

**PLC**

# **Khóa học Cơ Bản về GX Works2**

Khóa học đào tạo (e-learning) này được thiết kế  
cho những người lần đầu dùng phần mềm GX  
Works2 để tạo các chương trình PLC.

&gt;&gt; Giới thiệu

## Mục đích khóa học

TOC

Khóa học này cung cấp những kiến thức cơ bản về việc sử dụng phần mềm GX Works2 để lập trình, gỡ lỗi, và kiểm tra hoạt động của bộ điều khiển có thể lập trình (PLC). Khóa học này được thiết kế cho người lập các chương trình PLC cho các bộ điều khiển của sê-ri MELSEC-Q, sê-ri MELSEC-L và sê-ri MELSEC-F.

**Giới thiệu****Cấu trúc khóa học**

Nội dung của khóa học này như sau.

Chúng tôi khuyến cáo bạn nên bắt đầu từ Chương 1.

**Chương 1 - Phương pháp Điều khiển hệ thống PLC**

Ngôn ngữ lập trình và phần mềm dùng để lập trình sẽ được giới thiệu ở đây.

**Chương 2 - Thiết kế chương trình**

Bạn sẽ tìm hiểu cách thiết kế một chương trình dựa trên các mục điều khiển và cấu hình phần cứng.

**Chương 3 - Lập trình**

Bạn sẽ học cách lập trình bằng cách sử dụng phần mềm chuyên dụng GX Works2.

**Chương 4 - Gỡ lỗi**

Bạn sẽ tìm hiểu cách ghi chương trình PLC vào mô-đun CPU và gỡ lỗi cho chương trình đó.

**Chương 5 - Bài kiểm tra cuối khóa**

Mức đạt yêu cầu: 60% hoặc cao hơn.

## Giới thiệu

# Làm thế nào sử dụng Công cụ e-Learning

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

&gt;&gt; Giới thiệu

## Thận trọng khi sử dụng



### Biện pháp phòng ngừa an toàn

Khi bạn học tập bằng cách sử dụng các sản phẩm thực tế, hãy đọc kỹ các biện pháp phòng ngừa an toàn trong hướng dẫn sử dụng tương ứng.

### Biện pháp phòng ngừa trong khóa học này

- Màn hình hiển thị của phiên bản phần mềm mà bạn sử dụng có thể khác với các màn hình trong khóa học này.

# Chương 1 Phương pháp Điều khiển hệ thống PLC

Khóa học này nhằm dành cho những người làm việc bằng phần mềm kỹ thuật. Khóa học bao gồm các khái niệm cơ bản về quản lý các hệ thống của sê-ri MELSEC-Q, L và F.

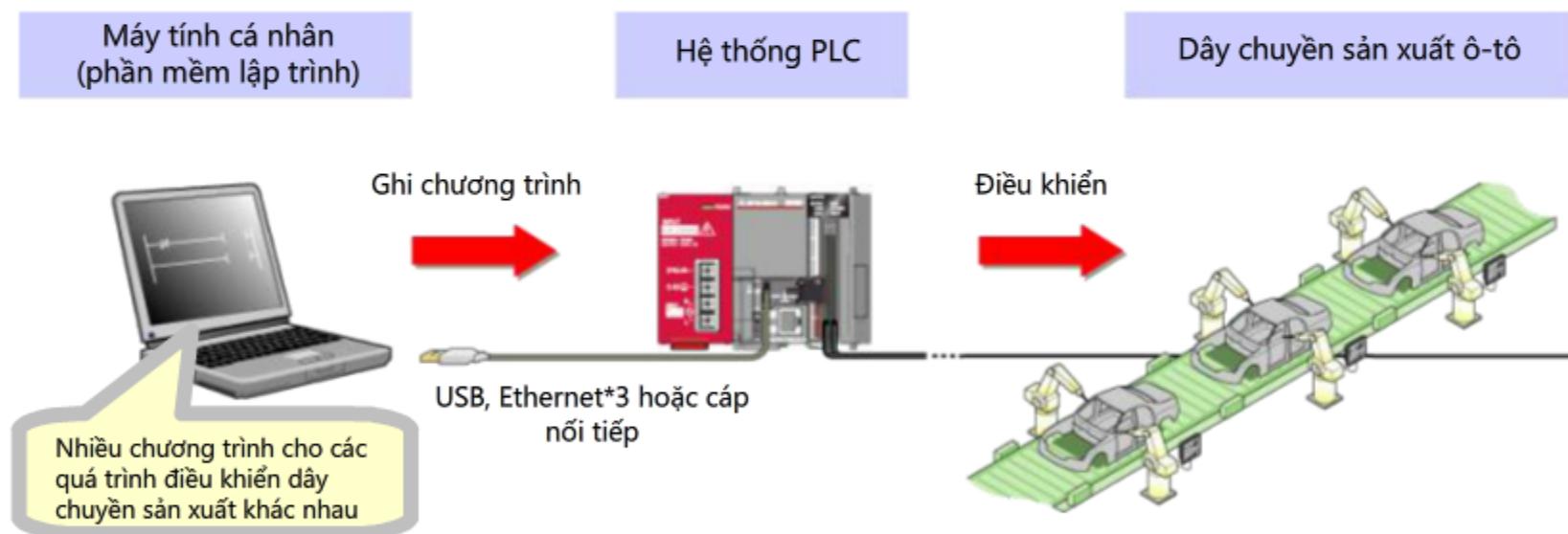
GX Works 2 (GXW2) sử dụng ngôn ngữ lập trình được tiêu chuẩn hóa của quốc tế bao gồm ngôn ngữ Sơ đồ chức năng trình tự (SFC), Danh sách lệnh (IL)\*1, Logic dạng thang, Sơ đồ Khối chức năng (FBD)\*2 và Văn bản có cấu trúc (ST).

Các chương trình được phát triển bằng máy tính cá nhân chạy "phần mềm kỹ thuật", GX works2 và thường được ghi vào bộ điều khiển khả trình CPU qua một USB, cáp Ethernet\*3 hoặc cáp nối tiếp. Mô-đun CPU có thể được lập trình lại nhiều lần sẽ rất cần thiết để thích ứng với bất cứ thay đổi bắt buộc cho việc điều khiển mong muốn.

\*1 Kế hoạch tương lai cho GX works2.

\*2 Hiện nay được gọi là Trình lập trình PLC dạng thang (ladder) có cấu trúc trong GX works2, dự kiến tuân thủ IEC.

\*3 Ethernet là thương hiệu đã đăng ký của Xerox Corp.



Trong khóa học này, logic dạng thang (một trong những ngôn ngữ lập trình PLC phổ biến nhất) sẽ được dùng trong chương trình ví dụ. Mặc dù ví dụ sẽ sử dụng PLC Dòng L, các nội dung của khóa học này vẫn áp dụng tương tự cho các hệ thống Dòng Q.

Phương pháp điều khiển cơ bản cũng giống như đối với sê-ri MELSEC-F nhưng khác ở một số vận hành và chức năng.

**1.1**

# Quy trình xây dựng hệ thống PLC

Khóa học e-learning này bao gồm các bước thiết kế phần mềm (thể hiện ở màu xanh lá) cần thiết cho việc triển khai một hệ thống điều khiển lập trình PLC.

## Thiết kế phần cứng

(1) Thiết kế hệ thống ..... Khóa học Cơ bản về MELSEC-Q/MELSEC-L



(2) Lựa chọn sản phẩm ..... Khóa học Cơ bản về MELSEC-Q/MELSEC-L



(3) Chuẩn bị trước ..... Khóa học Cơ bản về MELSEC-Q/MELSEC-L



(4) Lắp đặt và đấu dây ..... Khóa học Cơ bản về MELSEC-Q/MELSEC-L



(5) Kiểm tra đấu dây ..... Khóa học Cơ bản về MELSEC-Q/MELSEC-L

## Thiết kế phần mềm

(6) Thiết kế chương trình ..... Chương 2



(7) Lập trình ..... Chương 3



(8) Gỡ lỗi ..... Chương 4



(9) Vận hành

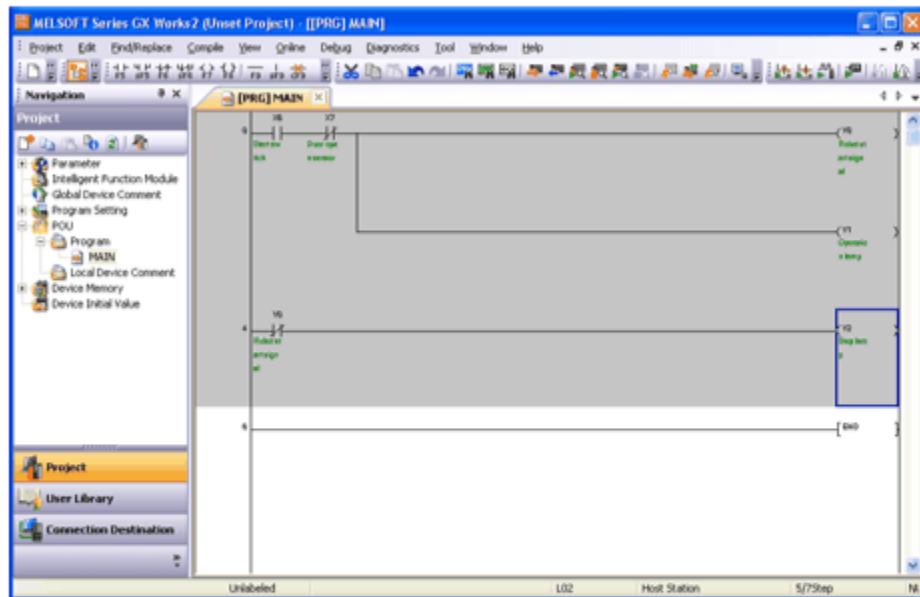
Nội dung của  
khóa học này

**1.2****Yêu cầu đối với lập trình**

Trong khóa học này, nội dung sẽ tập trung vào cách sử dụng phần mềm kỹ thuật bộ điều khiển lập trình GX Works2 để phát triển các chương trình hệ thống ví dụ.

Một vài chức năng chính của GX Works2 được liệt kê dưới đây.

- Quản lý bộ nhớ và tập tin
- Phát triển các chương trình điều khiển lập trình
- Quản lý tài liệu chương trình (các nhận xét, v.v...)
- Đọc ghi dữ liệu (đặc biệt là các chương trình) từ/đến mô-đun CPU
- Kiểm tra hoạt động chương trình
  - Phần mềm giả lập về phần cứng PLC
  - Cưỡng bức bật hoặc tắt I/O
  - Giám sát I/O và trạng thái địa chỉ bộ nhớ
- Thực hiện các trách nhiệm bảo trì và khắc phục sự cố



PLC\_GX\_Works2\_Basics\_VIE

## 1.3 Cấu hình màn hình GX Works2

Cấu hình màn hình GX Works2 được hiển thị dưới đây.

Đặt con trỏ chuột trong khung màu đỏ để hiển thị chức năng tương ứng.

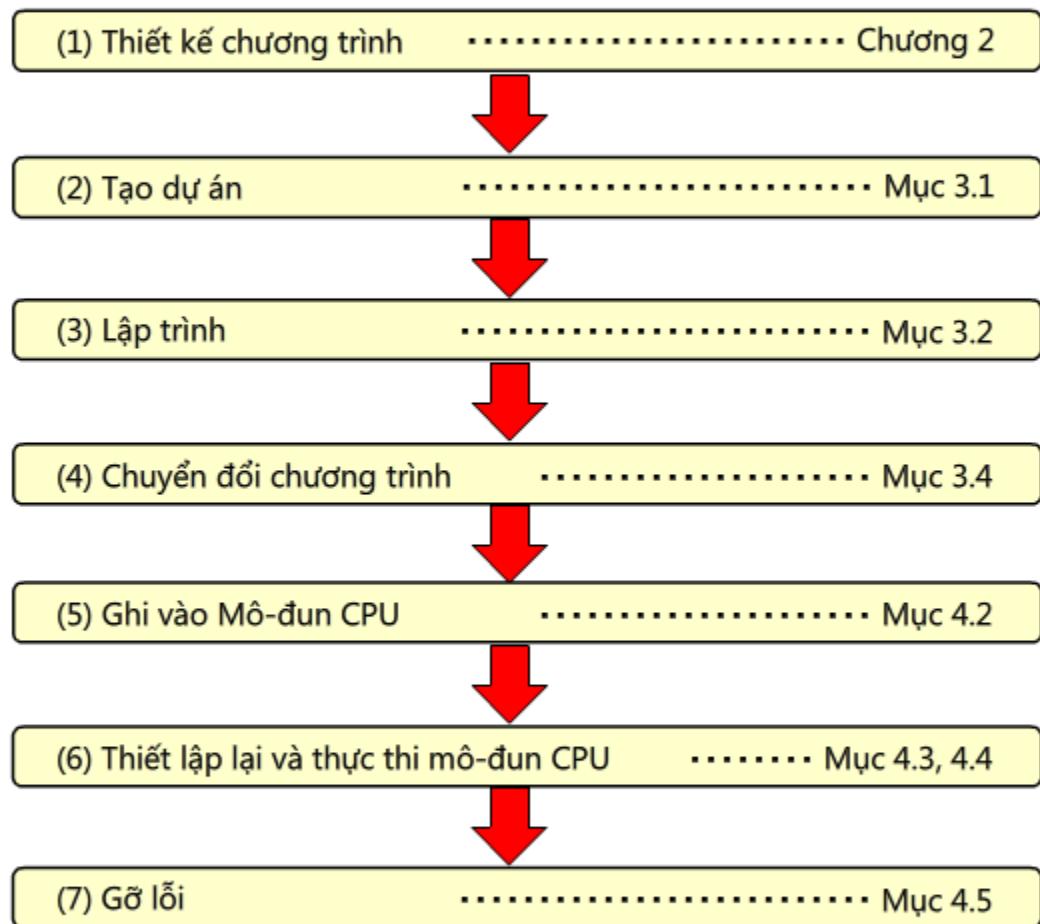
The screenshot shows the GX Works2 software interface with the following details:

- Title Bar:** MELSOFT Series GX Works2 ...s\EN93632\My Documents\le\_Learning\Robot\_Control
- Menu Bar:** Project, Edit, Find/Replace, Compile, View, Online, Debug, Diagnostics, Tool, Window, Help
- Toolbar:** Includes various icons for file operations, search, and project management.
- Navigation Area:** Shows the project structure with nodes like Parameter, Intelligent Function Module, Global Device Comment, Program Setting, POU, Program, Local Device Comment, Device Memory, and Device Initial Value. The "Program" node is expanded, showing "MAIN" and "SUB".
- Project Area:** Displays the "Project" icon.
- User Library Area:** Displays the "User Library" icon.
- Connection Destination Area:** Displays the "Connection Destination" icon.
- Main Area:** Contains two ladder logic programs:
  - [PRG] SUB:** A sub-program with one coil output (Y0) labeled "Robot start signal". It has one input (X6) labeled "StartSw" and one input (X7) labeled "Door open sensor".
  - [PRG] MAIN:** The main program. It calls the [PRG] SUB block at address 0. It has one coil output (Y0) labeled "Robot start signal" at address 4, which is connected to the X6 input of the [PRG] SUB block. It also has one coil output (Y1) labeled "Operational lamp" at address 6. At address 4, it has a normally closed contact (X6) labeled "Robot start signal" connected to the X7 input of the [PRG] SUB block. At address 6, it has a coil output (Y2) labeled "Stop lamp".
- Status Bar:** Unlabeled, L02, Host Station, 5/7Step, N13

A red box highlights the [PRG] MAIN window, and another red box highlights the coil output Y0 in the [PRG] MAIN window.

**1.4****Quy trình Tạo chương trình PLC**

Tạo một chương trình PLC theo quy trình sau đây.



## Chương 2 Tạo dữ liệu màn hình

Trong Chương 2, bạn sẽ tìm hiểu làm thế nào để thiết kế các chương trình, bao gồm việc xác định các nội dung điều khiển và chuyển đổi chúng thành một chương trình.

Thiết kế chương trình ..... Chương 2



Lập trình ..... Chương 3



Gỡ lỗi ..... Chương 4

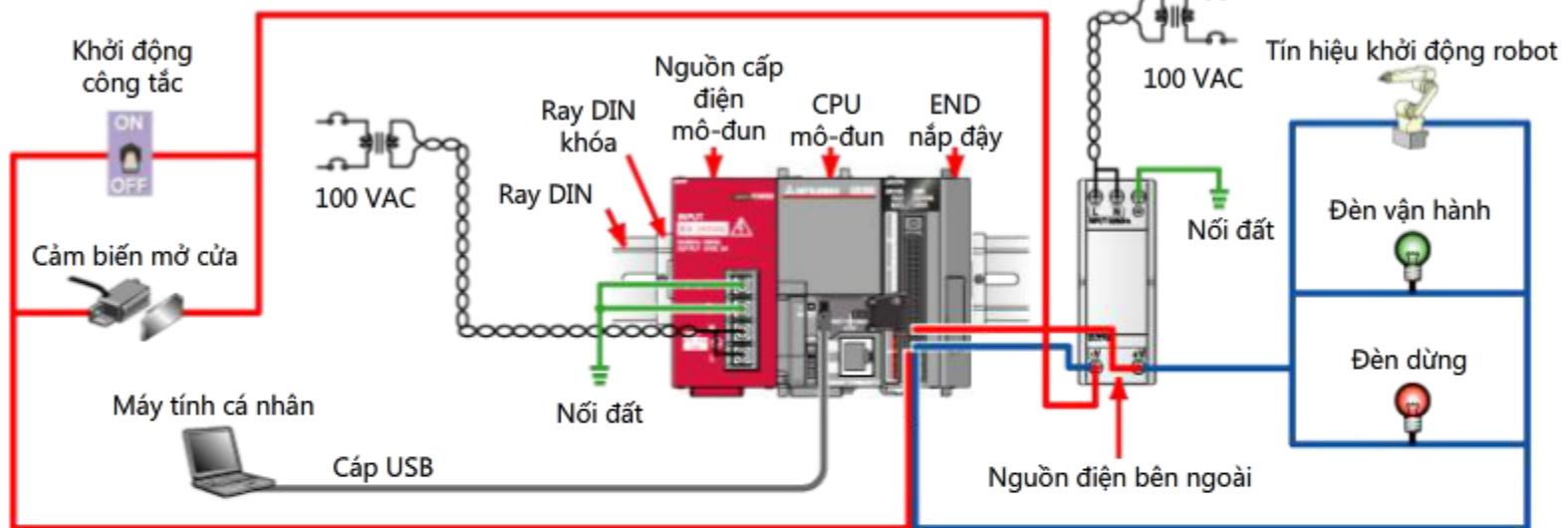
### Các bước học tập trong Chương 2

- 2.1 Cấu hình phần cứng ví dụ  
Hệ thống được dùng cho việc học tập
- 2.2 Định nghĩa Mục điều khiển
- 2.3 Tạo Bảng I/O tương ứng  
Thiết bị và Số hiệu thiết bị
- 2.4 Thiết kế chương trình

## 2.1 Cấu hình phần cứng của hệ thống ví dụ được dùng cho việc học tập

Trong khóa học này, bạn sẽ xây dựng một hệ thống PLC (được gọi là "hệ thống ví dụ" sau đây), trong đó khởi động các robot theo một quy trình.

Sơ đồ cấu hình phần cứng của hệ thống ví dụ được hiển thị dưới đây với một danh sách các thành phần phần cứng.



Mục	Thành phần	Kiểu máy	Mô tả
Hệ thống PLC	Mô-đun nguồn điện	L61P	Cung cấp nguồn cho các mô-đun bao gồm mô-đun CPU và mô-đun I/O.
	Mô-đun CPU	L02CPU	Điều khiển hệ thống PLC.
	Nắp KẾT THÚC	L6EC	Được gắn vào bên phải của khối hệ thống.
	Cáp USB	MR-J3USBCBL3M	Kết nối máy tính cá nhân, trong đó có cài đặt GX Works2, vào mô-đun CPU.
	Máy tính cá nhân	—	Chạy với GX Works2 đã cài đặt.
Nguồn điện bên ngoài	—	—	Nguồn cấp điện cho các thiết bị I/O gắn ngoài.
Thiết bị I/O gắn ngoài	Công tắc	—	Cài về ON để bắt đầu điều khiển.
	Cảm biến	—	Phát hiện xem cửa mở hay đóng.
	Robot	—	Vận hành theo các tín hiệu điều khiển.
	Hai đèn	—	Bật sáng theo tình trạng vận hành.

## 2.2

## Định nghĩa các mục điều khiển

Bước đầu tiên của việc thiết kế chương trình là nhận biết các thiết bị cần được điều khiển và các thiết bị I/O cần thiết cho việc điều khiển mong muốn. Trong hệ thống ví dụ, việc điều khiển các hoạt động khởi động và dừng của một robot sẽ được thực hiện. Robot sẽ được ngăn không cho khởi động nếu cửa đến hàng rào an toàn đang mở, và sẽ bị dừng lại nếu cửa đó được mở ra trong quá trình vận hành.

Xem ảnh động dưới đây để hiểu rõ hơn về cách hệ thống ví dụ sẽ hoạt động.

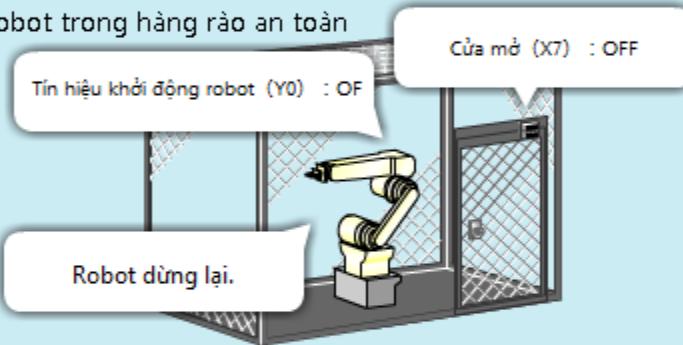
### Vận hành của hệ thống ví dụ

 Nhấp vào bên trong vòng tròn đỏ

Bảng điều khiển robot



Robot trong hàng rào an toàn



Khi bạn cài **công tắc khởi động (X6)** sang OFF, **Tín hiệu khởi động robot (Y0)** sẽ tắt để dừng vận hành robot. Đồng thời, **đèn vận hành (Y1)** trên bảng điều khiển sẽ tắt, và **đèn dừng (Y2)** sẽ bật.

[Chạy lại](#)

 Trước

**2.3****Tạo Bảng thiết bị I/O tương ứng và Số hiệu thiết bị**

Tốt nhất là nên tạo một bảng bao gồm tất cả các thiết bị và thanh ghi I/O được sử dụng trong PLC cũng như các thông tin tương ứng của chúng đối với bất cứ chương trình nào được tạo ra. Điều này sẽ giảm thiểu những sai lầm ngẫu nhiên xảy ra trong quá trình thiết kế và lập trình cũng như để nâng cao hiệu quả lập trình. Nếu đã tồn tại bảng tương ứng cho hệ thống, chẳng hạn như bảng được tạo bởi một người đã cấu hình phần cứng thì hãy tận dụng nó.

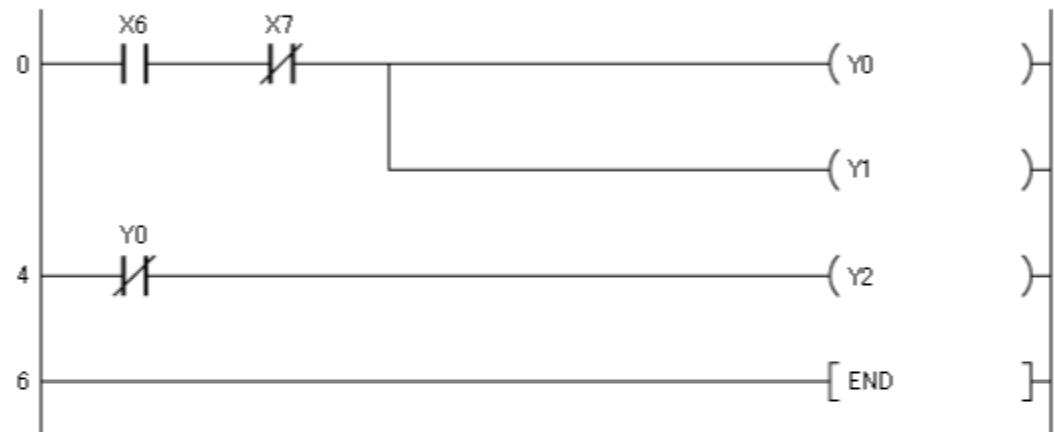
Bảng dưới đây là một bảng tương ứng cho hệ thống ví dụ được sử dụng trong khóa học này

Tên thiết bị I/O	Thiết bị số	I/O type	Loại thiết bị	Mô tả
Start switch (Khởi động công tắc)	X6	Đầu vào	Bit	Công tắc này sẽ khởi động hoặc dừng vận hành robot.
Door open sensor (Cảm biến mở cửa)	X7	Đầu vào	Bit	Cảm biến này sẽ kiểm tra xem cửa hàng rào an toàn của robot có đang mở hay không. Khi cửa mở ra, cảm biến này sẽ bật. Khi cửa đóng lại, cảm biến này sẽ tắt.
Robot start signal (Tín hiệu khởi động robot)	Y0	Đầu ra	Bit	Khi tín hiệu này bật, robot sẽ bắt đầu vận hành.
Operation lamp (Đèn vận hành)	Y1	Đầu ra	Bit	Đèn này sẽ bật sáng trong khi robot đang vận hành.
Stop lamp	Y2	Đầu ra	Bit	Đèn này sẽ bật sáng trong khi robot được dừng lại.

\* Nếu sử dụng dữ liệu dạng word, giá trị ban đầu, phạm vi thiết lập (các giới hạn trên và dưới), kiểu dữ liệu (đã xác thực, thực tế, v.v...), và các chú thích nên được đưa vào trong bảng. Những thông tin này sẽ hữu ích cho việc thiết kế và chỉnh sửa chương trình.

**2.4****Thiết kế chương trình**

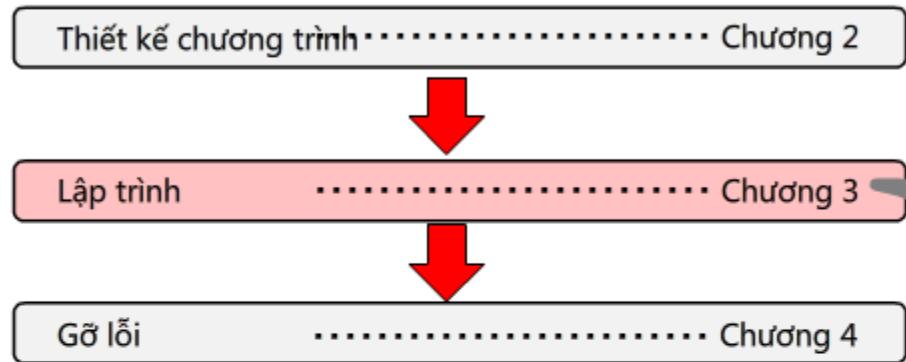
Thiết kế một chương trình sử dụng ngôn ngữ logic dạng thang dựa trên các mục điều khiển và bảng I/O tương ứng. Chương trình ladder và bảng I/O tương ứng thiết kế cho hệ thống ví dụ được trình bày bên dưới.

**Chương trình ladder****Bảng I/O tương ứng**

I/O device name	Loại	Thiết bị số
Start switch (Khởi động công tắc)	Đầu vào	X6
Door open sensor (Cảm biến mở cửa)	Đầu vào	X7
Robot start signal (Tín hiệu khởi động robot)	Đầu ra	Y0
Operation lamp (Đèn vận hành)	Đầu ra	Y1
Stop lamp (Đèn dừng)	Đầu ra	Y2

## Chương 3 Lập trình

Trong Chương 3, bạn sẽ học cách lập trình chương trình đã được thiết kế bằng GX Works2.



### Các bước học tập trong Chương 3

- 3.1 Tạo dự án
- 3.2 Tạo chương trình
- 3.3 Làm chương trình dễ hiểu
- 3.4 Chuyển đổi chương trình thành dạng thực thi
- 3.5 Lưu dự án

**3.1****Tạo dự án**

Bước đầu tiên để viết một chương trình là tạo một dự án.

Dự án là một tập hợp các dữ liệu GX Works2 sẽ sử dụng để quản lý các chương trình.

Bảng sau liệt kê các thành phần chính của một dự án.

Kiểu dữ liệu	Mô tả
Chương trình	Mã nguồn và mã biên dịch cho các hoạt động chuỗi của CPU.
Nhận xét	Một loại tài liệu hướng dẫn được hiển thị bên trong chương trình. Xem Mục 3.3 "Làm chương trình dễ hiểu" để biết chi tiết.
Tham số	Chứa hầu hết hoặc tất cả các thiết lập và thông tin cấu hình cho một hệ thống.
Chuyển giao thiết lập	Thông tin truyền kết nối cần thiết cho việc thiết lập giao tiếp giữa hệ thống đang chạy GX Works2 và mô-đun CPU.

**Chương trình ladder**

GX Works2 cho phép bạn chọn hai loại dự án sau.

Chương trình ví dụ trong khóa học này sử dụng loại "**simple project**".

Loại dự án	Mô tả
Simple project (Dự án đơn giản)	Loại dự án này tương thích ngược với các dự án của GX Developer. Dự án đơn giản có thể được chuyển đổi thành các dự án Có cấu trúc sau đó, nhưng không phải là cách hay dùng.
Structured project (Dự án có cấu trúc)	Các dự án này có khả năng sử dụng một ngôn ngữ lập trình bổ sung được gọi là Lập trình dạng thang có cấu trúc (Structrued Ladder). Ngoài ra, các chương trình có thể được tách ra thành nhiều phần nhỏ và các cụm mã thường xuyên sử dụng có thể dễ dàng được mô-đun hóa và tái sử dụng bằng một thư viện người dùng. Tương tự như vậy các nhãn cũng có thể được mô-đun hóa để dễ dàng sử dụng lại. Điều này có thể cải thiện việc lập trình và hiệu quả gỡ lỗi, đặc biệt là cho các dự án rất lớn.

**Nhãn**

Nhãn là các tên do người dùng tạo ra trở thành biệt danh cho các địa chỉ thiết bị. Chúng có thể được sử dụng toàn cục, cục bộ, hoặc toàn hệ thống khi được triển khai cùng với MELSOFT Navigator. Có thể tạo các dự án đơn giản có hoặc không có khả năng sử dụng nhãn. Đối với dự án ví dụ sẽ không sử dụng nhãn.

**3.1****Tạo dự án**

Để bắt đầu tạo dự án ví dụ, hãy thực hiện các thiết lập sau.

Trước khi tạo dự án, cần phải biết rõ dòng sản phẩm bộ điều khiển lập trình và tên model, cũng như loại dự án được sử dụng.

Mục	Mô tả
Loại dự án	Loại dự án sẽ xác định những tính năng nào sử dụng được khi viết chương trình. Trong ví dụ này, hãy chọn "dự án đơn giản".
Sử dụng nhãn	Nếu khả năng viết chương trình có sử dụng các nhãn là bắt buộc, hãy đánh dấu chọn mục này. Chương trình ví dụ không sử dụng các nhãn. Do đó, hãy để trống ô này không chọn.
Dòng PLC	Dòng PLC sẽ xác định các model có sẵn để lựa chọn trong danh sách thả xuống loại PLC. Trong ví dụ này, hãy chọn "LCPU".
Loại PLC	Loại PLC sẽ xác định cách trình biên dịch chuyển đổi các chương trình người dùng thành mã máy. Chọn model PLC sẽ được lập trình, trong trường hợp này là "L02".
Ngôn ngữ lập trình	Ngôn ngữ lập trình sẽ xác định loại chương trình của chương trình được tự động tạo ra đầu tiên (MAIN). Các chương trình bổ sung sử dụng các ngôn ngữ khác có thể được thêm vào sau đó. Trong ví dụ này, hãy chọn "Ladder".

Hãy xem trang tiếp theo có giải lập các quá trình tạo dự án mới.

## 3.1

## Tạo dự án

## MELSOFT Series GX Works2

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help



## Navigation

## Project



## New Project



## Project Type:

Simple Project

OK

Cancel

 Use Label

## PLC Series:

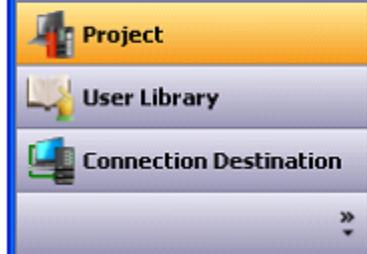
LCPU

## PLC Type:

L02

## Language:

Ladder



Bây giờ một dự án mới sẽ được tạo ra.

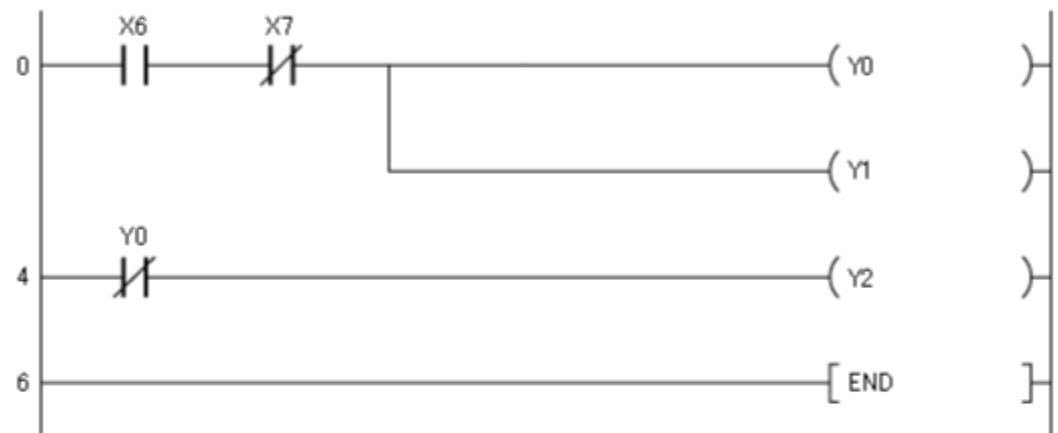
Nhấn để tiến hành.

**3.2****Tạo chương trình**

Sau khi tạo dự án, chúng ta hãy tạo một chương trình.

Tạo chương trình sau đây và tìm hiểu các thao tác cơ bản (nhập vào lệnh, thay đổi, xóa, sao chép & dán và nhập vào/xóa theo quy định của tuyến).

Chương trình được thiết kế cho hệ thống ví dụ trong Chương 2 được hiển thị dưới đây.

**Chương trình cho hệ thống ví dụ**

Trên trang tiếp theo, hãy thử tạo chương trình này bằng cửa sổ giả lập.



## 3.2

## Tạo chương trình

MELSOFT Series GX Works2 (Unset Project) - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

```

    graph TD
        X8 --> Y0_top
        Y0_top --- Y1
        Y0 --- Y2
        Y2 --- END
        0 --- Y2
    
```

Lúc này chương trình mạch Ladder đã hoàn tất.  
Nhấn để tiến hành.

Unlabeled | L02 | Host Station | 0/1Step | N/A

**3.3****Làm chương trình dễ hiểu**

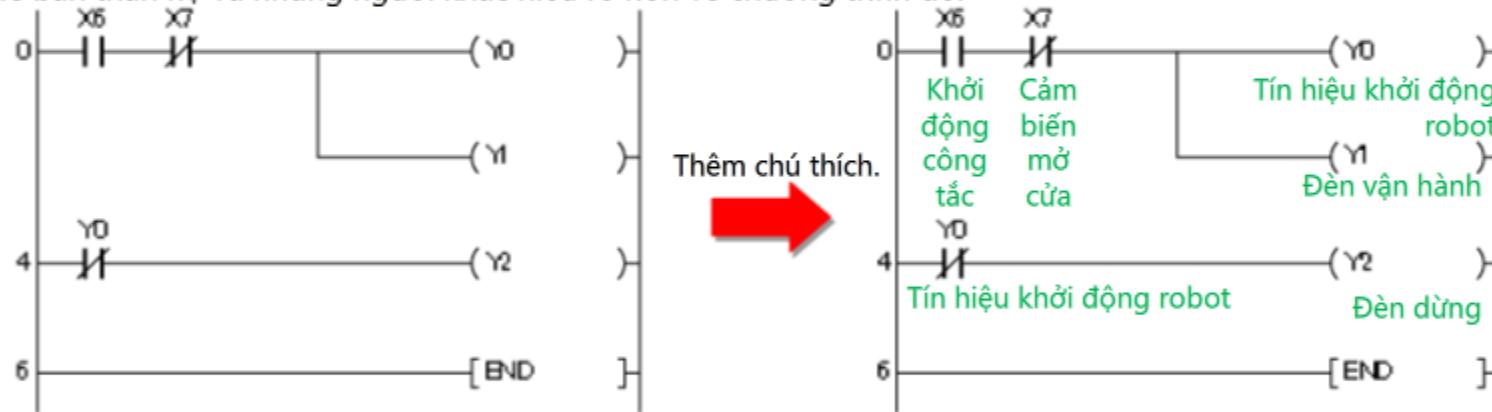
Trong trạng thái hiện tại của nó, bản trực quan của chương trình chỉ chứa các thiết bị, các lệnh, các đường và số bước. Khi nhìn vào một chương trình phức tạp, có thể rất khó khăn để xác định những gì chương trình đang làm.

- Khó tìm các lỗi lập trình như số thiết bị không chính xác hoặc các lệnh.
- Nhìn chung, sẽ khó thực hiện phân tích hoạt động, gỡ lỗi và mở rộng chương trình.
- Nếu nhà phát triển chương trình ban đầu không còn có thể duy trì chương trình, công việc tìm hiểu chương trình vận hành như thế nào đối với bất cứ ai khác đều rất khó khăn và có lẽ là bất khả.

**Biện pháp ứng phó**

Bao gồm **tài liệu hướng dẫn** trong chương trình, cho phép bất cứ ai có thể nhanh chóng hiểu được cách thức chương trình hoạt động.

Theo quy định thực hành đúng chuẩn, tất cả các lập trình viên cần phải thêm các chú thích chi tiết trong chương trình của mình để bản thân họ và những người khác hiểu rõ hơn về chương trình đó.



GX Works2 cho phép sử dụng ba loại chú thích khác nhau.

Để biết thêm thông tin chi tiết, hãy tham khảo sách hướng dẫn sử dụng Các dự án đơn giản GX Works2.

Loại chú thích	Nội dung chú thích
Device comment (Ghi chú thiết bị)	Nhập vào lên đến 32 ký tự sẽ được hiển thị dưới thiết bị đã chọn (I/O hoặc địa chỉ bộ nhớ khác.)
Statement (Câu lệnh)	Nhập vào lên đến 64 ký tự trên mỗi câu lệnh được thêm vào ở trên cùng của khối dạng thang đã chọn (vượt trên số bước). Mỗi khối dạng thang có thể có nhiều câu lệnh.
Note (Lưu ý)	Nhập vào lên đến 32 ký tự sẽ được hiển thị trên cuộn được chọn hoặc lệnh ứng dụng.

Trang tiếp theo sẽ giả lập quá trình thêm ghi chú thiết bị vào chương trình ví dụ.

## 3.3

## Làm chương trình dễ hiểu

MELSOFT Series GX Works2 (Unset Project) - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

```

    X8      X7
    |       |
    +---+---+
    | Start sw   Door open
    |             sensor
    +---+---+
                    |
                    +---+
                    | Y0
                    | Robot start signal
                    +---+
                    |
                    +---+
                    | Y0
                    | Robot start signal
                    +---+
                    |
                    +---+
                    | Y2
                    | Stop lamp
                    +---+
                    |
                    +---+
                    | END
                    +---+
    
```

Project

- Parameter
- Intelligent Function Module
- Global Device Comment
- Program Setting
- POU
  - Program
  - MAIN
- Local Device Comment
- Device Memory
- Device Initial Value

Project User Library Connection Destination

Unlabeled L02 Host Station 5/7Step N0.5

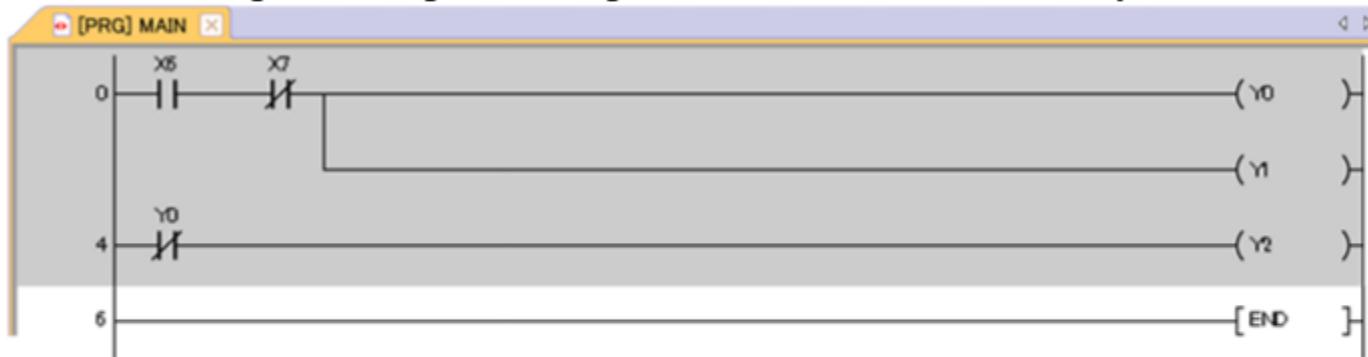
Đầu vào ghi chú thiết bị đã hoàn tất.  
Nhấn để tiến hành.

## 3.4

## Chuyển đổi chương trình thành dạng thực thi

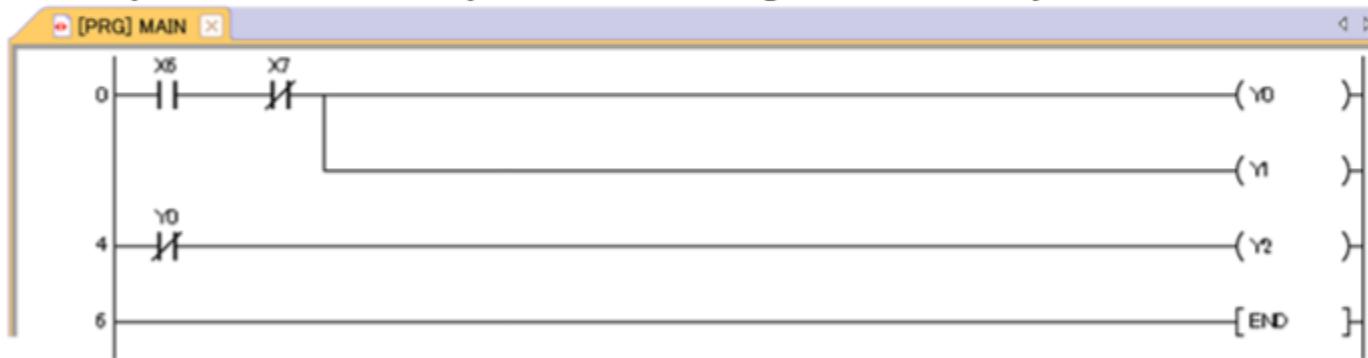
Sau khi hoàn tất chương trình, bạn cần phải chuyển đổi nó thành dạng có thể thực thi trong mô-đun CPU. Không thể thực thi hoặc lưu các chương trình không thê' đa'o ngươ.c.

Màu nền của chương trình không thê' đa'o ngươ.c có màu xám như hình dưới đây.



Chuyển đổi

Sau khi chuyển đổi, màu nền sẽ thay đổi thành màu trắng như hình dưới đây.



Trên trang tiếp theo, hãy thử chuyển đổi chương trình bằng cửa sổ giả lập.

## 3.4

## Chuyển đổi chương trình thành dạng thực thi

MELSOFT Series GX Works2 (Unset Project) - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

Project

- Parameter
- Intelligent Function Module
- Global Device Comment
- Program Setting
- POU
  - Program
  - MAIN
- Local Device Comment
- Device Memory
- Device Initial Value

Project User Library Connection Destination

```

    X0 ---|--- Y0 (Robot start signal)
           |----- Y1 (Operator lamp)
           |----- Y2 (Stop lamp)
           |
           +---- X7 (Door open)
           |
           +---- Y0 (Robot start signal)
           |
           +---- END
  
```

Khi chương trình được chuyển đổi, màu nền sẽ thay đổi từ màu xám sang màu trắng.

Chương trình đã được chuyển đổi.  
Nhấn để tiến hành.

Unlabeled L02 Host Station 5/7Step N0.5

**3.5****Lưu dự án**

Sau khi kết thúc chuyển đổi chương trình, hãy lưu dự án bao gồm các chương trình. Nếu GX Works2 chấm dứt mà không lưu dự án, các chương trình liên quan sẽ bị bỏ, vì vậy bạn nên lưu dự án của mình thường xuyên.

Khi lưu một dự án mới, hãy chỉ định các loại thông tin dự án sau đây. (Không cần thiết để lưu có ghi đè.)

Bạn nên bao gồm những thông tin giúp người khác dễ dàng hiểu được nội dung của điều không chương trình, tên hệ thống, v.v...

Mục	Bắt buộc	Mô tả
Đường dẫn đích lưu	✓	Chỉ định thư mục mà không gian làm việc sẽ được phân bổ vào đó.
Danh sách không gian làm việc/dự án		Nếu một hoặc nhiều không gian làm việc đã tồn tại trong thư mục chỉ định tại "Đường dẫn đích lưu", các không gian làm việc hiện tại sẽ được liệt kê.
Tên không gian làm việc	✓	Chỉ định tên không gian làm việc lên đến 128 ký tự.
Tên dự án	✓	Chỉ định một tên dự án lên đến 128 ký tự.
Tiêu đề		Chỉ định tiêu đề dự án lên đến 128 ký tự. Tham số này rất hữu ích khi bạn muốn gán một tên dài không vừa với trường "Tên dự án".

**Không gian làm việc** là một thư mục để quản lý nhiều dự án.

Ví dụ của việc sử dụng không gian làm việc được hiển thị dưới đây. (Các dự án được quản lý đối với từng loại xe trong dây chuyền sản xuất ô-tô.)

Tên không gian làm việc	Tên dự án	Tiêu đề
Dây chuyền sản xuất ô-tô	Dây chuyền sản xuất loại A	Chương trình vận hành bình thường cho việc điều khiển dây chuyền sản xuất loại A
	Dây chuyền sản xuất loại B	Chương trình vận hành bình thường cho việc điều khiển dây chuyền sản xuất loại B
	Dây chuyền sản xuất loại C	Chương trình vận hành bình thường cho việc điều khiển dây chuyền sản xuất loại C

**Lưu ý:**

- Nếu lưu dự án có chứa một chương trình không thể đảo ngược, chỉ có chương trình không thể đảo ngược đó bị loại bỏ. Trước khi lưu dự án, hãy thực hiện chuyển đổi chương trình như bạn đã học ở Mục 3.4.
- Chỉ định đường dẫn đích lưu, tên không gian làm việc và tên dự án để tổng số ký tự không vượt quá 150.

Trên trang tiếp theo, hãy cố gắng lưu dự án bằng cửa sổ giả lập.



## 3.5

## Lưu dự án

MELSOFT Series GX Works2 C:\SequenceProgram\le\_Learning\Robot\_Control - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

```

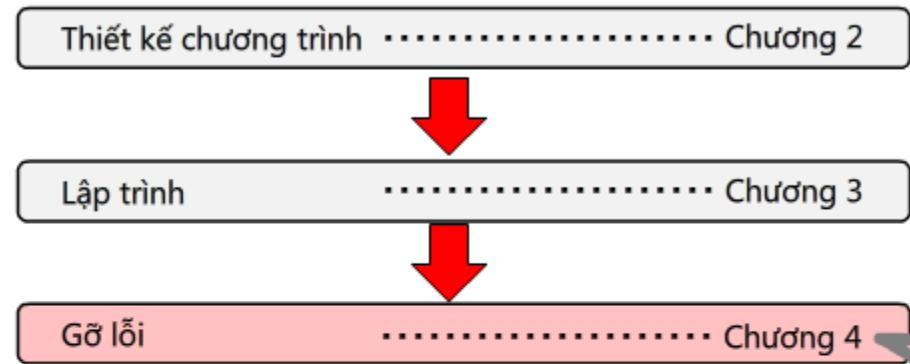
    graph TD
        StartStrt((X8)) ---|>| OpenDoorSensor((X7))
        OpenDoorSensor ---|>| RobotStartSignal((Y0))
        
        RobotStartSignal ---|>| StopLamp((Y2))
        StopLamp ---|>| End((END))
    
```

Dự án đã được lưu.  
Nhấn để tiến hành.

Unlabeled | L02 | Host Station | 6/7Step | N0.5

## Chương 4 Gỡ lỗi

Trong Chương 4, bạn sẽ tìm hiểu cách ghi các chương trình PLC vào mô-đun CPU và gỡ lỗi chúng.



### Các bước học tập trong Chương 4

- 4.1 Gỡ lỗi
  - 4.1.1 Gỡ lỗi chương trình không cần mô-đun CPU
  - 4.1.2 Thay đổi trạng thái một thiết bị I/O
  - 4.1.3 Giám sát trạng thái thiết bị
- 4.2 Ghi chương trình vào mô-đun CPU
- 4.3 Kích hoạt chương trình đã ghi
- 4.4 Chạy chương trình
- 4.5 Gỡ lỗi chương trình
- 4.6 Kiểm tra Vận hành hệ thống PLC
- 4.7 Vận hành hệ thống PLC
- 4.8 Kết luận

## 4.1

## Gỡ lỗi là gì?

Sau khi đã viết được một chương trình hay đoạn chương trình, cần phải kiểm tra mã để đảm bảo rằng chương trình hoạt động đúng như mong đợi.

Các lỗi phần mềm (khi mã đã viết không thực hiện đúng như dự kiến) được gọi là "**bugs**", và quá trình tìm kiếm nguyên nhân của hành vi không mong muốn và hiệu chỉnh nó được gọi là "**debugging**".

Kiểm tra và gỡ lỗi là những bước cần thiết trong việc tạo các chương trình.

Đặc biệt là ở các bộ điều khiển lập trình bởi nếu xuất hiện lỗi chúng có thể làm cho hệ thống dừng lại, gây hư hỏng thiết bị, hoặc gây ra các tai nạn khác.

Bảng sau liệt kê một vài chức năng trong GX Works2 có thể giúp quá trình gỡ lỗi.

Tên chức năng	Mô tả
Bộ giả lập	Chức năng này được sử dụng để giả lập việc thực thi chương trình ngay cả khi không có mô-đun CPU. Chức năng này có thể được dùng để gỡ lỗi trong môi trường không có sẵn mô-đun CPU.
Trình giám sát	Chức năng này cho phép theo dõi tình trạng thực thi và tình trạng của mỗi thiết bị trong quá trình thực thi của mô-đun CPU. Có thể sử dụng nhiều chức năng của trình giám sát tùy thuộc vào ứng dụng, chẳng hạn như giám sát trình lập trình PLC dạng thang (ladder), theo dõi riêng các thiết bị đã đăng ký và giám sát tất cả các thiết bị theo lô.
Thay đổi giá trị hiện tại	Chức năng này có thể buộc thay đổi trạng thái thiết bị (bit; ON ↔ OFF, word: giá trị hiện tại) trong quá trình thực thi của mô-đun CPU. Chức năng này rất hữu ích cho việc thay đổi giá trị hiện tại của một thiết bị word hoặc tình trạng của rờ-le gắn trong.
Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức	Chức năng này có thể buộc thay đổi trạng thái (ON ↔ OFF) của một thiết bị I/O đã đăng ký trong quá trình thực thi của mô-đun CPU. Để gỡ lỗi hoặc xác minh hoạt động với một mô-đun CPU duy nhất, chức năng này có thể được sử dụng thay thế cho công tắc.

Các chức năng này sẽ được giải thích chi tiết hơn liên quan đến quá trình gỡ lỗi suốt phần còn lại của chương này.

### Ghi chú về gỡ lỗi

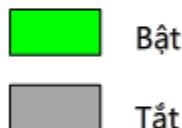
Không được thực hiện các tác vụ gỡ lỗi trong khi bộ điều khiển có thể lập trình được kết nối với các thiết bị I/O vật lý. Các lỗi trong chương trình, thiết bị I/O cưỡng bức, hoặc những thay đổi về giá trị word có thể dẫn đến hư hỏng thiết bị bên ngoài hoặc tồi tệ hơn.

Nếu một hệ thống PLC bị ngắt kết nối không sử dụng được, hãy dùng chức năng giả lập.

**4.1.1****Gỡ lỗi chương trình không cần Mô-đun CPU**

Nếu không có sẵn mô-đun CPU để gỡ lỗi, hãy sử dụng **chức năng bộ giả lập**.

Một chương trình có thể chạy trên mô-đun CPU ảo được cung cấp bởi phần mềm mà không cần dùng đến mô-đun CPU thực sự.



Mục	Trạng thái	Mô tả
Switch (Công tắc)	RUN	Chạy mô-đun CPU ảo.
	STOP	Dừng mô-đun CPU ảo.
	RESET	Cài lại mô-đun CPU ảo. (Được bật chỉ ở trạng thái STOP)
LED (Đèn LED)	MODE	Cho biết tình trạng MODE (CHẾ ĐỘ) của CPU ảo.
	RUN	Cho biết tình trạng hoạt động của CPU ảo. •Bật: Trạng thái RUN •Tắt: Trạng thái STOP
	ERR	Cho biết tình trạng lỗi của mô-đun CPU ảo. Nếu lỗi đang xuất hiện, đèn LED sẽ bật hoặc nhấp nháy.
	USER	Cho biết có lỗi người dùng đã xảy ra trong CPU ảo hay không. Bật hoặc nhấp nháy khi xảy ra lỗi.

**Ghi chú về việc sử dụng các chức năng giả lập**

- Gỡ lỗi bằng cách sử dụng chức năng giả lập không đảm bảo rằng chương trình PLC sẽ vận hành chính xác sau khi gỡ lỗi.
- Chức năng bộ giả lập sẽ thực thi đầu vào/đầu ra của dữ liệu với mô-đun I/O sử dụng bộ nhớ giả lập.  
Chức năng này không hỗ trợ một số lệnh, hàm và bộ nhớ thiết bị. Vì vậy, kết quả vận hành với chức năng bộ giả lập có thể khác với tình huống có mô-đun CPU thực sự.

Trên trang tiếp theo, hãy thử sử dụng chức năng bộ giả lập với cửa sổ giả lập.

## 4.1.1

## Gỡ lỗi chương trình không cần Mô-đun CPU

TOC

MELSOFT Series GX Works2 C:\SequenceProgram\le\_Learning\Robot\_Control - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

```

    graph TD
        StartStrt((X8)) ---|AND| S0[Step 0]
        DoorOpen[X7  
Door open  
sensor] ---|AND| S0
        S0 ---|OUT| RobotStart[Y0  
Robot start  
signal]
        
        RobotStart ---|AND| S4[Step 4]
        X0[Robot start  
signal] ---|AND| S4
        S4 ---|OUT| OperatorLamp[Y1  
Operator  
lamp]
        
        RobotStart ---|AND| S6[Step 6]
        StopLampP[X1  
Stop lamp  
p] ---|AND| S6
        S6 ---|OUT| END[END]
    
```

Bây giờ bạn đã học cách sử dụng tính năng giả lập.  
Nhấn để tiến hành.

Unlabeled L02 Host Station 6/7Step N0.5

## 4.1.2

## Thay đổi trạng thái thiết bị I/O

Khi gỡ lỗi một chương trình PLC bằng mô-đun CPU mà không có thiết bị I/O nào được kết nối vào hoặc bằng chức năng bộ giả lập, hãy sử dụng chức năng **Forced Input Output Registration/Cancellation** (Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức) để thay đổi trạng thái ON/OFF của một thiết bị I/O.

Trạng thái của các thiết bị I/O đã đăng ký có thể được cưỡng bức thay đổi sang ON hoặc OFF bằng phần mềm.

(Sê-ri MELSEC-Q và MELSEC-L): Từ màn hình "Forced Input Output Registration/Cancellation"  
(Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức)

(Sê-ri MELSEC-F): Từ màn hình "Modify Value" (Thay đổi giá trị)



Màn hình Forced Input Output Registration/Cancellation  
(Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức)  
(Sê-ri MELSEC-Q và MELSEC-L)

### Để thay đổi trạng thái của các thiết bị khác

Để thay đổi thiết bị hiện tại của một thiết bị word hoặc trạng thái ON/OFF của một rờ-le gắn trong, hãy sử dụng **current value change function** (chức năng thay đổi giá trị hiện tại).

Để biết chi tiết, hãy tham khảo sách hướng dẫn.



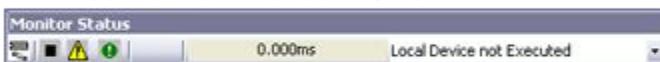
Màn hình Modify Value (Thay đổi giá trị)  
(Sê-ri MELSEC-F)

**4.1.3****Giám sát trạng thái thiết bị**

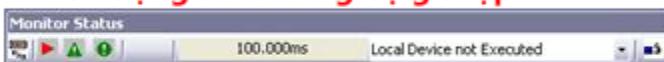
Khi khởi động giả lập, việc giám sát sẽ tự động bắt đầu. Để vào chế độ giám sát khi kết nối vào một CPU bộ điều khiển có thể lập trình thực sự, chỉ cần nhấp vào Online, Monitor, và sau đó là Start Monitoring. Hoặc sử dụng phím tắt bàn phím, F3.

Trong chế độ trình giám sát các giá trị và trạng thái của tất cả các thiết bị được sử dụng trong chương trình đều có thể được nhìn thấy phủ lên trên mã chương trình. Điều này cho phép người dùng thấy được việc thay đổi các giá trị bao gồm cả những ảnh hưởng của việc sử dụng chức năng "Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức".

Ngoài ra, Thanh trạng thái Trình giám sát sẽ xuất hiện và bao gồm những thông tin cơ bản để xác định tình trạng CPU hoặc CPU ảo. Hãy tham khảo bảng dưới đây để hiểu rõ các thông tin được cung cấp bởi Thanh trạng thái Trình giám sát.

**Khi kết nối với mô-đun CPU**

Trạng thái	Biểu tượng/chỉ báo
Tình trạng kết nối	Khi kết nối với mô-đun CPU Khi sử dụng chức năng bộ giả lập
Trạng thái RUN/STOP	RUN STOP
Trạng thái ERR.	ERR. tắt ERR. bật ⇨  ERR nhấp nháy
Trạng thái USER	USER tắt USER bật ⇨  USER nhấp nháy
Thời gian quét	0.000ms
Trạng thái có/không có lệnh được hỗ trợ	Tồn tại các lệnh không được hỗ trợ hay không khi thực thi chức năng bộ giả lập. Lệnh không được hỗ trợ không tồn tại.

**Khi sử dụng chức năng bộ giả lập**

Mô tả
Hiển thị trạng thái của kết nối với mô-đun CPU hoặc chức năng bộ giả lập.
Hiển thị trạng thái hoạt động của CPU (RUN hoặc STOP).
Hiển thị trạng thái lỗi của mô-đun CPU.
Hiển thị trạng thái lỗi người dùng của mô-đun CPU.
Hiển thị thời gian quét tối đa của mô-đun CPU được theo dõi.
Hiển thị xem có tồn tại lệnh không được hỗ trợ hay không khi thực thi chức năng bộ giả lập. Nhấp vào biểu tượng này sẽ mở cửa sổ Lệnh/Thiết bị không được hỗ trợ.

**4.1.3****Giám sát trạng thái thiết bị**

Trong chế độ giám sát, trạng thái hiện tại của tất cả các thiết bị trong chương trình đều nhìn thấy được.

**Hiển thị trạng thái thiết bị bit (ON/OFF)**

Trạng thái ON/OFF được hiển thị trong quá trình giám sát như hình dưới đây.

Trạng thái OFF

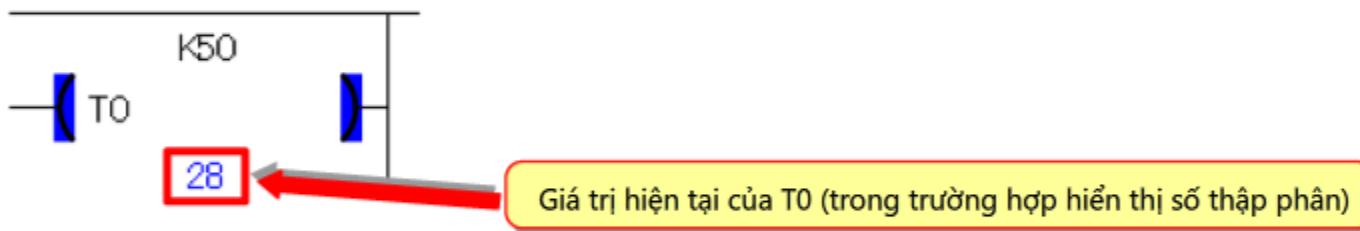
Trạng thái ON

\* Loại hiển thị này chỉ áp dụng cho các lệnh SET, RST, PLS, PLF, SFT, SFTP, MC và các lệnh so sánh loại tiếp điểm.

Lưu ý rằng đối với lệnh RST, chỉ hiển thị được trạng thái ON/OFF.

**Hiển thị giá trị hiện tại của thiết bị word (hiển thị số hệ thập phân/thập lục phân)**

Giá trị hiện tại trong quá trình giám sát được hiển thị như hình dưới đây.

**Chi giám sát các thiết bị cụ thể**

Khi giám sát một chương trình rất lớn và phức tạp, có thể sẽ hiệu quả hơn khi chỉ theo dõi một số thiết bị nhất định cần quan tâm. Để thực hiện điều này, GX Works2 bao gồm các cửa sổ theo dõi cho phép người dùng dễ dàng thêm các thiết bị mà họ quan tâm, thấy được trạng thái hiện tại của chúng, và sửa đổi các giá trị của các thiết bị đó trong quá trình giám sát. Để biết chi tiết, hãy tham khảo Hướng dẫn sử dụng GX Works2 (Bản thường dùng).

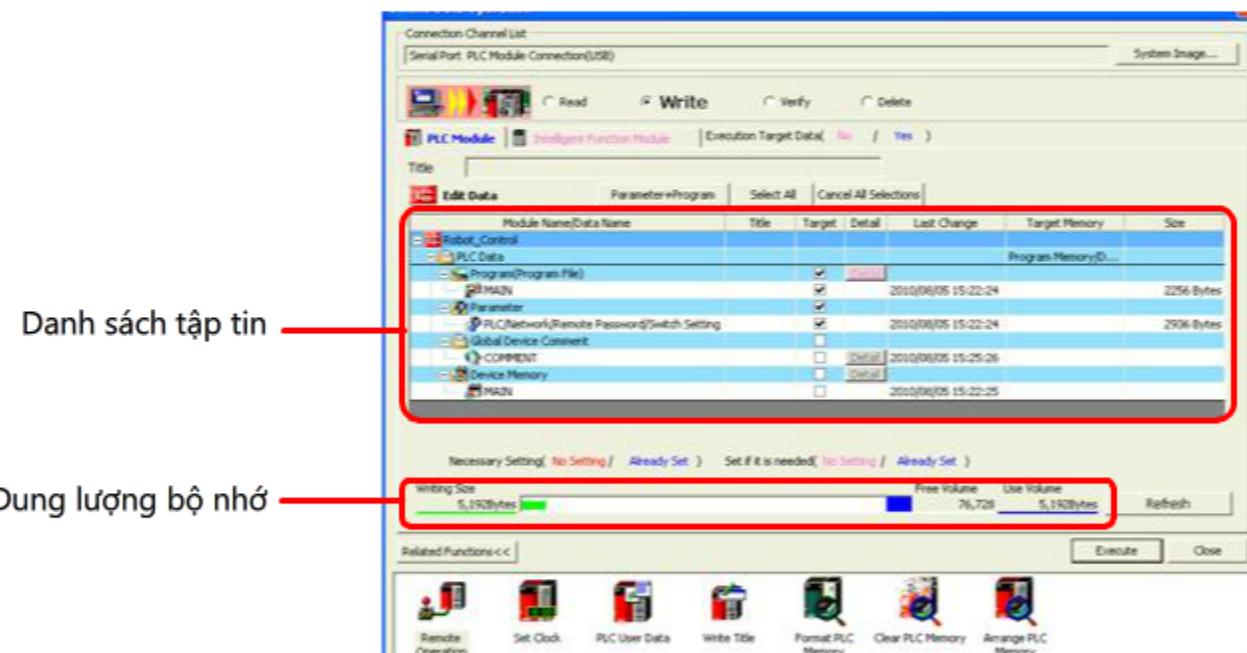
Watch 1					
Device/Label	Current Value	Data Type	Class	Device	Comment
X7	-	Bit		X7	Door open sensor
Y0	-	Bit		Y0	Robot start signal
Y1	-	Bit		Y1	Operation lamp
Y0	-	Bit		Y0	Robot start signal
Y2	-	Bit		Y2	Stop lamp
Y0	-	Bit		Y0	Robot start signal

## 4.2

## Ghi chương trình vào mô-đun CPU

Trước khi thực hiện bất cứ sửa lỗi nào bằng mô-đun CPU thực sự, hãy đặt CPU vào **chế độ STOP**, đảm bảo kết nối với CPU đã được thành lập, và ghi các chương trình và tham số vào bộ nhớ chương trình.

Như trong hình dưới đây, các chức năng chính của cửa sổ **Write to PLC** (Ghi sang PLC) sẽ cho phép người dùng lựa chọn các tập tin mong muốn cần ghi, chọn vị trí của chúng cũng như xác nhận dung lượng bộ nhớ của CPU. Ba nút trên danh sách tập tin sẽ cho phép người dùng nhanh chóng chọn các tập tin mong muốn cần ghi. Phổ biến nhất, được sử dụng trong giả lập sau đây là "**Parameter+Program**".



Trên trang tiếp theo, hãy thử ghi sang mô-đun CPU bằng cách sử dụng cửa sổ giả lập.

## 4.2

**Ghi chương trình vào mô-đun CPU**

MELSOFT Series GX Works2 C:\SequenceProgram\le\_Learning\Robot\_Control - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN

**Project**

- Parameter
- Intelligent Function Module
- Global Device Comment
- Program Setting
- POU
  - Program
  - MAIN
- Local Device Comment
- Device Memory
- Device Initial Value

Project User Library Connection Destination

```

    graph TD
      StartStrt((X8)) --- Parallel[Parallel]
      Parallel --- Y0_RobotStart[Robot start signal]
      Parallel --- StopSignal[X0]
      StopSignal --- RobotStop[Robot stop signal]
      RobotStop --- OperatorLamp[Y1 Operator lamp]
      RobotStop --- StopLamp[Stop lamp]
      StopLamp --- StopLamp[Stop lamp]
      StopLamp --- END[END]
  
```

Chương trình lúc này sẽ được ghi vào mô-đun PLC.  
Nhấn để tiến hành

Unlabeled | L02 | Host Station | 6/7Step | NL7

**4.3****Kích hoạt chương trình đã ghi**

(Sê-ri MELSEC-F): Không cần vận hành sau đây.

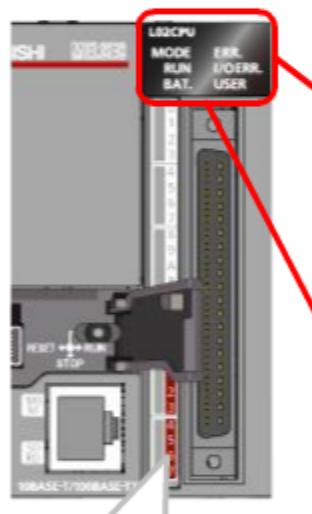
(Sê-ri MELSEC-Q và MELSEC-L): Vận hành sau đây là cần thiết.

Sau khi ghi chương trình vào mô-đun CPU, hãy **thiết lập lại** mô-đun CPU.

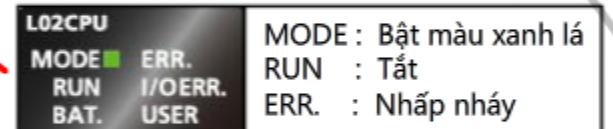
Chương trình đã ghi sẽ không được kích hoạt trừ khi mô-đun CPU được thiết lập lại.

\* Thao tác này là không cần thiết nếu sử dụng chức năng bộ giả lập để gỡ lỗi.

Thiết lập lại mô-đun CPU như sau:



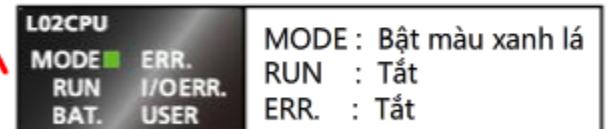
- (1) Bấm và giữ công tắc RESET/STOP/RUN ở mặt trước của mô-đun CPU sang vị trí RESET (trong 1 giây hoặc hơn).  
[Đang tiến hành thiết lập lại]



Nhấn giữ trong 1 giây  
hoặc hơn.



- (2) Nhả công tắc sau khi cả hai đèn MODE LED đã sáng và ERR. nhấp nháy đều tắt đi.  
[Thiết lập lại hoàn tất]



- (3) Công tắc này sẽ trở về vị trí STOP để hoàn tất việc thiết lập lại.

## 4.4

## Chạy chương trình

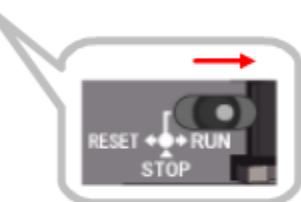
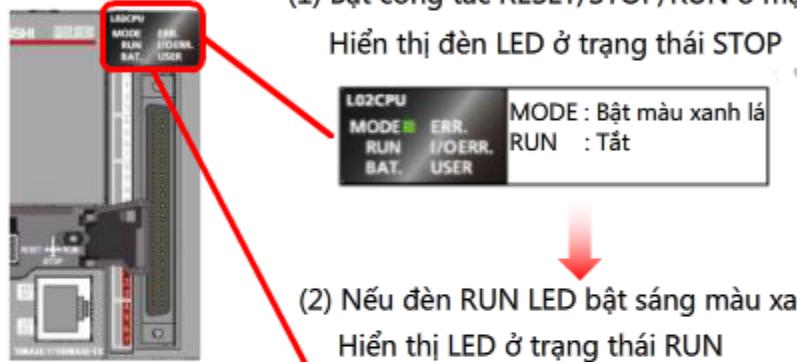
## Sê-ri MELSEC-Q và MELSEC-L

Sau khi thiết lập lại hoàn tất, hãy chạy chương trình.

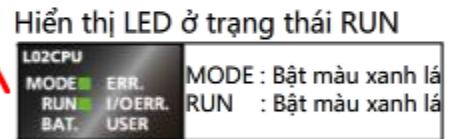
Đưa mô-đun CPU về **trạng thái RUN** như sau để chạy chương trình.

\* Thao tác này là không cần thiết nếu sử dụng chức năng bộ giả lập để gỡ lỗi.

(1) Bật công tắc RESET/STOP/RUN ở mặt trước của mô-đun CPU đến vị trí RUN.



(2) Nếu đèn RUN LED bật sáng màu xanh lá, chương trình đang chạy bình thường.



## Sê-ri MELSEC-F

Sau khi ghi chương trình vào thiết bị chính, chuyển thiết bị chính về trạng thái RUN như dưới đây để chạy chương trình. (Không cần thao tác cài lại).

(1) Bật công tắc RUN/STOP trên bảng điều khiển trước của thiết bị chính tới vị trí RUN.



Hiển thị đèn LED ở trạng thái STOP

(2) Nếu các đèn LED RUN sáng lên, chương trình đang chạy bình thường.



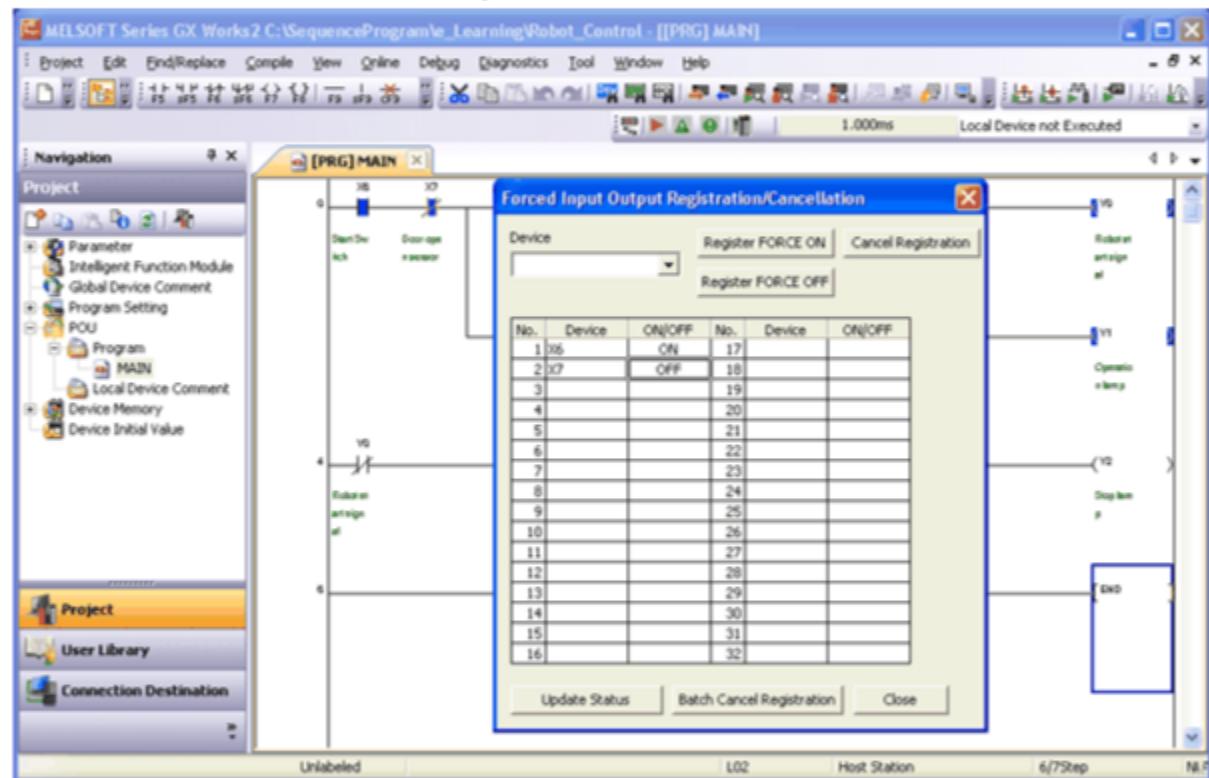
Hiển thị LED ở trạng thái RUN

## 4.5

**Gỡ lỗi chương trình**

Sau khi chạy mô-đun CPU, hãy sử dụng chức năng Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức để thay đổi trạng thái từng thiết bị và theo dõi kết quả (đầu ra) trên trình lập trình PLC dạng thang (ladder).

(Mẫu màn hình của sê-ri MELSEC-Q và MELSEC-L)



Trên trang tiếp theo, hãy thử gỡ lỗi chương trình bằng cửa sổ giả lập.

## 4.5

## Gỡ lỗi chương trình

MELSOFT Series GX Works2 C:\SequenceProgram\le\_Learning\Robot\_Control - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN Project User Library Connection Destination

1.000ms Local Device not Executed

```

    graph TD
        X8[0: X8] --> DOOR_OPEN[Door open]
        DOOR_OPEN --> DOOR_SENSOR[Door sensor]
        DOOR_SENSOR --> Y0_1[Y0: 2]
        
        Y0_1 --> ROBOT_START[Robot start signal]
        ROBOT_START --> STOP_LAMP[Stop lamp]
        STOP_LAMP --> Y2[Y2: 6]
        
        Y2 --> END[END: 7]
    
```

Gỡ lỗi của chương trình được hoàn tất.  
Nhấn để tiến hành.

Unlabeled | L02 | Host Station | 6/7Step | NL2

## 4.6

## Kiểm tra Vận hành hệ thống PLC

Sau khi hoàn tất gỡ lỗi chương trình, hãy ghi chương trình vào hệ thống PLC thực sự để kiểm tra vận hành lần cuối.

Vận hành thiết bị I/O thực tế để xác nhận rằng nó hoạt động như thiết kế.

Ngay cả khi vận hành thiết bị I/O, có thể kiểm tra trạng thái của mỗi thiết bị bằng chức năng giám sát của GX Works2.

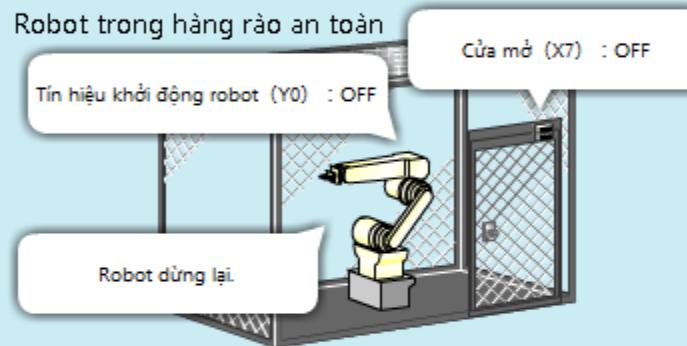
## Vận hành của hệ thống ví dụ

Nhấn vào bên trong vòng tròn đỏ

Bảng điều khiển robot



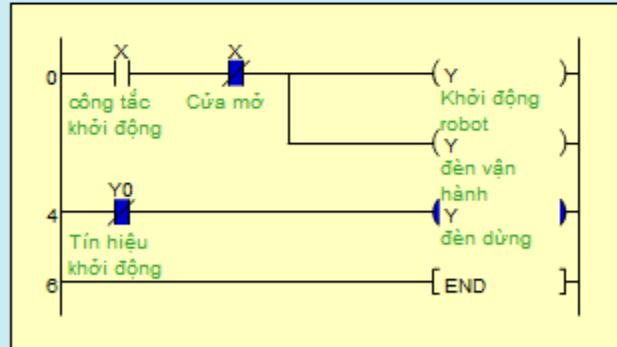
Robot trong hàng rào an toàn



Khi bạn cài công tắc khởi động (X6) sang OFF, Tín hiệu khởi động robot (Y0) sẽ tắt để dừng vận hành robot. Đồng thời, đèn vận hành (Y1) trên bảng điều khiển sẽ tắt, và đèn dừng (Y2) sẽ bật.

**Chạy lại**

Trước



## 4.7

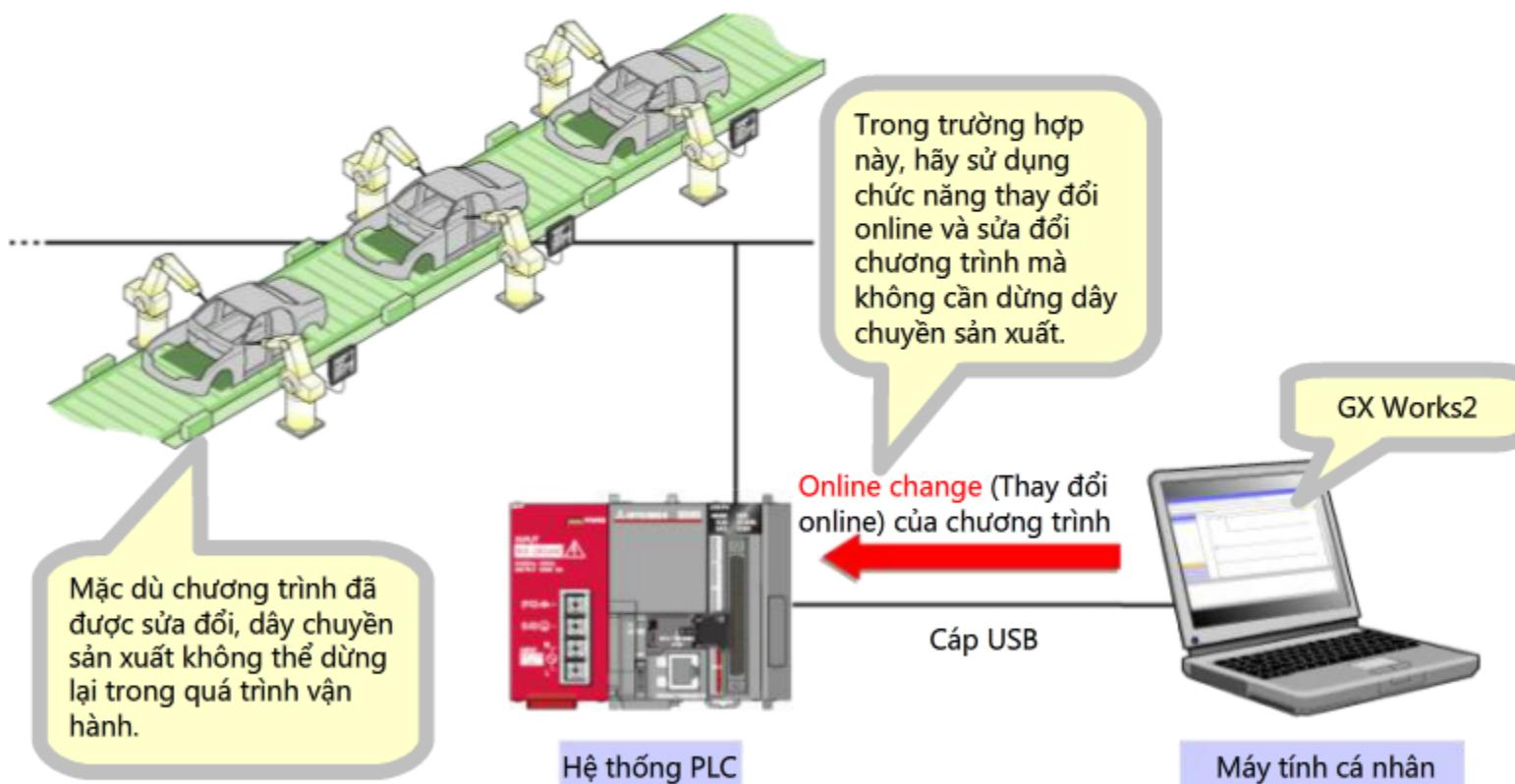
## Vận hành hệ thống PLC

Sau khi hoàn tất việc xác minh vận hành, hãy chạy hệ thống PLC để bắt đầu vận hành.

### Nếu chương trình cần được sửa đổi trong hệ thống đang chạy

Việc sửa đổi chương trình như hiệu chỉnh lỗi hoặc mở rộng hệ thống có thể được yêu cầu sau khi bắt đầu vận hành hệ thống. Thông thường, hệ thống (mô-đun CPU) cần phải được dừng lại để ghi chương trình sửa đổi, nhưng điều này không phải lúc nào cũng làm được. Để giải quyết vấn đề này, GX Works cung cấp một chức năng thay đổi online, được sử dụng để ghi các chương trình mà không cần dừng mô-đun CPU đang chạy.

#### Ví dụ: Dây chuyền sản xuất ô-tô chạy liên tục 24 giờ



Trên trang tiếp theo, hãy thử chức năng thay đổi online bằng cửa sổ giả lập.

## 4.7

## Vận hành hệ thống PLC

TOC

MELSOFT Series GX Works2 C:\SequenceProgram\le\_Learning\Robot\_Control - [[PRG] MAIN]

Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help

Navigation [PRG] MAIN Project User Library Connection Destination

1.000ms Local Device not Executed

Robot start signal

Start switch

Robot stop signal

Stop button

Operation lamp

Y0 Y1 Y2

END

Thay đổi online của chương trình sửa đổi đã hoàn tất.  
Nhập để tiến hành.

```

    graph TD
        StartSwitch[X8] ---|NO| StartSignal[Robot start signal]
        StopButton[Y0] ---|NC| StopSignal[Robot stop signal]
        StartSignal ---|Set| Y1[Operation lamp]
        StopSignal ---|Reset| Y1
        Y1 ---|Set| End[END]
    
```

Unlabeled L02 Host Station 2/7Step N0.5

**4.8****Kết luận**

Khóa học này đã hoàn tất việc giải thích cơ bản về thiết kế phần mềm bộ điều khiển có thể lập trình.

Trong khóa học này bạn đã học được:

- Các yếu tố cần thiết cho việc lập trình một hệ thống PLC
- Một số nguyên tắc cơ bản để thiết kế chương trình bao gồm việc sử dụng các chú thích
- Cách thức sử dụng GX Works2 để thực hiện các tác vụ cơ bản khi lập trình máy tính
- Một vài kỹ thuật được dùng để gỡ lỗi chương trình PLC

## Kiểm tra

## Bài kiểm tra cuối khóa

Vì bạn đã hoàn thành tất cả các bài học của **Khóa học Cơ Bản về PLC GX Works2**, bạn đã sẵn sàng tham gia bài kiểm tra cuối khóa. Nếu bạn không rõ về bất cứ chủ đề nào được trình bày, vui lòng nhân cơ hội này xem xét lại các chủ đề đó.

**Có tổng cộng 5 câu hỏi (15 mục) trong Bài kiểm tra cuối khóa này.**

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

### Làm thế nào ghi điểm bài kiểm tra

Sau khi chọn câu trả lời, hãy chắc chắn đã nhấp vào nút **Answer**. Câu trả lời của bạn sẽ bị mất nếu bạn tiếp tục mà không nhấp vào nút Answer. (Coi như là câu hỏi chưa được trả lời.)

### Kết quả điểm số

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

Câu trả lời đúng: **2**

Tổng số câu hỏi: **9**

Tỷ lệ phần trăm: **22%**

Để vượt qua bài kiểm tra, bạn  
phải trả lời đúng **60%** các câu  
hỏi.

Tiếp tục

Xem lại

Thư lai

- Nhấp vào nút **Tiếp tục** để thoát khỏi bài kiểm tra.
- Nhấp vào nút **Xem lại** để xem lại bài kiểm tra. (Kiểm tra câu trả lời đúng)
- Nhấp vào nút **Thư lai** để làm lại bài kiểm tra một lần nữa.

## Kiểm tra Bài kiểm tra cuối khóa 1

Chương trình mà bạn phụ trách đã được người khác tiếp nhận, và họ thấy khó mà hiểu được các mục điều khiển cho chương trình. Những biện pháp ứng phó thích hợp nào để ngăn chặn vấn đề này?

- Bằng cách sử dụng chức năng chú thích của GX Works2, hãy đưa ra cho chương trình tiêu đề và lời giải thích phù hợp.
- Giải thích bằng lời các mục điều khiển cho người mới.
- Tránh tiếp nhận một chương trình lớn phức tạp.
- Chuyển giao bảng tương ứng cho các thiết bị I/O và số hiệu thiết bị cùng với chương trình.

[Điểm số](#)[Lùi](#)

## Kiểm tra Bài kiểm tra cuối khóa 2

Hoàn tất đúng quy trình lập trình.

Bước 1 Thiết kế chương trình

Bước 2 ( Q1  )

Bước 3 ( Q2  )

Bước 4 Chuyển đổi chương trình

Bước 5 Lưu dự án

Bước 6 ( Q3  )

Bước 7 ( Q4  )

Bước 8 Chạy mô-đun CPU (RUN)

Bước 9 ( Q5  )

Bước 10 Kiểm tra Vận hành hệ thống PLC

## Kiểm tra Bài kiểm tra cuối khóa 3

Điền vào chỗ trống để hoàn tất giải thích về những gì cần phải làm sau khi hoàn thành chương trình.

Sau khi chương trình đã được viết xong, nó phải được kiểm tra để đảm bảo rằng chương trình hoạt động như mong đợi.

A ( ) (khi mã được viết không thực hiện đúng như dự kiến) được gọi là

a ( ) và quá trình tìm kiếm nguyên nhân và hiệu chỉnh nó được gọi là  
( )

Quá trình này là một bước quan trọng trong việc tạo chương trình.

Điểm số

Lùi

## Kiểm tra Bài kiểm tra cuối khóa 4

Chọn ứng dụng thích hợp cho từng chức năng của GX Works.

Chức năng	Ứng dụng
Giả lập	--Select-- ▾
Đăng ký/hủy bỏ đầu ra theo đầu vào cưỡng bức	--Select-- ▾
Thay đổi giá trị hiện tại	--Select-- ▾
Trình giám sát lập trình PLC dạng thang (ladder)	--Select-- ▾
Theo dõi	--Select-- ▾

Điểm số

Lùi

## Kiểm tra Bài kiểm tra cuối khóa 5

Chọn mô tả chính xác về chức năng thay đổi online.

- Chức năng này sẽ tự động dừng CPU, ghi chương trình vào CPU, và sau đó tự động chạy CPU.
- Chức năng này sẽ so sánh chương trình trong mô-đun CPU đang chạy với chương trình do GX Works2 mở.
- Chức năng này có thể ghi một chương trình vào mô-đun CPU sau khi dừng mô-đun CPU đang chạy một cách an toàn.
- Chức năng này thể ghi một chương trình vào mô-đun CPU đang chạy mà không cần dừng lại.

Điểm số

Lùi

## Kiểm tra ĐIỂM KIỂM TRA



Bạn đã hoàn thành Bài kiểm tra cuối khóa. Các kết quả của bạn được tóm lược như sau.  
Để kết thúc Bài kiểm tra cuối khóa, hãy tiếp tục đến trang tiếp theo.

Câu trả lời đúng: 0

Tổng số câu hỏi: 5

Tỷ lệ phần trăm: 0%

[Tiếp tục](#)

[Xem lại](#)

[Thư lai](#)

**Bạn đã không vượt qua bài kiểm tra.**

Bạn đã hoàn thành **khóa học Cơ bản về PLC GX Works2.**

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**