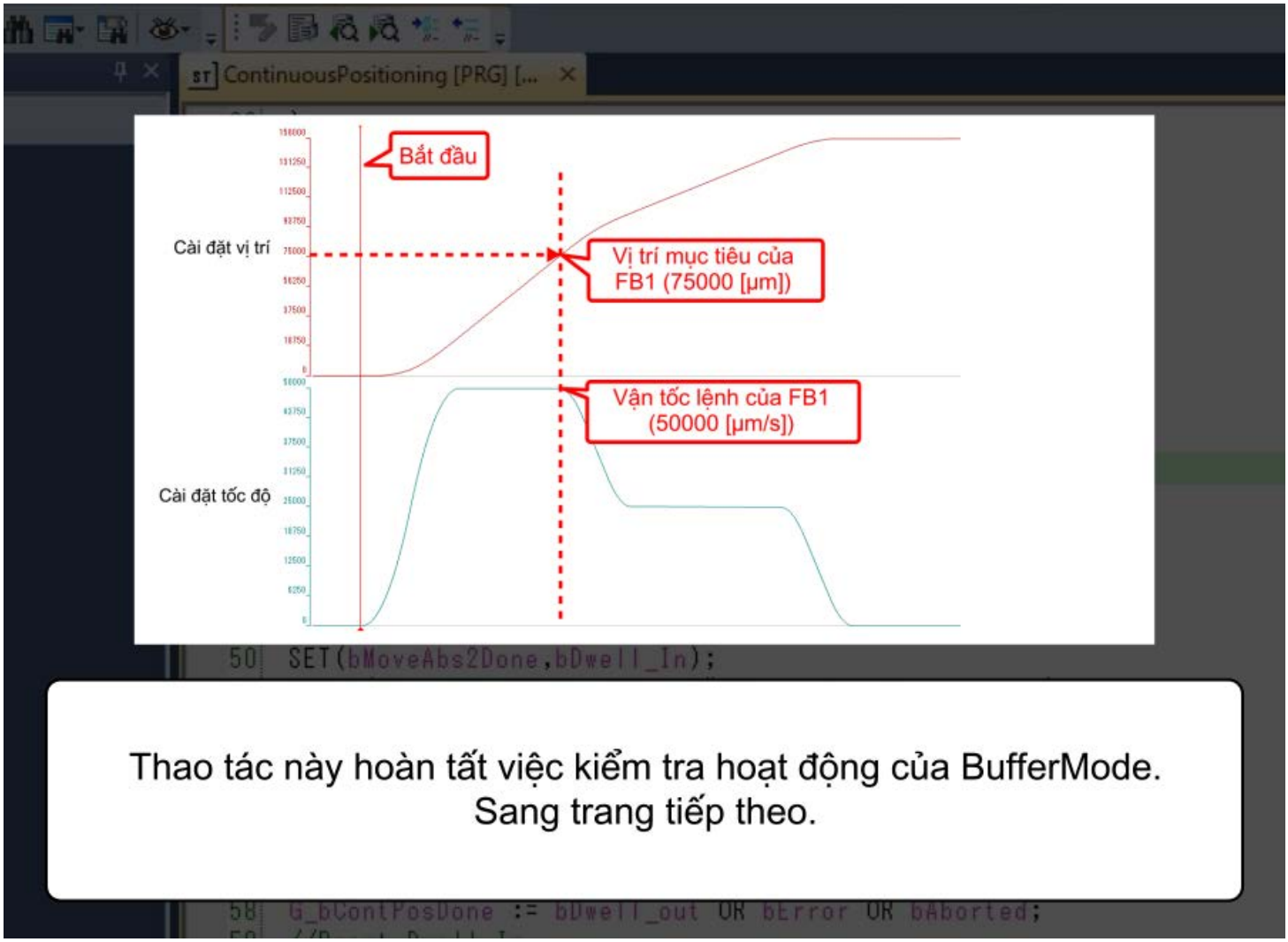
3.5 Kiểm tra hoạt động



Mô-tơ chạy khi X25 bật.
Biểu đồ hiển thị Vị trí đã đặt và Vận tốc đã đặt.



Trong mcBlendingHigh, vận tốc lệnh nhanh hơn được áp dụng khi đạt đến vị trí mục tiêu của FB1.
Trong ví dụ của khóa học này, vì vận tốc lệnh của FB1 nhanh hơn, dạng sóng vận tốc sẽ tương tự như BlendingPrevious.


ContinuousPositioning [PRG] [...
Bắt đầu
Cài đặt vị trí
Vị trí mục tiêu của FB1 (75000 [µm])
Vận tốc lệnh của FB1 (50000 [µm/s])
Cài đặt tốc độ
50 SET(bMoveAbs2Done,bDwell_In);
Thao tác này hoàn tất việc kiểm tra hoạt động của BufferMode.
Sang trang tiếp theo.
58 G_bContPosDone := bDwell_out OR bError OR bAborted;

3.6 Tóm tắt chương này

Trong chương này, bạn đã học về:

- Hủy bỏ
- Đệm
- Phối hợp
- Ví dụ về chương trình
- Kiểm tra hoạt động

Những điểm quan trọng

Hủy bỏ	• Khi FB loại hoạt động đang chạy và FB loại hoạt động tiếp theo được thực thi, quy trình Hủy bỏ sẽ ngưng FB đang được thực thi và thực thi FB tiếp theo.
Đệm	• Khi FB loại hoạt động đang chạy và FB loại hoạt động tiếp theo được thực thi, quy trình Đệm sẽ đợi cho đến khi FB đang được thực thi hoàn thành và thực thi FB tiếp theo.
Phối hợp	• Khi FB loại hoạt động đang chạy và FB loại hoạt động tiếp theo được thực thi, quy trình Phối hợp sẽ thực thi FB tiếp theo mà không dừng hoạt động của FB đang được thực thi. • Có bốn phương pháp chuyển đổi vận tốc trong quy trình Phối hợp: BlendingLow, BlendingHigh, BlendingPrevious và BlendingNext.
Ví dụ về chương trình	• Chọn chế độ đệm với đầu vào BufferMode của FB hoạt động.
Kiểm tra hoạt động	• Bạn đã kiểm tra sự khác biệt trong hoạt động của từng chế độ đệm trong video.

Chương 4 Vận hành với PLC CPU

Tải xuống chương trình mẫu sẽ được sử dụng trong chương này bằng liên kết bên dưới.
Nội dung chương trình này giống với chương trình mẫu được mô tả ở chương 2 và chương 3.
Chỉ có phương pháp lập trình là khác nhau.

RD78GBasic2_sample2.zip (1.39 MB)

4.1 Đăng ký Thư viện FB mô-đun chuyển động

(1) Tải xuống thư viện FB
Motion control FB có thể được sử dụng trong chương trình của PLC CPU bằng cách đăng ký thư viện FB cho mô-đun Chuyển động với GX Works3.
Tải xuống thư viện FB bằng liên kết bên dưới và giải nén tệp ZIP đến đích mong muốn.

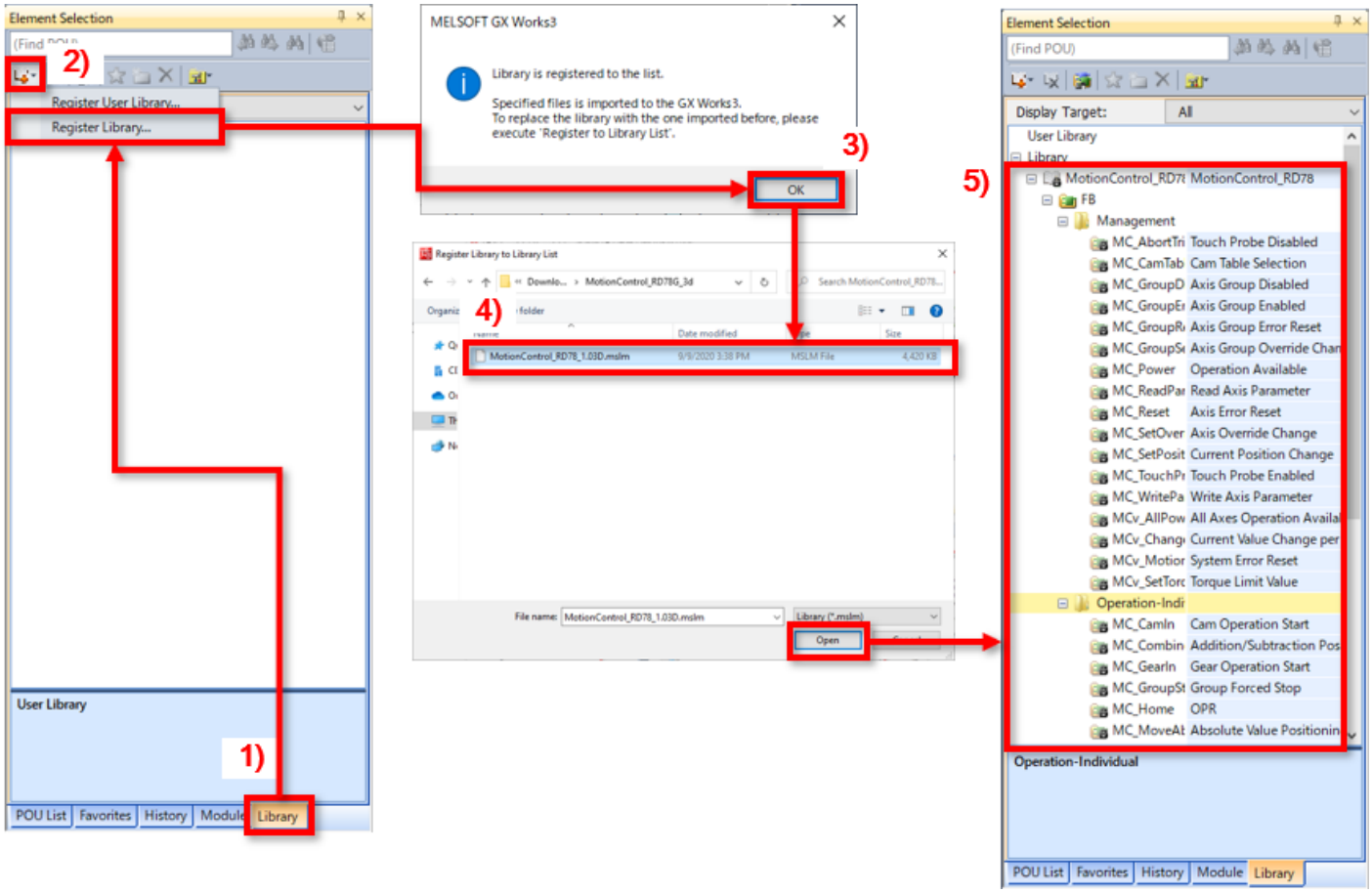
MotionControl_RD78G_3d.zip(4.29 MB)

(Lưu ý) Bạn có thể tải xuống phiên bản mới nhất của thư viện FB từ Trang web toàn cầu của MITSUBISHI ELECTRIC FA.

4.1 Đăng ký Thư viện FB mô-đun chuyển động

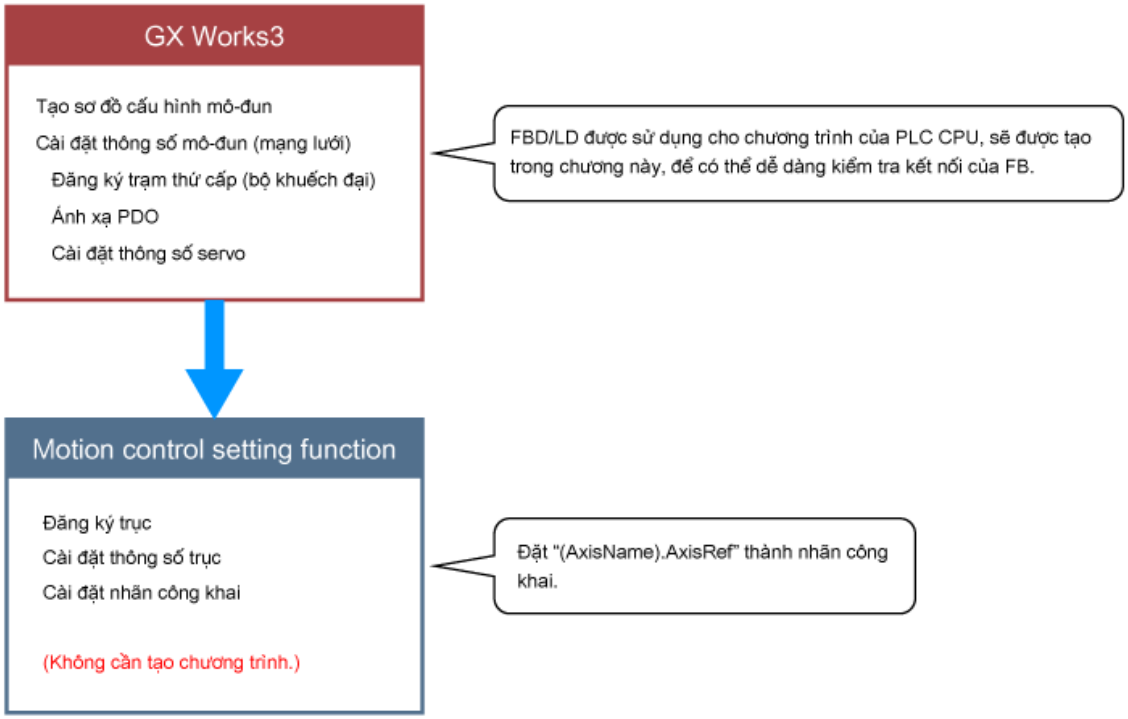
(2) Đăng ký thư viện FB

1) Mở dự án bất kỳ trong GX Works3 và mở thẻ Thư viện trong cửa sổ Element Selection.
2) Nhấp vào nút [Register to Library List] ở phần bên trên và chọn [Register Library].
3) Khi thông báo "Thư viện đã được đăng ký vào danh sách" hiện lên, hãy nhấp vào [OK].
4) Chọn tệp thư viện FB "MotionControl_RD78_****.mslm" và nhấp vào [Open].
   (**** biểu thị phiên bản.)
5) Motion control FB được đăng ký vào thư viện trong cửa sổ Chọn yếu tố.

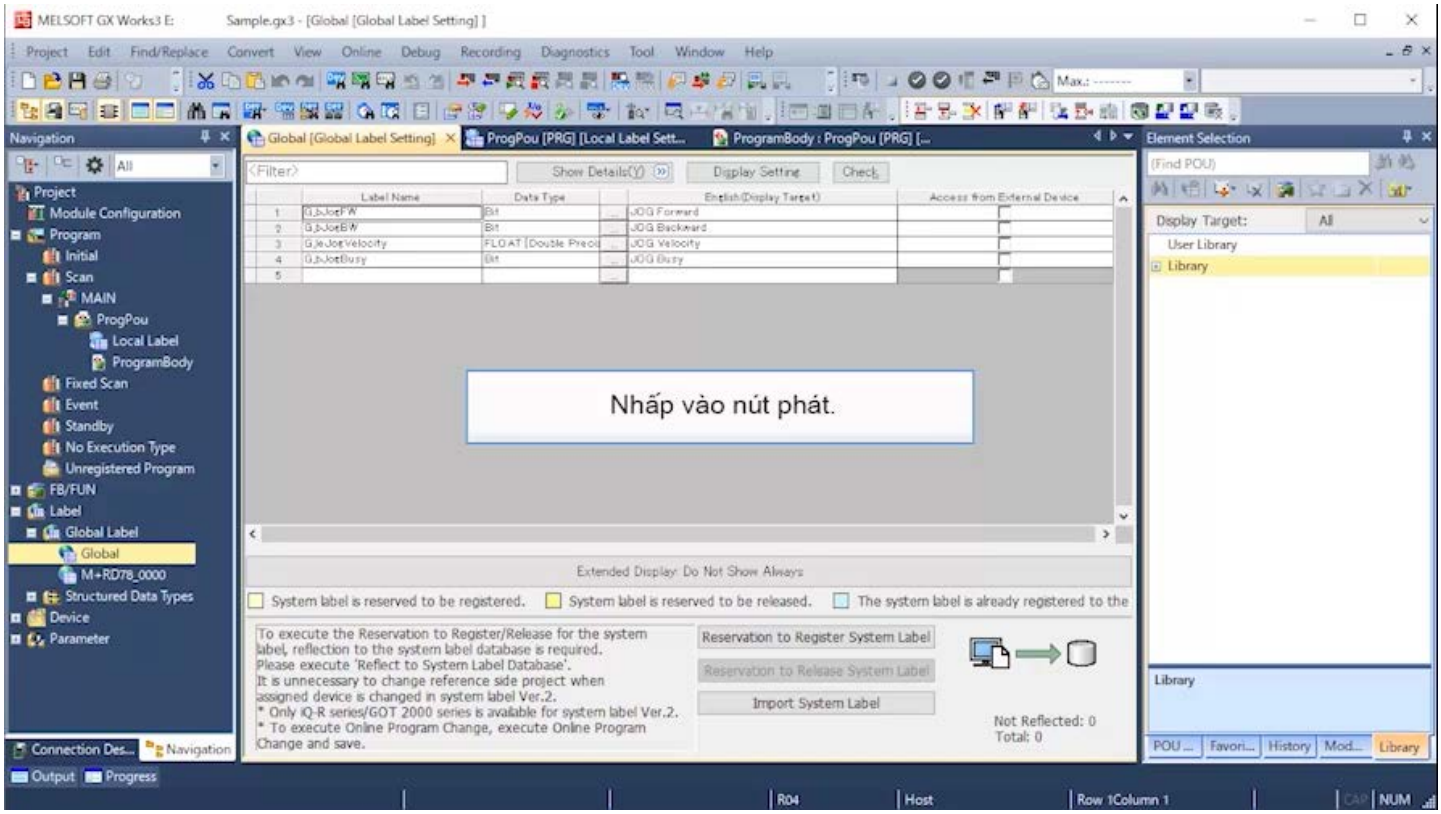


4.2 Tạo dự án

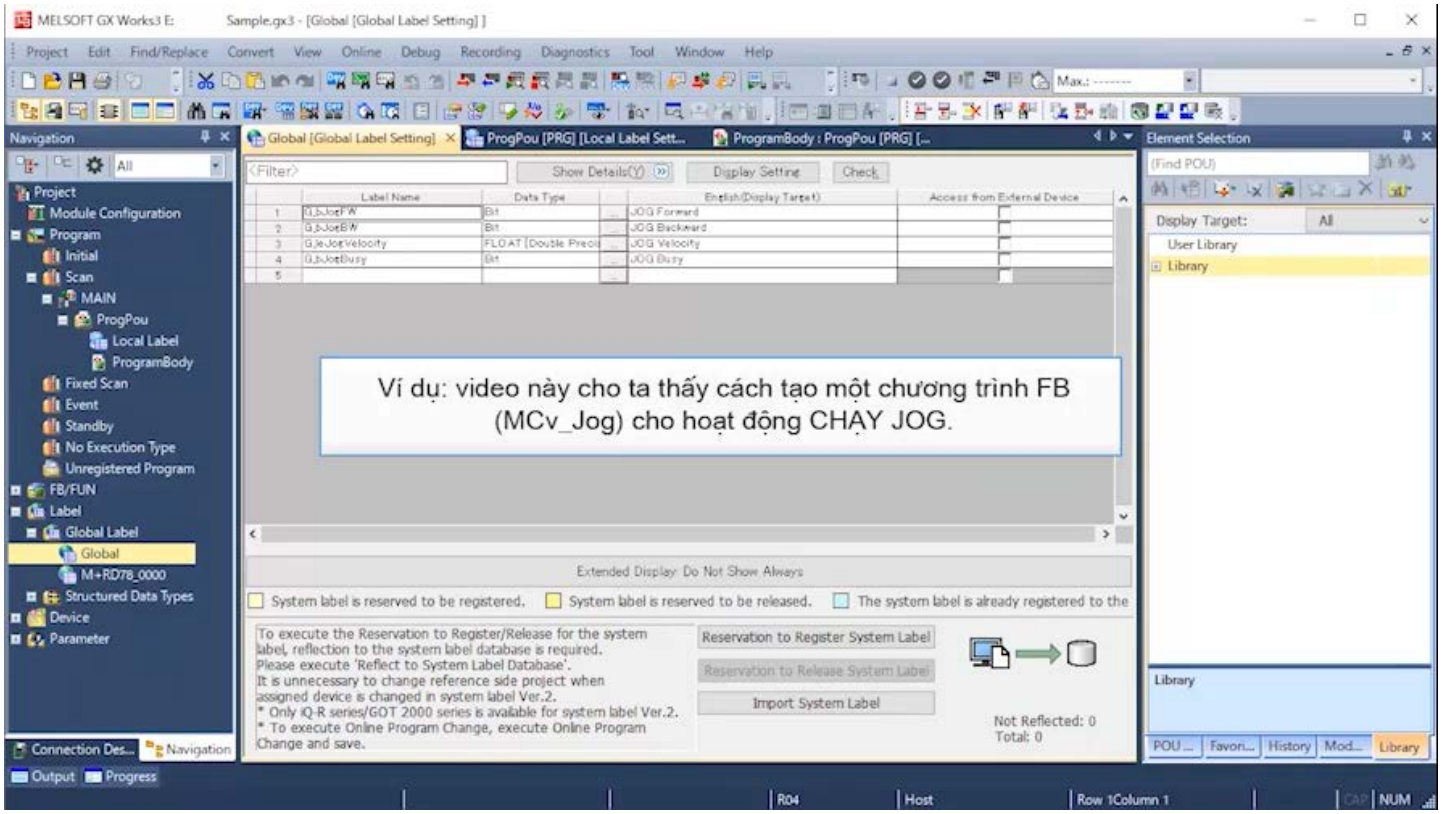
Quy trình tạo dự án giống với mô tả ở phần trước.



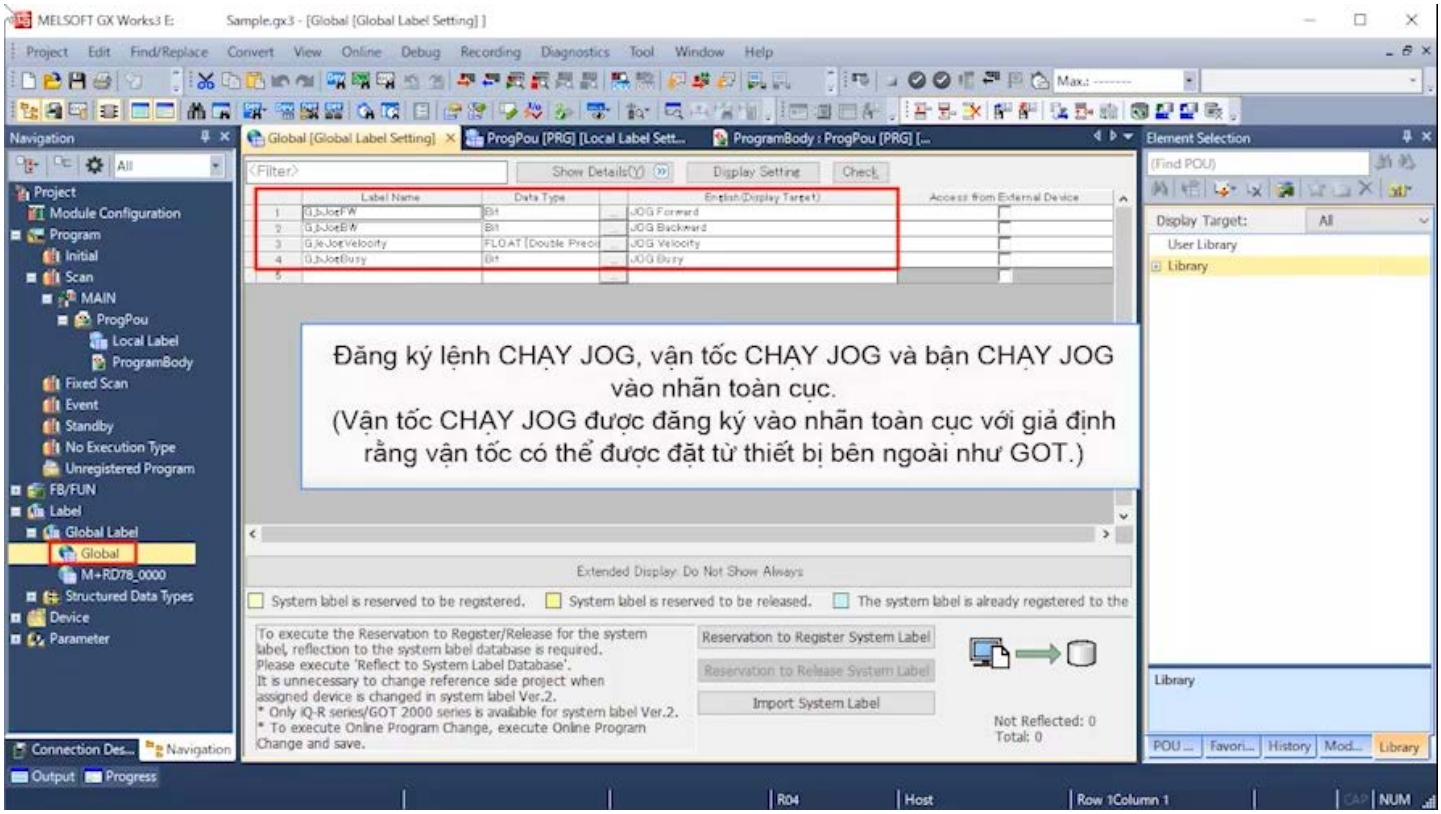
4.3 Cách sử dụng Motion Control FB



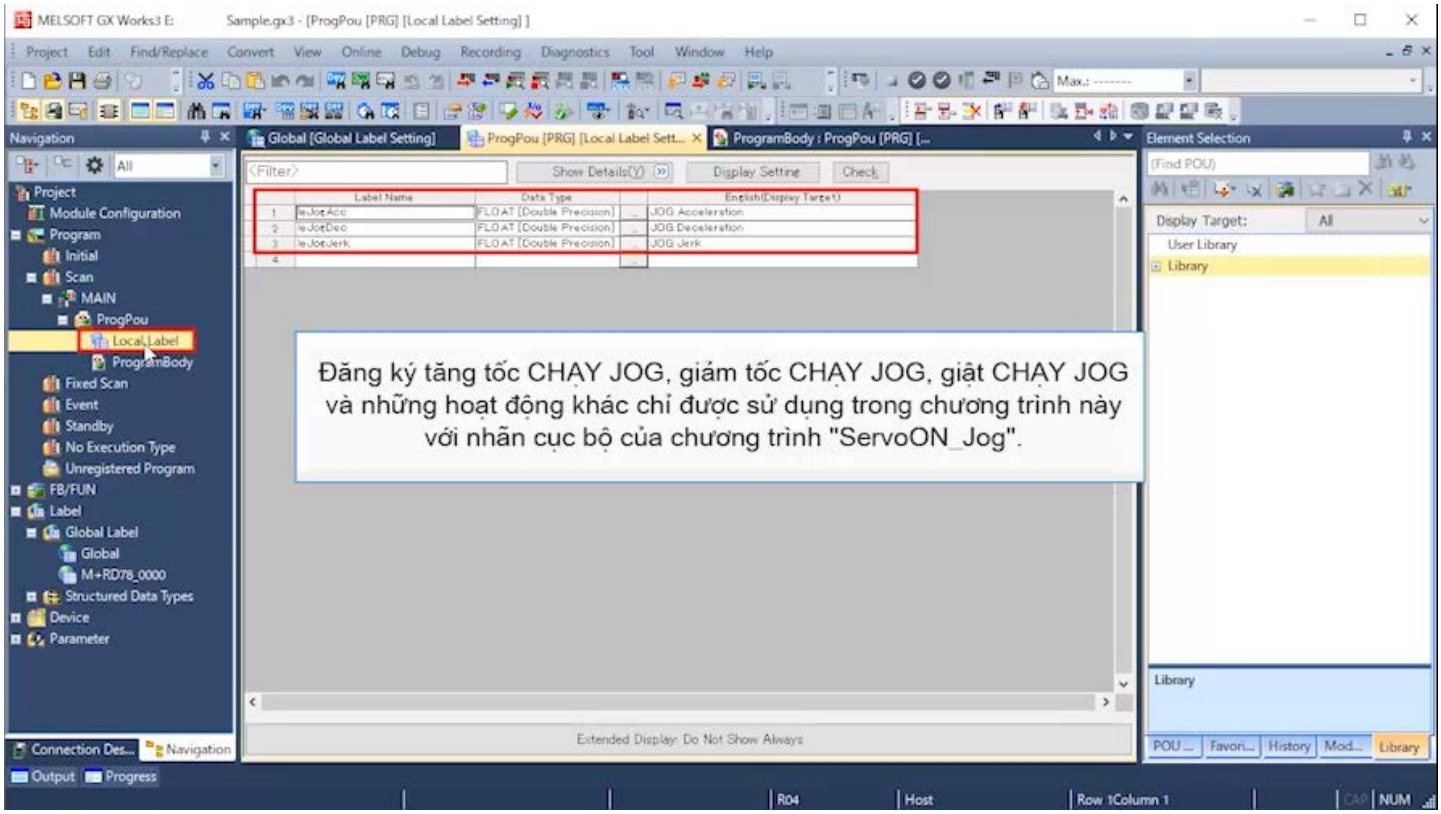
4.3 Cách sử dụng Motion Control FB



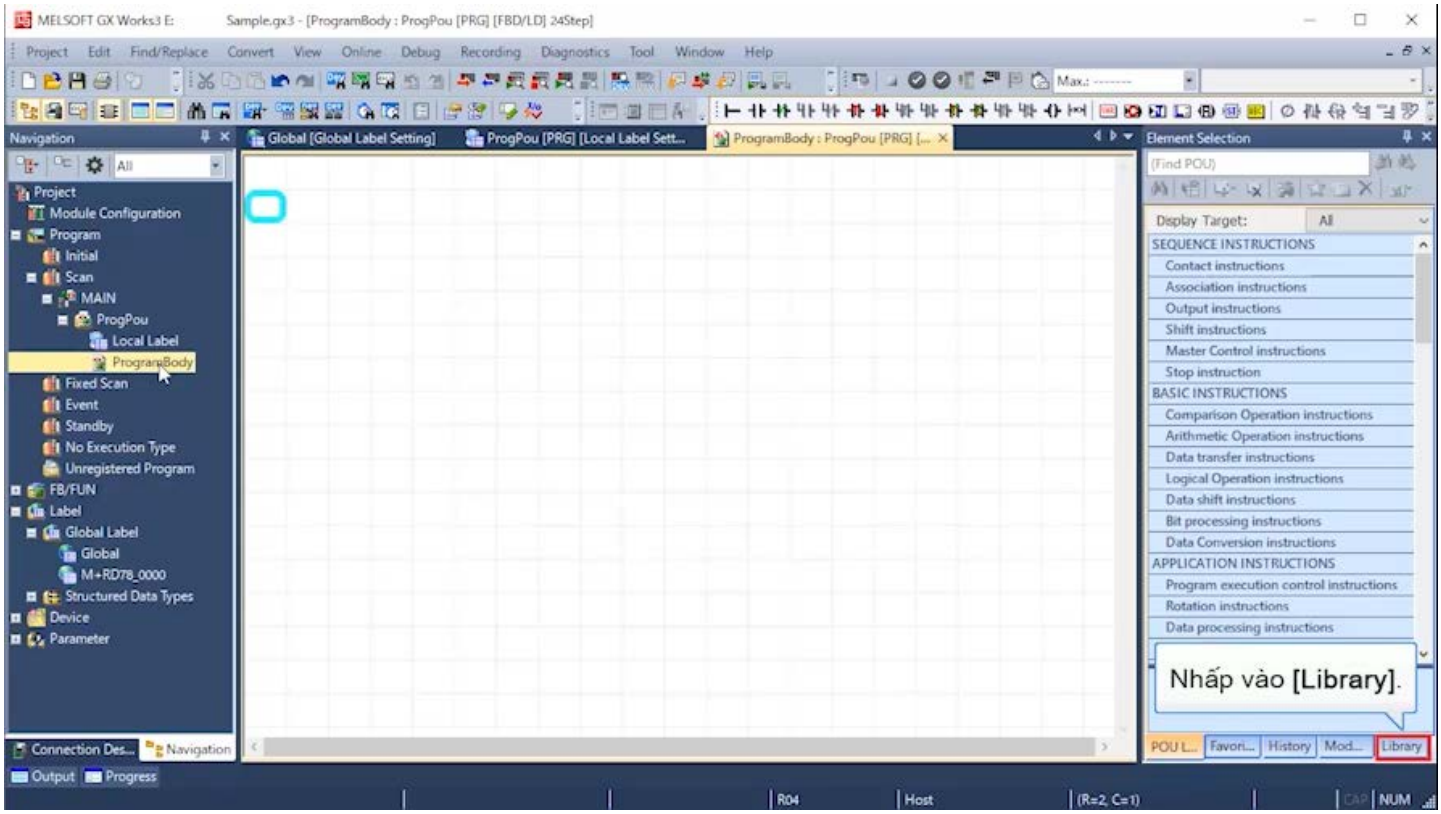
4.3 Cách sử dụng Motion Control FB



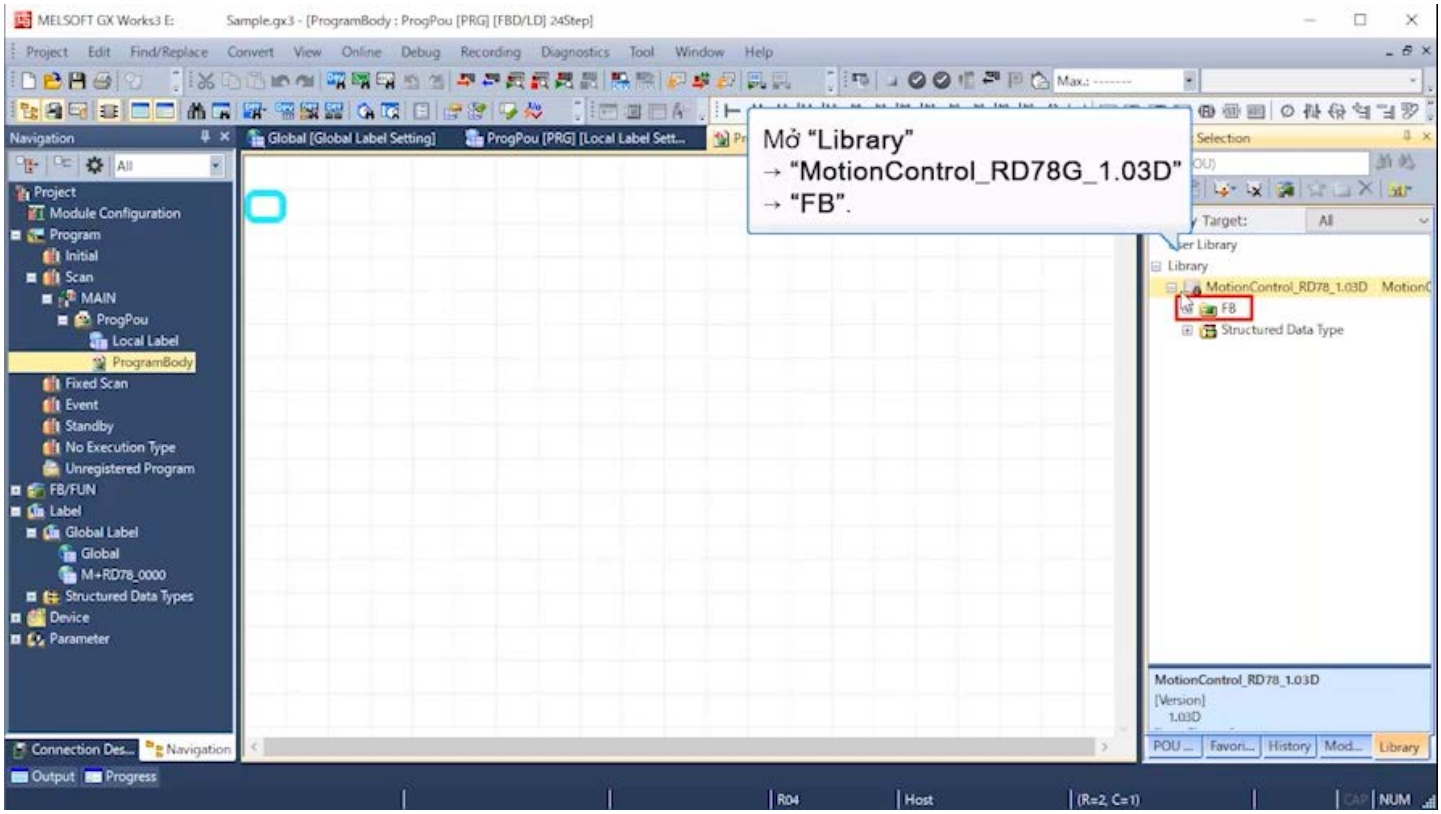
4.3 Cách sử dụng Motion Control FB



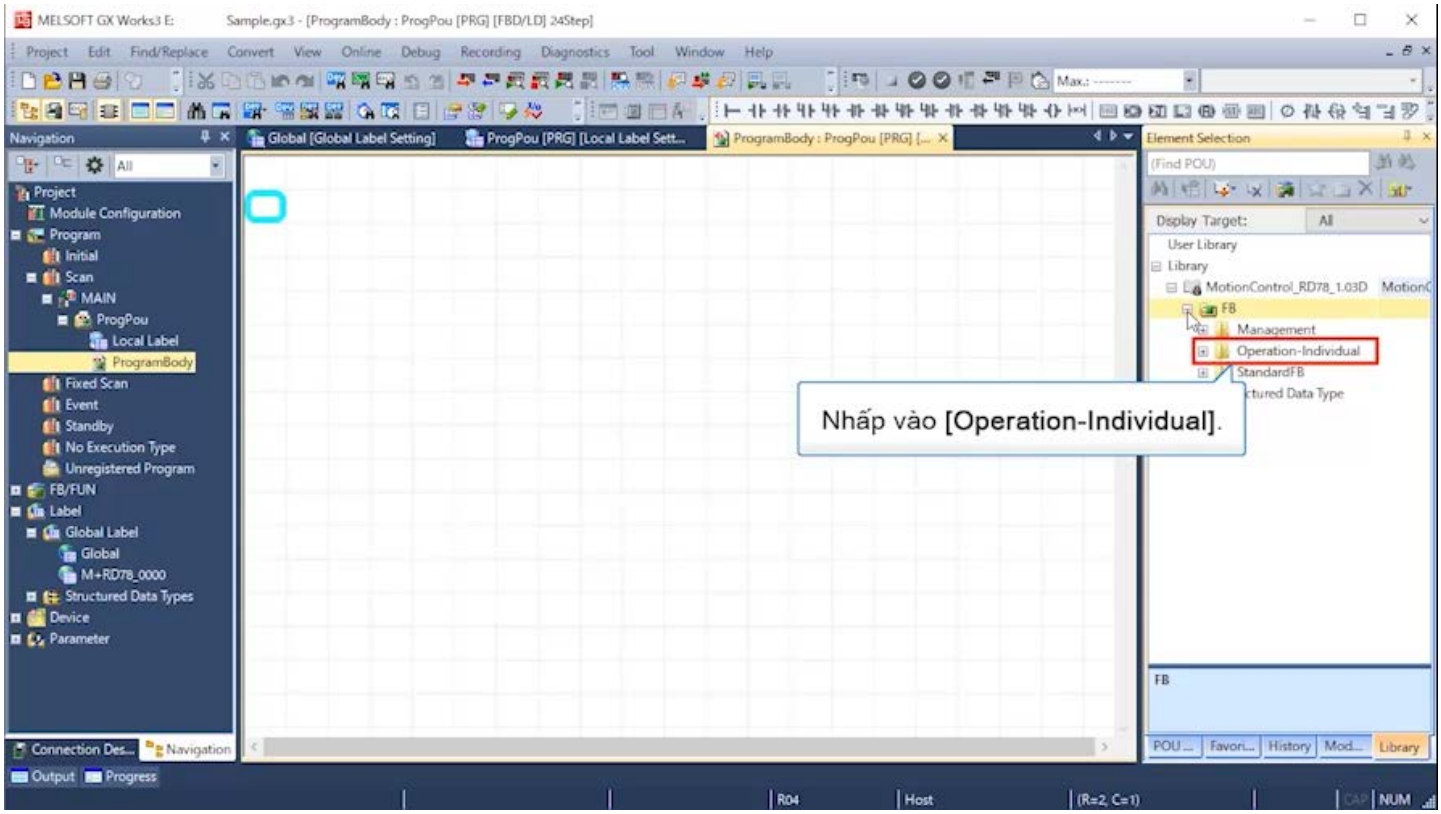
4.3 Cách sử dụng Motion Control FB



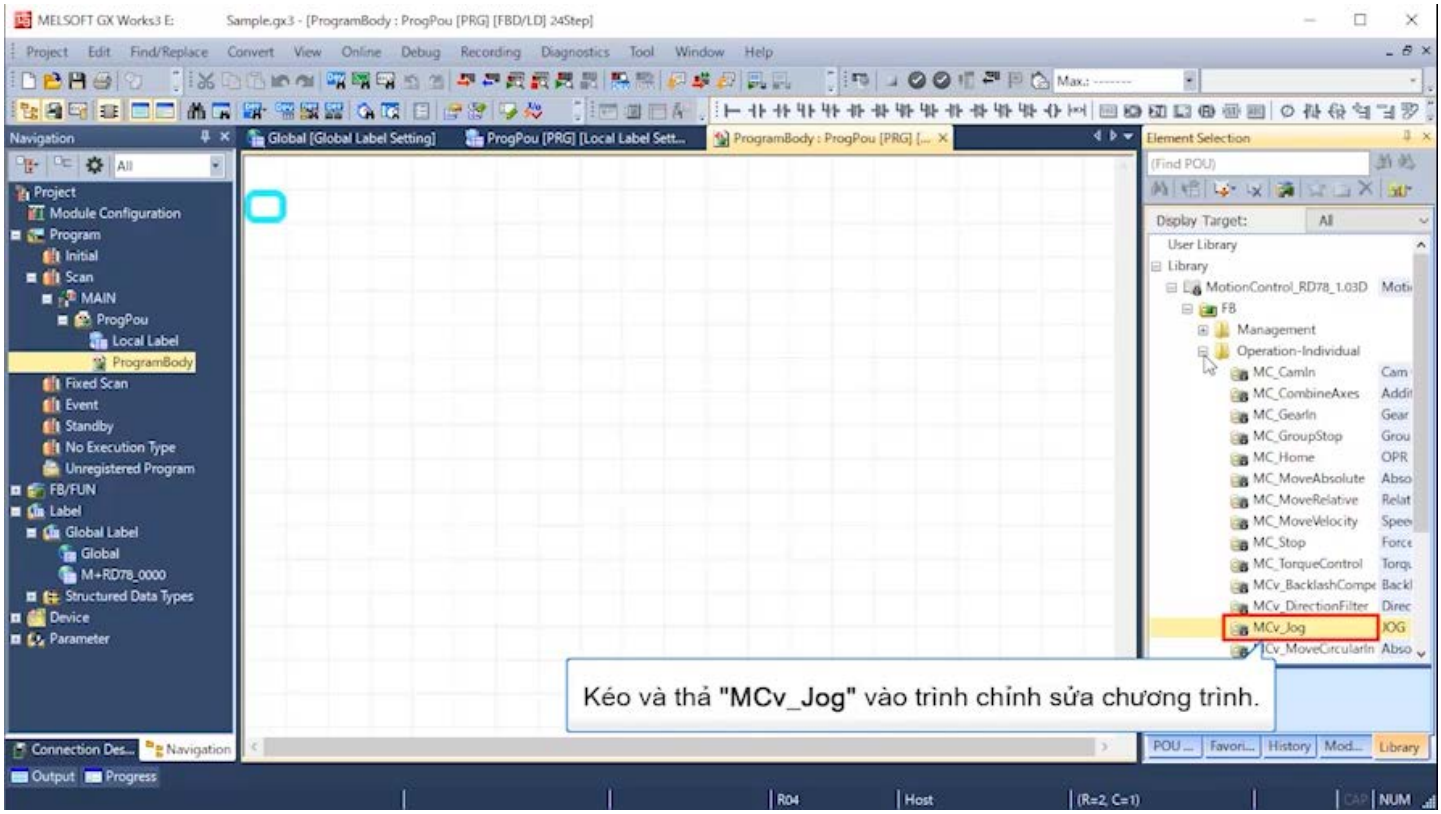
4.3 Cách sử dụng Motion Control FB



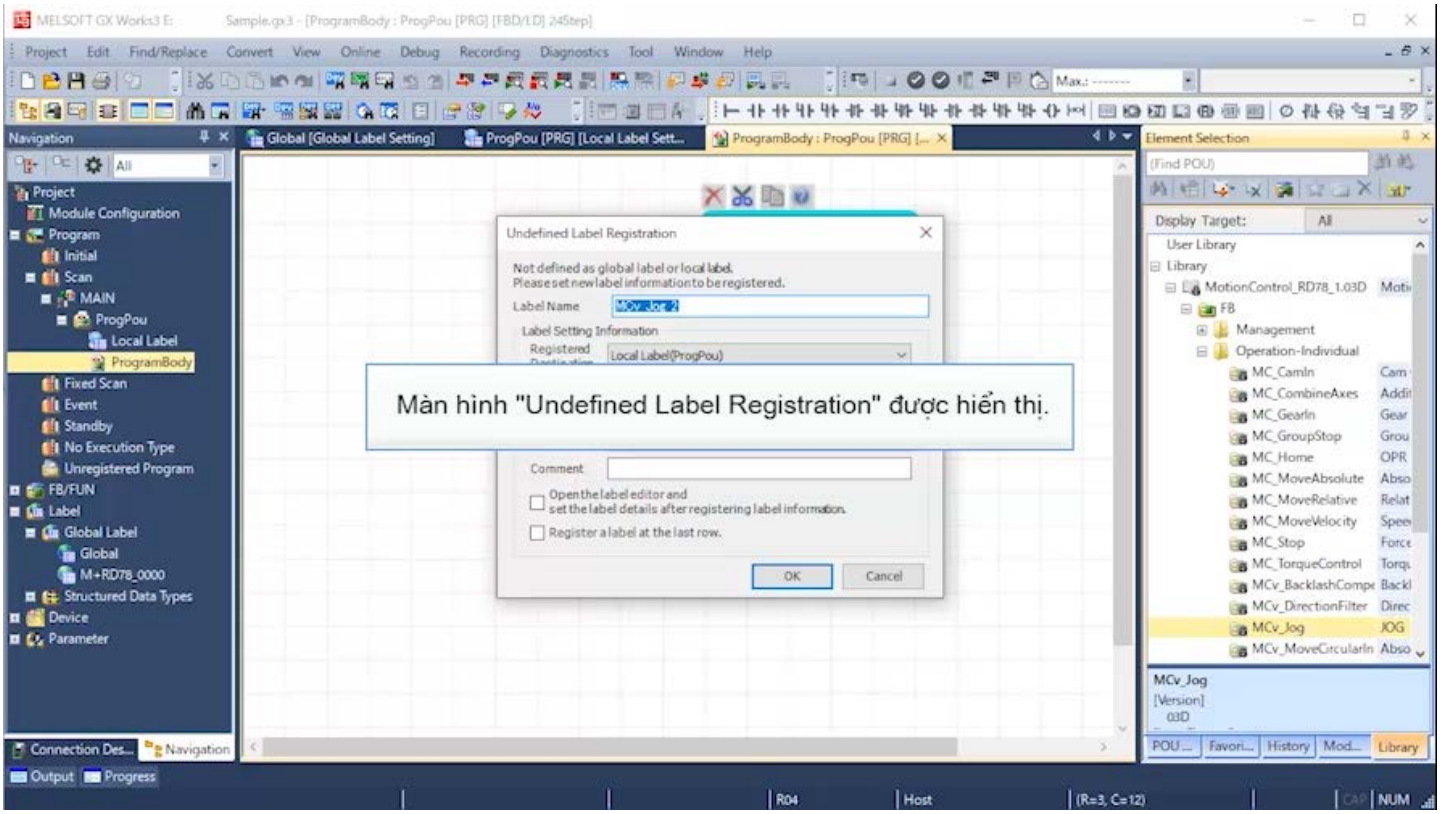
4.3 Cách sử dụng Motion Control FB



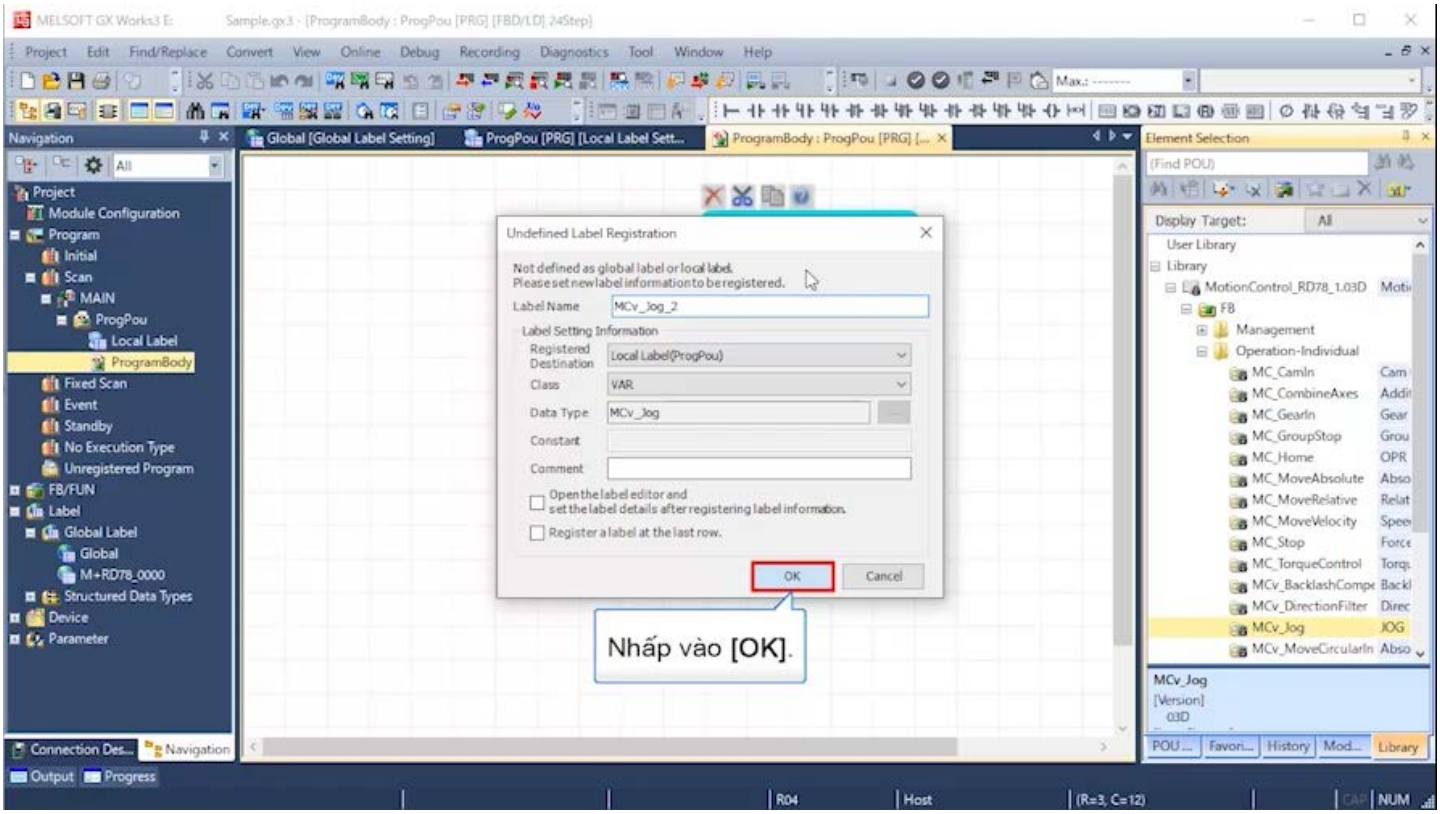
4.3 Cách sử dụng Motion Control FB



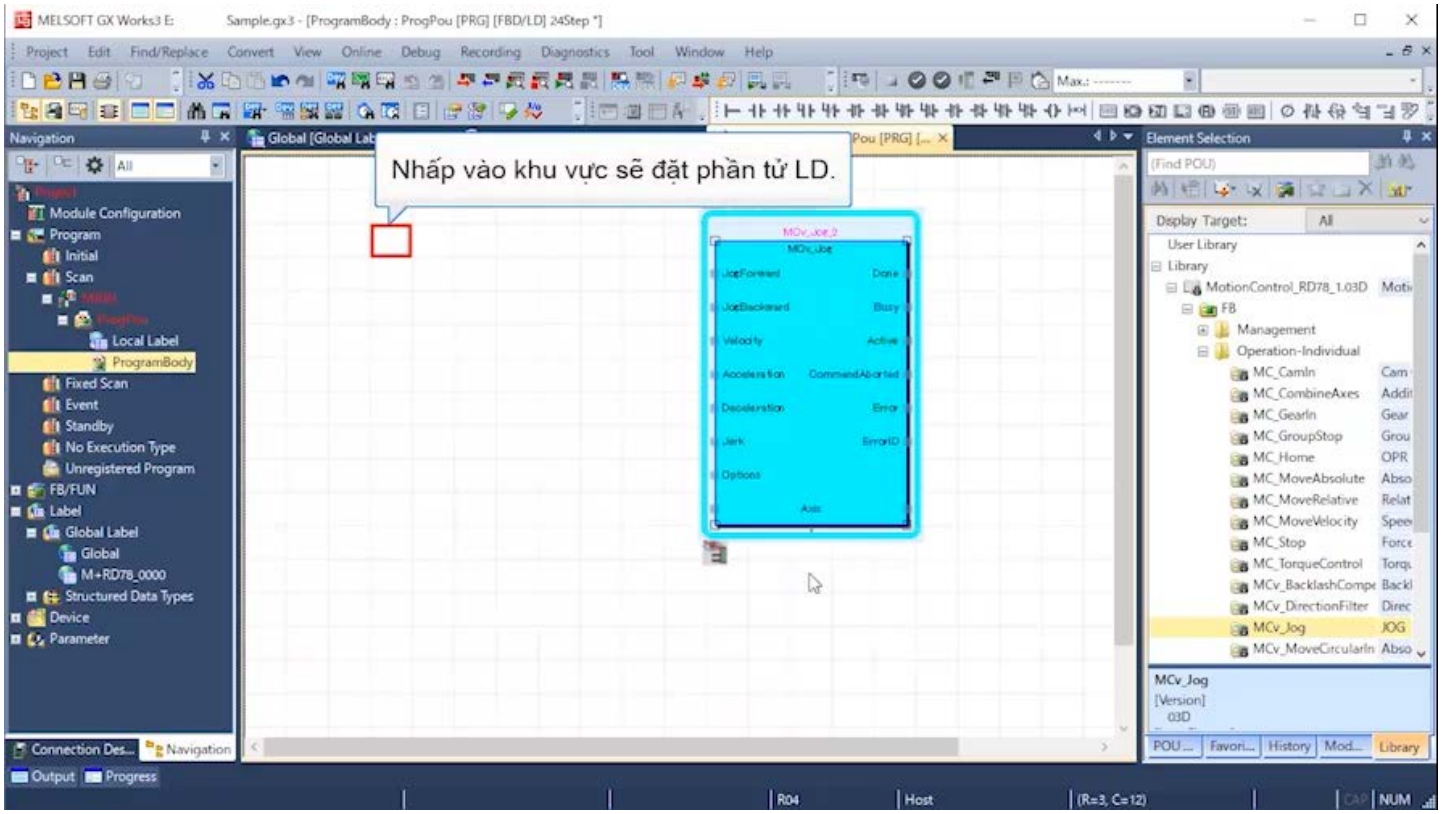
4.3 Cách sử dụng Motion Control FB



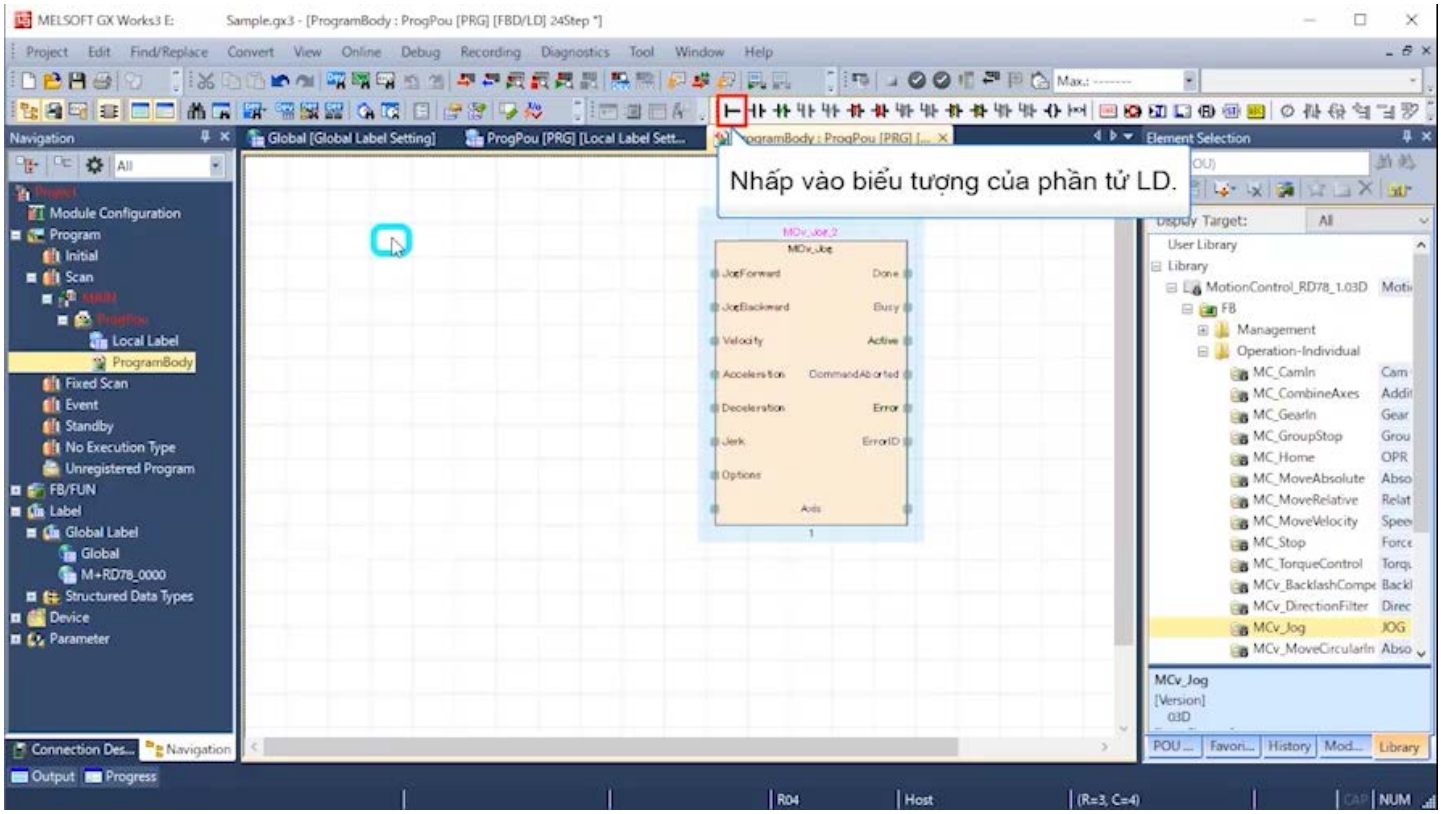
4.3 Cách sử dụng Motion Control FB



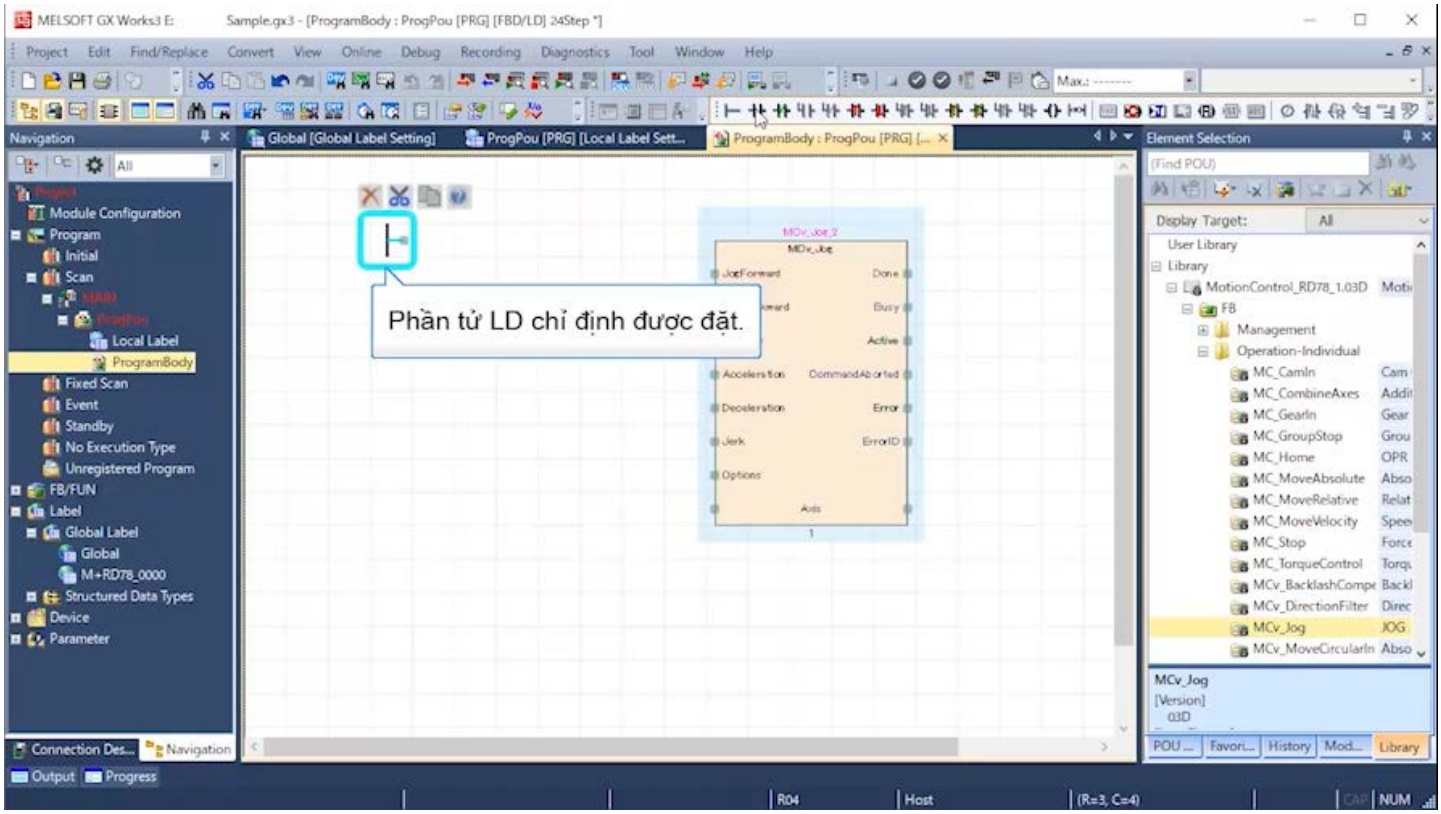
4.3 Cách sử dụng Motion Control FB



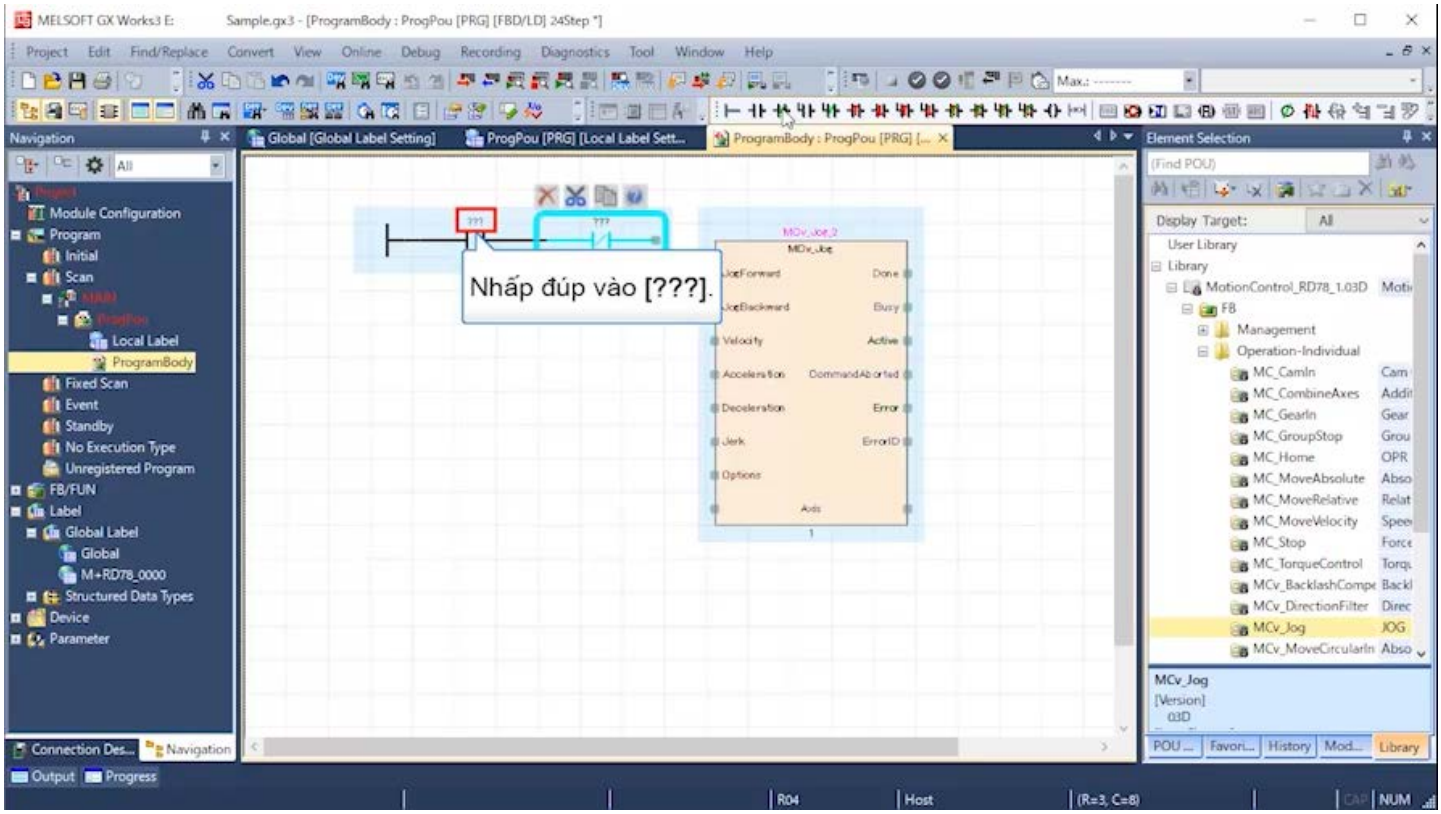
4.3 Cách sử dụng Motion Control FB



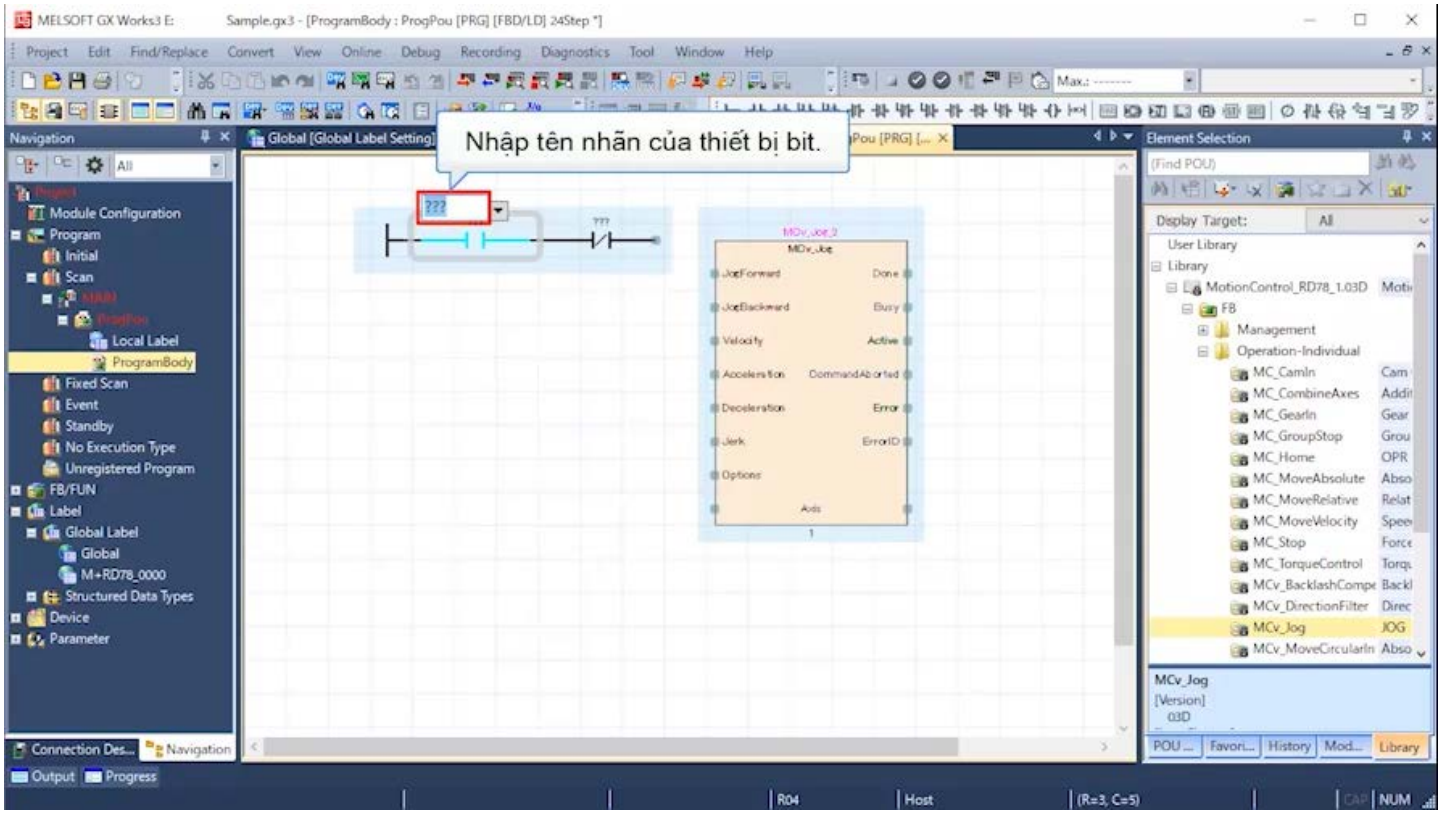
4.3 Cách sử dụng Motion Control FB



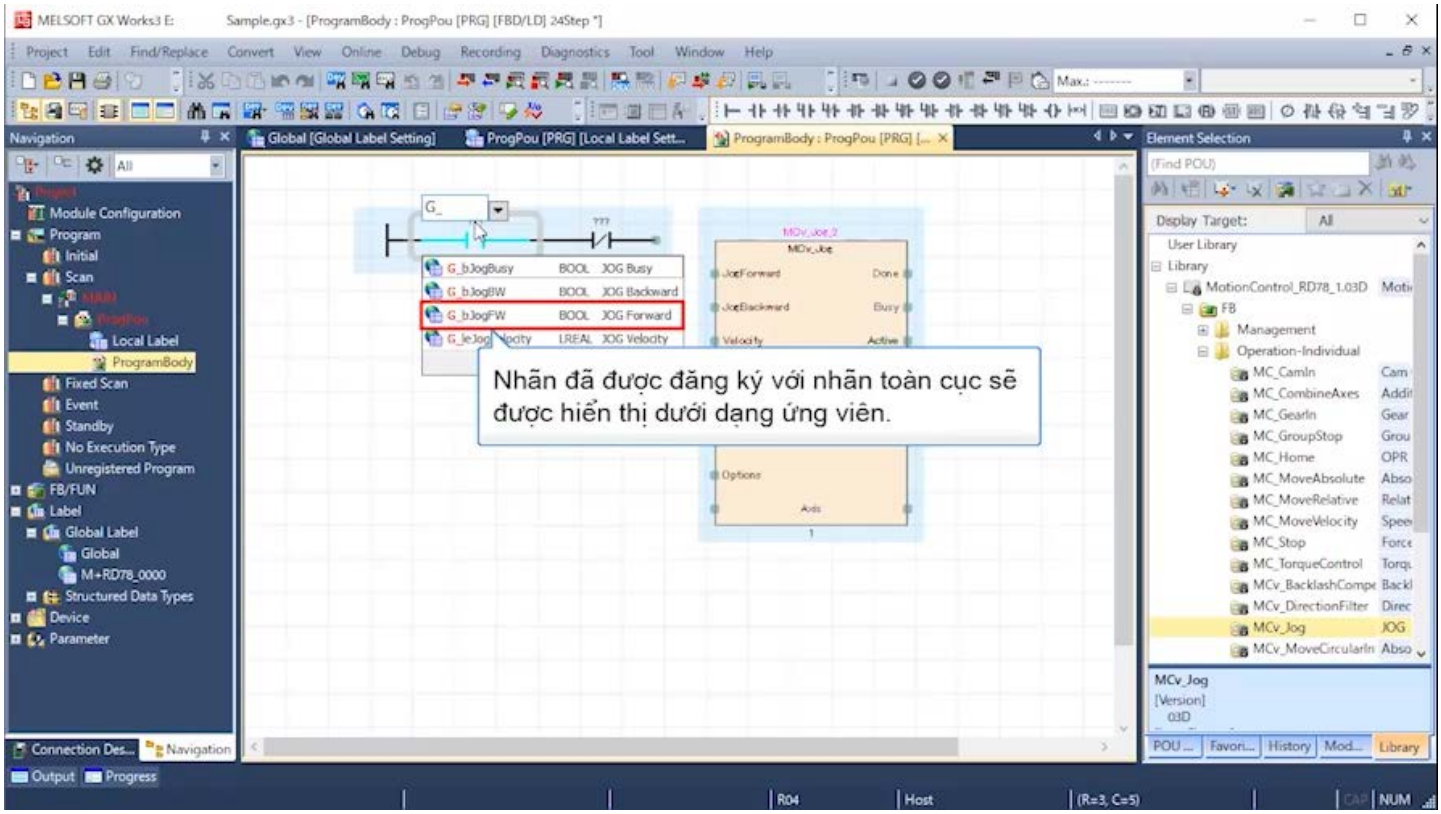
4.3 Cách sử dụng Motion Control FB



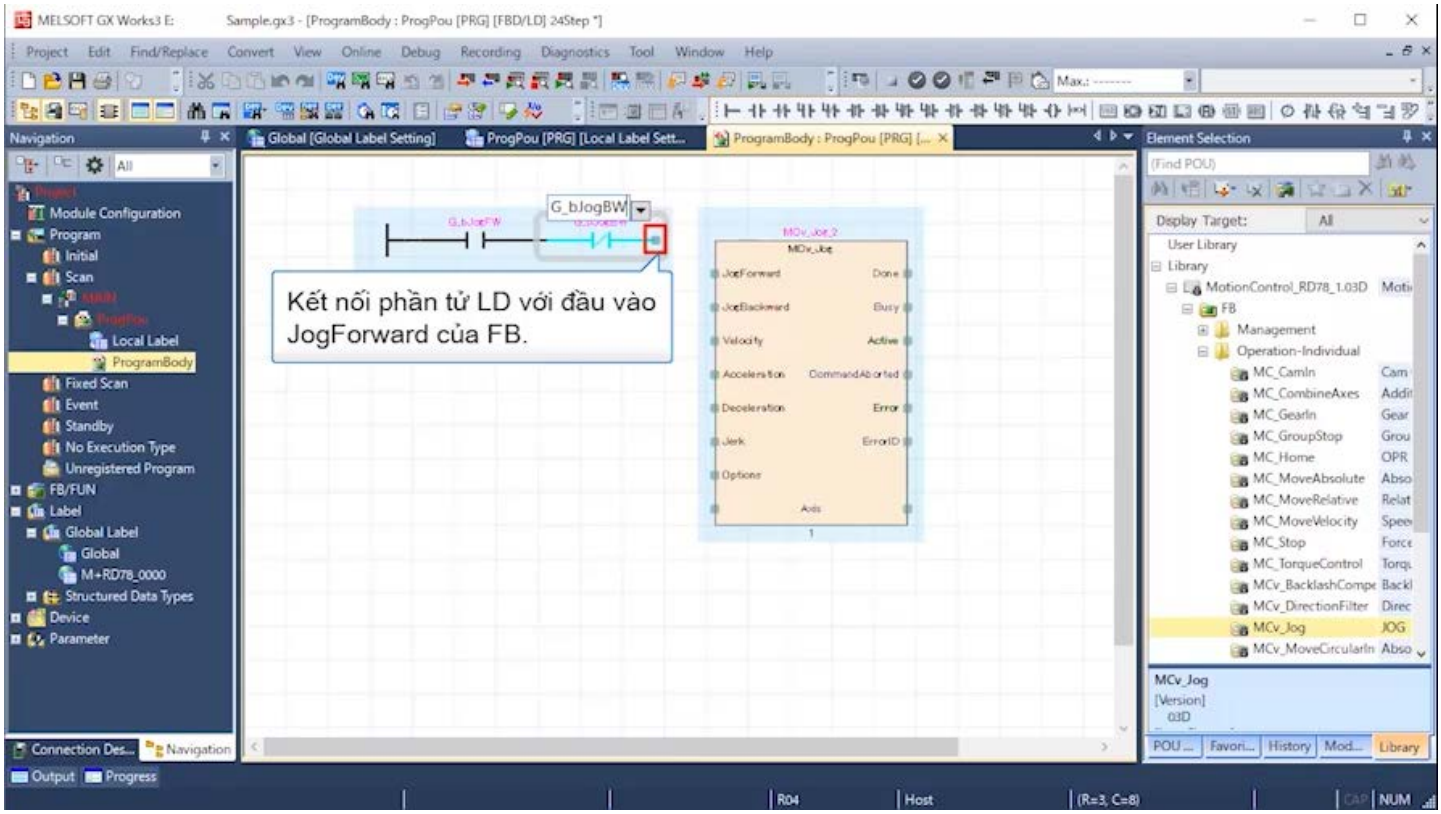
4.3 Cách sử dụng Motion Control FB



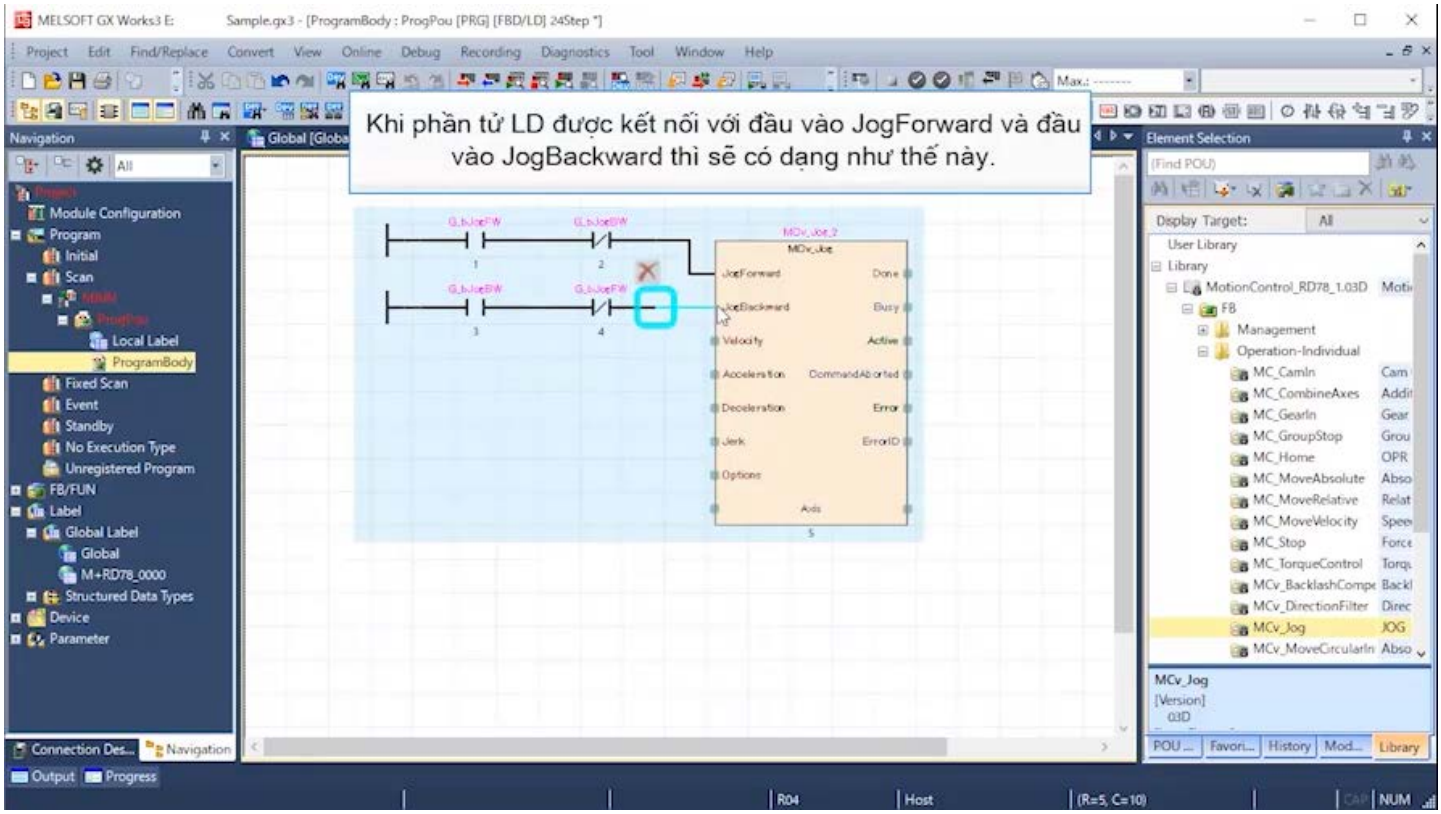
4.3 Cách sử dụng Motion Control FB



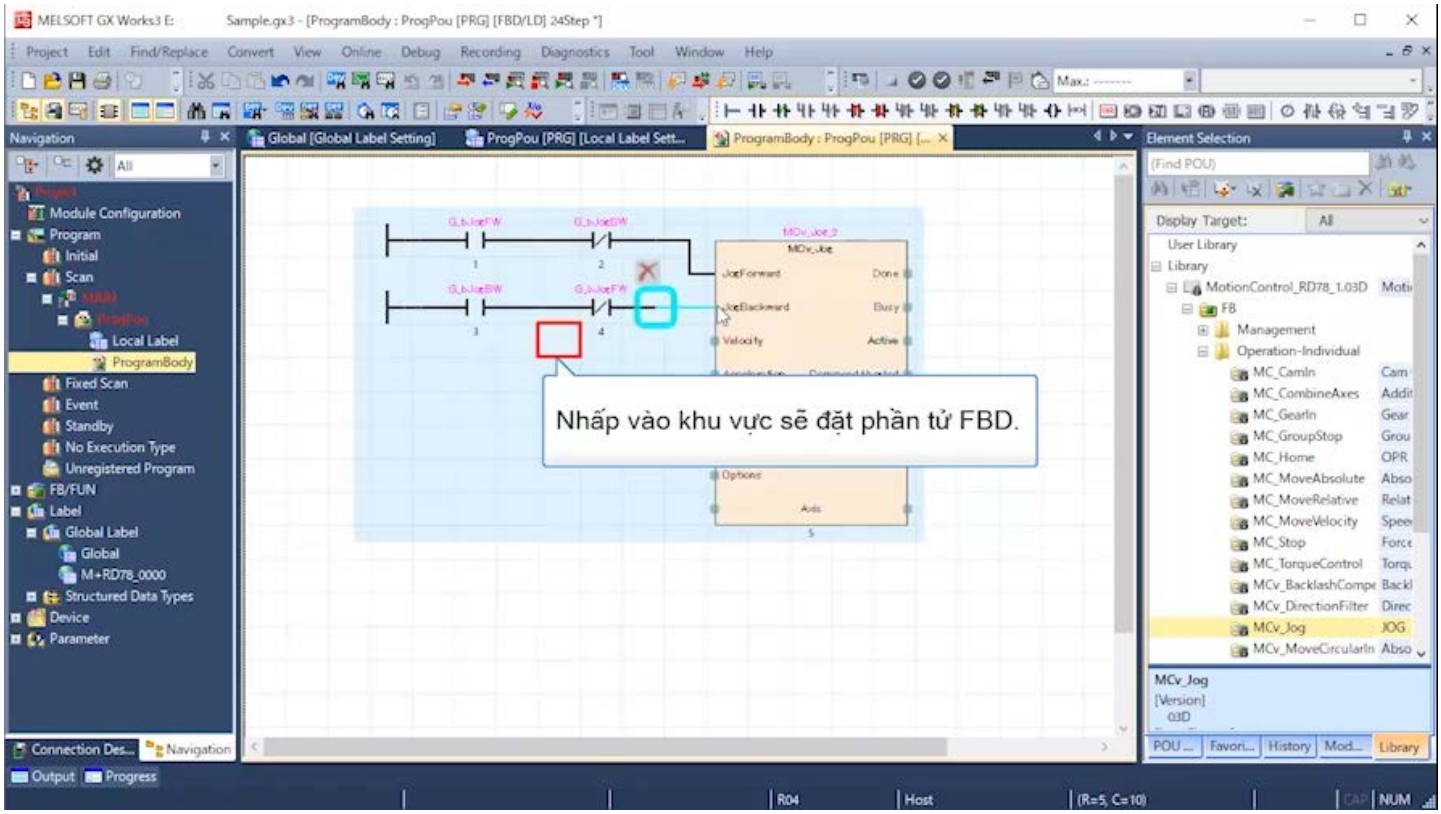
4.3 Cách sử dụng Motion Control FB



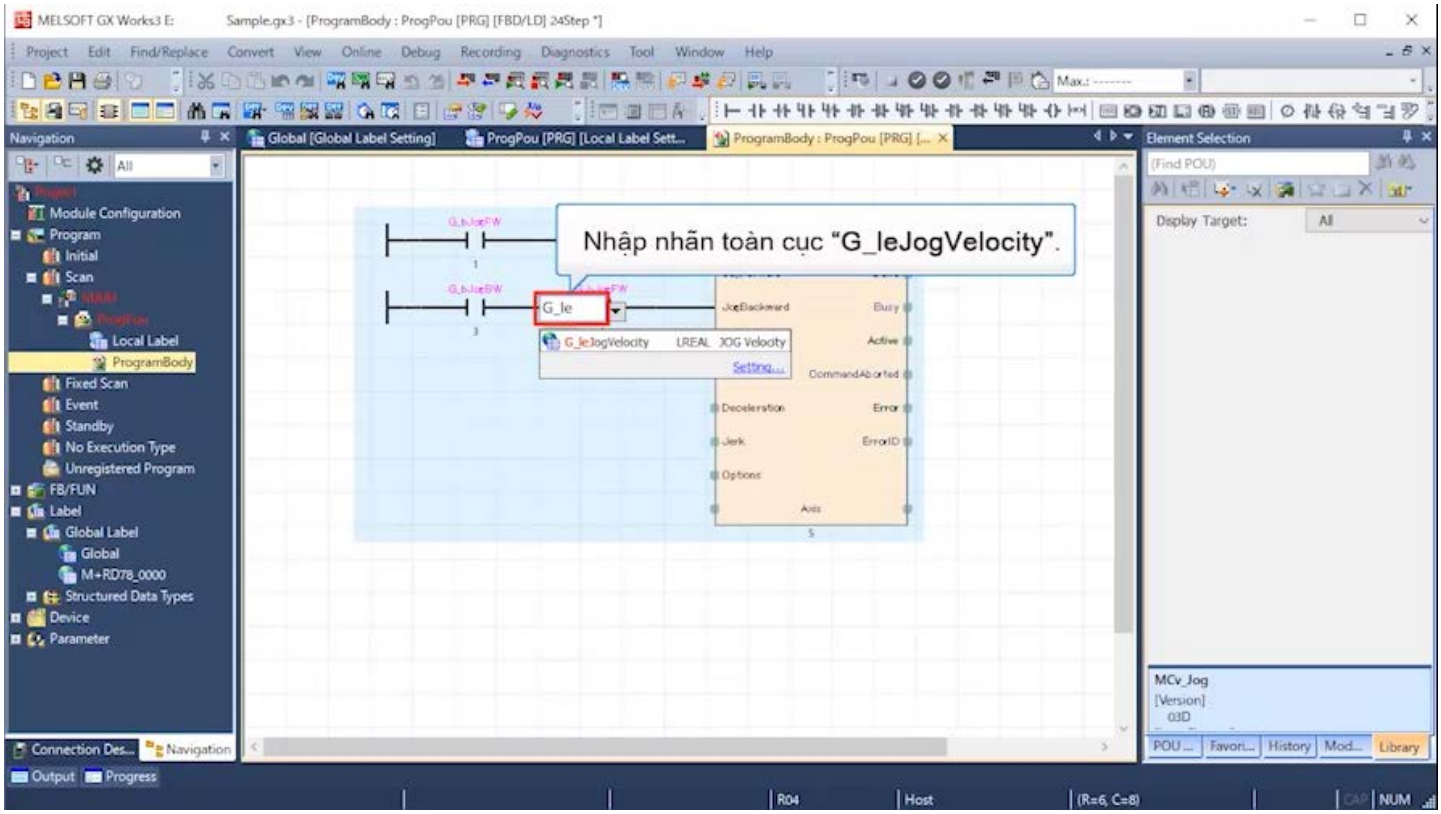
4.3 Cách sử dụng Motion Control FB



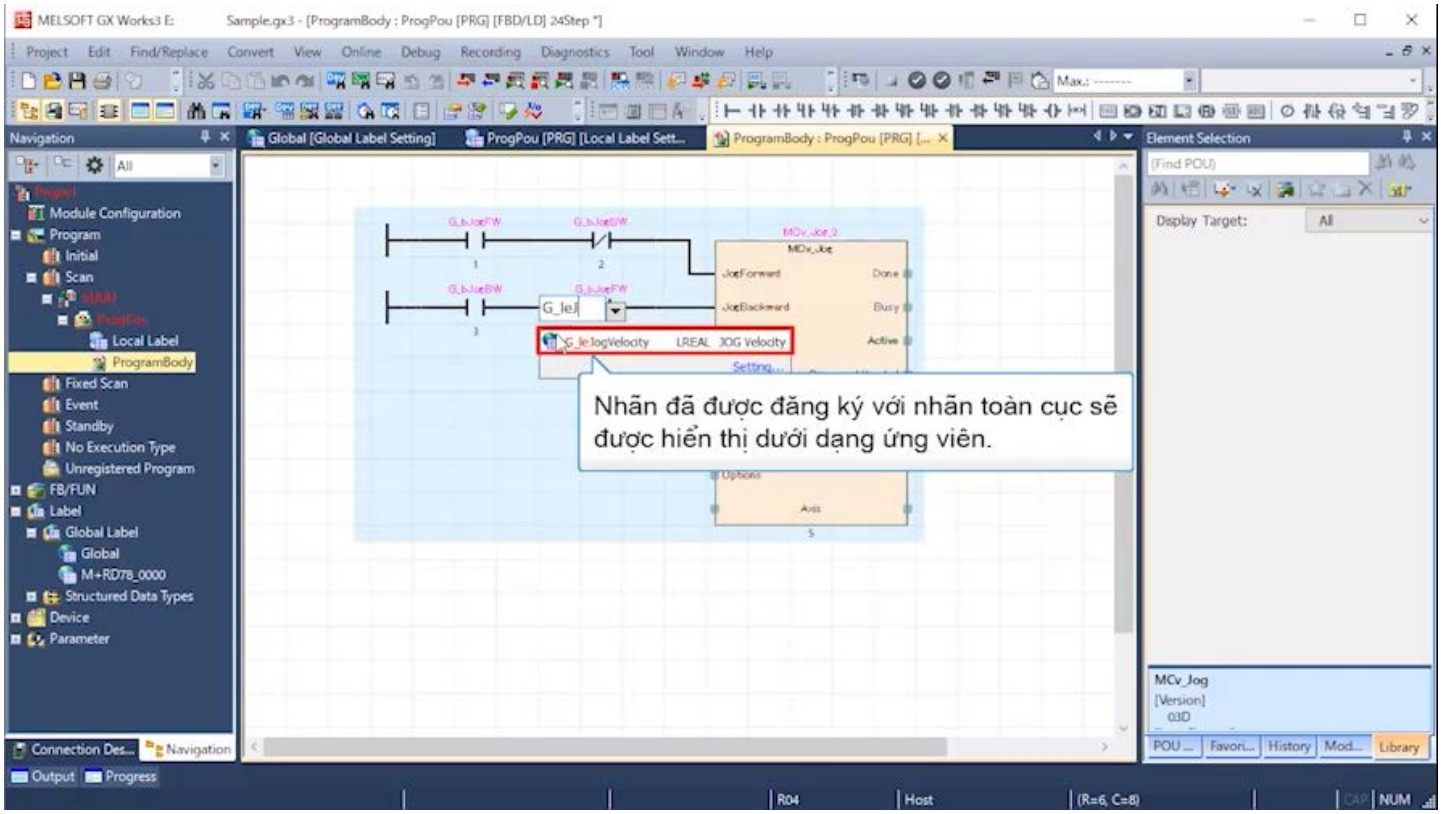
4.3 Cách sử dụng Motion Control FB



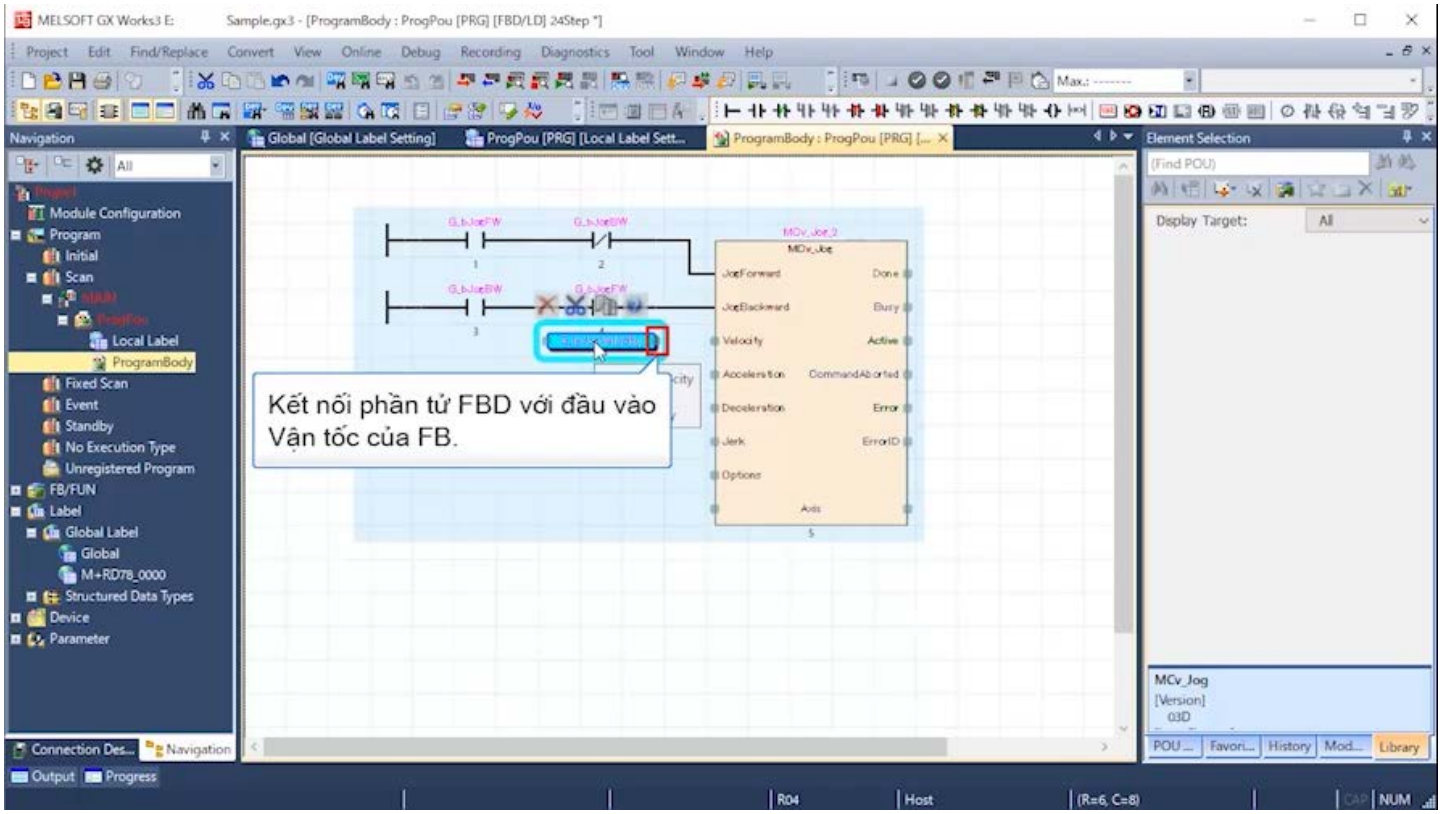
4.3 Cách sử dụng Motion Control FB



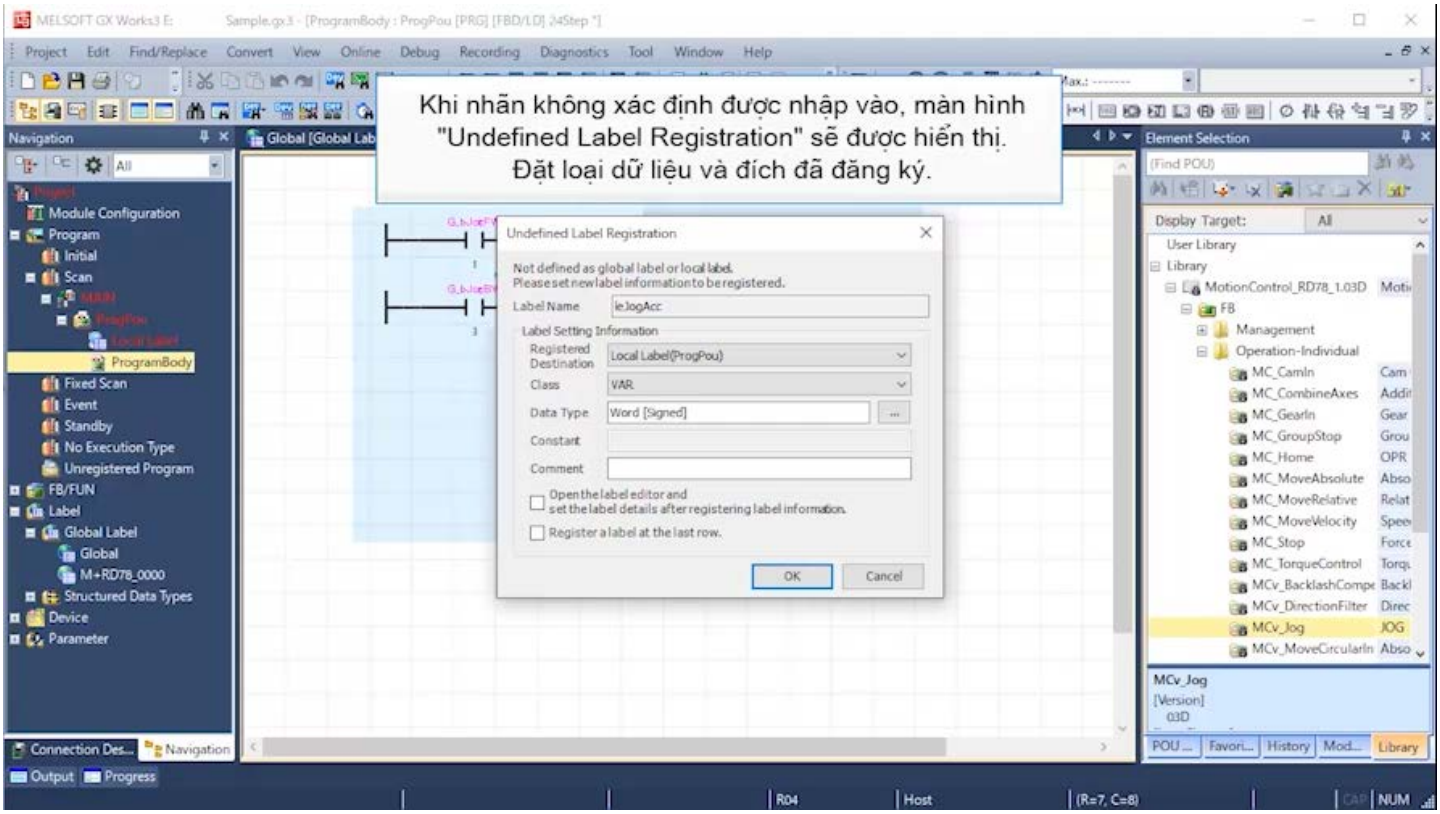
4.3 Cách sử dụng Motion Control FB



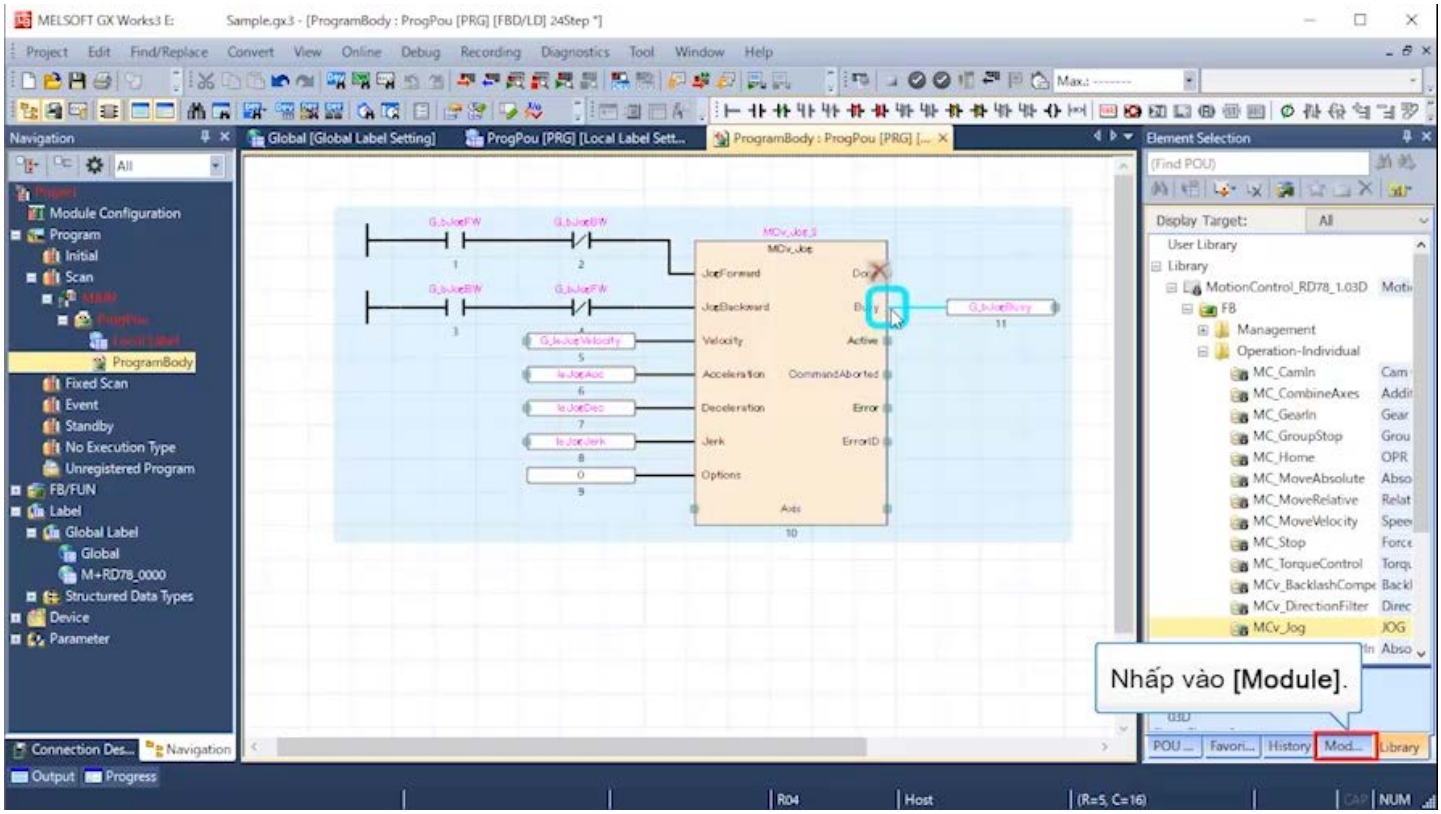
4.3 Cách sử dụng Motion Control FB



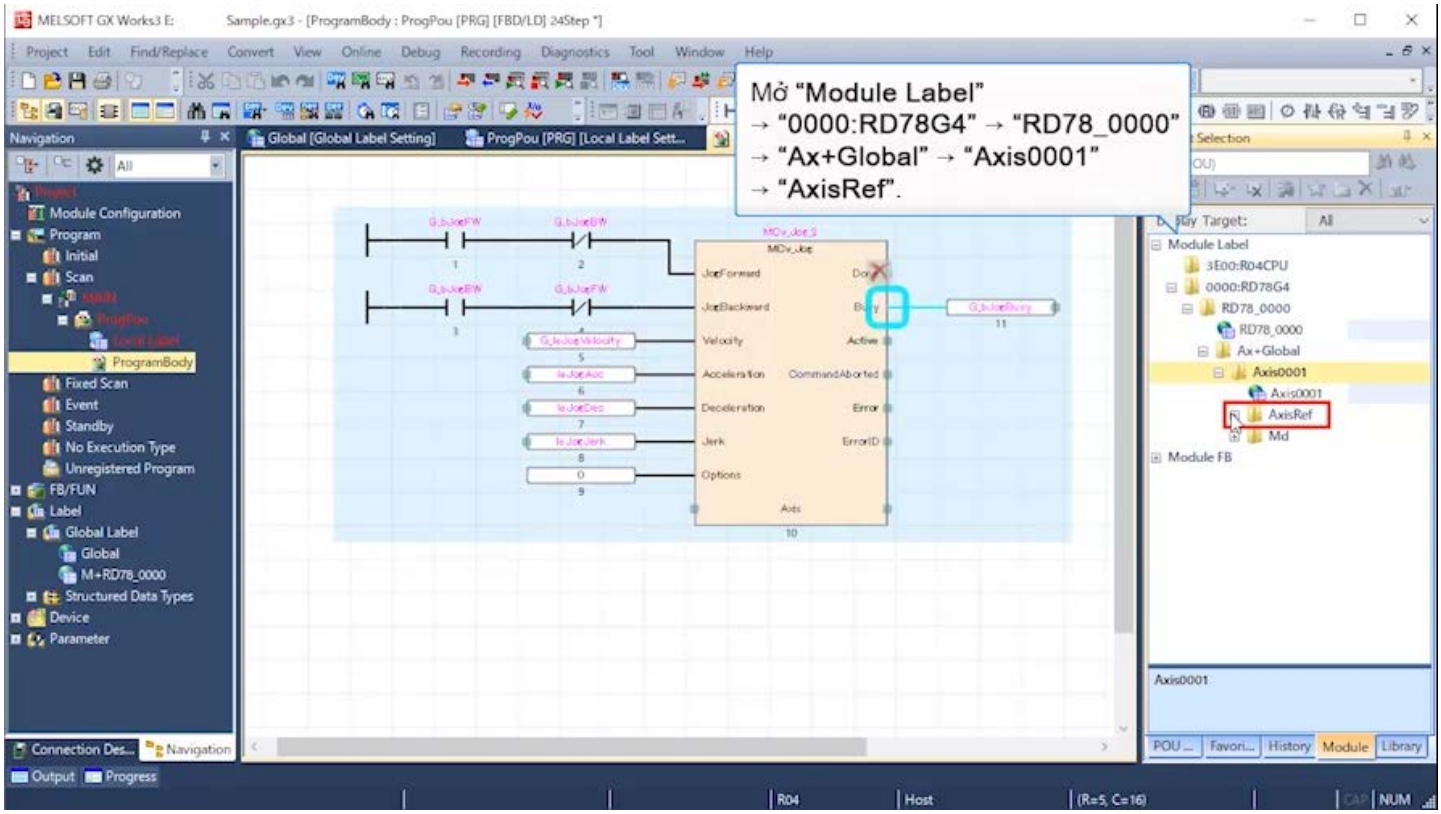
4.3 Cách sử dụng Motion Control FB



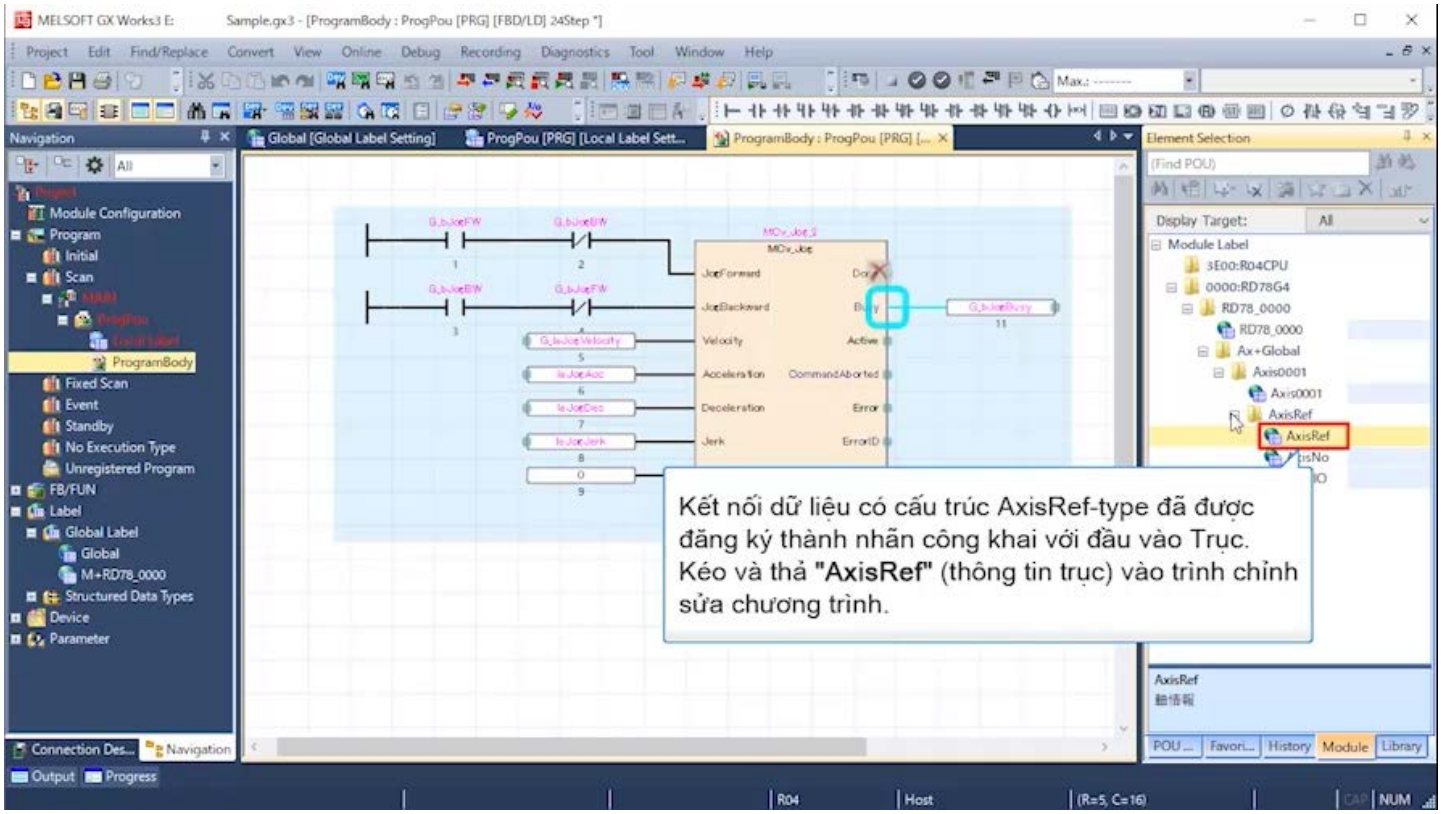
4.3 Cách sử dụng Motion Control FB



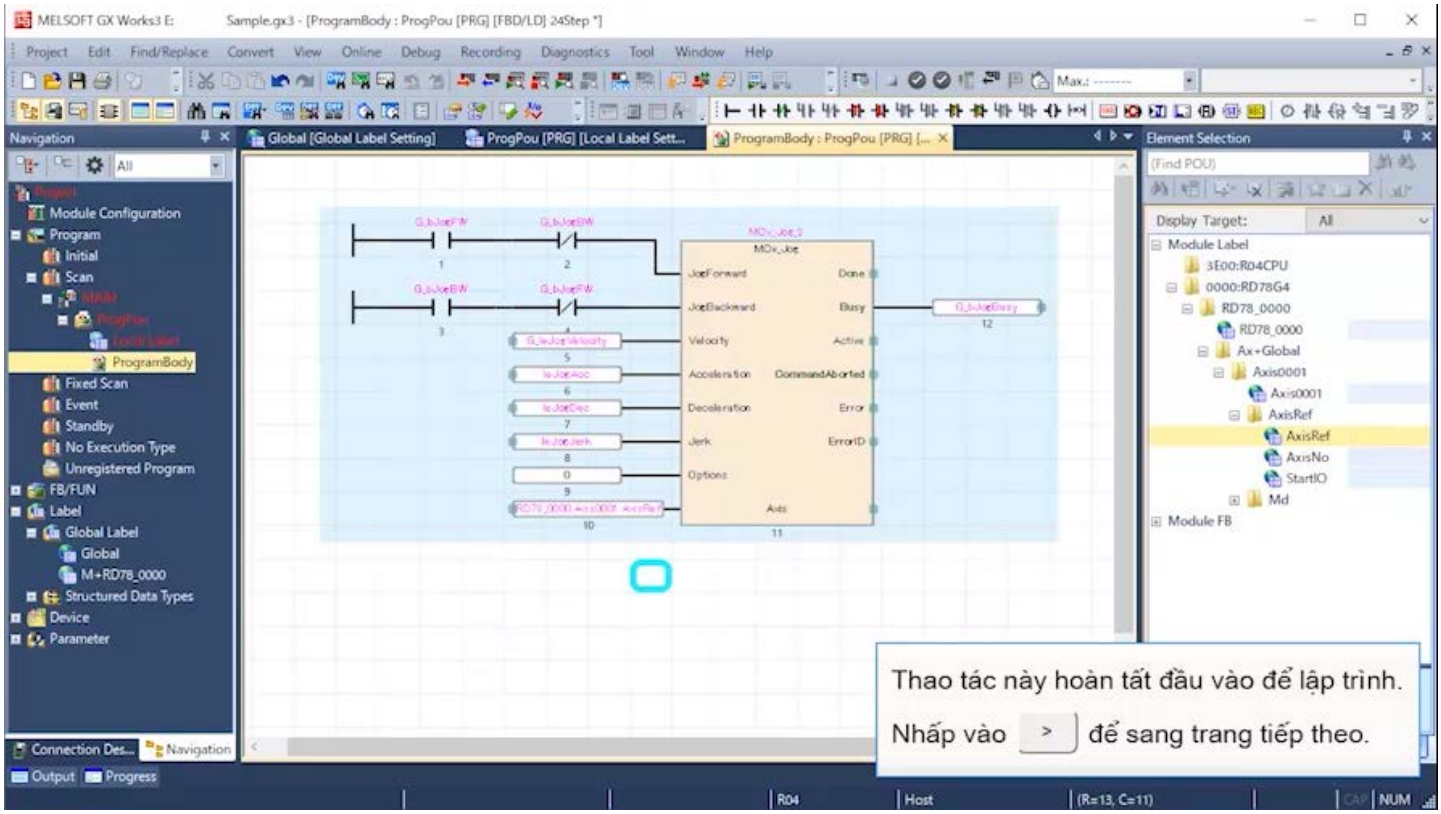
4.3 Cách sử dụng Motion Control FB



4.3 Cách sử dụng Motion Control FB



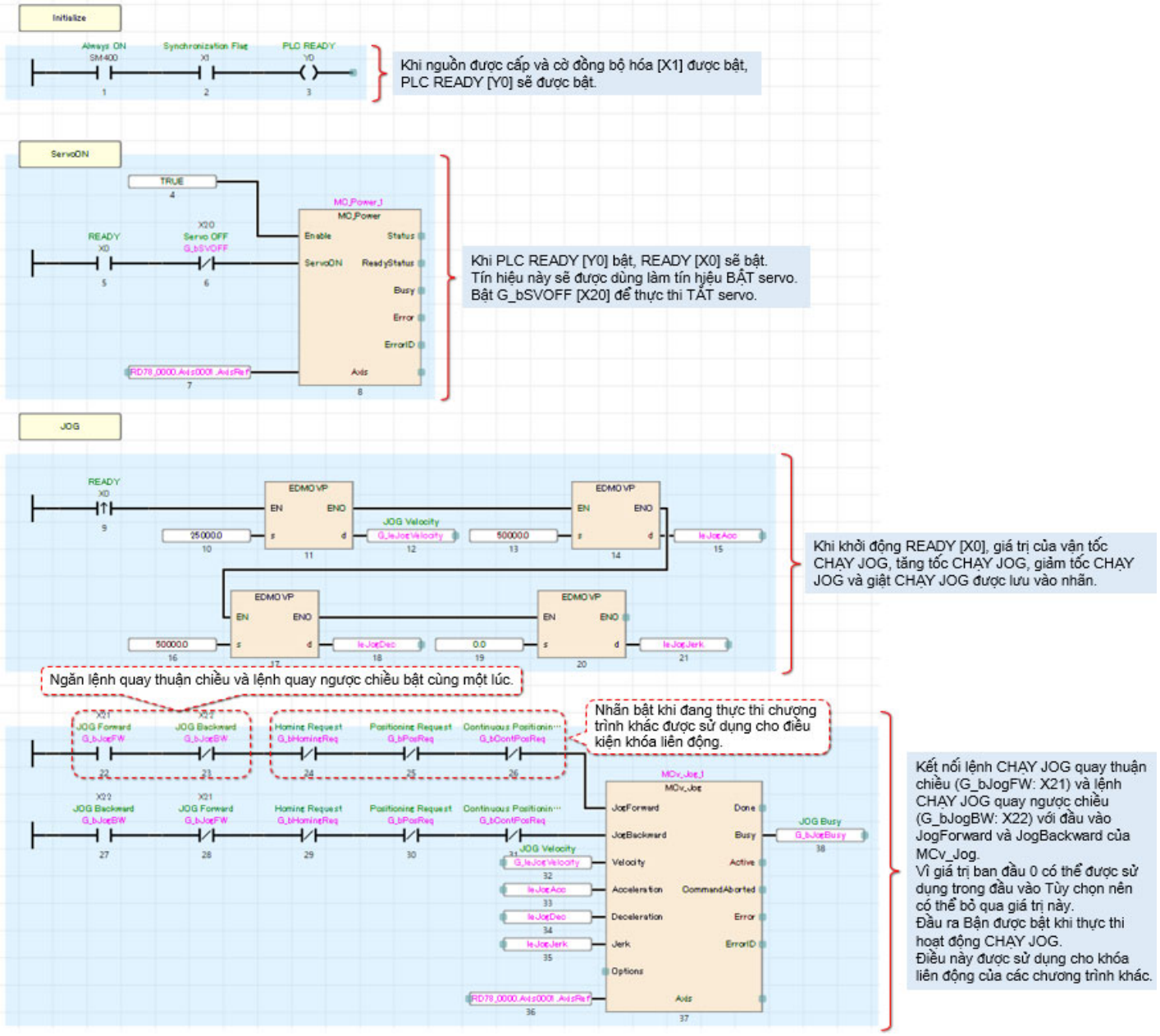
4.3 Cách sử dụng Motion Control FB



4.4 Mô tả về chương trình mẫu

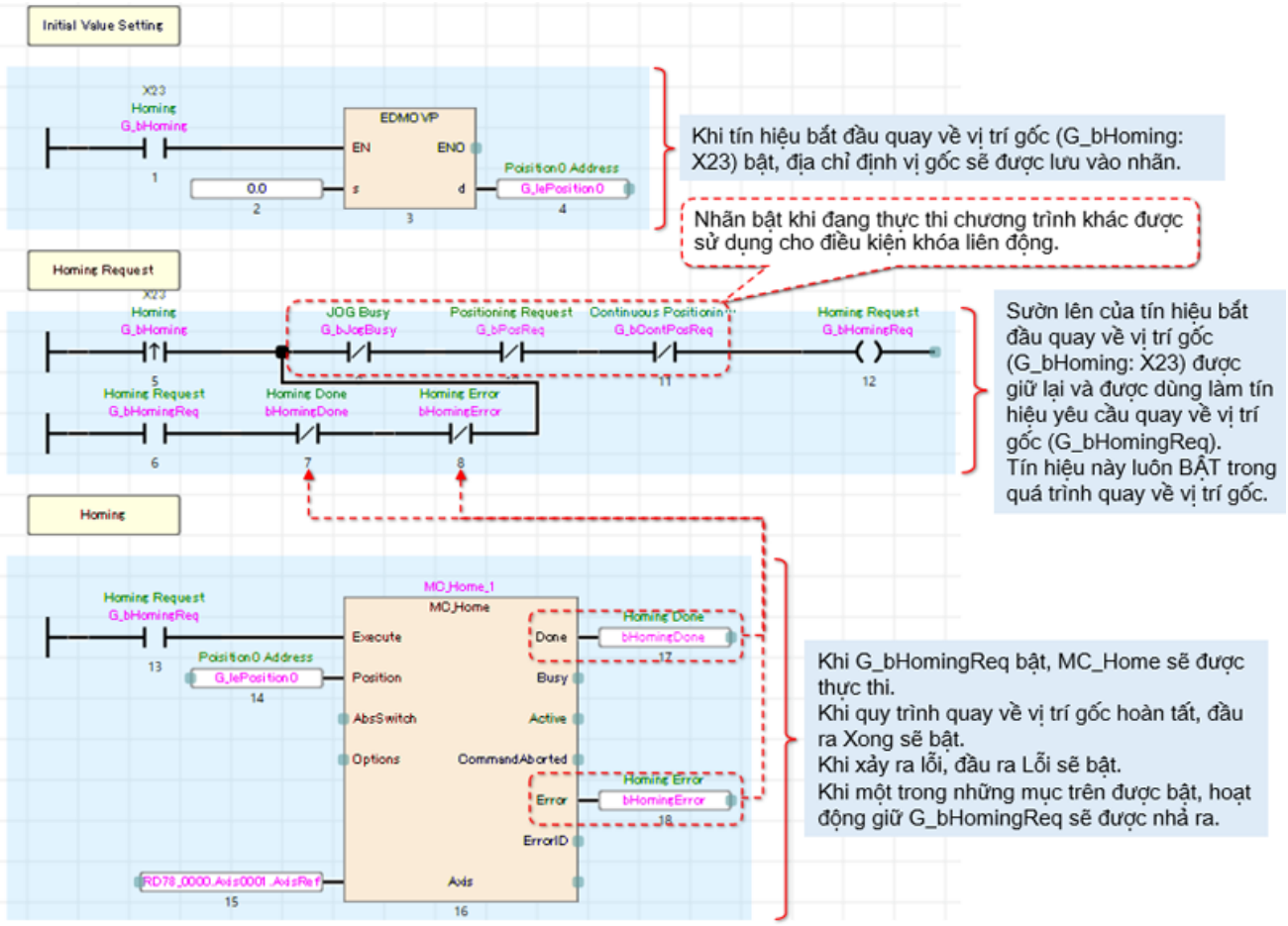
(1) ServoON_Jog

Chương trình này BẬT chế độ PLC sẵn sàng, BẬT servo và thực thi hoạt động CHẠY JOG.



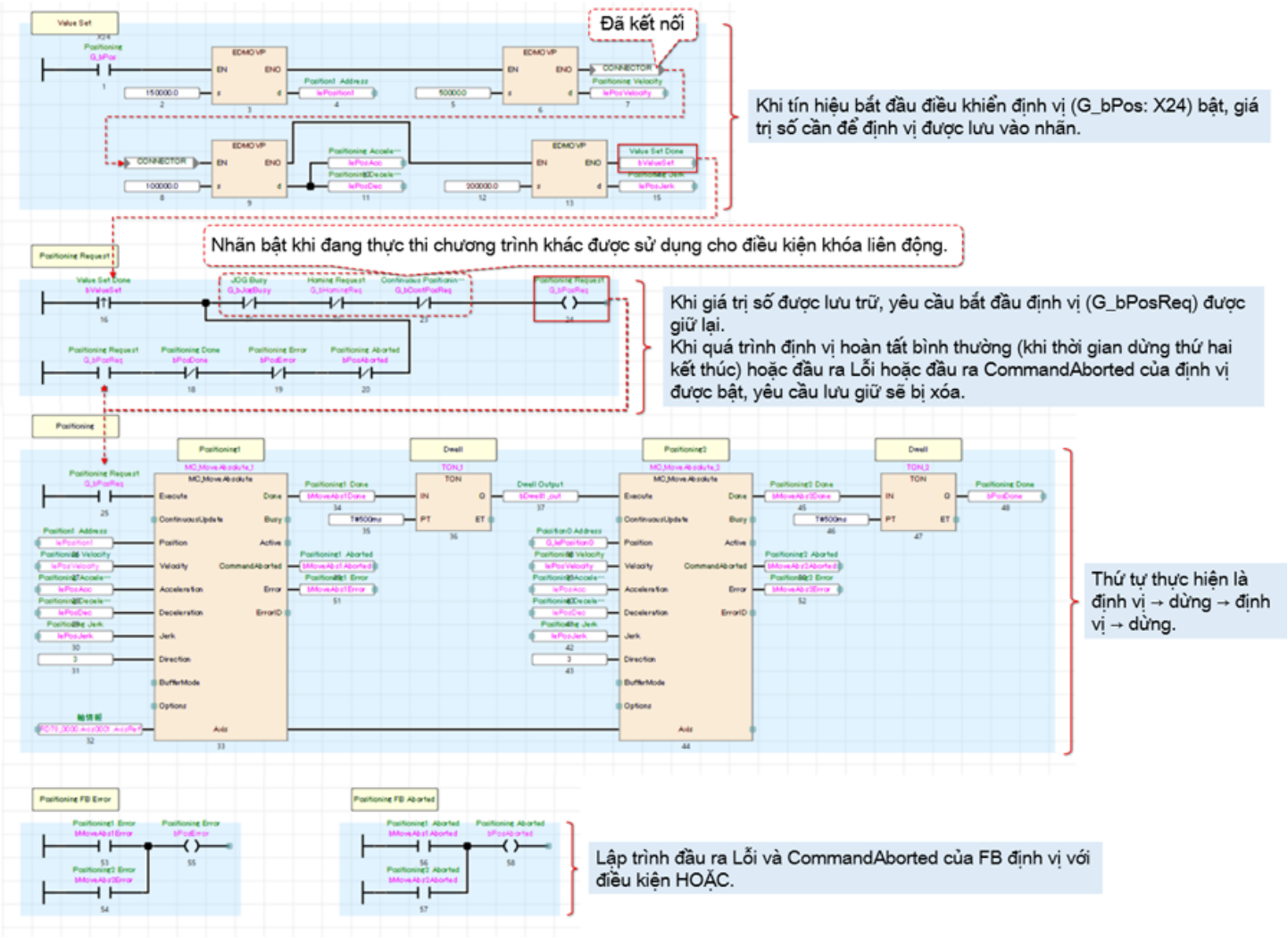
(2) Quay về vị trí gốc

Chương trình này thực thi hoạt động quay về vị trí gốc.



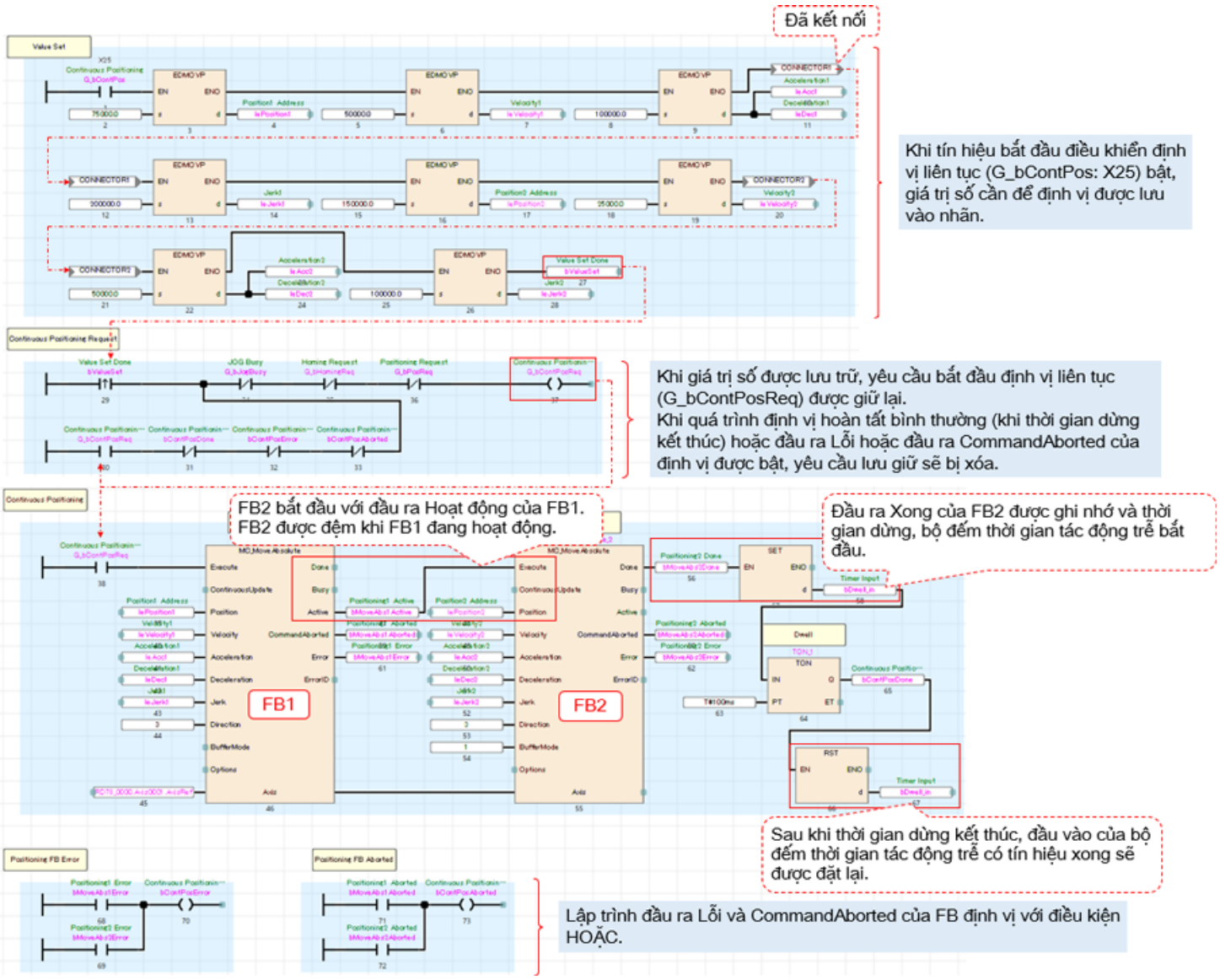
(3) Định vị

Chương trình này thực thi hoạt động định vị trục đơn.



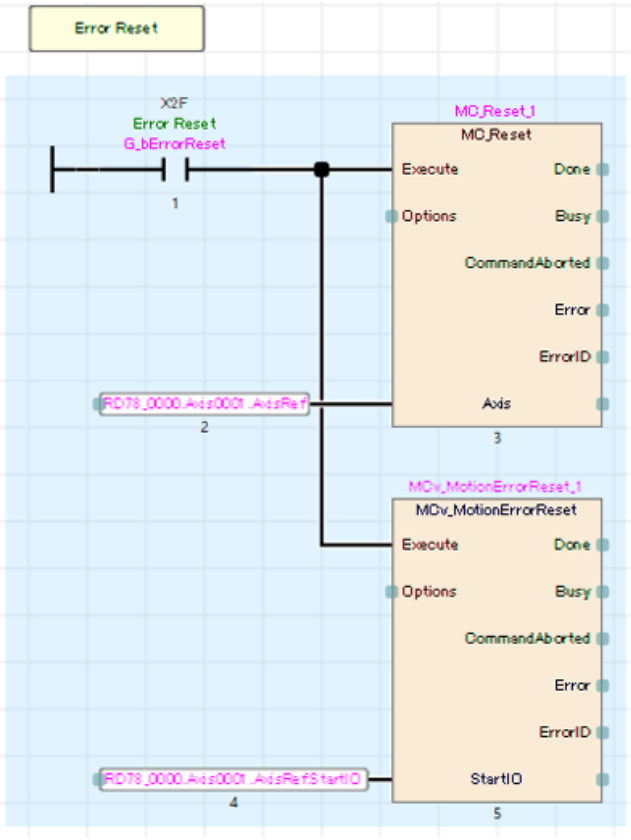
(4) ContinuousPositioning

Chương trình này thực thi định vị liên tục bằng chế độ đệm.



(5) ErrorReset

Chương trình này thiết lập lại lỗi.



Khi tín hiệu đặt lại lỗi (G_bErrorReset: X2F) bật, đặt lại lỗi trục (MC_Reset) và đặt lại lỗi hệ thống (MCv_MotionErrorReset) sẽ được thực thi.

(6) Giám sát

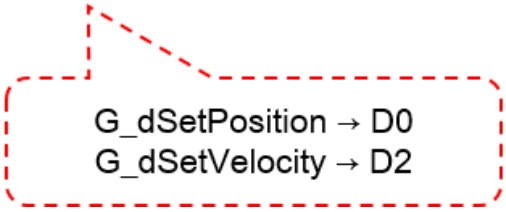
Chương trình này lưu trữ SetPosition (Vị trí đã đặt) và SetVelocity (Vận tốc đã đặt) của nhãn toàn cục giám sát trục được chỉ định cho D0 và D2 của PLC CPU.

Vì SetPosition và SetVelocity là loại số thực có độ chính xác kép nên chúng được chuyển đổi thành loại từ kép có dấu để PLC CPU có thể dễ dàng xử lý. (Lưu ý)

Những thiết bị từ này không được sử dụng trong đối tượng.

Chúng được sử dụng để hiển thị trên chương trình tuần tự khác và GOT, cũng như cho các mục đích khác. Chương trình này được mô tả bằng ST.

```
1  G_dSetPosition := LREAL_TO_DINT(RD78_0000.Axis0001.Md.SetPosition);
2  G_dSetVelocity := LREAL_TO_DINT(RD78_0000.Axis0001.Md.SetVelocity);
3
```



(Lưu ý) Khi loại số thực có độ chính xác kép được chuyển đổi thành loại từ kép có dấu, nếu giá trị cần chuyển đổi nằm ngoài phạm vi từ -2147483648 đến 2147483647 thì sẽ xảy ra lỗi tính toán.

4.5 Ghi chương trình

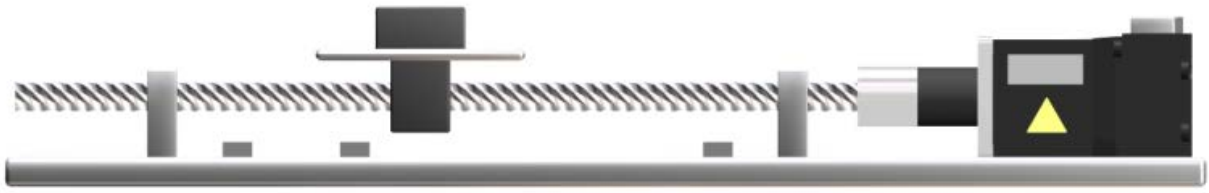
Ghi chương trình và tham số vào PLC CPU và mô-đun Chuyển động.
Chương trình chỉ được ghi vào mô-đun CPU. Thông số trục và cài đặt nhãn công khai phải được ghi vào phía mô-đun Chuyển động.

1) Sau khi tất cả chương trình trong PLC CPU được xây dựng lại, hãy chọn [Online] → [Write to PLC] trong thanh công cụ của GX Works3 để ghi tất cả dữ liệu vào PLC CPU.

2) Khi tham số được ghi vào PLC CPU, giao tiếp với mô-đun Chuyển động được bật.
Chọn [Online] → [Write to Module] trong thanh công cụ của Motion Control Setting Function để ghi tất cả dữ liệu vào mô-đun Chuyển động.

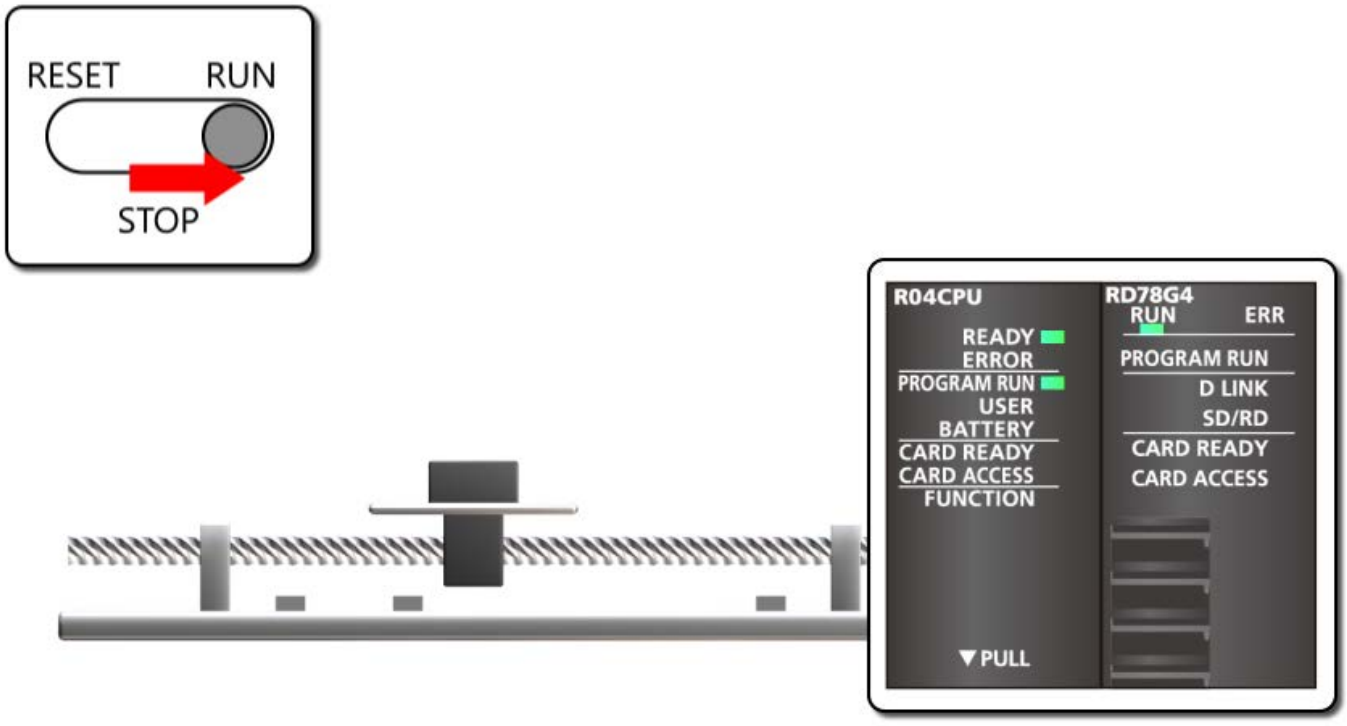
3) Đặt lại PLC CPU để hoàn tất hoạt động ghi.

Nhấp vào nút phát ở phần dưới bên trái của cửa sổ.

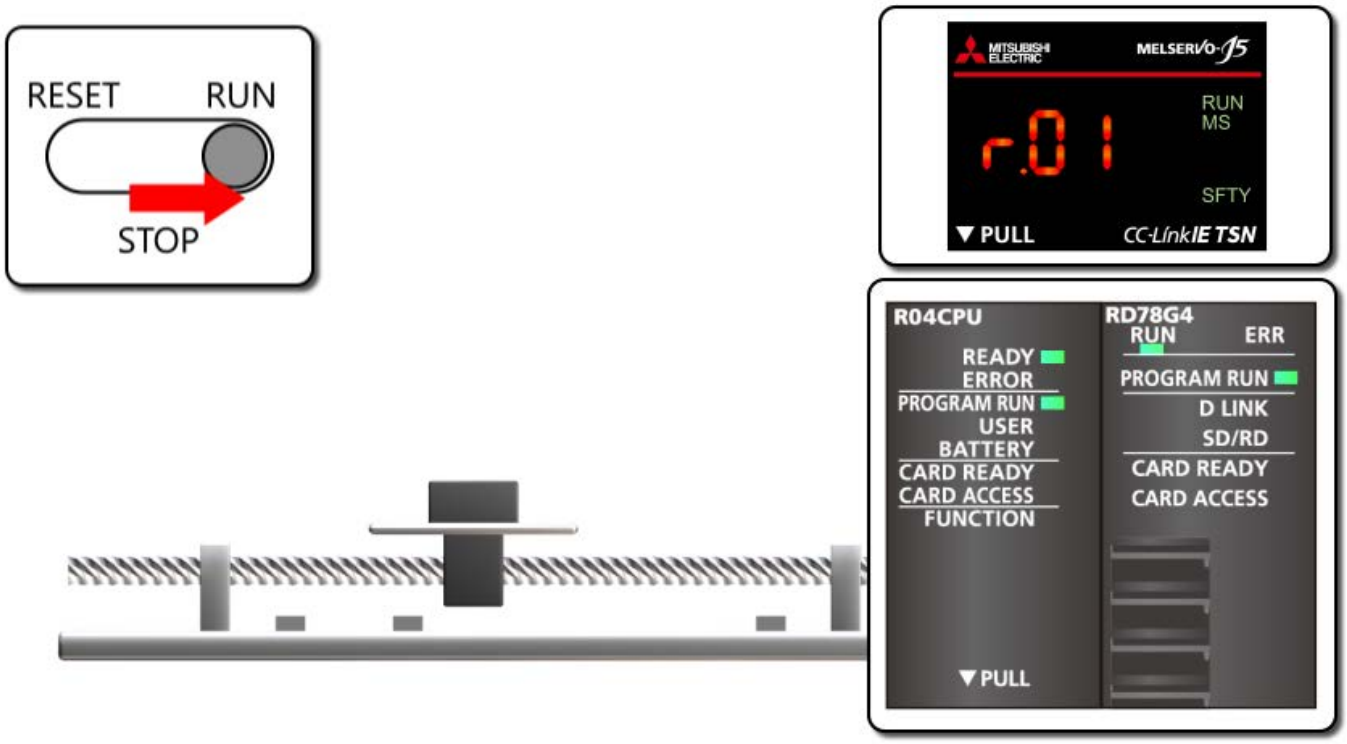


Kiểm tra hoạt động của chương trình mẫu.
Trước khi bắt đầu hoạt động, hãy đảm bảo chương trình và thông số đã được ghi vào PLC CPU và mô-đun Chuyển động.

4.6 Kiểm tra hoạt động của chương trình mẫu

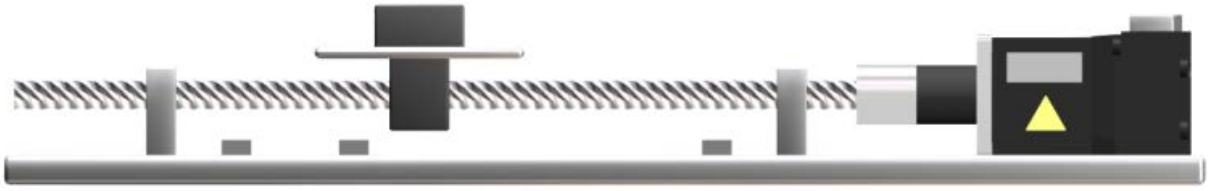


Đặt công tắc RUN/STOP/RESET của PLC CPU thành CHẠY.
Đèn READY và đèn PROGRAM RUN của bộ điều khiển khả trình đã bật.
Đèn RUN của mô-đun Chuyển động bật.



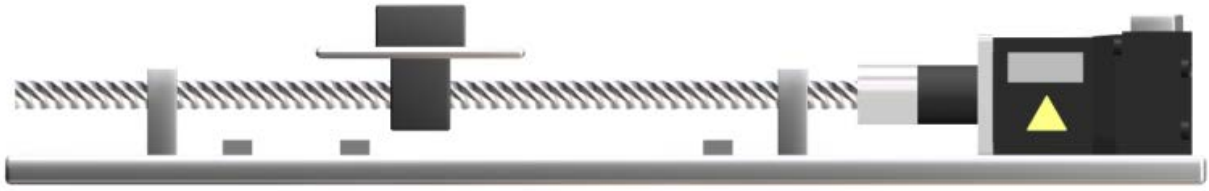
Chờ cho đến khi đèn PROGRAM RUN của mô-đun Chuyển động bật. "r.01" được hiển thị trên bộ khuếch đại servo. (Các điểm sáng lên.) Mô-tơ servo tiến vào trạng thái BẬT servo.

4.6 Kiểm tra hoạt động của chương trình mẫu



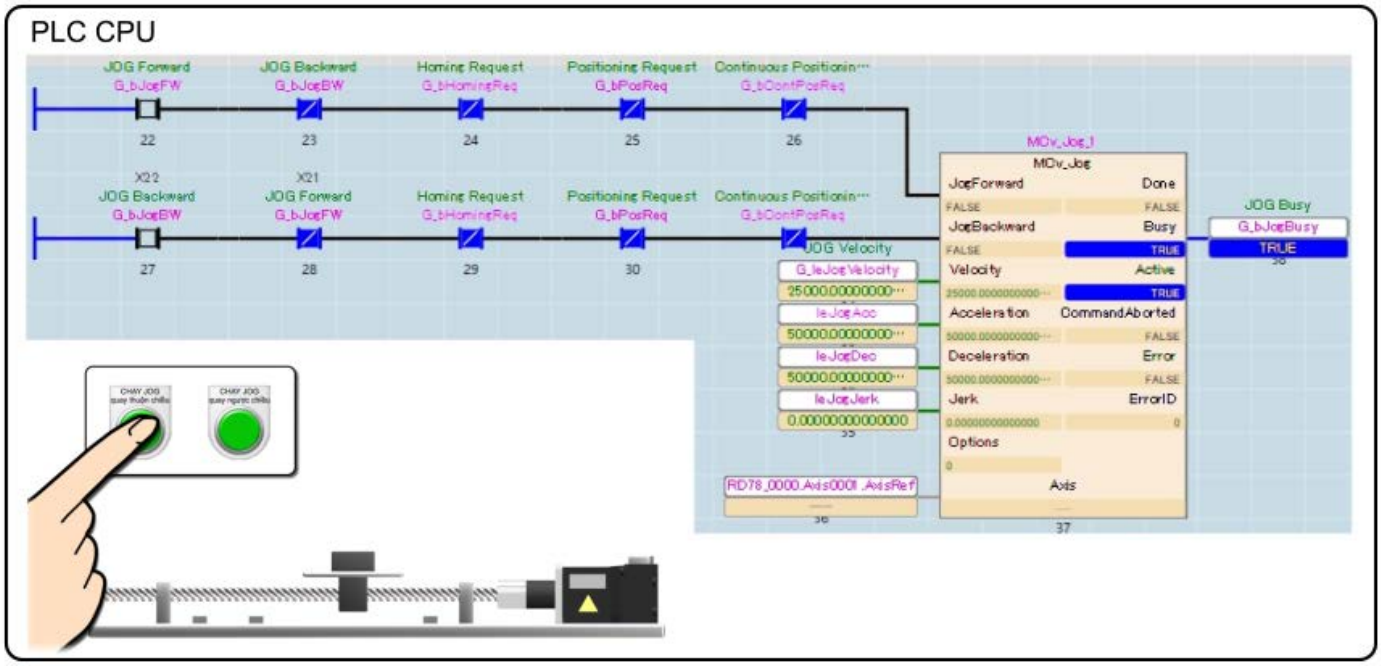
Bật X20 để thực thi TẮT servo.
"r.01" được hiển thị trên bộ khuếch đại servo. (Các điểm nhấp nháy.)
Tắt X20 để thực hiện lại hoạt động BẬT servo.

4.6 Kiểm tra hoạt động của chương trình mẫu



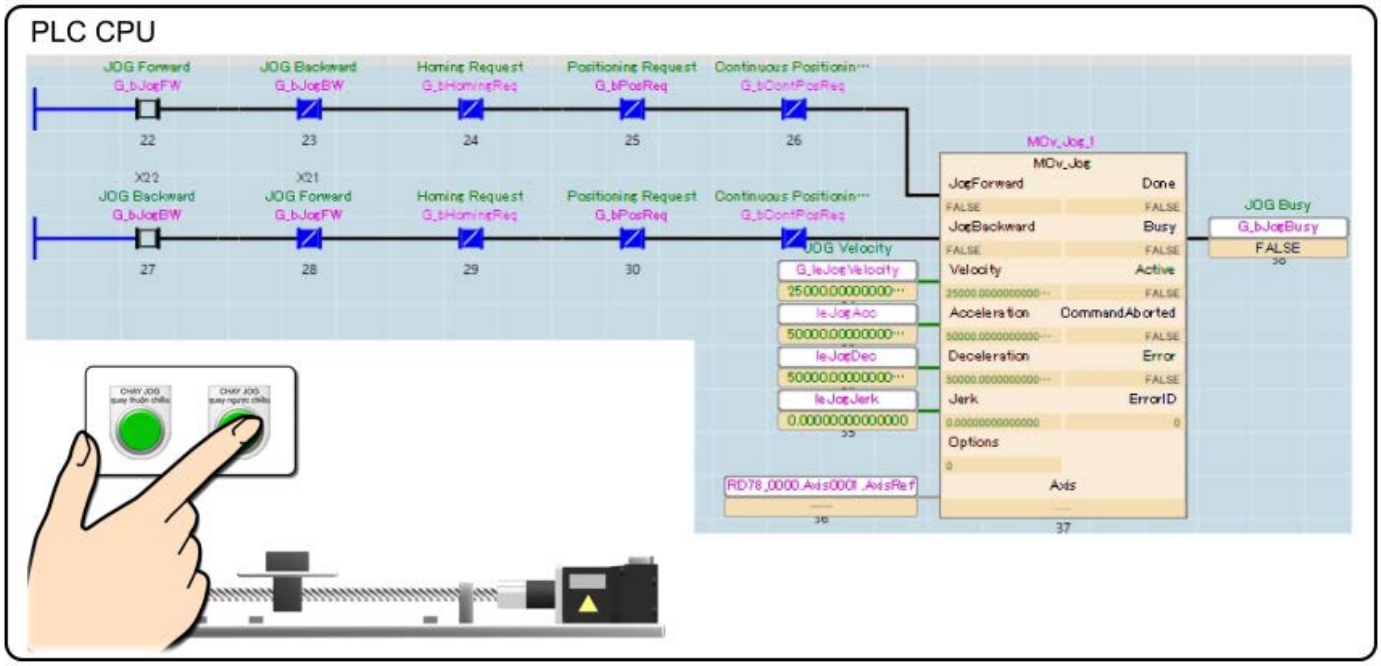
Bật CHẠY JOG quay thuận chiều (X21) để di chuyển theo hướng tăng địa chỉ và tắt để dừng lại.
Bật CHẠY JOG quay ngược chiều (X22) để di chuyển theo hướng giảm địa chỉ và tắt để dừng lại.

4.6 Kiểm tra hoạt động của chương trình mẫu



Kiểm tra hoạt động giám sát chương trình.
Khi X21 bật, đầu vào JogForward của MCv_Jog_1 sẽ bật.
Hoạt động CHẠY JOG quay bình thường được thực hiện.
Đầu ra Bận và "G_bJogBusy" bật trong khi hoạt động.

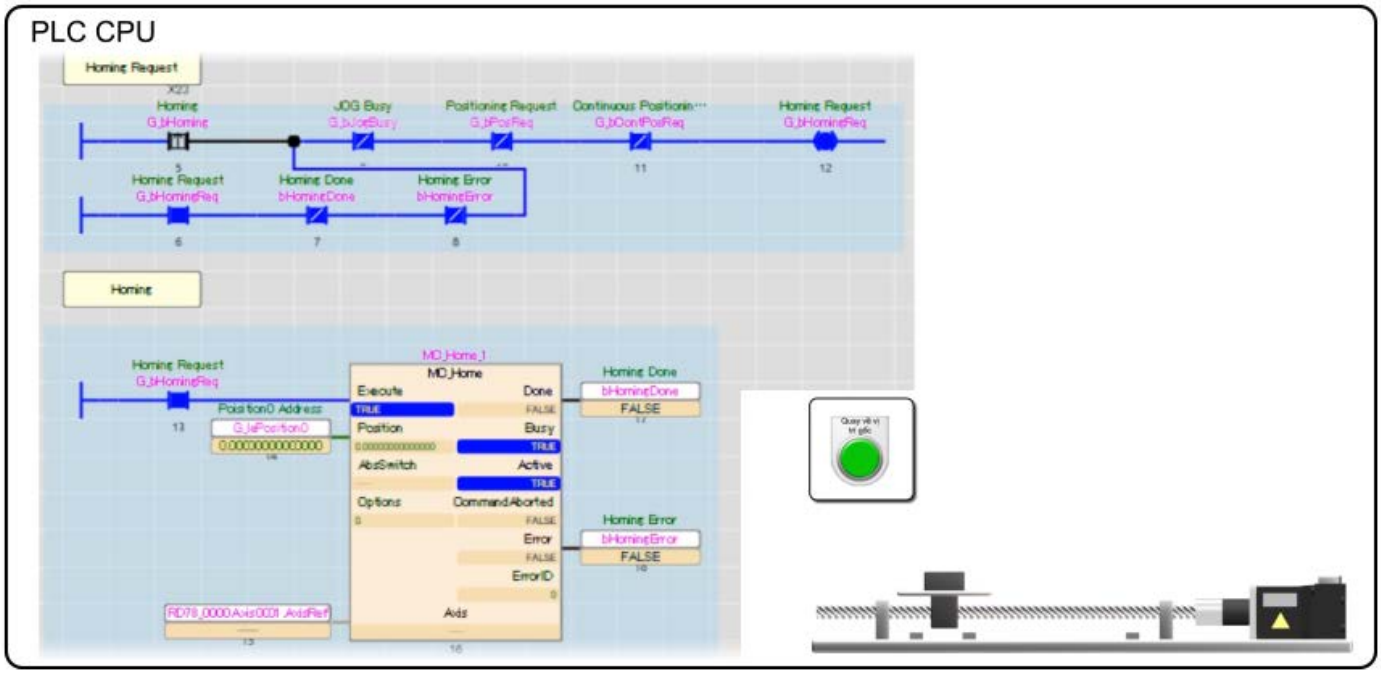
4.6 Kiểm tra hoạt động của chương trình mẫu



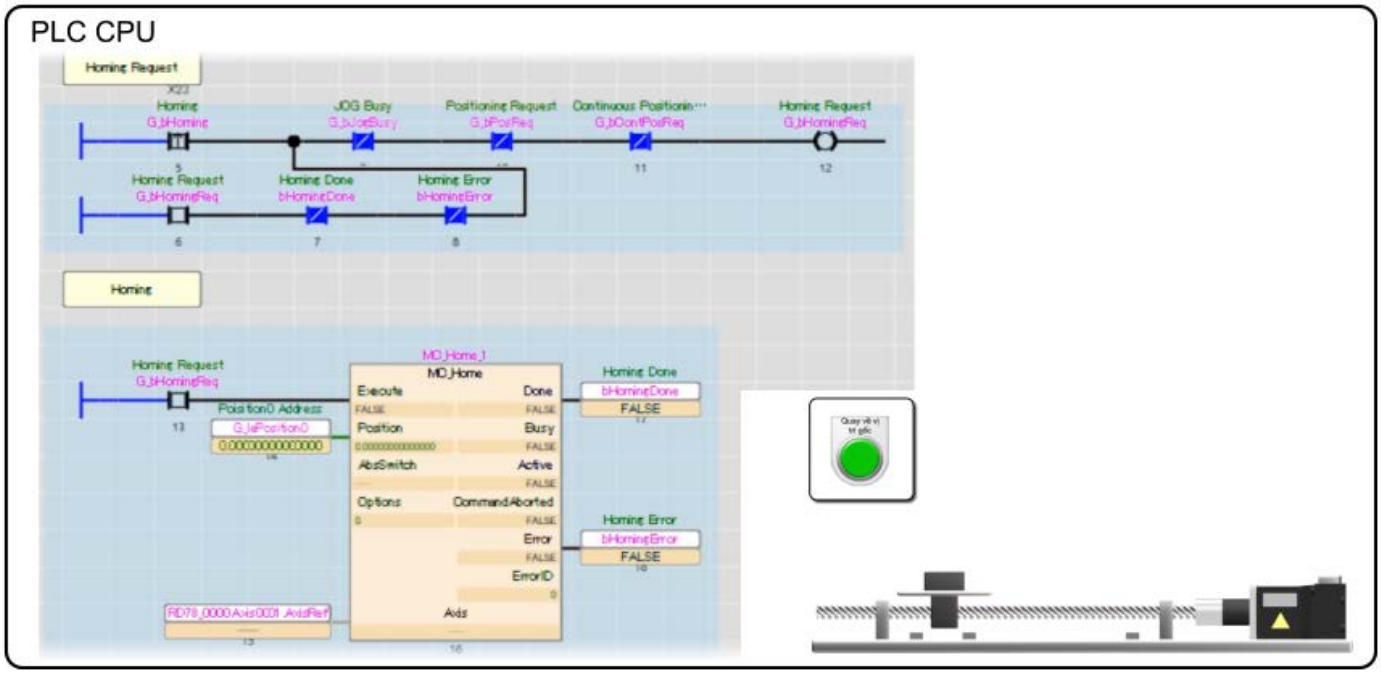
Khi X22 bật, đầu vào JogBackward của MCv_Jog_1 sẽ bật.
Hoạt động CHẠY JOG quay ngược chiều được thực hiện.
Đầu ra Bận và "G_bJogBusy" bật trong khi hoạt động.

Bật quay về vị trí gốc (X23) để bắt đầu quy trình quay về vị trí gốc.
Thực thi hoạt động quay về vị trí gốc với phương pháp proximity dog
(33 được trừ từ Pr.PT45)
Trục dừng xa hơn một chút so với dog và đặt điểm đó làm vị trí gốc.

4.6 Kiểm tra hoạt động của chương trình mẫu



Kiểm tra hoạt động giám sát chương trình.
Khi X23 bật, địa chỉ định vị gốc sẽ được lưu vào nhãn.
"G_bHomingReq", lệnh thực thi của MC_Home_1, được bật và giữ lại.

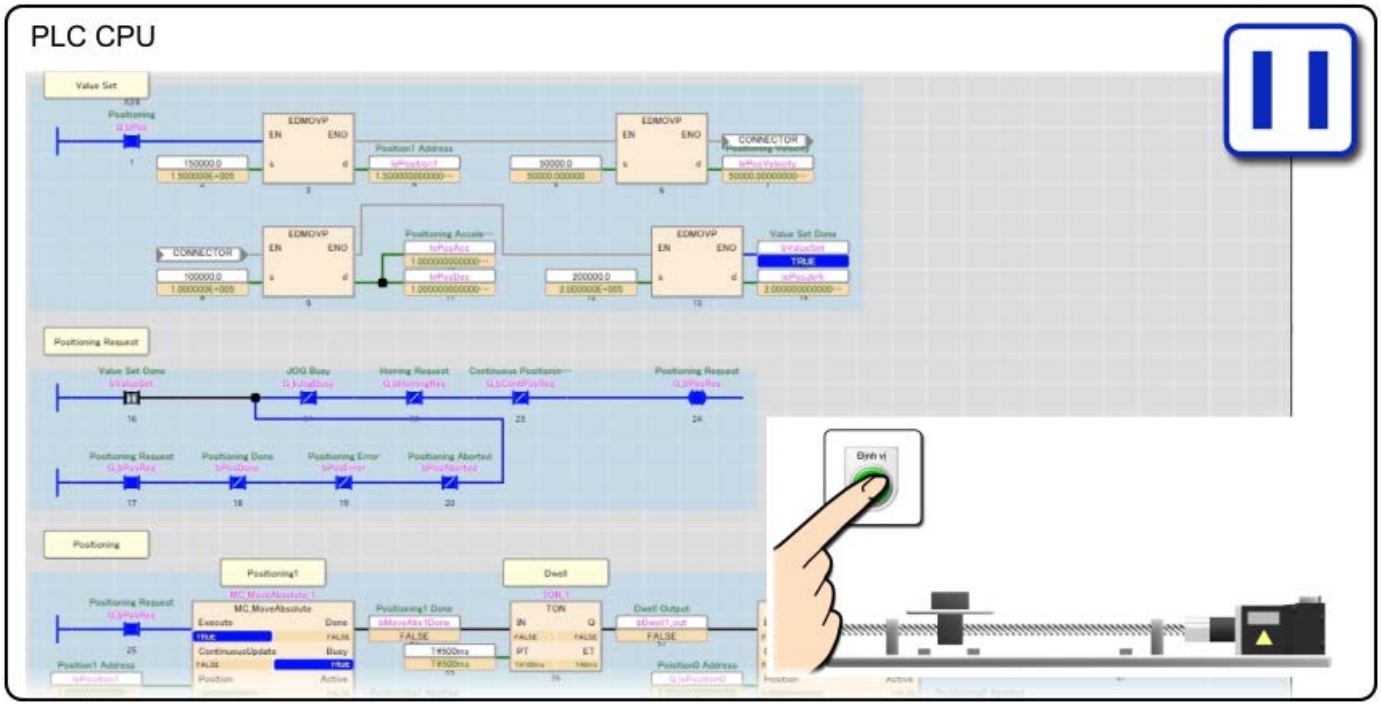


Hoạt động quay về vị trí gốc bắt đầu.
Khi quy trình quay về vị trí gốc hoàn tất, đầu ra Xong và "bHomingDone" sẽ bật và việc giữ lại "G_bHomingReq" bị hủy.

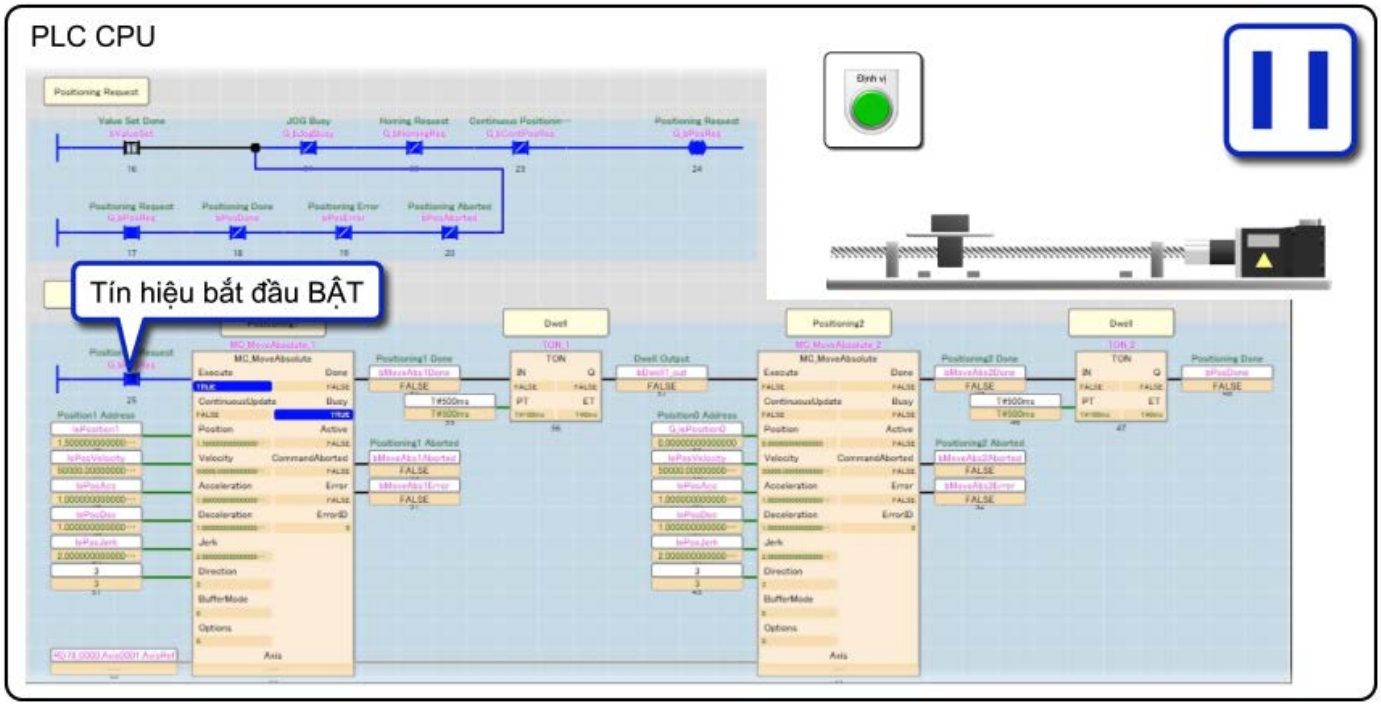
4.6 Kiểm tra hoạt động của chương trình mẫu

Bật quy trình bắt đầu định vị (X24) sẽ bắt đầu chuyển động tịnh tiến. Trục tiến lên 150 mm rồi dừng trong 0,5 giây và lùi lại 150 mm rồi dừng trong 0,5 giây.

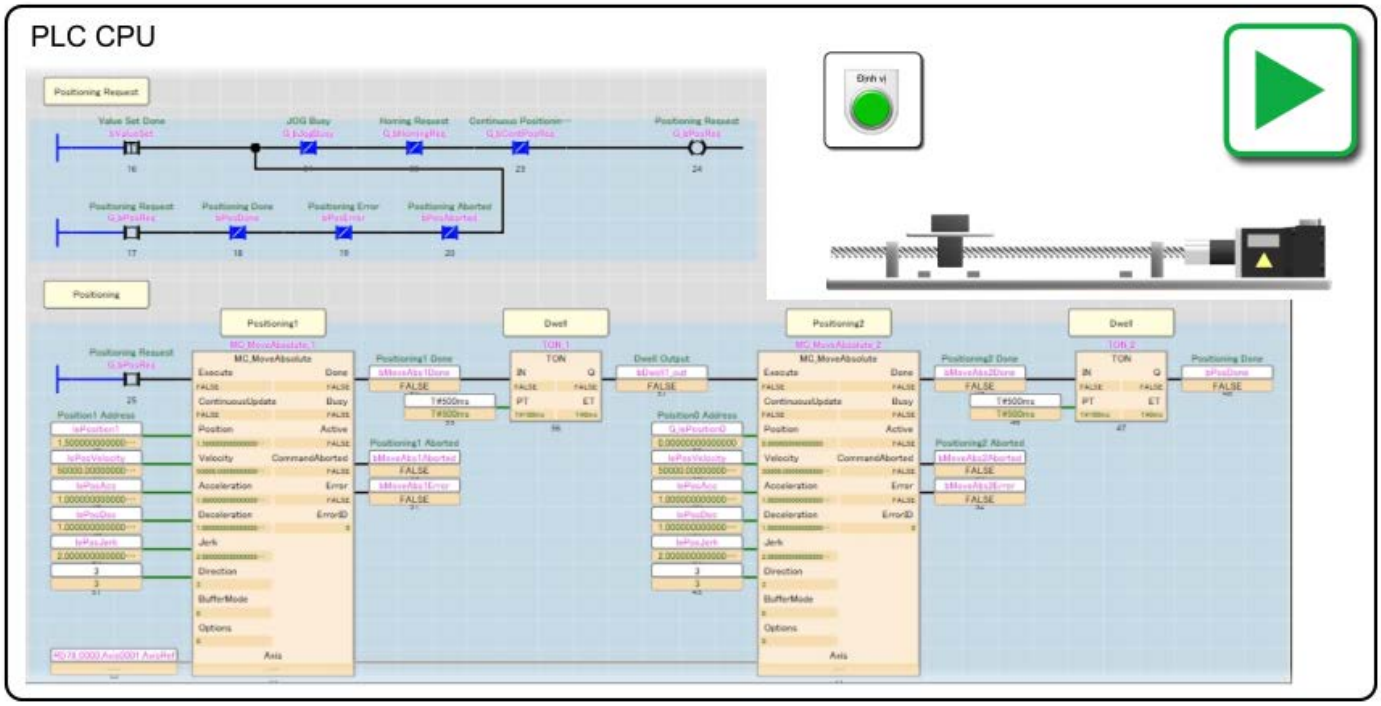
4.6 Kiểm tra hoạt động của chương trình mẫu



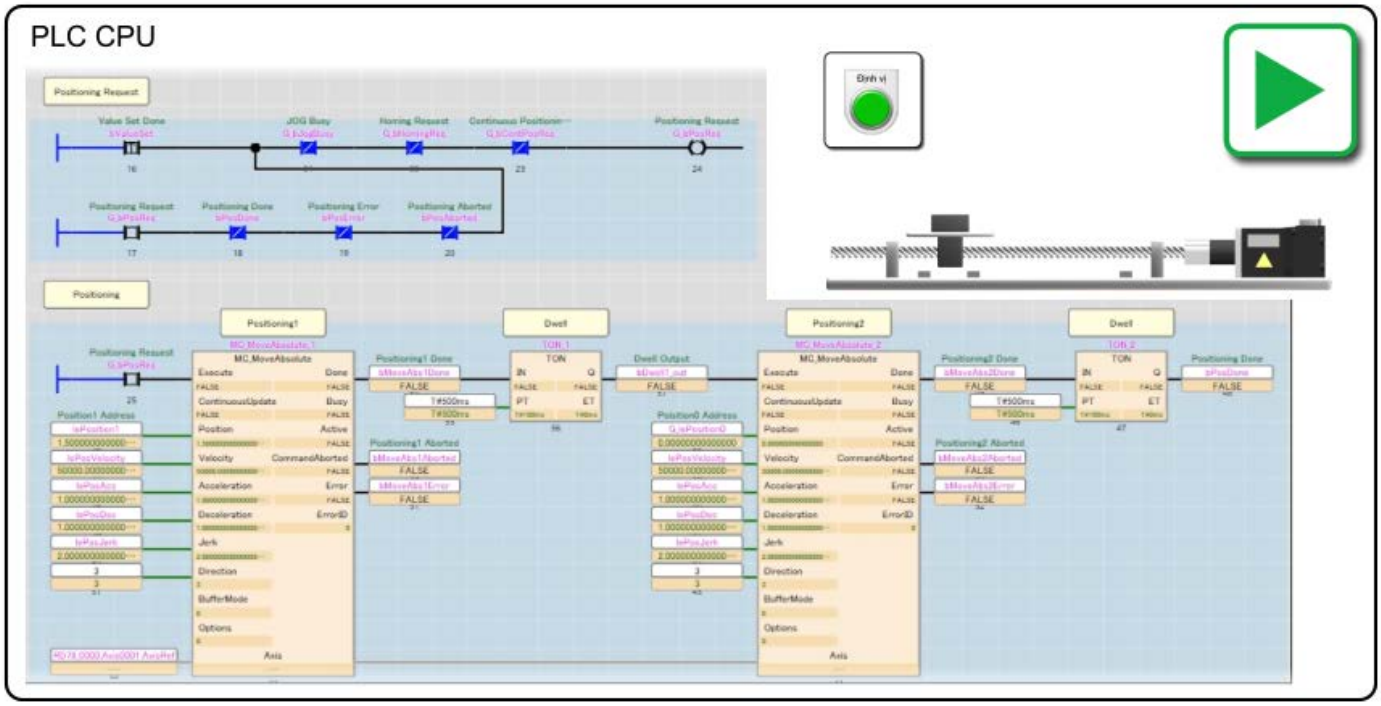
Kiểm tra hoạt động giám sát chương trình.
Khi X24 bật, dữ liệu để định vị được lưu trữ vào từng nhãn và "bValueSet" bật.
"G_bPosReq", lệnh thực thi của MC_MoveAbsolute_1, được bật và giữ lại tại sườn lên của "bValueSet".



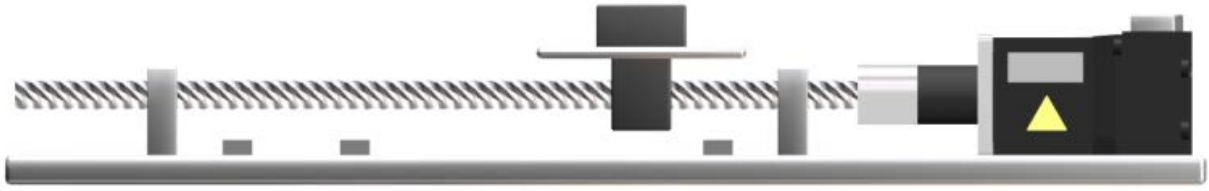
Khi "G_bPosReq" bật, MC_MoveAbsolute_1 bắt đầu và mô tơ servo bắt đầu chạy.



Khi việc định vị bằng MC_MoveAbsolute_1 hoàn tất, TON_1, tức là dừng, sẽ hoạt động.
Khi hết 500 ms, MC_MoveAbsolute_2 được thực thi và mô tơ servo bắt đầu chạy.

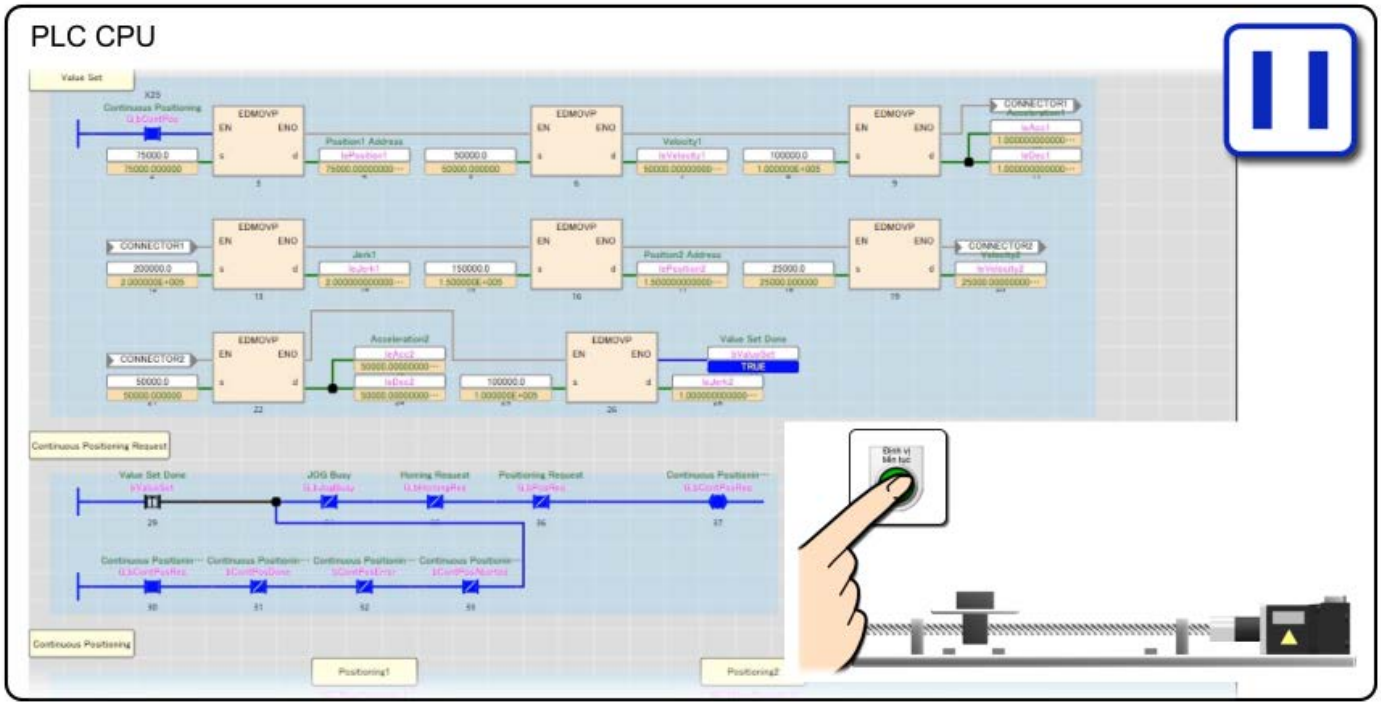


Khi việc định vị bằng MC_MoveAbsolute_2 hoàn tất, TON_2, tức là dừng, sẽ hoạt động.
Khi hết 500 ms, việc lưu giữ "G_bPosReq" sẽ bị xóa và đặt lại về trạng thái ban đầu.

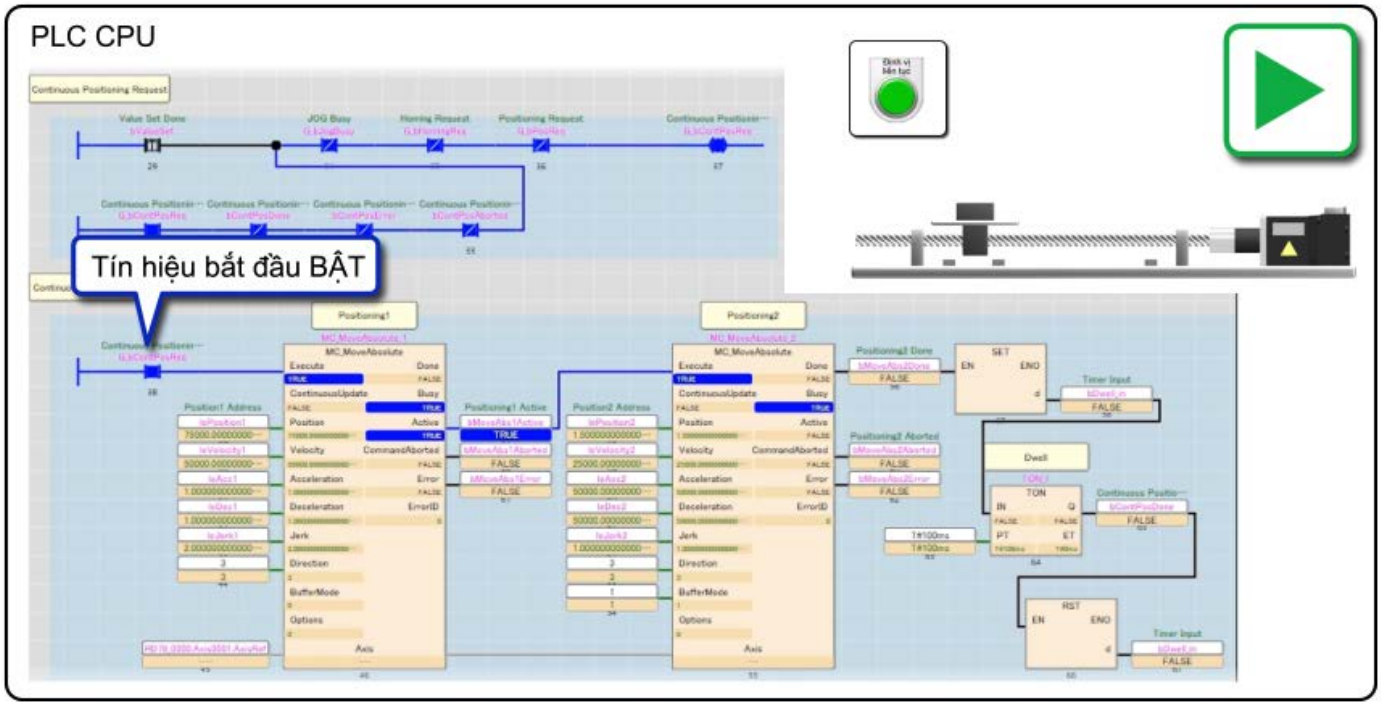


Bật bắt đầu định vị tiếp nối (X25) để bắt đầu vận hành chế độ đệm (mc_Buffered).

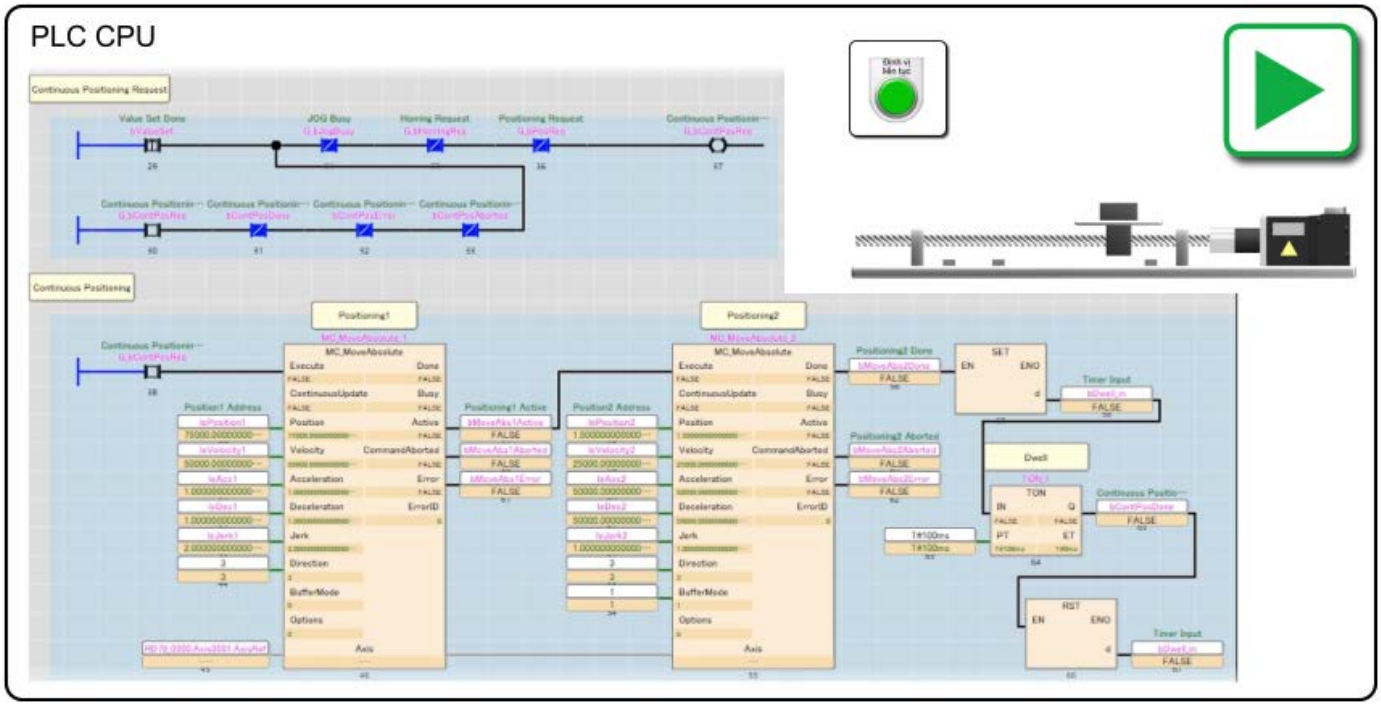
4.6 Kiểm tra hoạt động của chương trình mẫu



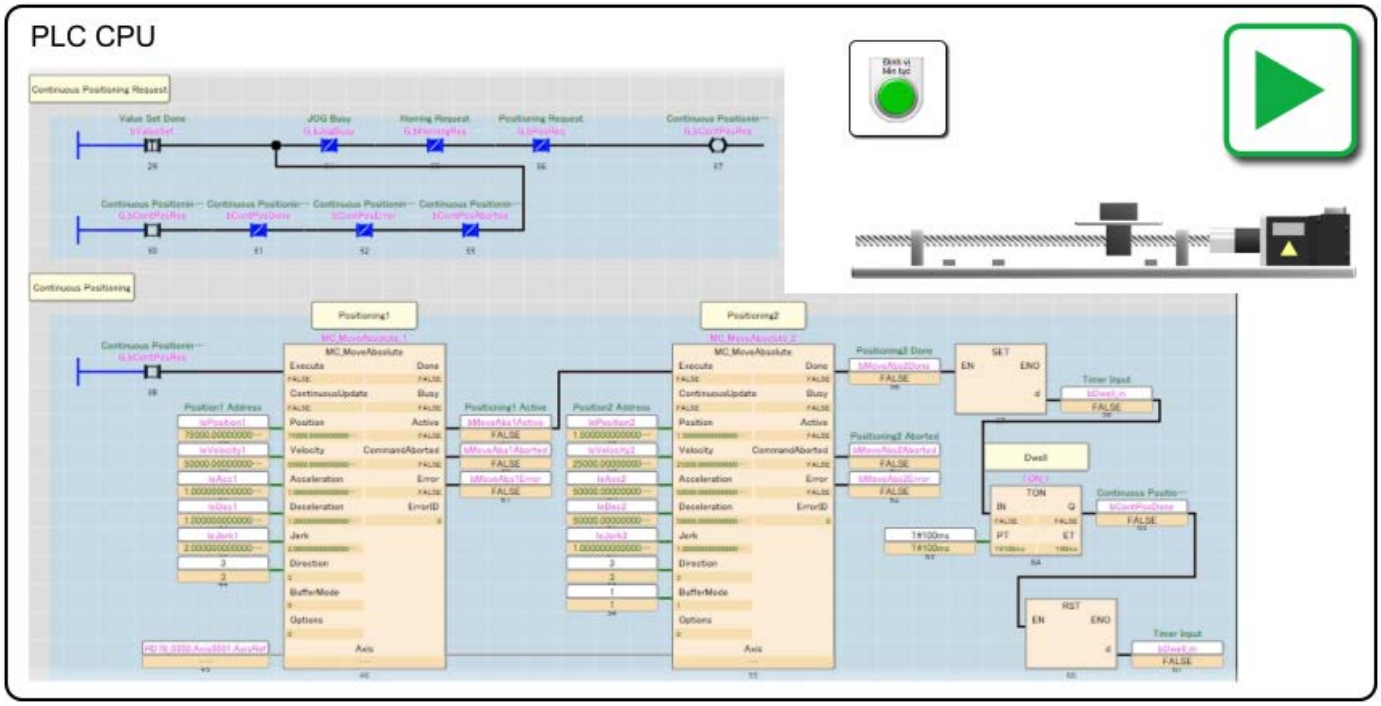
Kiểm tra hoạt động giám sát chương trình.
Khi X25 bật, dữ liệu để định vị được lưu trữ vào từng nhãn và "bValueSet" bật.
"G_bContPosReq", lệnh thực thi của MC_MoAbsolute_1, được bật và giữ lại tại sườn lên của "bValueSet".



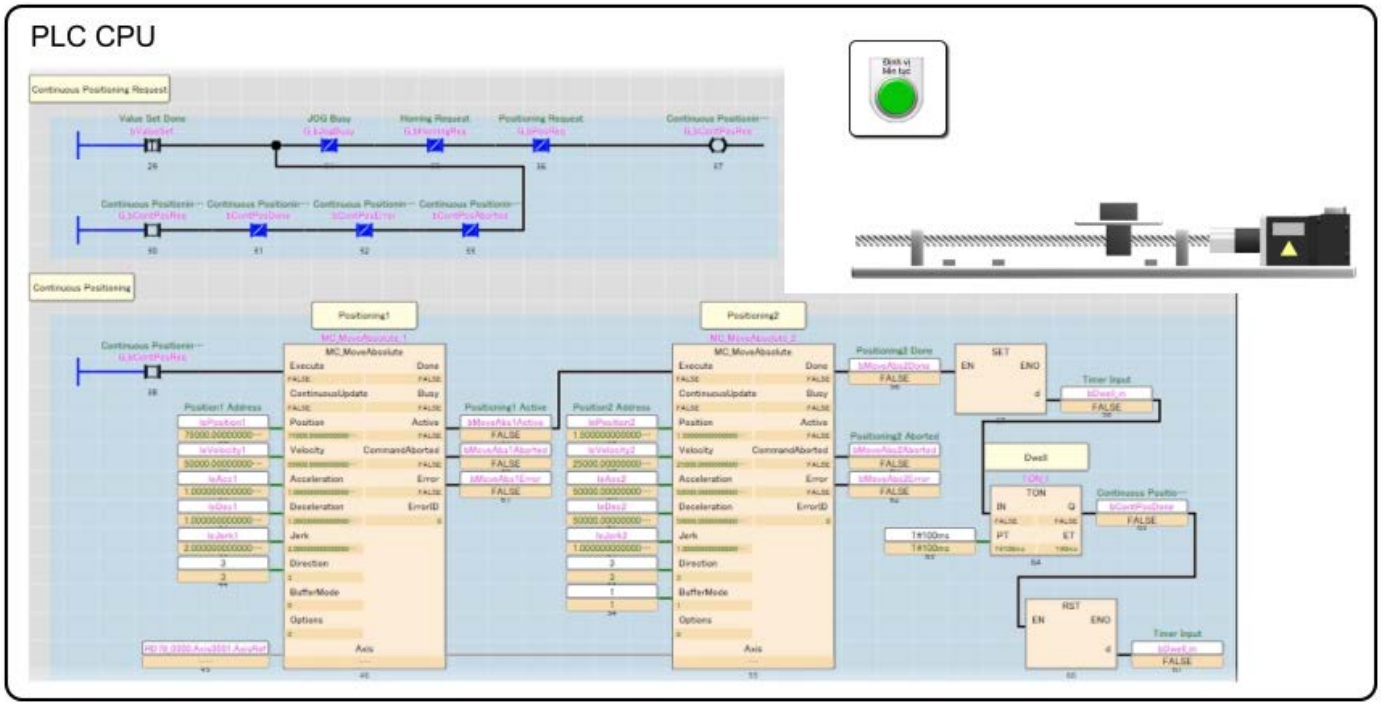
Khi "G_bContPosReq" bật, MC_MoveAbsolute_1 bắt đầu và mô tơ servo bắt đầu chạy.
Tại thời điểm này, do đầu ra Hoạt động là lệnh thực thi của MC_MoveAbsolute_2 nên MC_MoveAbsolute_2 được đệm.



Khi hoạt động của MC_MoveAbosolute_1 hoàn tất, MC_MoveAbsolute_2 được đệm sẽ được thực thi.
Khi hoạt động của MC_MoveAbosolute_2 hoàn tất, TON_1, tức là dừng, sẽ được thực thi.



Khi hết 100 ms, việc lưu giữ "G_bContPosReq" sẽ bị xóa và đặt lại về trạng thái ban đầu.



Thao tác này hoàn tất việc kiểm tra hoạt động.
Sang trang tiếp theo.

4.7 Tóm tắt chương này

Trong chương này, bạn đã học về:

- Đăng ký Thư viện FB mô-đun chuyển động
- Tạo dự án
- Cách sử dụng Motion Control FB
- Mô tả về chương trình mẫu
- Kiểm tra hoạt động của chương trình mẫu

Những điểm quan trọng

Đăng ký Thư viện FB mô-đun chuyển động	• Thư viện FB phải được đăng ký với GX Works3 để sử dụng Motion control FB trong PLC CPU.
Tạo dự án	• Cấu hình tham số trục và cài đặt khác như khi lập trình mô-đun Chuyển động.
Cách sử dụng Motion Control FB	• Có thể đặt FB điều khiển chuyển động vào trình biên tập chương trình bằng cách kéo và thả từ thẻ Thư viện của cửa sổ Chọn yếu tố của GX Works3. • Kết nối tiếp điểm và nhãn với đầu vào/đầu ra của FB.
Mô tả về chương trình mẫu	• Bạn đã tạo một chương trình tương tự như các chương trình mẫu trong chương 2 và chương 3 chỉ bằng cách sử dụng PLC CPU.
Kiểm tra hoạt động của chương trình mẫu	• Bạn đã kiểm tra hoạt động của chương trình mẫu trong video.

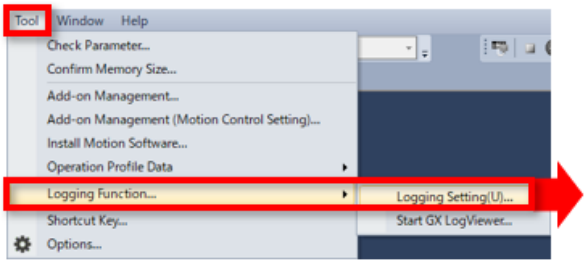
Chương 5 Ghi nhật ký

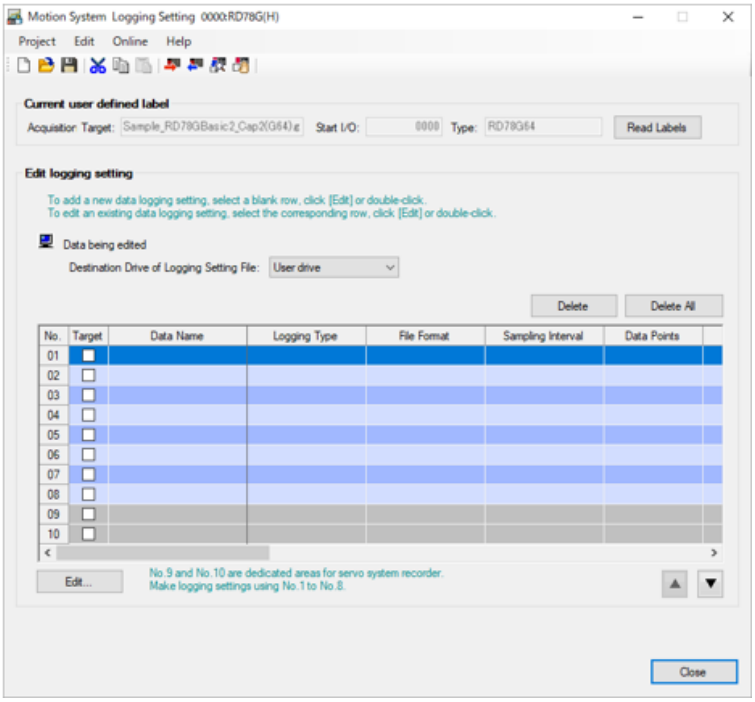
Chương này mô tả cách ghi nhật ký dữ liệu của mô-đun Chuyển động và hiển thị dữ liệu trong biểu đồ.
Trong khóa học này, chương trình khởi động định vị thuộc chương trình mẫu ở chương 2 và chương 3 sẽ được ghi nhật ký làm ví dụ.

(Lưu ý) Không thể ghi nhật ký chương trình trong chương 4 bằng quy trình được mô tả trong chương này.
Phải sử dụng "CPU Module Logging Configuration Tool".

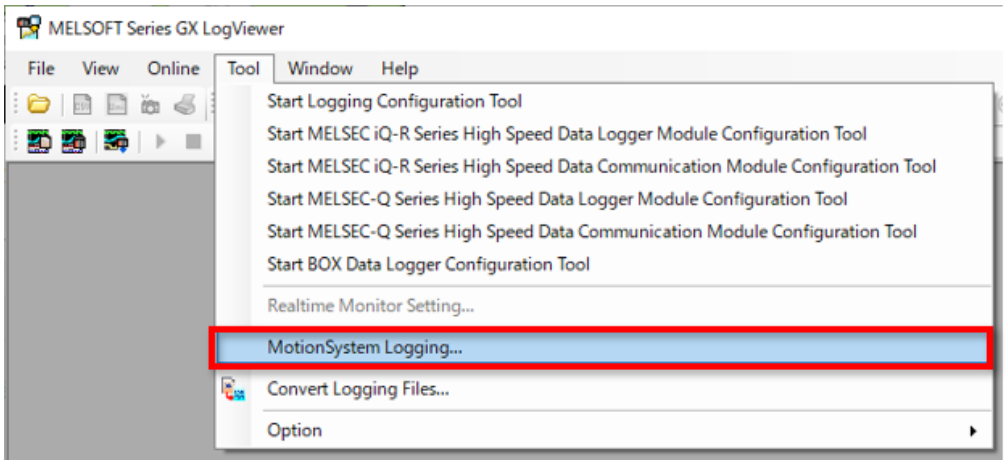
5.1 Khởi động Công cụ cấu hình ghi nhật ký

Chọn [Tool] → [Logging Function] → [Logging Setting] từ thanh công cụ của màn hình Motion Control Setting Function.
Công cụ cài đặt ghi nhật ký hệ thống chuyển động khởi động.



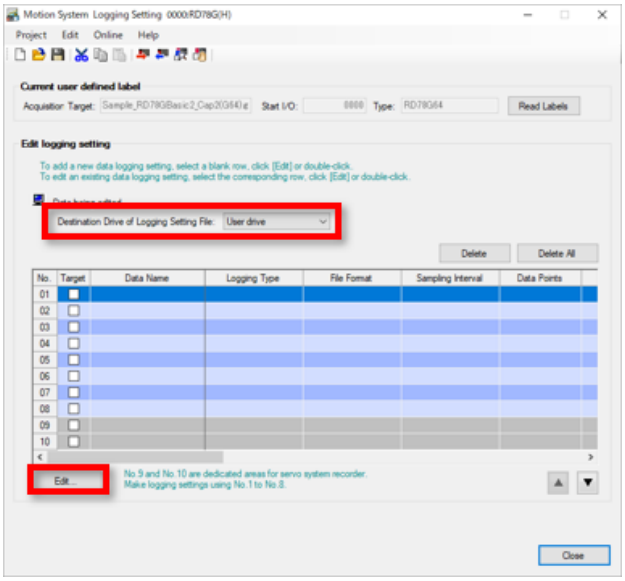


[Điểm]
Bạn có thể khởi động Công cụ cài đặt ghi nhật ký hệ thống chuyển động từ [Tool] → [MotionSystem Logging] trong GX LogViewer.

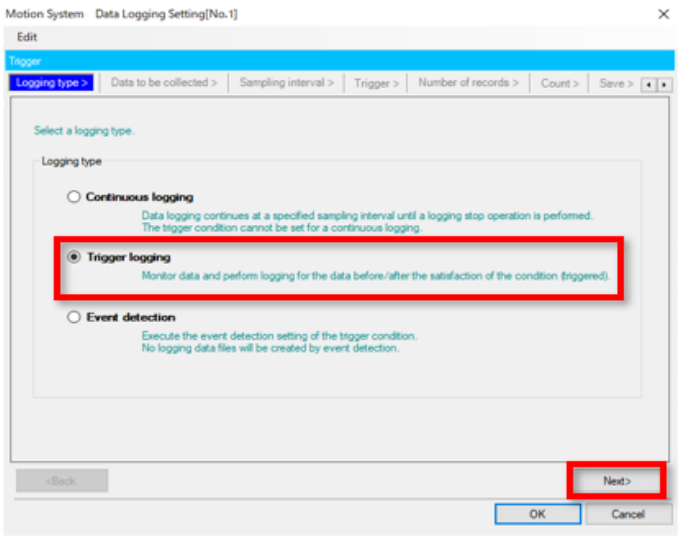


5.2 Cài đặt dữ liệu sẽ được ghi nhật ký

(1) Đặt đích lưu cho dữ liệu ghi nhật ký trong trường chỉnh sửa cài đặt ghi nhật ký của công cụ cài đặt ghi nhật ký hệ thống chuyển động.
Sau đó nhấp vào nút [Edit].
Màn hình Cài đặt ghi nhật ký dữ liệu sẽ được hiển thị.



(2) Chọn [Logging type] từ ghi nhật ký liên tục, kích hoạt ghi nhật ký và phát hiện sự kiện.
Khóa học này sẽ mô tả thông tin chi tiết về kích hoạt ghi nhật ký.
Chọn kích hoạt ghi nhật ký, sau đó chọn nút [Next].



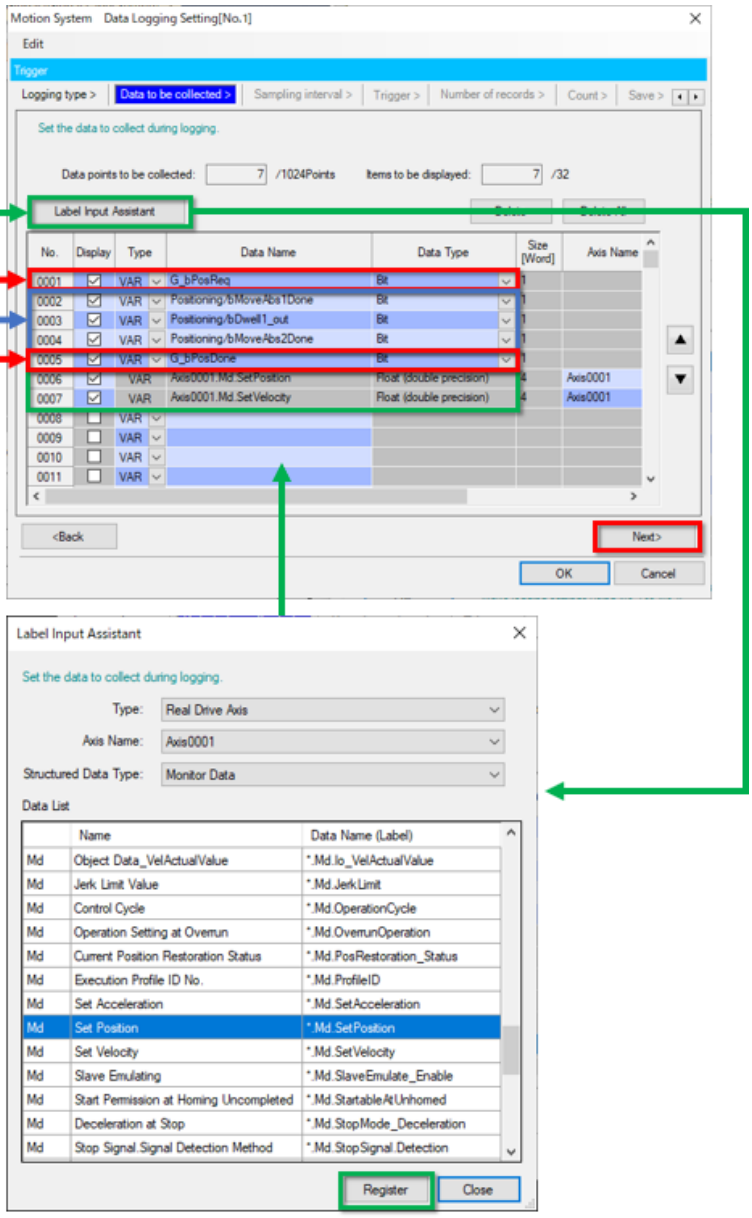
5.2 Cài đặt dữ liệu sẽ được ghi nhật ký

(3) Nhãn của dữ liệu cần ghi nhật ký được đăng ký trong [Data to be collected].

1) Nhãn toàn cục
Nhập tên nhãn toàn cục vào trường tên dữ liệu.
Chọn loại dữ liệu của nhãn trong trường loại dữ liệu.

2) Nhãn cục bộ
Nhập tên dữ liệu bằng định dạng "tên chương trình/tên nhãn cục bộ".
Chọn loại dữ liệu của nhãn trong trường loại dữ liệu.

3) Loại dữ liệu có cấu trúc
Nhấp vào nút [Label Input Assistant] và chọn thành phần của loại dữ liệu có cấu trúc từ danh sách.
Chọn thành phần từ danh sách và nhấp vào nút [Register] để ánh xạ dữ liệu sẽ được thu thập.



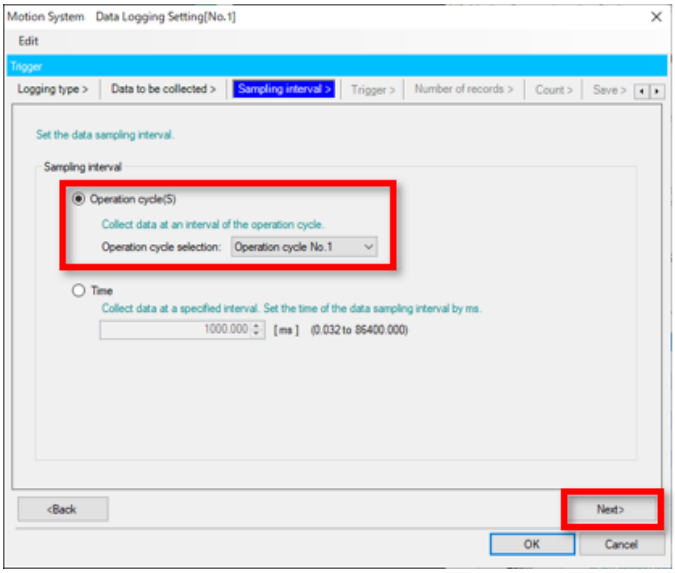
Trong khóa học này, dữ liệu sau đây được ghi nhật ký làm ví dụ.

Tên dữ liệu
G_bPosReq
Positioning/bMoveAbs1Done
Positioning/bDwell1_out
Positioning/bMoveAbs2Done
G_bPosDone
Axis0001.Md.SetPosition
Axis0001.Md.SetVelocity

Nhấp vào nút [Next] khi đăng ký hoàn tất.

(4) Đặt khoảng thời gian lấy mẫu trong [Sampling Interval].
Sử dụng chu trình hoạt động số 1 để lấy mẫu trong khóa học này.

Sau khi chọn khoảng thời gian lấy mẫu, hãy nhấp vào nút [Next].



5.2 Cài đặt dữ liệu sẽ được ghi nhật ký (Tiếp theo)

(5) Điều kiện để bắt đầu ghi nhật ký được đặt trong [Trigger].
Bit khởi động, là tín hiệu bắt đầu định vị, được sử dụng làm bộ kích hoạt trong khóa học này.

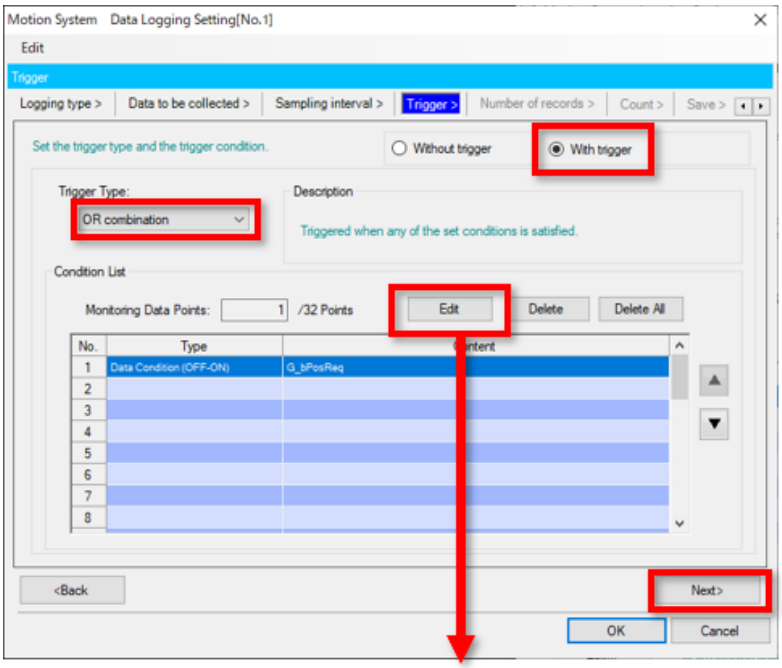
1) Chọn [With trigger].

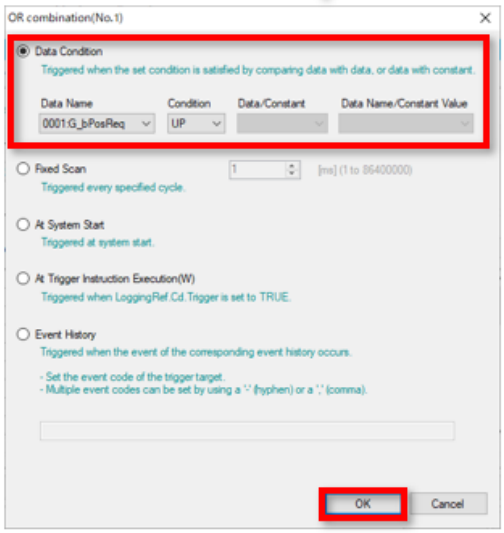
2) Chọn "OR combination" làm loại kích hoạt.

3) Chọn Số 1 trong danh sách điều kiện và nhấp vào nút chỉnh sửa. Cửa sổ phụ sẽ hiện ra.

4) Chọn "Data Condition" và chọn "0001:G_bPosReq" làm tên dữ liệu.
Chọn "UP" làm điều kiện.
Khi lựa chọn hoàn tất,
hãy nhấp vào nút [OK].

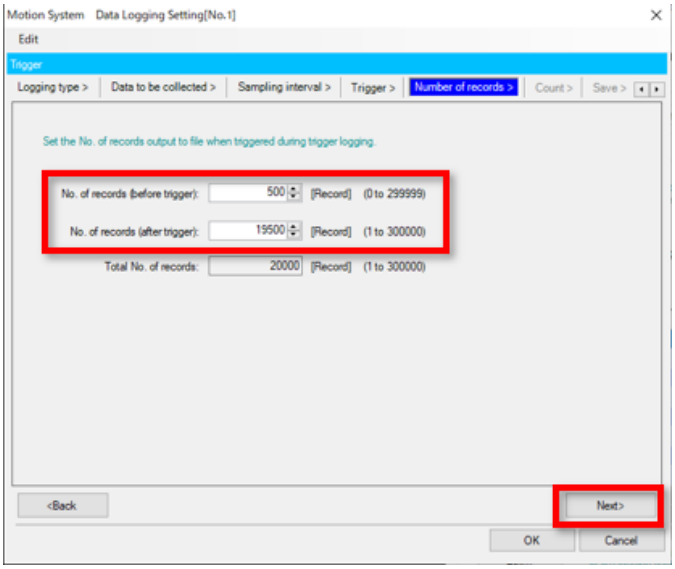
5) Sau khi bạn quay về màn hình ban đầu, hãy nhấp vào nút [Next].



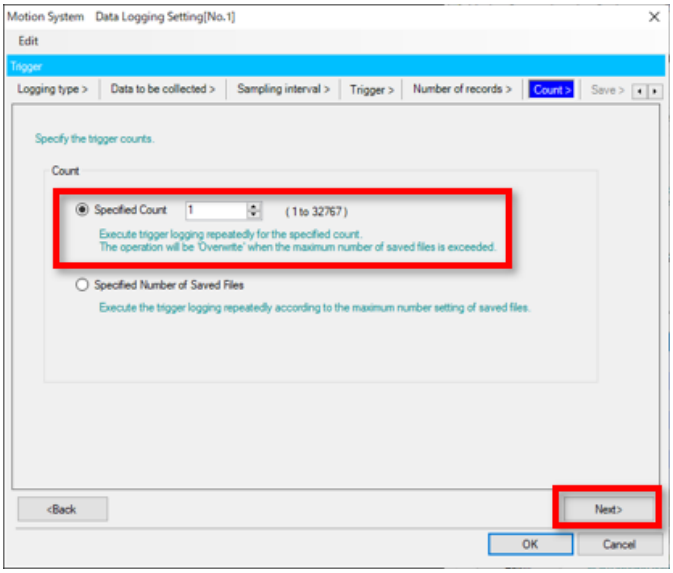


5.2 Cài đặt dữ liệu sẽ được ghi nhật ký (Tiếp theo)

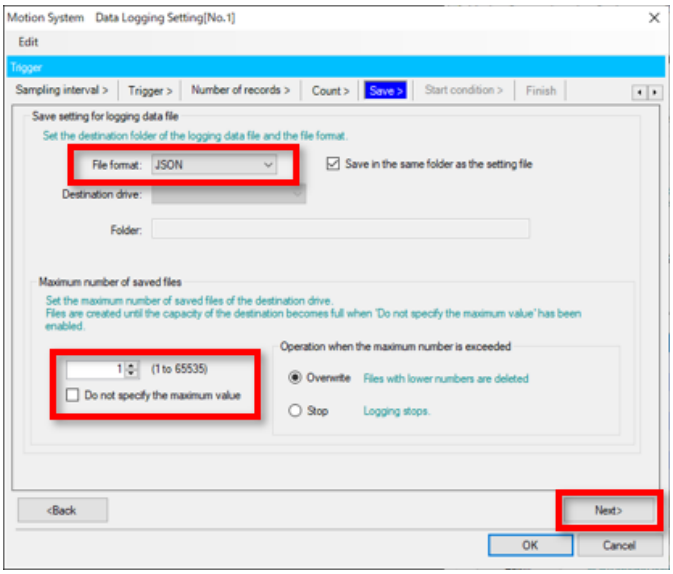
(6) Số lượng điểm lấy mẫu được đặt trong [Number of records].
Trong khóa học này, Số bản ghi (trước khi kích hoạt) được đặt thành "500" và Số bản ghi (sau khi kích hoạt) được đặt thành "19500".
Khi cài đặt hoàn tất, hãy nhấp vào nút [Next].



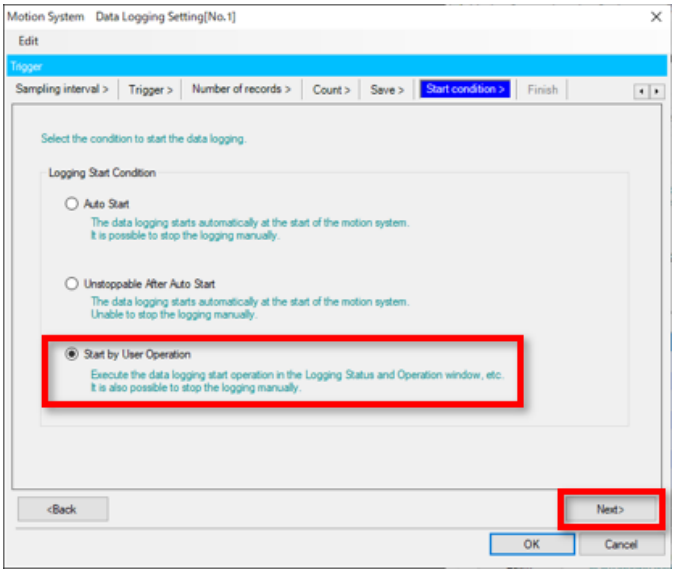
(7) Số lần ghi nhật ký được đặt trong [Count]. Số lần ghi được đặt thành 1 trong khóa học này.
Khi cài đặt hoàn tất, hãy nhấp vào nút [Next].



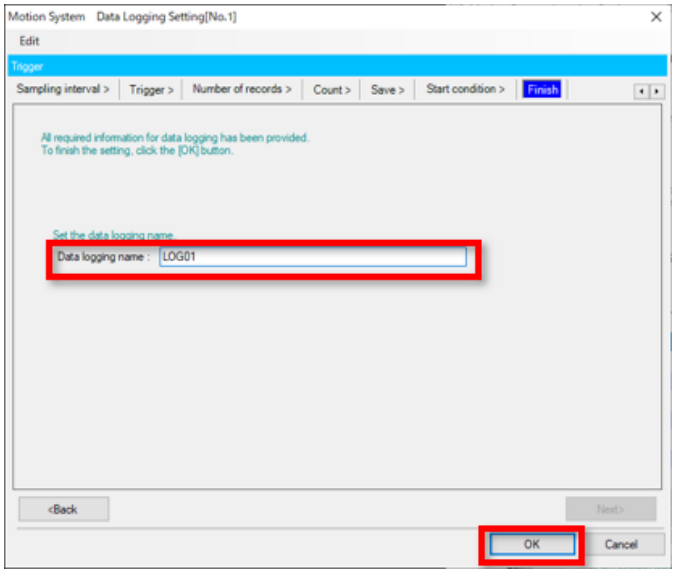
(8) Định dạng tệp và số tệp dữ liệu ghi nhật ký đã lưu sẽ được đặt trong [Save].
Trong khóa học này, giá trị mặc định (định dạng: JSON, số tệp đã lưu: 1) được đặt.
Khi cài đặt hoàn tất, hãy nhấp vào nút [Next].



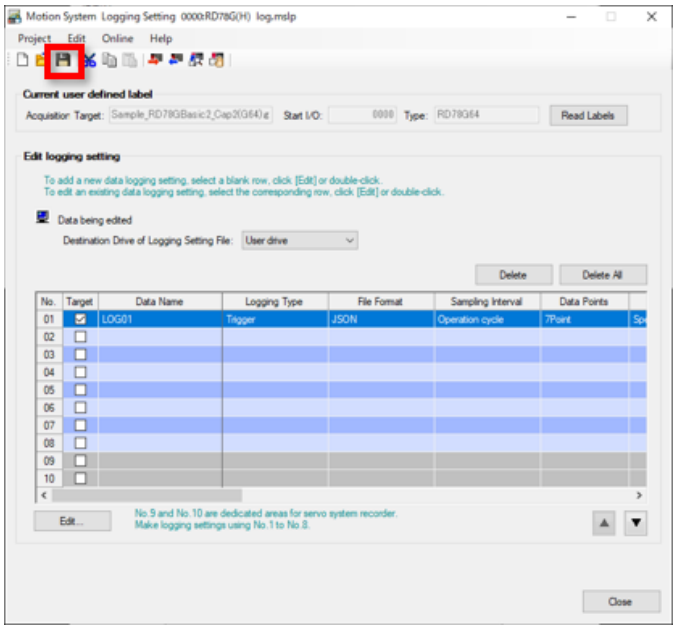
(9) Điều kiện để bắt đầu ghi nhật ký được đặt trong [Start condition].
"Start by User Operation" được đặt trong khóa học này.
Khi cài đặt hoàn tất, hãy nhấp vào nút [Next].



(10) Tên của nhật ký dữ liệu được đặt trong [Finish].
Giá trị mặc định (LOG01) được đặt trong khóa học này.
Khi cài đặt hoàn tất, hãy nhấp vào nút [Next].

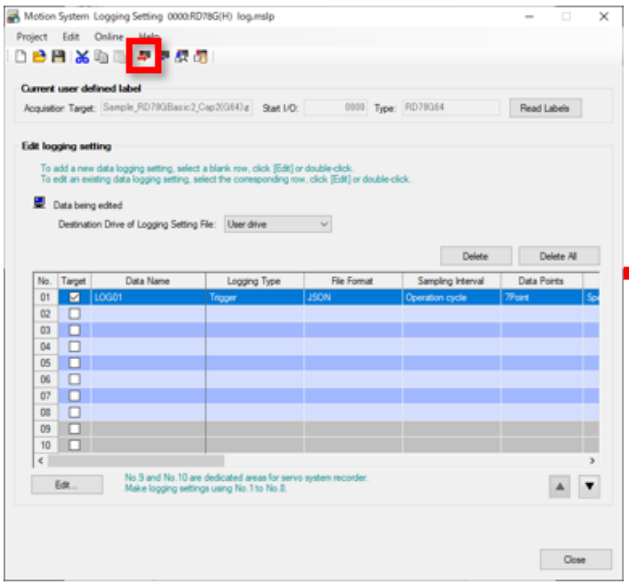


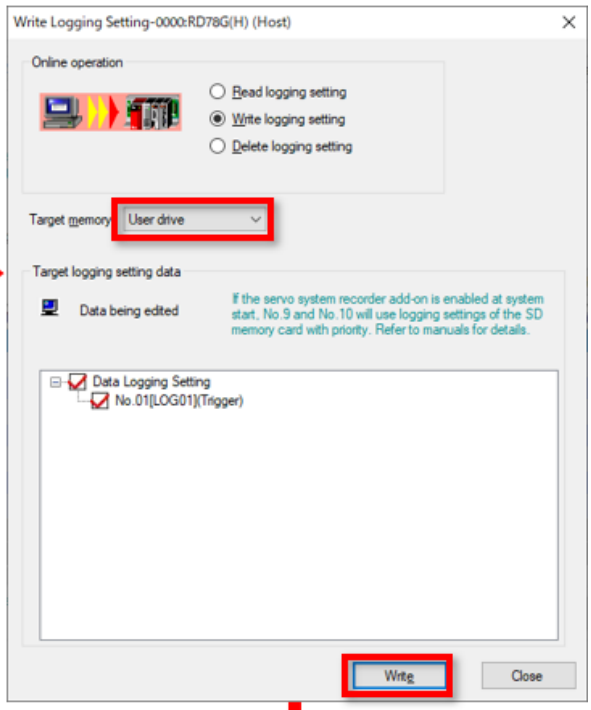
(11) Quay lại công cụ cài đặt ghi nhật ký hệ thống chuyển động.
Bạn có thể lưu lại cài đặt đã được cấu hình.
Nhấp vào biểu tượng lưu và lưu vào đích tùy chọn.

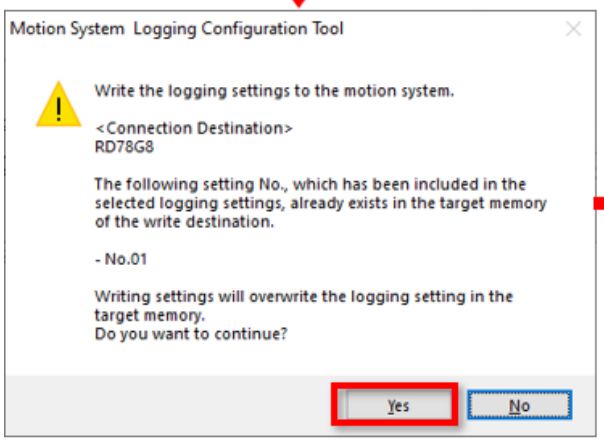


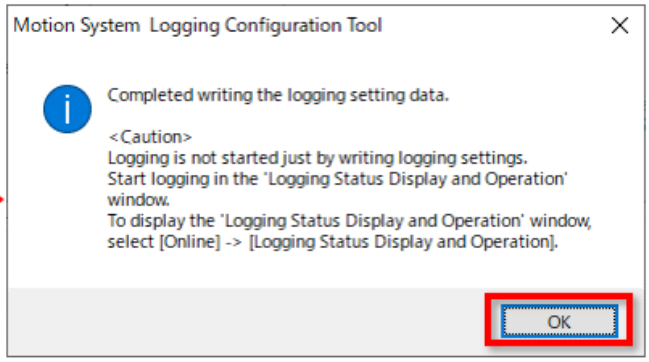
5.3 Ghi cài đặt ghi nhật ký

Thông tin cài đặt của nhật ký được ghi.
Nhấp vào biểu tượng cài đặt ghi nhật ký, chọn bộ nhớ đích và nhấp vào nút [Write].
Một cửa sổ xác nhận sẽ xuất hiện.
Nhấp vào nút [Yes] và tiếp tục. Khi ghi xong, nhấp vào nút [OK] và đóng màn hình.



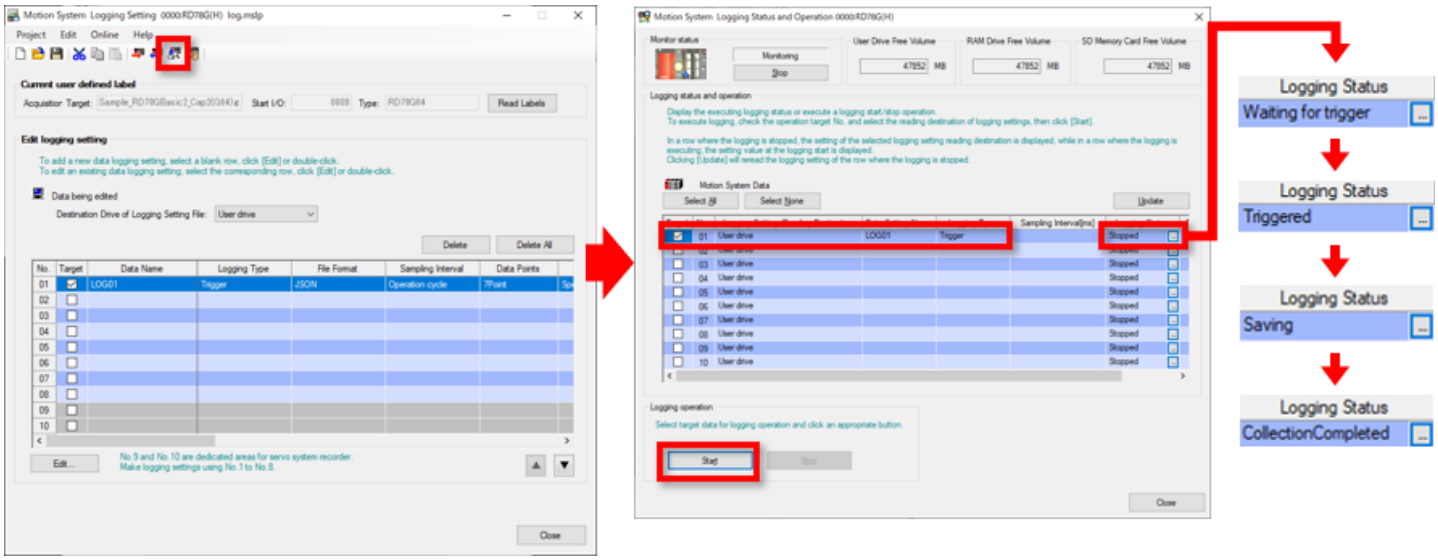






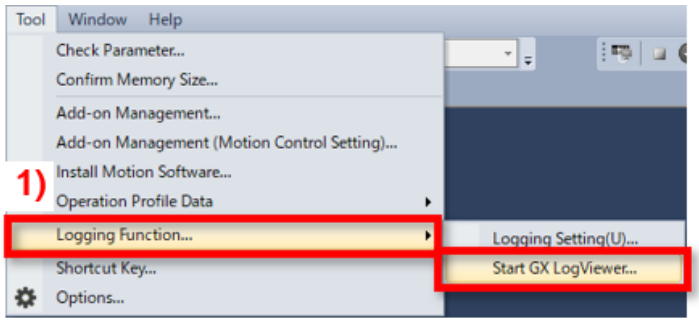
5.4 Bắt đầu ghi nhật ký

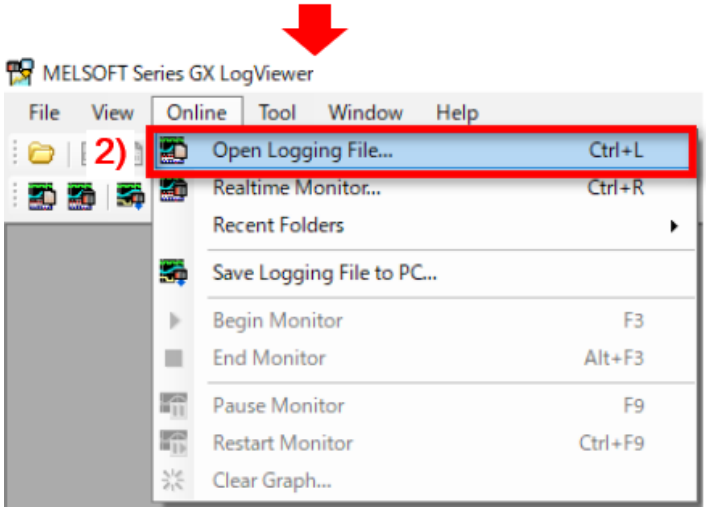
Khi "Start by User Operation" được đặt trong 5.2 (9), hãy nhấp vào biểu tượng [Logging Status and Operation] để hiển thị màn hình [Logging Status and Operation] và bắt đầu ghi nhật ký.
Khi chọn tên cài đặt dữ liệu ghi nhật ký sẽ được thực thi và nhấp vào nút [Start], LoggingStatus sẽ chuyển thành "Waiting for trigger".
Khi chương trình được thực thi ở trạng thái này và điều kiện kích hoạt (khi X24 được BẬT trong ví dụ của khóa học này) được thỏa mãn, trạng thái sẽ chuyển sang "Triggered".
Khi quá trình ghi nhật ký hoàn tất, trạng thái sẽ chuyển từ "Saving" thành "CollectionCompleted".

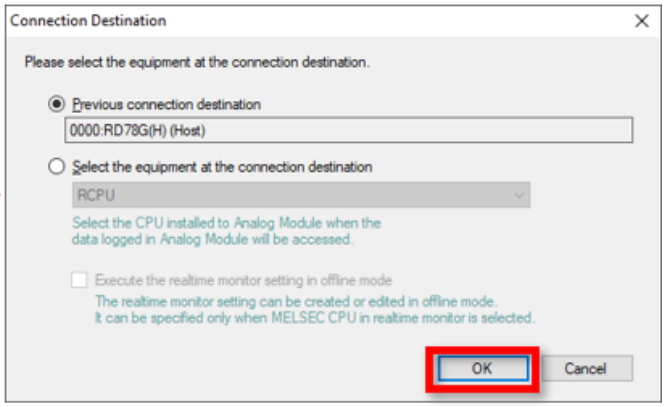


5.5 Đọc dữ liệu ghi nhật ký

GX LogViewer được sử dụng để đọc dữ liệu ghi nhật ký.
Chọn [Tool] → [Logging Function] → [Start GX LogViewer] từ thanh công cụ của màn hình Motion Control Setting Function.
Khi GX LogViewer khởi động, hãy chọn [Online] → [Open Logging File].
Chọn "0000:RD78G(H) (Host)" trong màn hình Đích kết nối. (Lưu ý)



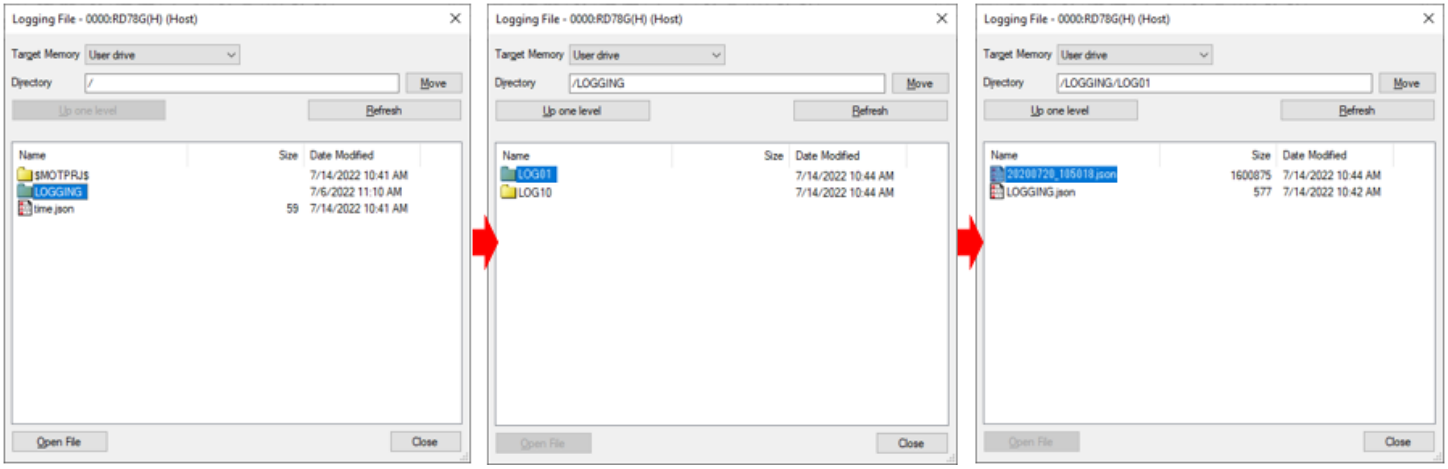




(Lưu ý) Nếu GX LogViewer đã khởi động và giao tiếp với mô-đun Chuyển động đã thiết lập thì màn hình này sẽ không được hiển thị.

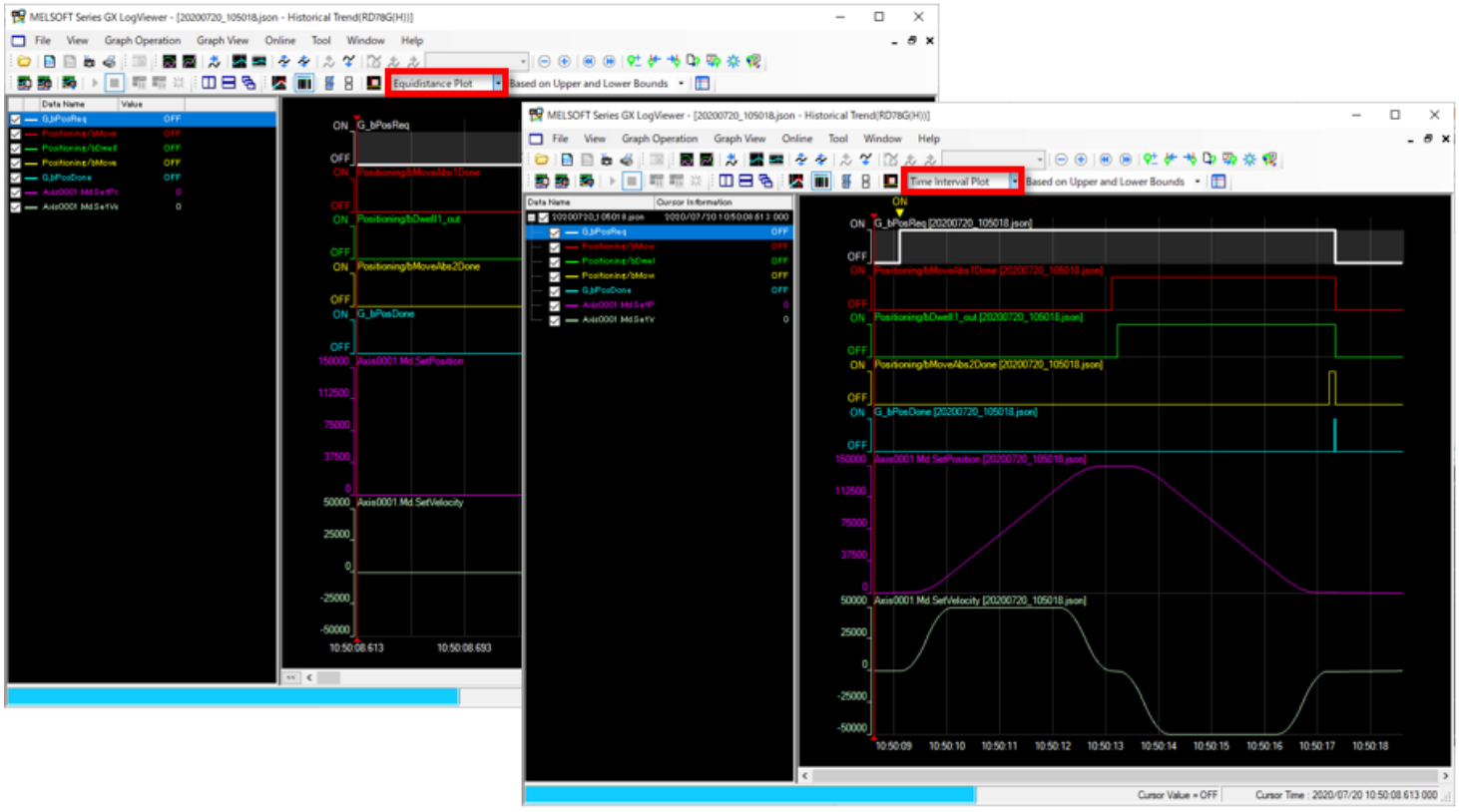
5.5 Đọc dữ liệu ghi nhật ký

Chọn tệp ghi nhật ký sẽ được đọc.
Ổ đĩa người dùng trong "LOGGING" → "LOG1" → "(Logged date and time).json" được chọn trong ví dụ của chương này.
Chọn tên tệp và nhấp vào nút [Open File].



5.5 Đọc dữ liệu ghi nhật ký

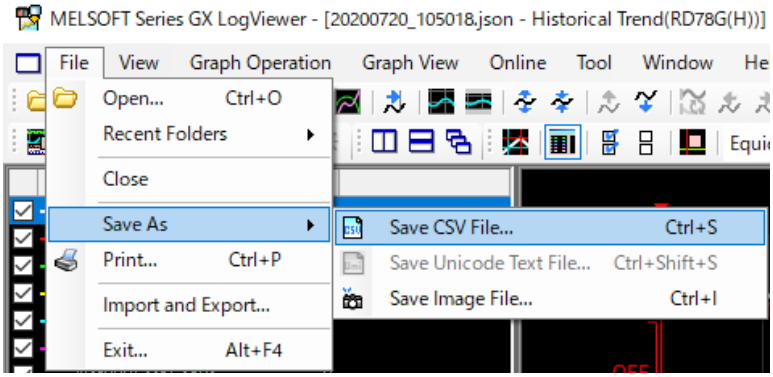
Dữ liệu dạng sóng được ghi trong GX LogViewer được hiển thị.
Khi định dạng biểu đồ được đổi từ "Equidistance Plot" thành "Time Interval Plot", toàn bộ dạng sóng ghi nhật ký có thể được hiển thị.



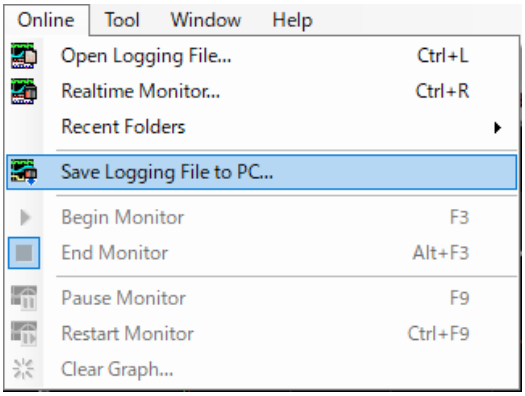
5.6 Lưu dữ liệu ghi nhật ký

Dữ liệu dạng sóng ghi nhật ký có thể được lưu dưới dạng tệp csv hoặc tệp json.
(Khi được ghi nhật ký ở định dạng CSV, dữ liệu có thể được lưu dưới dạng tệp CSV.)

1) Khi lưu dưới dạng tệp csv
   Chọn [File] → [Save As] → [Save CSV File] từ thanh công cụ của GX LogViewer.



2) Khi lưu dưới dạng tệp json
   Chọn [Online] → [Save Logging File to PC] từ thanh công cụ của GX LogViewer.



5.7 Tóm tắt chương này

Trong chương này, bạn đã học về:

- Khởi động Công cụ cấu hình ghi nhật ký
- Cài đặt dữ liệu sẽ được ghi nhật ký
- Ghi cài đặt ghi nhật ký
- Bắt đầu ghi nhật ký
- Đọc dữ liệu ghi nhật ký
- Lưu dữ liệu ghi nhật ký

Những điểm quan trọng

Khởi động Công cụ cấu hình ghi nhật ký	• Khởi động công cụ cài đặt ghi nhật ký hệ thống chuyển động từ chức năng cài đặt điều khiển chuyển động.
Cài đặt dữ liệu sẽ được ghi nhật ký	• Đặt dữ liệu sẽ được ghi nhật ký, điều kiện kích hoạt và dữ liệu khác bằng cách làm theo quy trình được hiển thị trong công cụ cài đặt ghi nhật ký hệ thống chuyển động.
Ghi cài đặt ghi nhật ký	• Ghi dữ liệu cài đặt ghi nhật ký vào mô-đun Chuyển động trước khi ghi nhật ký.
Bắt đầu ghi nhật ký	• Khi điều kiện bắt đầu ghi nhật ký được đặt thành "Start by User Operation", hãy nhấp vào nút bắt đầu trong màn hình "Logging Status and Operation" để bắt đầu ghi nhật ký.
Đọc dữ liệu ghi nhật ký	• GX LogViewer được sử dụng để đọc dữ liệu ghi nhật ký.
Lưu dữ liệu ghi nhật ký	• Dữ liệu dạng sóng ghi nhật ký có thể được lưu dưới dạng tệp csv hoặc tệp json.

Kiểm tra Bài kiểm tra cuối khóa

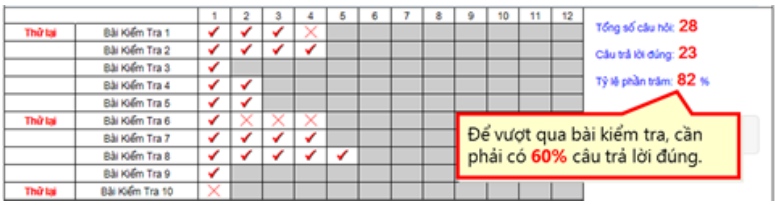
Bạn hiện đã hoàn thành tất cả bài học của Khóa học **Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Điều khiển định vị RD78G(H))**, bạn đã sẵn sàng để làm bài kiểm tra cuối khóa. Nếu bạn chưa rõ về bất kỳ chủ đề nào đã được hướng dẫn, vui lòng tận dụng cơ hội này để xem lại chủ đề đó.

Có tổng cộng 4 câu hỏi (7 mục) trong Bài kiểm tra cuối khóa này.

Bạn có thể làm bài kiểm tra cuối khóa nhiều lần tùy thích.

Kết quả điểm số

Số lượng câu trả lời đúng, số lượng câu hỏi, tỷ lệ câu trả lời đúng, và kết quả đạt/hỏng sẽ xuất hiện trên trang điểm số.



Kiểm tra **Bài kiểm tra cuối khóa 1**

Chọn (các) mô tả chính xác của nhãn công khai. (Bạn có thể chọn nhiều câu trả lời.)

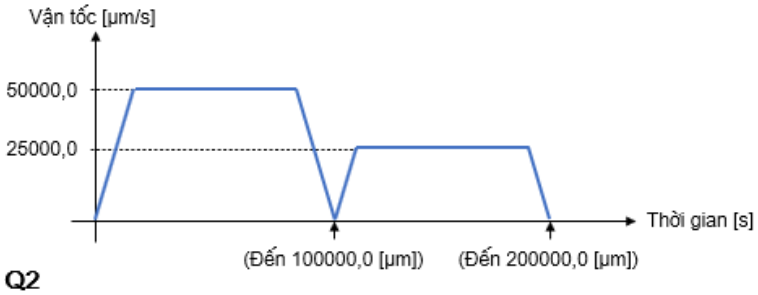
- [ ] Nhãn công khai là nhãn dùng chung có thể sử dụng trong cả mô-đun Chuyển động và PLC CPU.
- [ ] Nhãn công khai được đăng ký từ nhãn toàn cục của PLC CPU.
- [ ] Khi nhãn toàn cục được đặt thành nhãn công khai, hãy chọn xem nhãn có được đọc hay ghi từ/tới PLC CPU không.

Kiểm tra Bài kiểm tra cuối khóa 2

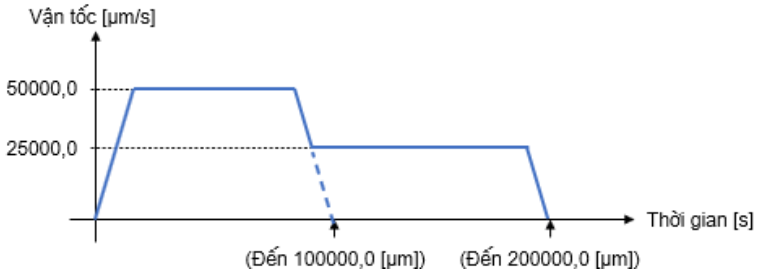
FB1 được thực thi trước, FB2 được thực thi sau.
Khi FB1 và FB2 có vị trí mục tiêu và vận tốc mục tiêu như được hiển thị trong bảng sau, hãy chế độ đệm được thực thi tiếp theo.

	Vị trí mục tiêu	Vận tốc mục tiêu
FB1	100000,0 [µm]	50000,0 [µm/s]
FB2	200000,0 [µm]	25000,0 [µm/s]

Q1



Q2



Q1 Chọn câu trả lời thích hợp.

Q2 Chọn câu trả lời thích hợp.

Q1:
- 1：mcAborting
- 2：mcBuffered
- 3：mcBlendingNext
- 4：mcBlendingPrevious

Q2:
- 1：mcBlendingNext và mcBlendingHigh
- 2：mcBlendingPrevious và mcBlendingHigh
- 3：mcBlendingNext và mcBlendingLow
- 4：mcBlendingPrevious và mcBlendingLow

Kiểm tra **Bài kiểm tra cuối khóa 3**

Chọn (các) câu đúng trong các câu sau để lập trình với PLC CPU. (Bạn có thể chọn nhiều câu trả lời.)

- [ ] Thư viện FB phải được đăng ký với GX Works3 để sử dụng Motion control FB cho mô-đun Chuyển động trong PLC CPU.
- [ ] Đặt motion control FB vào trình biên tập chương trình từ cây dự án của GX Works3.
- [ ] Không có tham số nào được đặt cho mô-đun Chuyển động.

Kiểm tra **Bài kiểm tra cuối khóa 4**

Chọn các đáp án thích hợp để điền vào chỗ trống.

- Khởi động (Q1) để đặt dữ liệu sẽ được ghi nhật ký.
- Ghi dữ liệu ghi nhật ký vào (Q2) để thực hiện ghi nhật ký.
- (Q3) được sử dụng để đọc dữ liệu ghi nhật ký và kiểm tra dạng sóng.

Q1 Chọn câu trả lời thích hợp.

Q2 Chọn câu trả lời thích hợp.

Q3 Chọn câu trả lời thích hợp.

Q1:
- 1：CPU module logging configuration tool
- 2：Công cụ cài đặt ghi nhật ký hệ thống chuyển động

Q2:
- 1：Mô-đun CPU
- 2：Mô-đun Chuyển động
- 3：Bộ khuếch đại servo

Q3:
- 1：MR Configurator2
- 2：GX LogViewer

Kiểm tra **Bài kiểm tra cuối khóa 1**

Chọn (các) mô tả chính xác của nhãn công khai. (Bạn có thể chọn nhiều câu trả lời.)

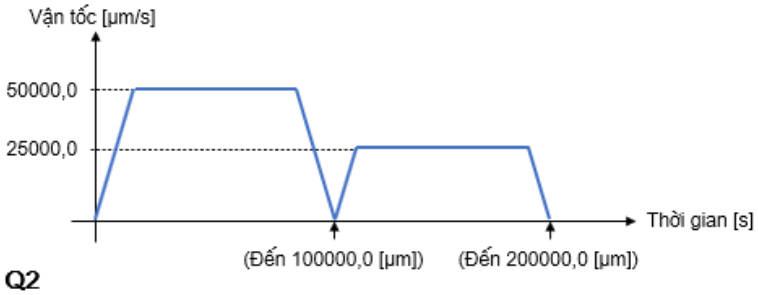
- [x] Nhãn công khai là nhãn dùng chung có thể sử dụng trong cả mô-đun Chuyển động và PLC CPU.
- [ ] Nhãn công khai được đăng ký từ nhãn toàn cục của PLC CPU.
- [x] Khi nhãn toàn cục được đặt thành nhãn công khai, hãy chọn xem nhãn có được đọc hay ghi từ/tới PLC CPU không.

Kiểm tra Bài kiểm tra cuối khóa 2

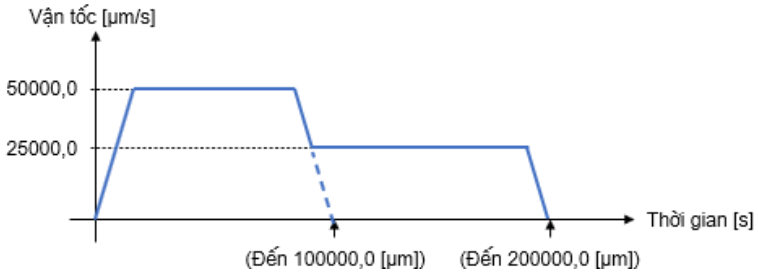
FB1 được thực thi trước, FB2 được thực thi sau.
Khi FB1 và FB2 có vị trí mục tiêu và vận tốc mục tiêu như được hiển thị trong bảng sau, hãy chế độ đệm được thực thi tiếp theo.

	Vị trí mục tiêu	Vận tốc mục tiêu
FB1	100000,0 [µm]	50000,0 [µm/s]
FB2	200000,0 [µm]	25000,0 [µm/s]

Q1



Q2



Q1 2: mcBuffered

Q2 3: mcBlendingNext và mcBlendingLow

Q1:
- 1：mcAborting
- 2：mcBuffered
- 3：mcBlendingNext
- 4：mcBlendingPrevious

Q2:
- 1：mcBlendingNext và mcBlendingHigh
- 2：mcBlendingPrevious và mcBlendingHigh
- 3：mcBlendingNext và mcBlendingLow
- 4：mcBlendingPrevious và mcBlendingLow

Kiểm tra **Bài kiểm tra cuối khóa 3**

Chọn (các) câu đúng trong các câu sau để lập trình với PLC CPU. (Bạn có thể chọn nhiều câu trả lời.)

- [x] Thư viện FB phải được đăng ký với GX Works3 để sử dụng Motion control FB cho mô-đun Chuyển động trong PLC CPU.
- [ ] Đặt motion control FB vào trình biên tập chương trình từ cây dự án của GX Works3.
- [ ] Không có tham số nào được đặt cho mô-đun Chuyển động.

Kiểm tra **Bài kiểm tra cuối khóa 4**

Chọn các đáp án thích hợp để điền vào chỗ trống.

- Khởi động (Q1) để đặt dữ liệu sẽ được ghi nhật ký.
- Ghi dữ liệu ghi nhật ký vào (Q2) để thực hiện ghi nhật ký.
- (Q3) được sử dụng để đọc dữ liệu ghi nhật ký và kiểm tra dạng sóng.

Q1	2: Công cụ cài đặt ghi nhật ký hệ thống chuyển động
Q2	2: Mô-đun Chuyển động
Q3	2: GX LogViewer

Q1:
- 1：CPU module logging configuration tool
- 2：Công cụ cài đặt ghi nhật ký hệ thống chuyển động

Q2:
- 1：Mô-đun CPU
- 2：Mô-đun Chuyển động
- 3：Bộ khuếch đại servo

Q3:
- 1：MR Configurator2
- 2：GX LogViewer

Kiểm tra Điểm kiểm tra

Bạn đã hoàn thành Bài kiểm tra cuối khóa. Kết quả của bạn như sau.
Để kết thúc Bài kiểm tra cuối khóa, hãy tiếp tục tới trang tiếp theo.

		1	2	3	4	5	6	7	8	9	10
	Bài kiểm tra cuối khóa 1	✓									
	Bài kiểm tra cuối khóa 2	✓	✓								
	Bài kiểm tra cuối khóa 3	✓									
	Bài kiểm tra cuối khóa 4	✓	✓	✓							

Tổng số câu hỏi: 7

Câu trả lời đúng: 7

Tỷ lệ phần trăm: 100 %

Xóa

Bạn đã hoàn thành Khóa học "Thông tin cơ bản về mô-đun chuyển động dòng MELSEC iQ-R (Điều khiển định vị RD78G(H))".

Cảm ơn bạn đã tham gia khóa học này.

Chúng tôi hy vọng bạn thích các bài học và những thông tin bạn có được trong khóa học này sẽ hữu ích trong tương lai.

Bạn có thể xem lại khóa học này nhiều lần tùy ý.

Xem lại

Đóng