MITSUBISHI ELECTRIC
Changes for the Better

PLC
GX Works2 Advanced

This course provides knowledge of the functions to improve the development environment of the design site having problems about "productivity," "quality," "project management," and "security measures." The course is intended for sequence programmers who already use MELSOFT GX Works2.

L(NA)00115ENG

PLC_GX_Works2_Advanced_ENG

Introduction Purpose of the Course

This course provides knowledge of the functions to improve the development environment of the design site having problems about "productivity," "quality," "project management," and "security measures." The course is intended for sequence programmers who already use MELSOFT GX Works2.

Introduction Course Structure

The contents of this course are as follows.
We recommend that you start from Chapter 1.

Chapter 1 - Improving Development Environment Using GX Works2

You will learn the problems facing the design site and the development environment required for solving them.

Chapter 2 - Programming

You will learn the functions used for programming.

Chapter 3 - Debugging

You will learn the functions used for debugging.

Chapter 4 - Project Management and Security Measures

You will learn the functions for project management and security measures at the stage of development and maintenance after the start of system operation.

Final Test

Passing grade: 60% and higher

Introduction How to Use This e-Learning Tool

Go to the next page	⏭	Go to the next page.
Back to the previous page	⏮	Back to the previous page.
Move to the desired page	TOC	“Table of Contents” will be displayed, enabling you to navigate to the desired page.
Exit the learning	✕	Exit the learning. Window such as "Contents" screen and the learning will be closed.

Introduction Cautions for Use

Safety precautions

When you learn by using actual products, please carefully read the safety precautions in the corresponding manuals.

Precautions in this course

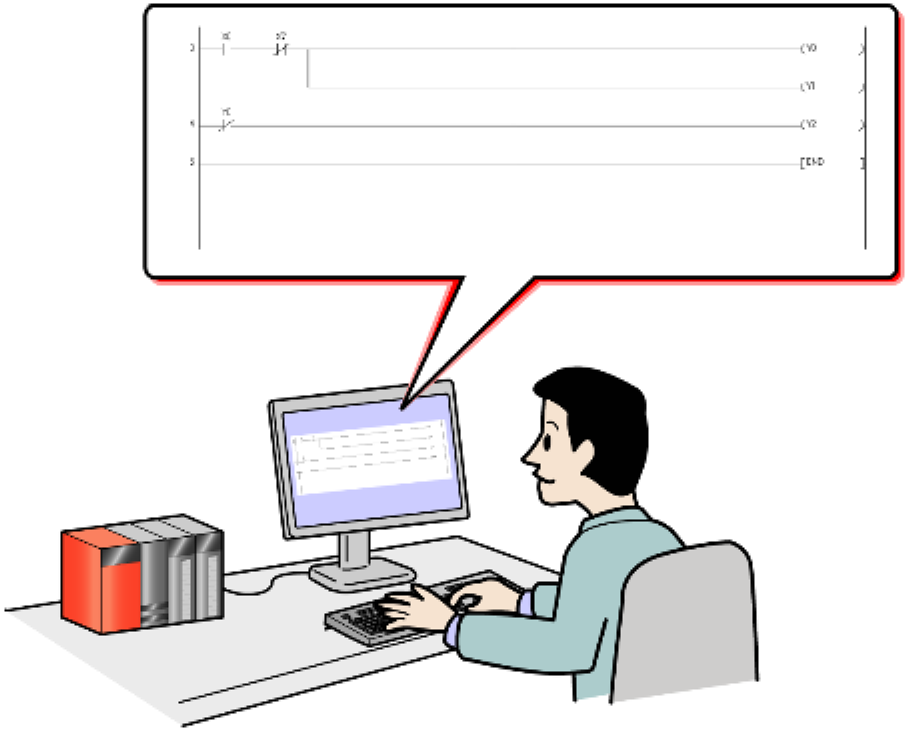
- The displayed screens of the software version that you use may differ from those in this course.

Chapter 1 Improving Development Environment Using GX Works2

Learning steps in Chapter 1

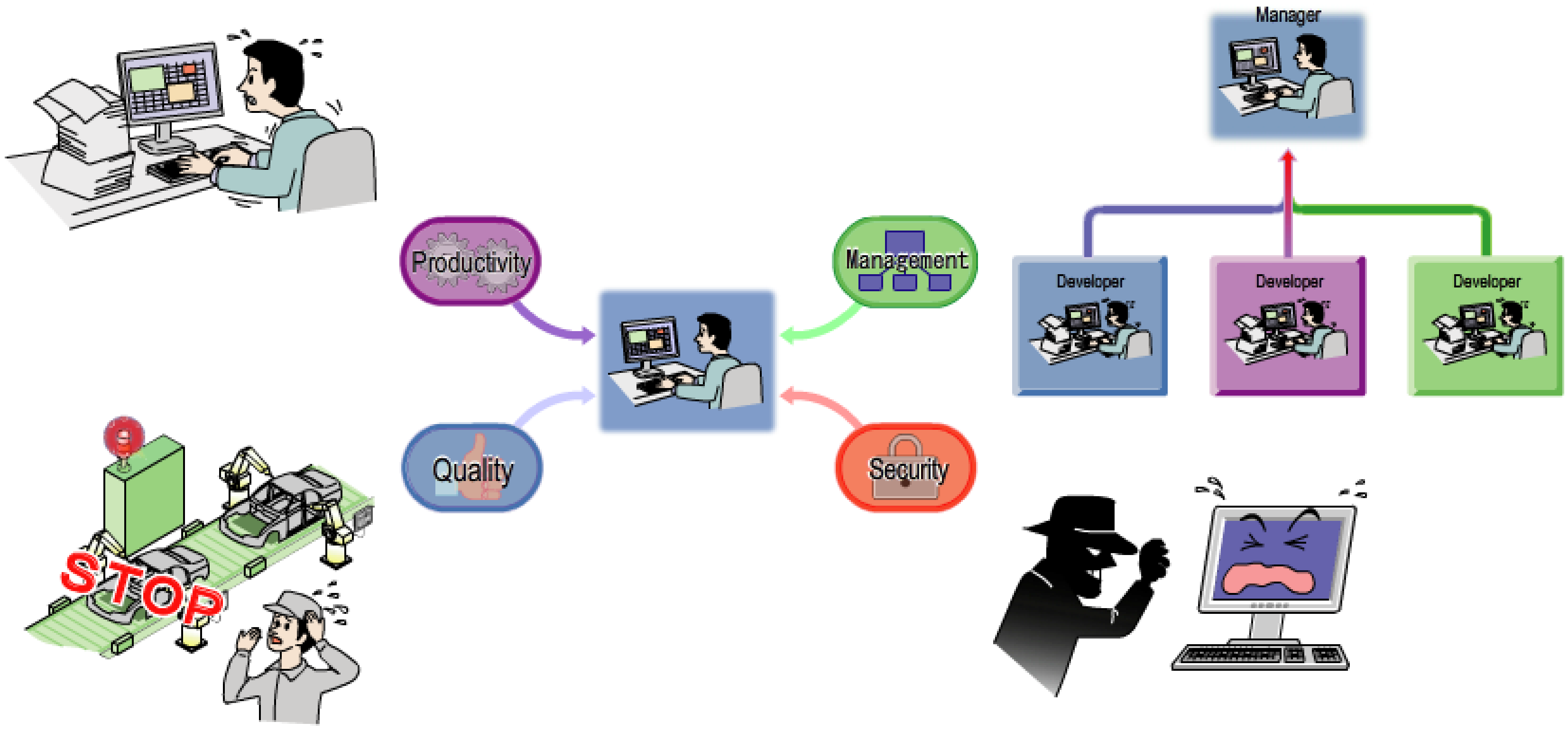
This course is intended for programmers working on the development of sequence programs. You will learn how to use the excellent functions in **MELSOFT GX Works2** to solve design site problems related to **"productivity," "quality," "project management,"** and **"security."**

In Chapter 1, you will learn about the problems facing the design site and the development environment required for solving these problems.



1.1 Problems Faced by the Design Site

While the design site is required to improve productivity to reduce development costs, it must also ensure that the developed programs are of high quality.
The design site must also conduct project management that allows team development and ensures quick recovery in the case of trouble.
Security measures are also required because sequence programs include important knowledge and data.



1.1.1 Improving Productivity

The design site is required to develop sequence programs to handle large-scale, complex processing.
However, the cost of development increases in line with the length of the development period.
Thus, you are asked to improve the productivity of program development in order to reduce the development cost.

GX Works2 provides the following functions to solve this problem:

- Label
- Function block
- Device initial value and device memory*
- Inline structured text
- Import from sample comment

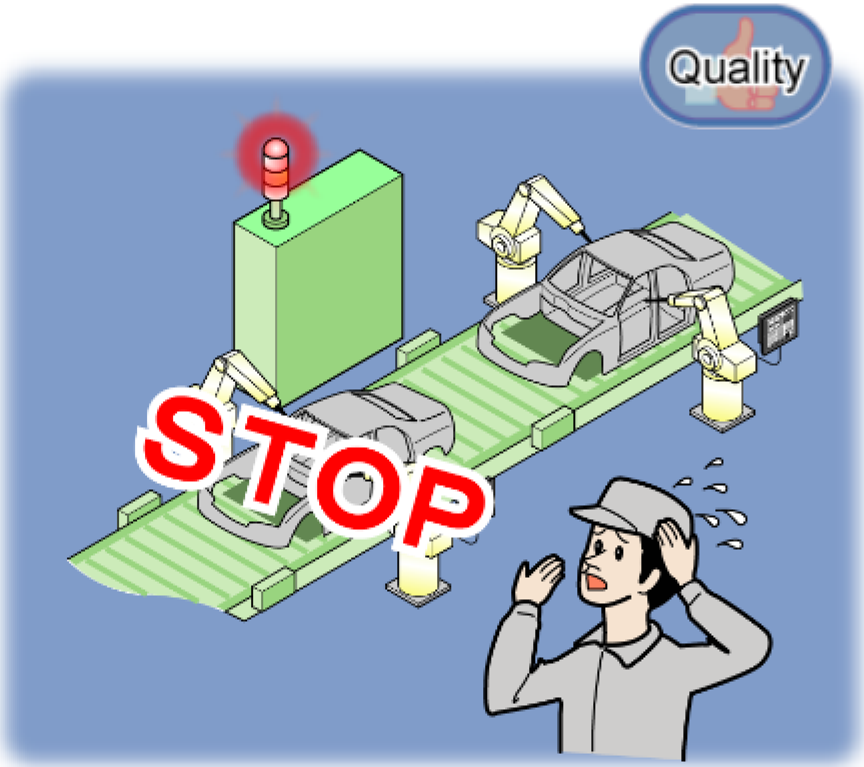
* The MELSEC-F series cannot configure device initial values.

1.1.2 Improving Quality

A bug in the sequence program could cause a system stop, fault, or accident, which would halt production and result in the loss of profit and reliability.
You are asked to develop quality programs that are free of any bugs.

GX Works2 provides the following functions to solve this problem:

- Label
- Function block
- Device initial value and device memory*
- Comment
- Import from sample comment
- Watch
- Cross reference
- Sampling trace*
- Executional conditioned device test*
- Step execution function*
- I/O system setting



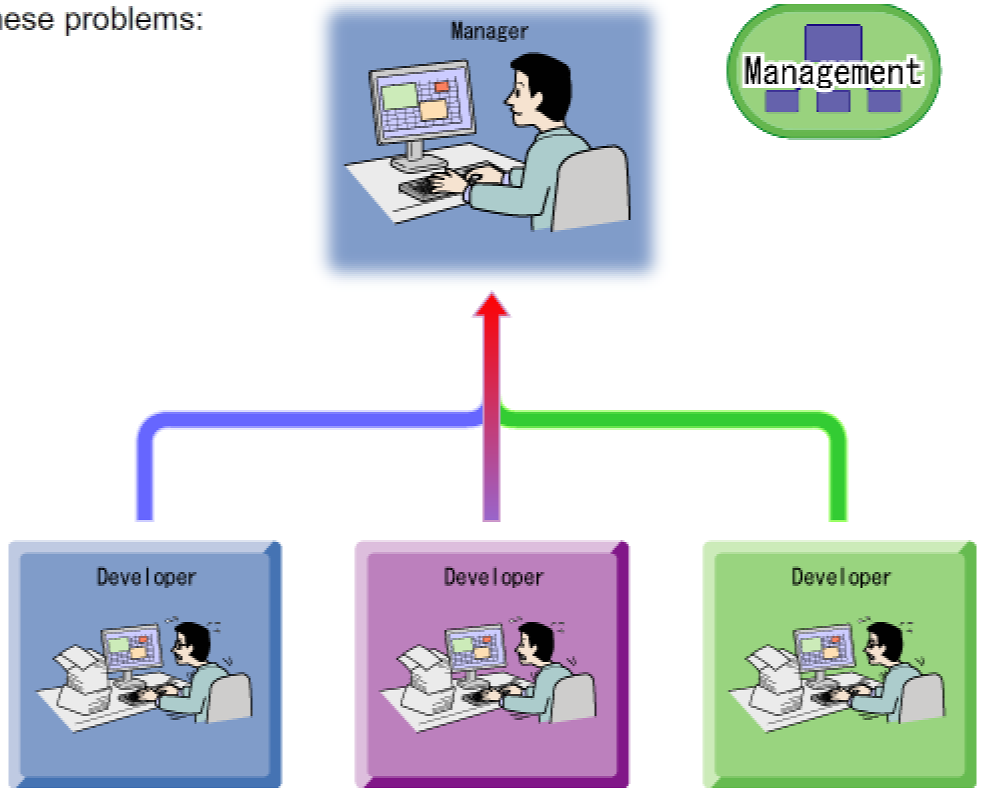
* The MELSEC-F series cannot use device initial value settings, sampling trace, executional conditioned device testing, and the step execution function.

1.1.3 Project Management

A large-scale sequence program is often developed by a team of programmers.
From the viewpoint of individual management authority and data confidentiality, limits must be established for the range of accessible data and usable functions.
In addition, to ensure quick recovery in the case of program loss due to programmable controller failure, you are asked to implement program version management and periodic program backup.

GX Works2 provides the following functions to solve these problems:

- Security
- Revision history
- Verify with PC



PLC_GX_Works2_Advanced_ENG

1.1.4 Security Measures

The sequence program includes strategically important knowledge and data.
The **leakage** of know-how and data from the program to the outside could have a devastating impact on business.
Unauthorized modification of the program could lead to production problems such as by stopping the system.
The appropriate security measures must be taken to prevent these problems.

GX Works2 provides the following function to solve these problems:

- Security

1.2 Learning Procedure

In this course, you will learn about the GX Works2 functions by following the procedure for actual system development.

(1) Programming .... Chapter 2

↓

(2) Debugging .... Chapter 3

↓

(3) Project management and security measures .... Chapter 4

<Explanation of icons>

The icons displayed on the pages in Chapters 2, 3 and 4 correspond to the respective improvement functions, as shown below.

Productivity	Function used for improving program productivity
Quality	Function used for improving program quality
Management	Function used for project management
Security	Function used for security measures

Chapter 2 Programming

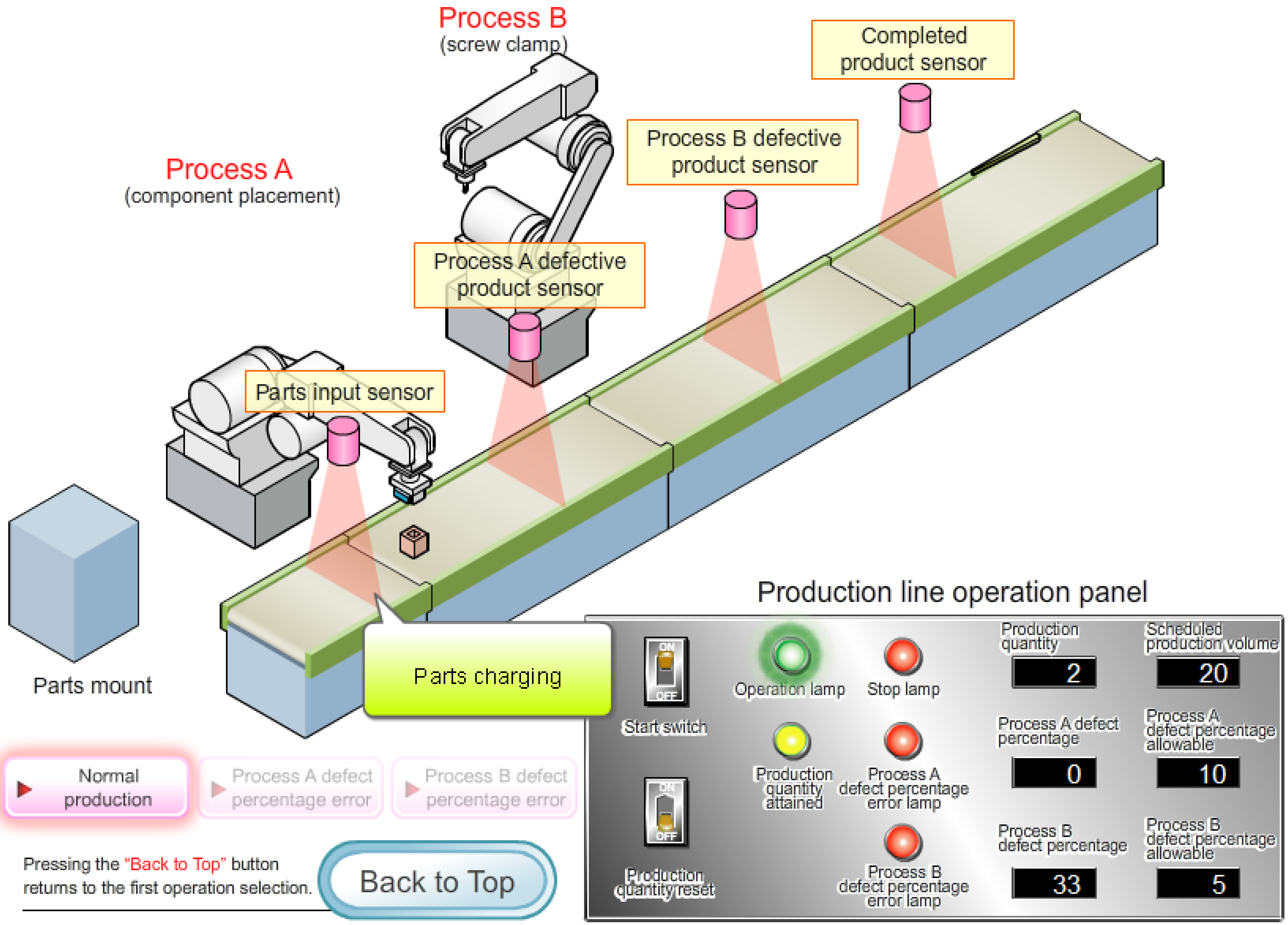
Learning steps in Chapter 2

In Chapter 2, you will learn about the functions used for programming.
GX Works2 provides many different functions to ensure efficient programming and improve the quality of programs.

PLC_GX_Works2_Advanced_ENG

Chapter 2 Introduction of System Used for Learning in this Course

This course employs the following hypothetical system to help you learn how to use the GX Works2 functions:


Productivity
Quality
Process B
(screw clamp)
Completed product sensor
Process B defective product sensor
Process A
(component placement)
Process A defective product sensor
Parts input sensor
Parts mount
Parts charging
Production line operation panel
Start switch
ON
OFF
Operation lamp
Stop lamp
Production quantity
2
Scheduled production volume
20
Production quantity attained
Process A defect percentage error lamp
Process A defect percentage
0
Process A defect percentage allowable
10
Production quantity reset
Process B defect percentage error lamp
Process B defect percentage
33
Process B defect percentage allowable
5
Normal production
Process A defect percentage error
Process B defect percentage error
Back to Top

Pressing the "Back to Top" button returns to the first operation selection.

2.1 Replacing the Device Name with a Name Associated with the Application

Productivity Quality

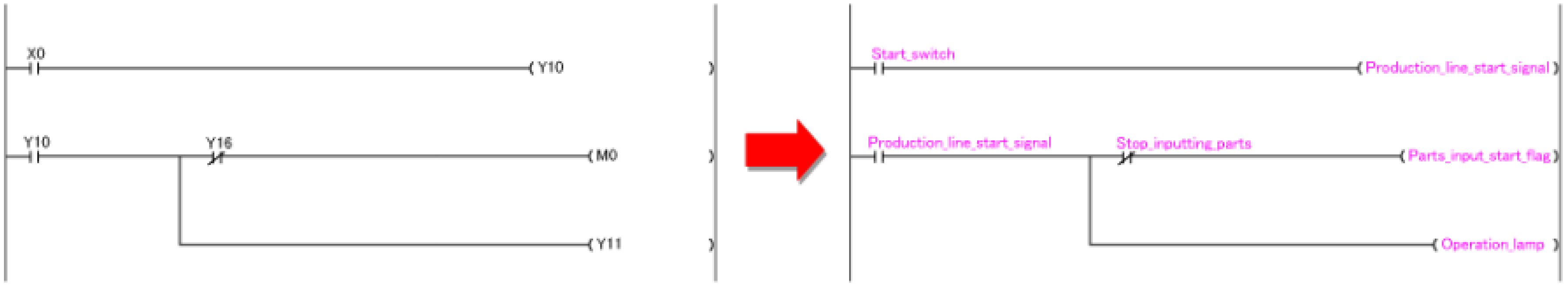
A **device** used in the sequence program is given a name that consists of a letter followed by a number, for example “M0” or “D5”.
Thus, the device name does not provide any clues as to the application of the device.
A large-scale program uses many different types of devices, which means that during programming, you must continually check the system design documents to determine the application of devices. This reduces the work efficiency and adversely affects the program quality due to errors in selecting devices.

“Labels” can be used to solve these problems.
Instead of using a device name, you can use a name (label) that indicates the actual application, for example **“Production line start signal.”**
For this type of name, Japanese (hiragana, katakana, and kanji) characters can be used in addition to alphanumeric characters.

<Statement for setting “M0” containing the production line start signal to ON by SET instruction>

Device	—[SET M0]—
Label	—[SET **Production line start signal**]—

Using this type of label is effective for creating an easy-to-read program, improving the efficiency of program development, and preventing device input errors.



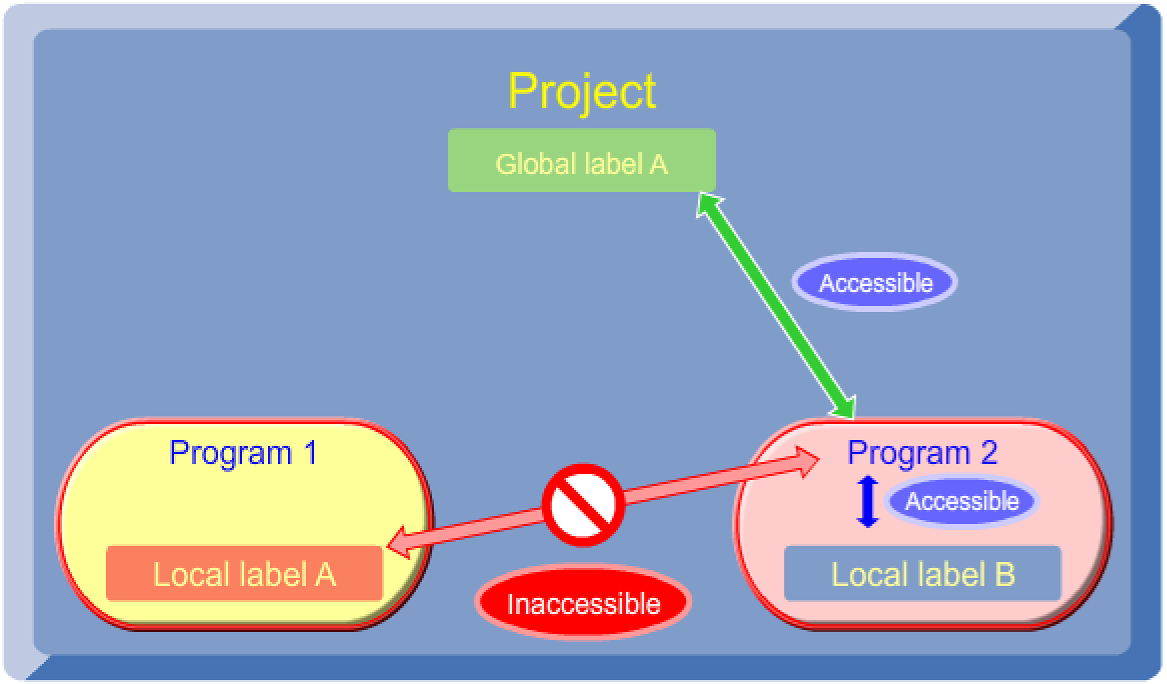
2.1.1 Label Types

Productivity Quality

There are two types of labels: **"global label"** and **"local label."**

<Global label>
Global labels are used for an entire project and can be accessed by any program in that project.

<Local label>
Local labels are used in a specific program and can only be accessed by the program in which the label is registered.



"Program 1" and "Program 2" are buttons.
Press either program to see if it can access the two types of labels.

2.1.2 Types of Label Applications and Stored Values

When registering a label, specify the label application and the type of value that can be stored by using **"Class"** and **"Data type."**

<Class>
The class indicates the use range and application of a label.
The classes that can be selected vary depending on the type of label.

Class	Intended use	Label setting area		
		Global label	Program local label	Function block local label
VAR_GLOBAL	Common label that can be used by programs and function blocks in a project	○	×	×
VAR_GLOBAL_ CONSTANT	Common label with a constant that can be used by programs and function blocks in a project	○	×	×
VAR	Label that can be used by programs and function blocks for which the label is intended	×	○	○
VAR_CONSTANT	Label with a constant that can be used by programs and function blocks for which the label is intended	×	○	○
VAR_RETAIN	Label (latch type) that can be used by programs and function blocks for which the label is intended	×	○	○
VAR_INPUT	Label used for the input of a function block for which the label is intended * The value cannot be changed in a program component.	×	×	○
VAR_OUTPUT	Label used for the output of a function block for which the label is intended	×	×	○
VAR_IN_OUT	Label used for the input and output of a function block for which the label is intended * The value can be changed in a program component.	×	×	○

2.1.2 Types of Label Applications and Stored Values

Productivity Quality

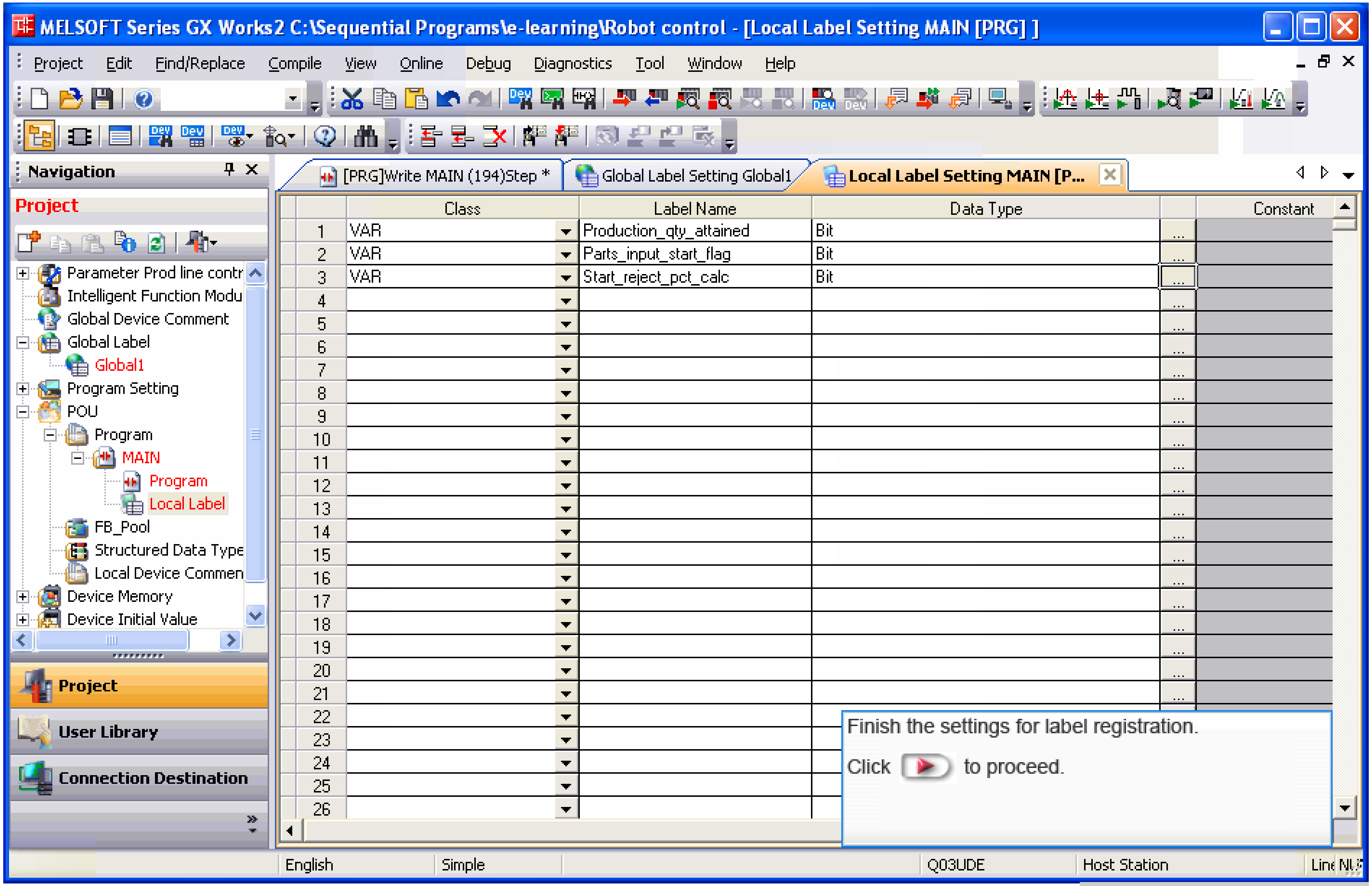
<Data type>

The data type refers to the type of value stored in the label.
The data type assigned to a label indicates the type and range of values that can be stored in the label and the corresponding device.
The data types that can be used with ladder programs are listed below.

Data type	Description	Bit length	Range of values stored
Bit	ON or OFF is stored. Corresponds to device "M"	1 bit	1: ON, 0: OFF
Word (signed)	An integer without fractions is stored. Corresponds to device "D"	16 bits	-32768 to 32767
Double-word (signed)		32 bits	-2147483648 to 2147483647
FLOAT (single precision)	A real number including fractions is stored. Corresponds to device "D"	16 bits	-2^{128} to -2^{-126}, 0, 2^{-126} to 2^{128}
FLOAT (double precision)		32 bits	-2^{1024} to -2^{-1022}, 0, 2^{-1022} to 2^{1024}
String	A character string is stored. Corresponds to device "D"	Variable	Up to 255 characters
Timer	Turns ON when the specified time is reached. Corresponds to timer device "T"	—	—
Retentive timer	Turns ON when the specified time is reached. Corresponds to retentive timer device "ST"	—	—
Counter	Turns ON when the specified count is reached. Corresponds to counter device "C"	—	—
Pointer	A subroutine start position is stored. Corresponds to pointer device "P"	—	—

PLC_GX_Works2_Advanced_ENG

2.1.3 Label registration


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [Local Label Setting MAIN [PRG]]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Global1
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Type
Local Device Commen
Device Memory
Device Initial Value
Project
User Library
Connection Destination
[PRG]Write MAIN (194)Step *
Global Label Setting Global1
Local Label Setting MAIN [P...
English
Simple
Q03UDE
Host Station

	Class	Label Name	Data Type	Constant
1	VAR	Production_qty_attained	Bit	
2	VAR	Parts_input_start_flag	Bit	
3	VAR	Start_reject_pct_calc	Bit	
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				

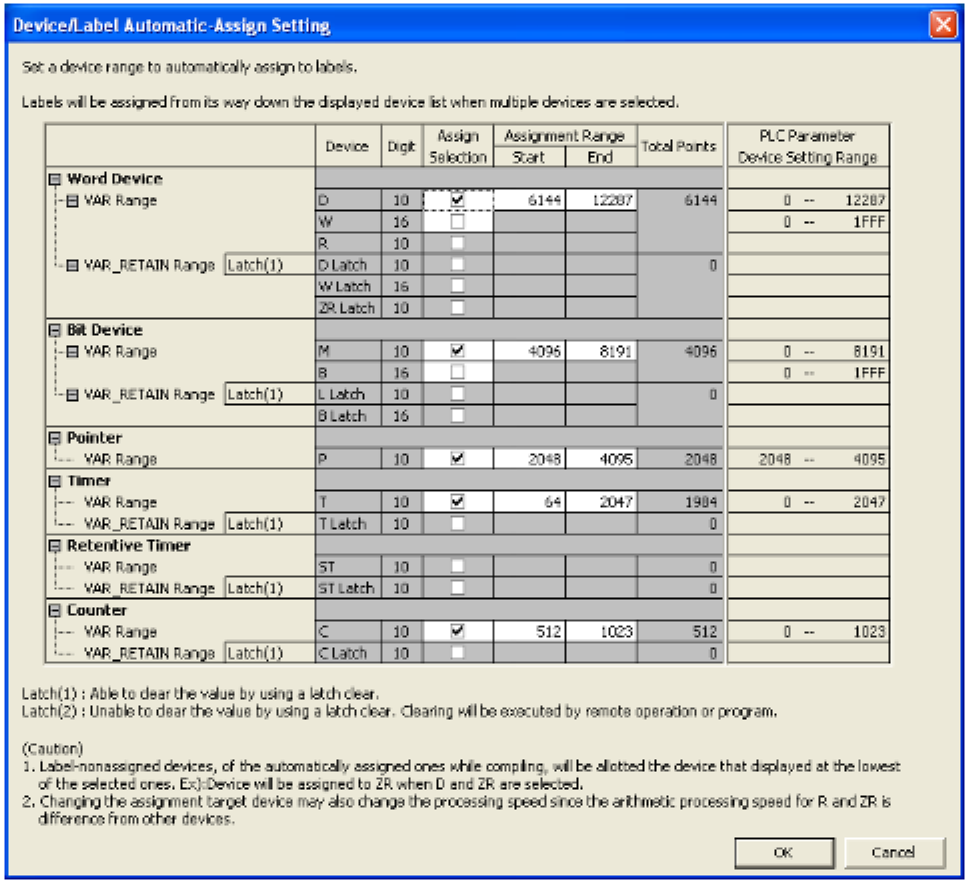
Finish the settings for label registration.

Click ▶ to proceed.

2.1.4 Automatic Assignment of Labels to Devices

Productivity Quality

Converting a program automatically assigns the appropriate device to the label according to the class and data type.
When using a label, it is not necessary to know which device is assigned.
Use **"Device/Label Automatic-Assign Setting"** to change the range of devices assigned to the label.
* This screen is the automatic assignment device setting window for the MELSEC-Q and MELSEC-L series. The screen may differ for the MELSEC-F series.



Device/Label Automatic-Assign Setting

Set a device range to automatically assign to labels.

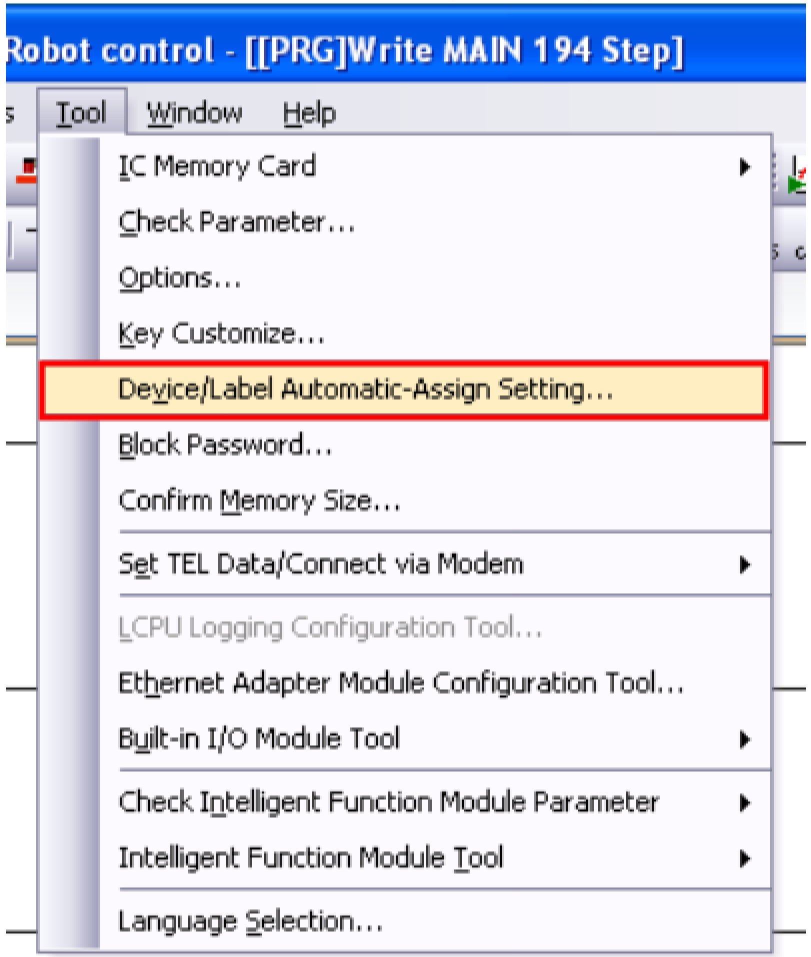
Labels will be assigned from its way down the displayed device list when multiple devices are selected.

		Device	Digit	Assign Selection	Assignment Range Start	Assignment Range End	Total Points	PLC Parameter Device Setting Range
Word Device								
VAR Range		D	10	✓	6144	12287	6144	0 -- 12287
		W	16	☐				0 -- 1FFF
		R	10	☐				
VAR_RETAIN Range	Latch(1)	D Latch	10	☐			0	
		W Latch	16	☐				
		ZR Latch	10	☐				
Bit Device								
VAR Range		M	10	✓	4096	8191	4096	0 -- 8191
		B	16	☐				0 -- 1FFF
VAR_RETAIN Range	Latch(1)	L Latch	10	☐			0	
		B Latch	16	☐				
Pointer								
VAR Range		P	10	✓	2048	4095	2048	2048 -- 4095
Timer								
VAR Range		T	10	✓	64	2047	1984	0 -- 2047
VAR_RETAIN Range	Latch(1)	T Latch	10	☐			0	
Retentive Timer								
VAR Range		ST	10	☐			0	
VAR_RETAIN Range	Latch(1)	ST Latch	10	☐			0	
Counter								
VAR Range		C	10	✓	512	1023	512	0 -- 1023
VAR_RETAIN Range	Latch(1)	C Latch	10	☐			0	

Latch(1) : Able to clear the value by using a latch clear.
Latch(2) : Unable to clear the value by using a latch clear. Clearing will be executed by remote operation or program.

(Caution)
1. Label-nonassigned devices, of the automatically assigned ones while compiling, will be allotted the device that displayed at the lowest of the selected ones. Ex):Device will be assigned to ZR when D and ZR are selected.
2. Changing the assignment target device may also change the processing speed since the arithmetic processing speed for R and ZR is difference from other devices.

<Starting "Device/Label Automatic-Assign Setting">
From the GX Works2 menu, select **[Tool]** – **[Device/Label Automatic-Assign Setting]**.



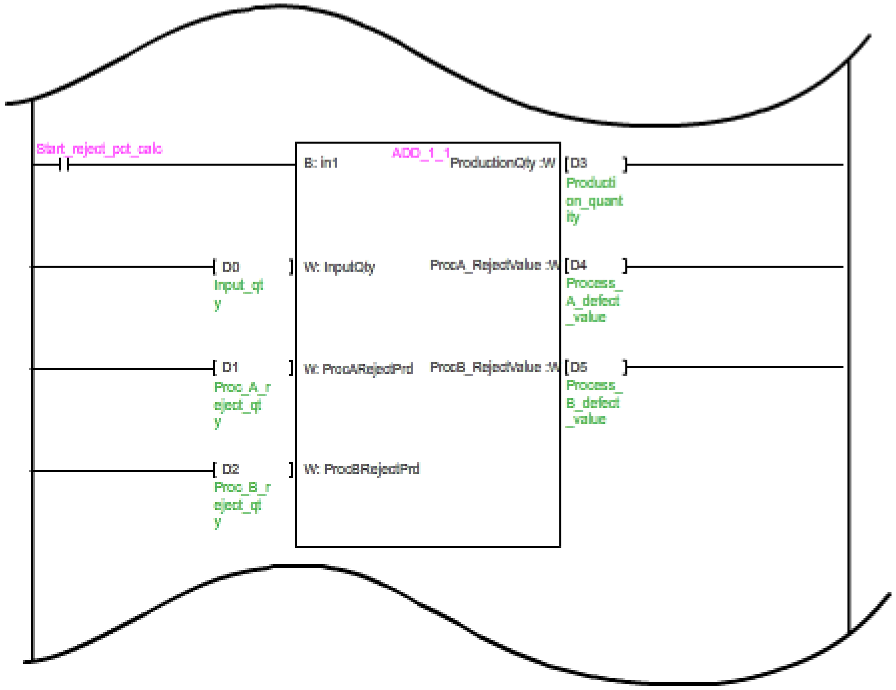
Robot control - [[PRG]Write MAIN 194 Step]

Tool
Window
Help
IC Memory Card
Check Parameter...
Options...
Key Customize...
Device/Label Automatic-Assign Setting...
Block Password...
Confirm Memory Size...
Set TEL Data/Connect via Modem
LCPU Logging Configuration Tool...
Ethernet Adapter Module Configuration Tool...
Built-in I/O Module Tool
Check Intelligent Function Module Parameter
Intelligent Function Module Tool
Language Selection...

2.2 Arranging Repeatedly Used Ladder Blocks as Function Blocks for Diversion

In a large-scale program, some ladder blocks may be used repeatedly.
Or, some of the same ladder blocks may be used in different programs.
Work efficiency cannot be improved if you must input the same ladder block every time it is needed.
In addition, if a defect is found in a common ladder block, each block must be located and corrected. This causes a significant loss of time.

Use **"Function Block (FB)"** to solve these problems.
A ladder block that is used repeatedly can be **arranged as a function block**, which can be used in every program. This greatly improves the work efficiency.
If a defect is found in a common block, all you need to do is correct the function block.
This not only saves time, but also prevents the risk of not correcting one of the blocks.

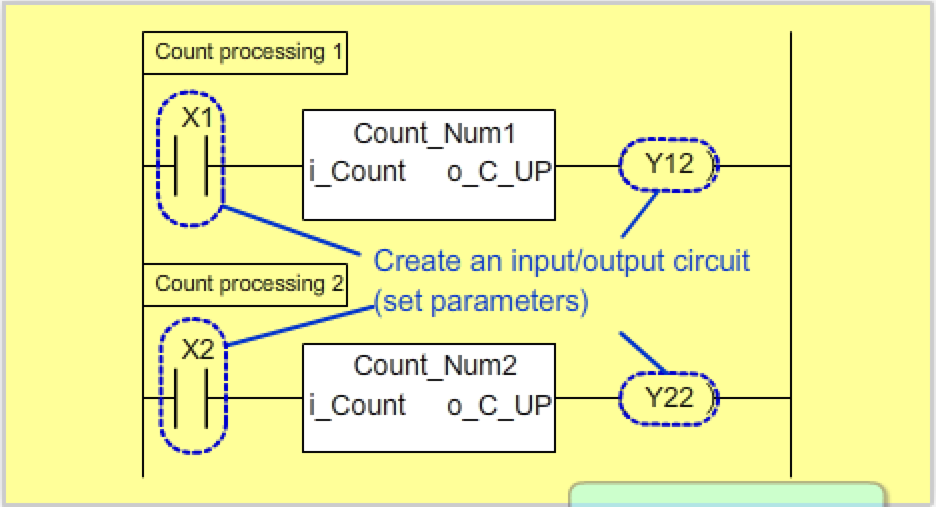


Press the button to watch the animation from the beginning.

2.2.1 Creating and Placing Function Blocks

Since the function block will be used in different programs, real devices (such as X, Y, and D) cannot be used in the function block. These devices must be replaced by **labels** as you learned in Section 2.1.
The procedure for arranging ladder blocks into a function block is shown by animation.

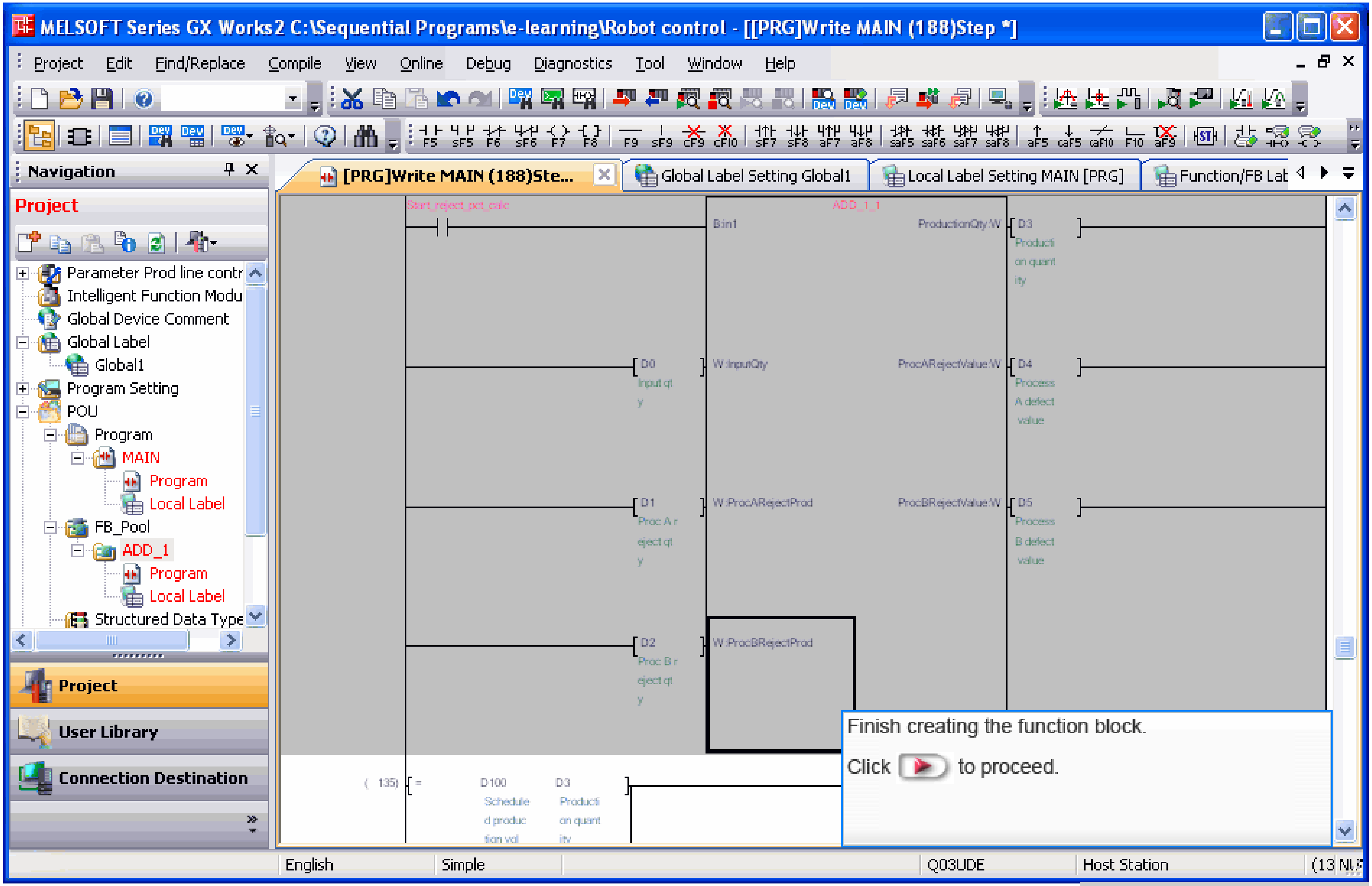
1. Prepare a program to be arranged into a function block.
2. Divide the program into input and output, and replace the internal devices by internal labels to make a function block.
3. Place the created function block in the program using drag and drop.
4. Assign parameters to the input and output labels of the function block that you placed in the program.
5. Create an input circuit that passes the parameter to the input label and an output circuit that receives the parameter from the output label, before and after the function block.



Press the button to return to the first flow.

2.2.1 Creating and Placing Function Blocks

Productivity Quality



Finish creating the function block.

Click ▶ to proceed.

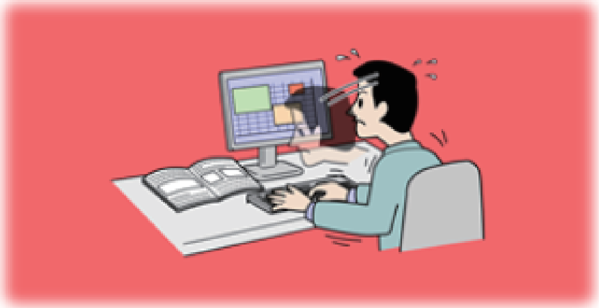
2.2.2 Using the Function Block Library

Productivity Quality

You have now learned how to create a function block.
Mitsubishi Electric has arranged the control programs of various modules such as CPUs, analog input/output modules, networks, and positioning modules as function blocks, which are available free of charge as the **"FB Library."**
Using the FB library eliminates the need to develop module control programs, which was traditionally the users' responsibility.
Even modules that you are not familiar with can be easily introduced.

The FB library can be downloaded from the **"MELSOFT Library Download"** page on the **MITSUBISHI ELECTRIC FA Website**.

<Before>



<If you use the FB library>

* There is no FB library provided for the MELSEC-F series.
The supported models are listed on the download page. Please check the supported model before downloading.

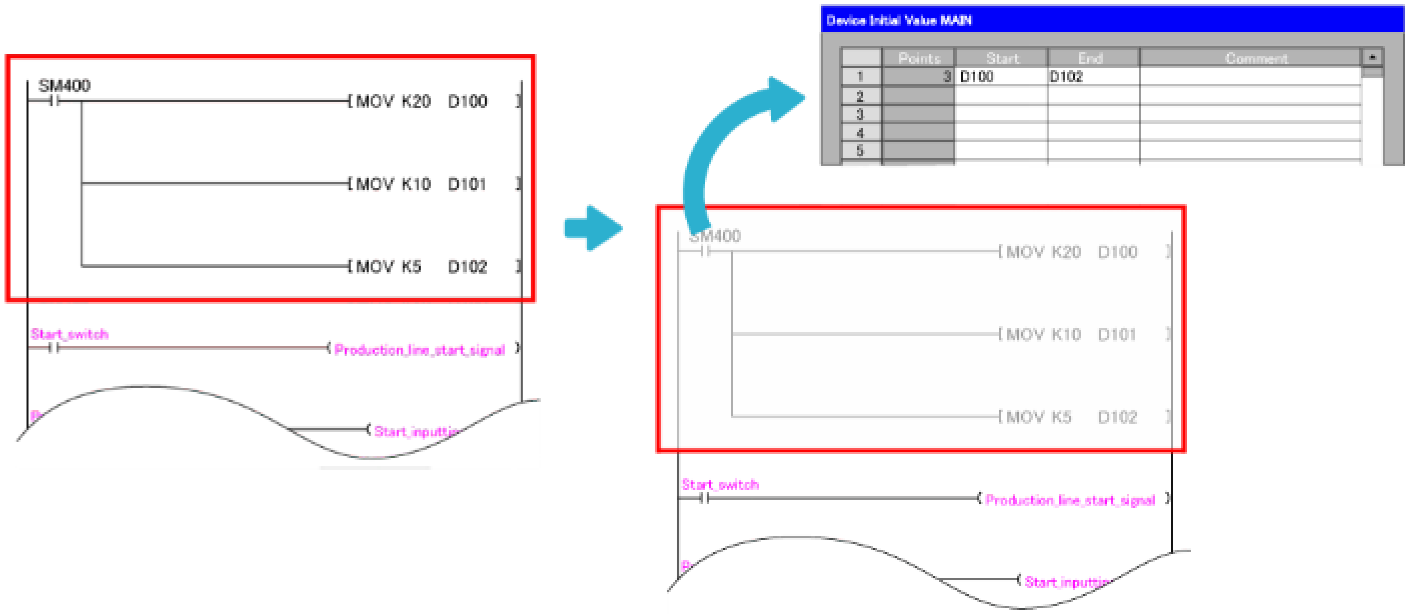
2.3 Changing Device Initial Values without Correcting the Program

Productivity Quality

The initial value or constant of a device is normally set using the MOV instruction before the main program processing.
In this case, the program must be directly corrected each time the program operation is changed according to the system application.
Not only is this method time-consuming, but there is also the risk of correction errors or failure to make a correction.

Use **"Device Initial Value"** to solve these problems.
Using the GX Works2 function ensures proper management of device initial values and eliminates the need to make program corrections, thus allowing you to create programs much more efficiently.
In addition, this function eliminates the need for an initial device value setting program, which would reduce the program volume (memory usage) and decrease the risk of failure.

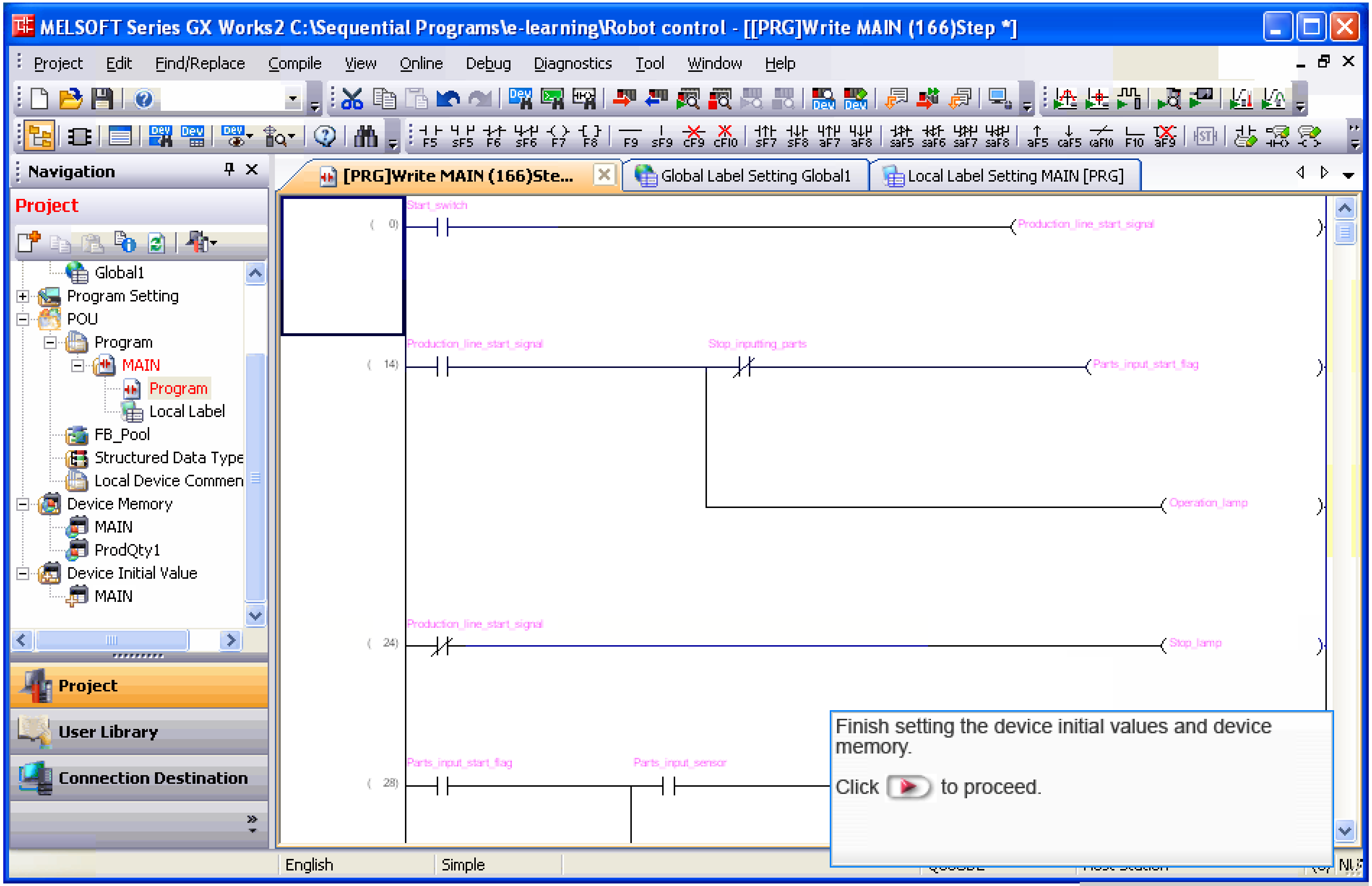
For the device initial value, specify the range of devices for setting the initial values.
Actual initial values are stored in the **device memory** and are assigned to the device range specified.
With device memory areas prepared according to the system applications, the device initial values can be changed by simply changing the device memory area to be assigned.



Device Initial Value MAIN

	Points	Start	End	Comment
1	3	D100	D102	
2				
3				
4				
5				

2.3 Changing Device Initial Values without Correcting the Program



Finish setting the device initial values and device memory.

Click ▶ to proceed.

PLC_GX_Works2_Advanced_ENG

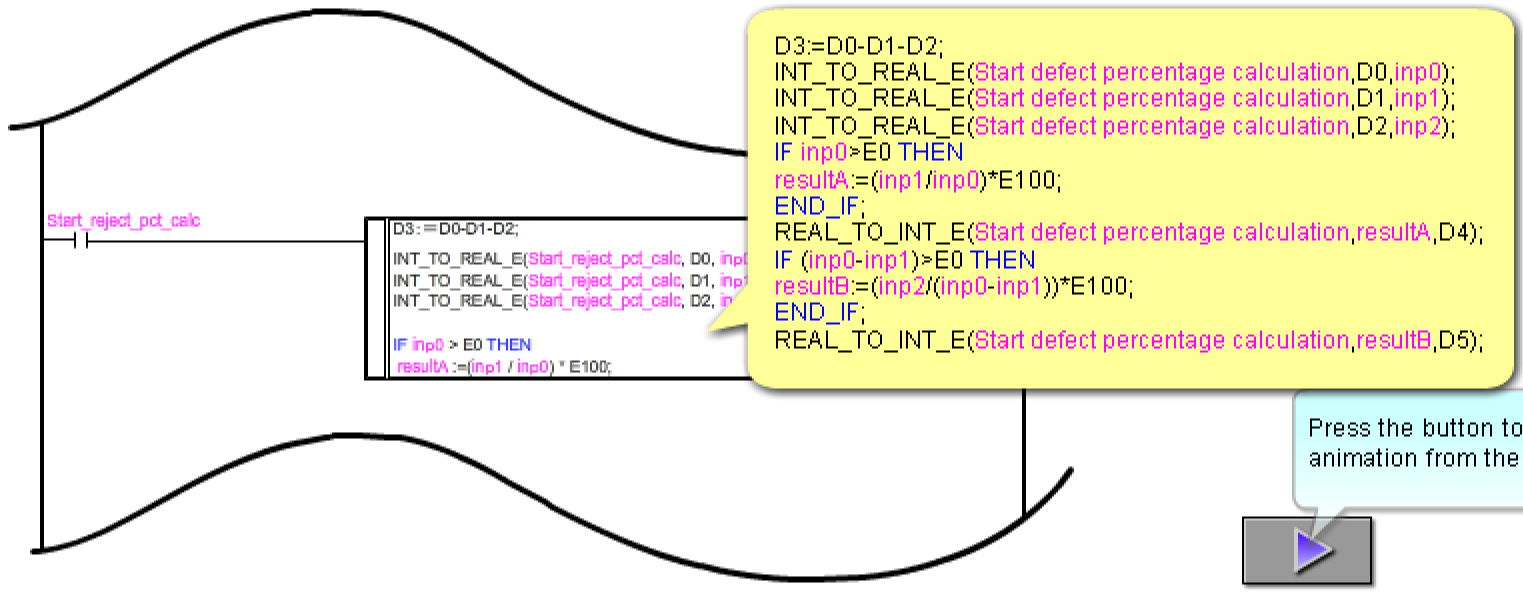
2.4 Simplifying Ladder Programs

Productivity Quality

In a large-scale ladder program, the connection of devices, instructions, and ruled lines can become quite complicated, making it difficult to grasp what processing the program performs.
In particular, numerical calculations, such as a formula completed on a single line, must be programmed using a combination of instructions.

Use **"Inline Structured Text"** to solve these problems.
A ladder program is replaced by a program that is partially written in **structure text (ST) language**.
The ST language is a sequence control programming language similar to C language used for computer software programming. Numerical calculations can be written using formulas, so even programmers who are not familiar with C can use inline structured text.

The following figure shows an example in which part of the ladder program for the system is replaced with inline structured text.
You can see that the complicated latter program is now easy to understand.



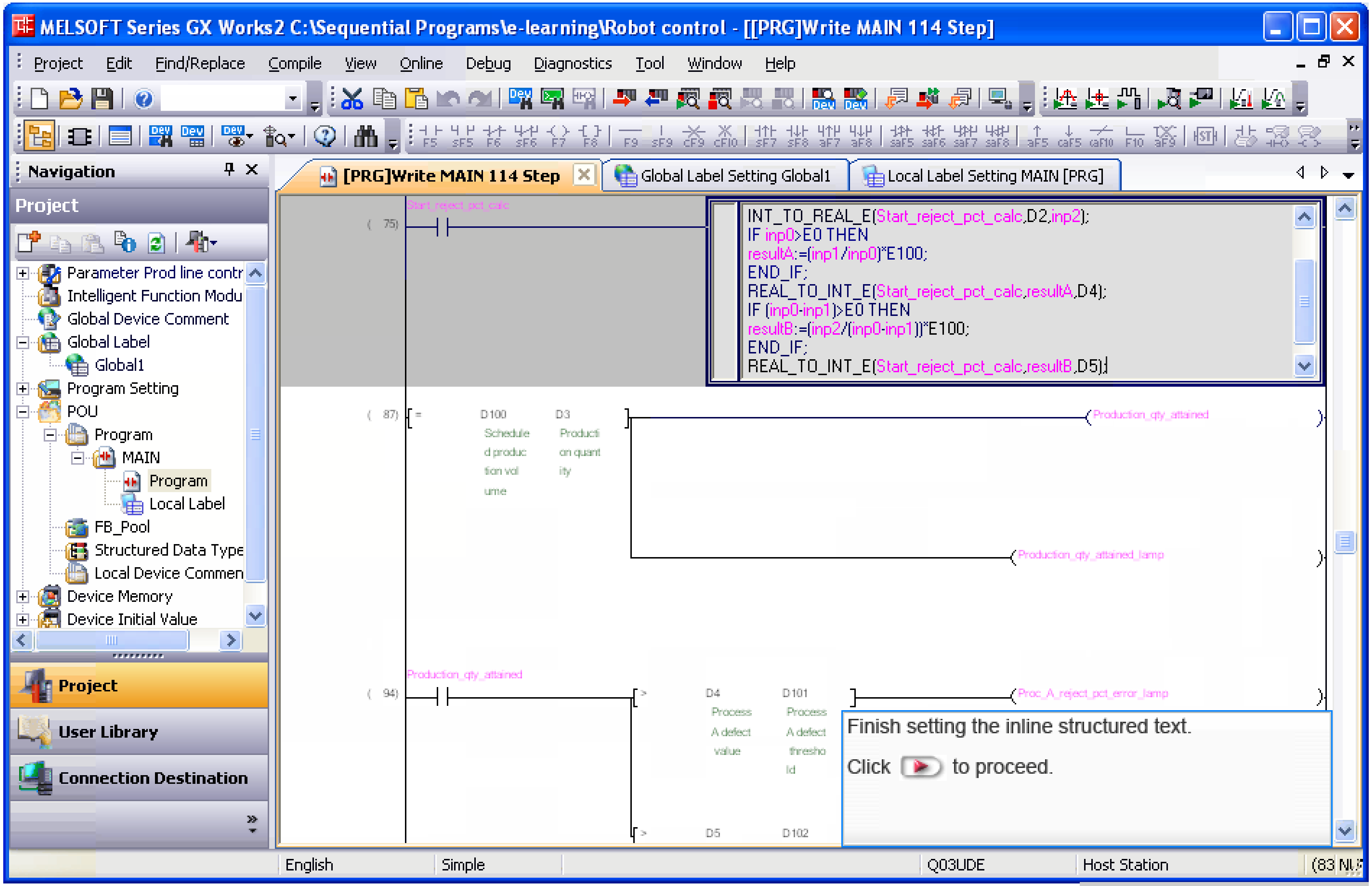
```
D3:=D0-D1-D2;

INT_TO_REAL_E(Start_reject_pct_calc, D0, inp
INT_TO_REAL_E(Start_reject_pct_calc, D1, inp
INT_TO_REAL_E(Start_reject_pct_calc, D2, in

IF inp0 > E0 THEN
 resultA :=(inp1 / inp0) * E100;
```

```
D3:=D0-D1-D2;
INT_TO_REAL_E(Start defect percentage calculation,D0,inp0);
INT_TO_REAL_E(Start defect percentage calculation,D1,inp1);
INT_TO_REAL_E(Start defect percentage calculation,D2,inp2);
IF inp0>E0 THEN
resultA:=(inp1/inp0)*E100;
END_IF;
REAL_TO_INT_E(Start defect percentage calculation,resultA,D4);
IF (inp0-inp1)>E0 THEN
resultB:=(inp2/(inp0-inp1))*E100;
END_IF;
REAL_TO_INT_E(Start defect percentage calculation,resultB,D5);
```

Press the button to watch the animation from the beginning.

PLC_GX_Works2_Advanced_ENG

2.4 Simplifying Ladder Programs


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Write MAIN 114 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
[PRG]Write MAIN 114 Step
Global Label Setting Global1
Local Label Setting MAIN [PRG]
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Global1
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Type
Local Device Commen
Device Memory
Device Initial Value
Project
User Library
Connection Destination
English
Simple
Q03UDE
Host Station

```
( 75)  Start_reject_pct_calc
INT_TO_REAL_E(Start_reject_pct_calc,D2,inp2);
IF inp0>E0 THEN
resultA:=(inp1/inp0)*E100;
END_IF;
REAL_TO_INT_E(Start_reject_pct_calc,resultA,D4);
IF (inp0-inp1)>E0 THEN
resultB:=(inp2/(inp0-inp1))*E100;
END_IF;
REAL_TO_INT_E(Start_reject_pct_calc,resultB,D5);

( 87)  [=  D100 Scheduled production volume   D3 Production quantity ]
       ( Production_qty_attained )
       ( Production_qty_attained_lamp )

( 94)  Production_qty_attained
       [>  D4 Process A defect value   D101 Process A defect threshold ]
       ( Proc_A_reject_pct_error_lamp )
       [>  D5   D102
```

Finish setting the inline structured text.

Click ▶ to proceed.

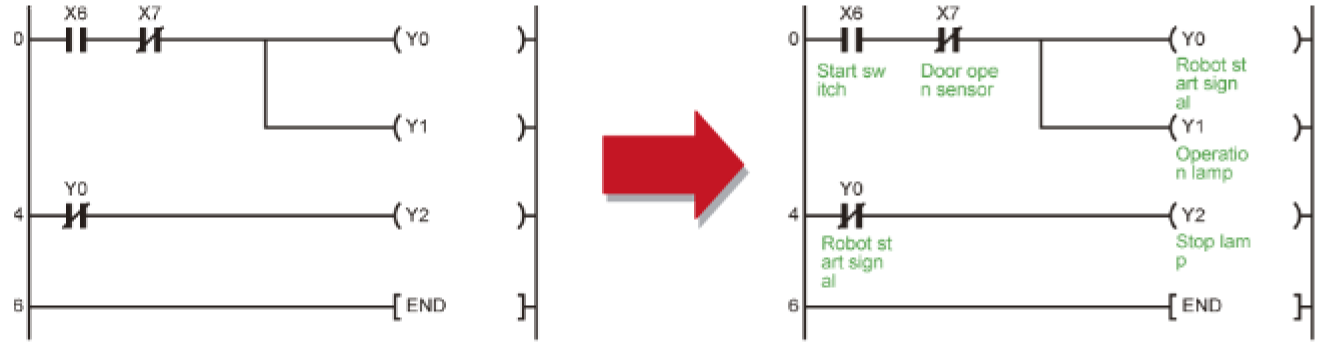
2.5 Creating Programs that are Easy to Understand and Read

Productivity Quality

You may find it difficult to understand the details of control in a large-scale program by just looking at the program. The following problems can occur as a result:

- You make program errors (such as input of incorrect instructions or devices).
- You are unable to find the causes of program errors.
- Someone taking over the programming cannot understand the details of control.

Use **"Comment"** to solve these problems.
Memos such as control information and device names can be attached to the program so that the details of control can be easily understood.
These comments should be input wherever possible to create programs that are easy to understand not only for you, but also for others.

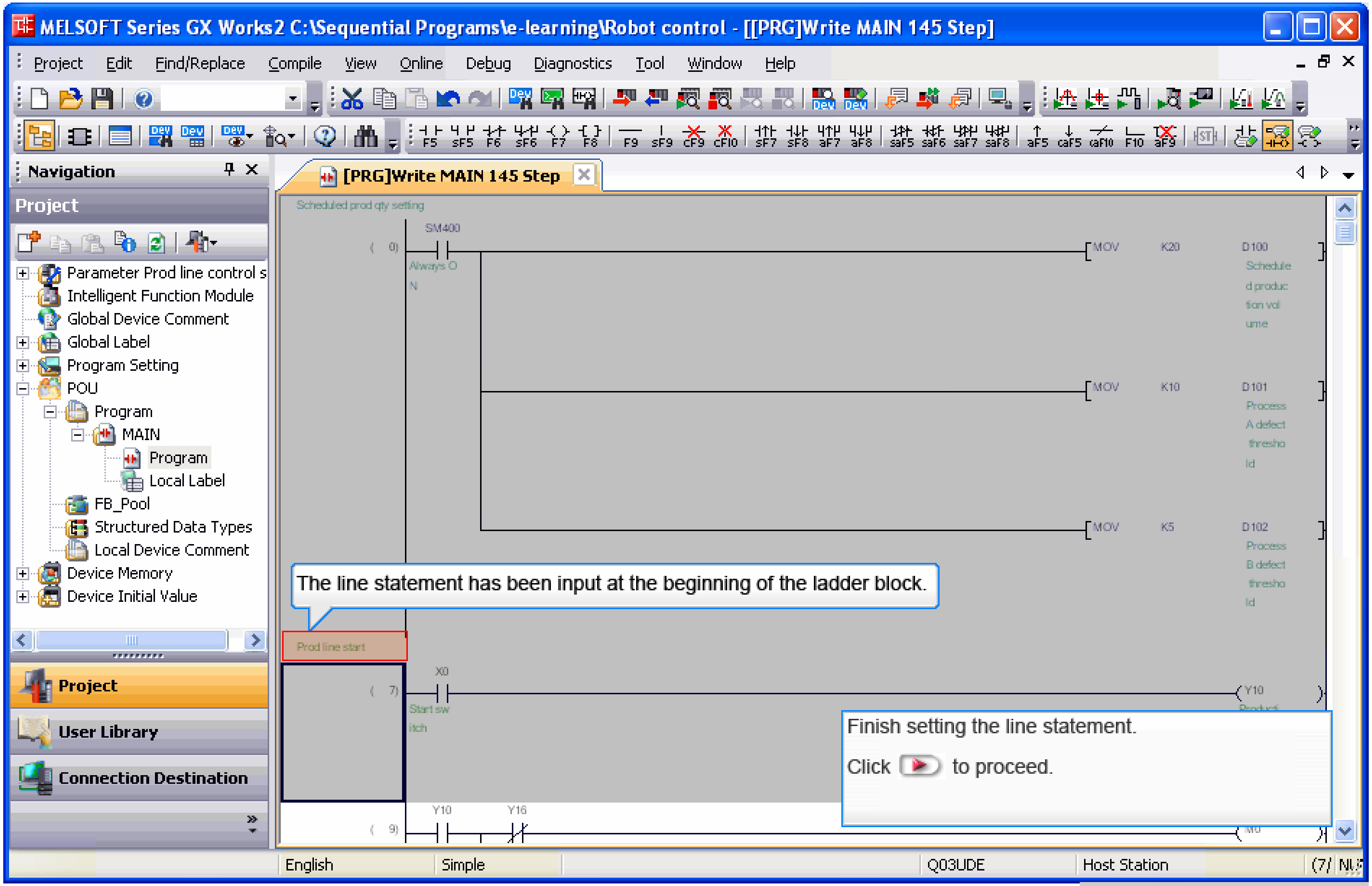


GX Works2 allows the following comments to be input.

Comment type	Comment range
Device comment	A comment can be attached to a device. This comment indicates the application of each device and the type of connected I/O device.
Statement	A comment can be attached to a ladder block. This comment makes the flow of processing easy to understand.
Note	A comment can be attached to a coil/application instruction in the program. This comment makes the contents of a coil (output) and an application instruction easy to understand.

PLC_GX_Works2_Advanced_ENG

2.5.1 Writing a comment for each ladder block



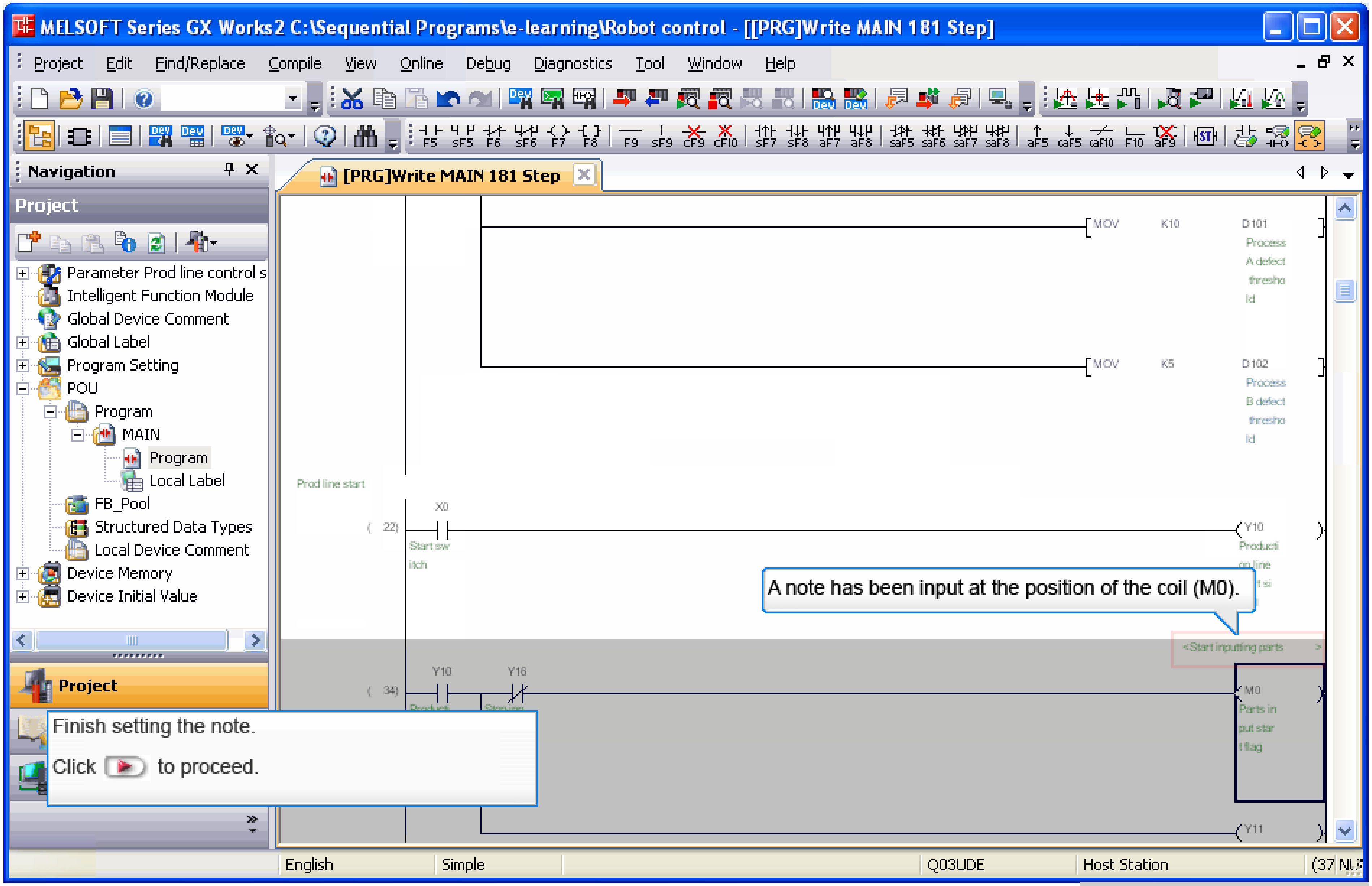
```
Scheduled prod qty setting
( 0)  SM400                                   [MOV  K20  D100 ]
      Always ON                                          Scheduled production volume
                                              [MOV  K10  D101 ]
                                                         Process A defect threshold
                                              [MOV  K5   D102 ]
                                                         Process B defect threshold
Prod line start
( 7)  X0                                      ( Y10 )
      Start switch
( 9)  Y10   Y16                               ( M0 )
```

The line statement has been input at the beginning of the ladder block.

Finish setting the line statement.

Click ▶ to proceed.

2.5.2 Writing a comment for each output (coil, application instruction)


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Write MAIN 181 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line control s
Intelligent Function Module
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Types
Local Device Comment
Device Memory
Device Initial Value
Project
[PRG]Write MAIN 181 Step
MOV K10 D101
Process A defect threshold
MOV K5 D102
Process B defect threshold
Prod line start
(22) X0 Start switch
Y10 Production line
(34) Y10 Y16
<Start inputting parts >
M0 Parts input start flag
Y11
English Simple Q03UDE Host Station

A note has been input at the position of the coil (M0).

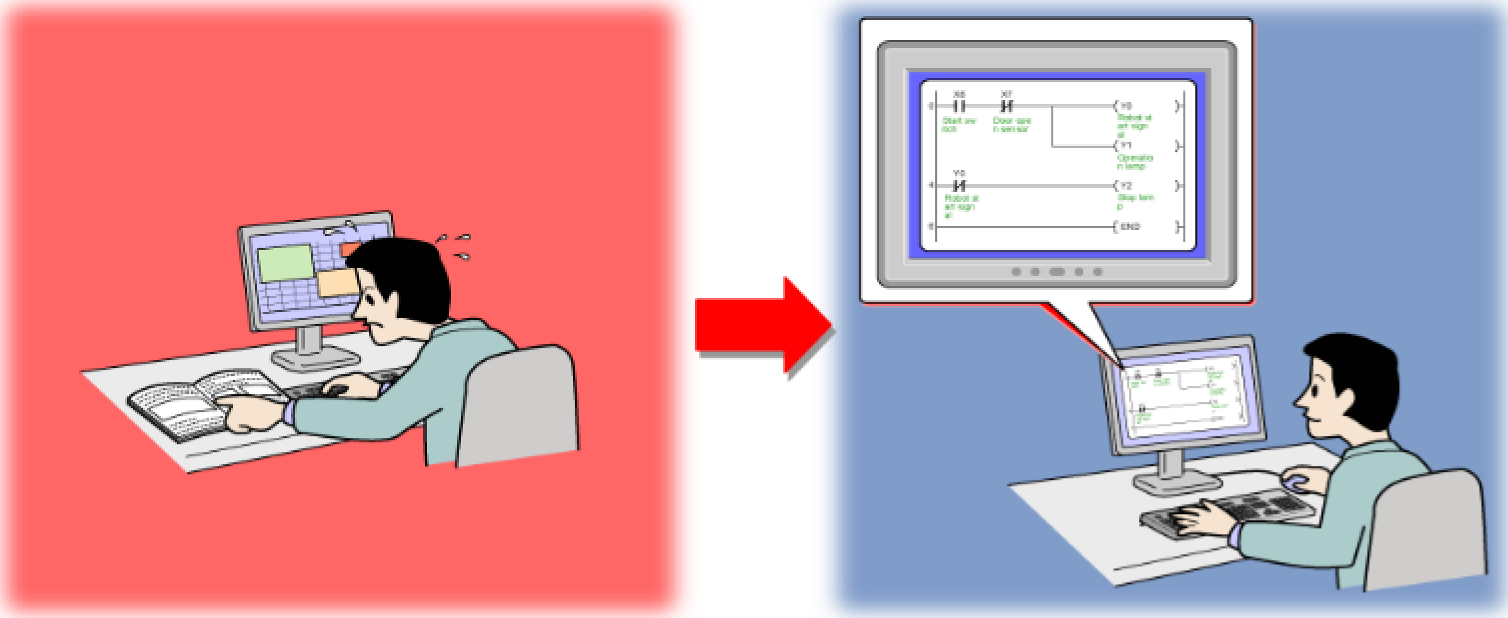
Finish setting the note.

Click ▶ to proceed.

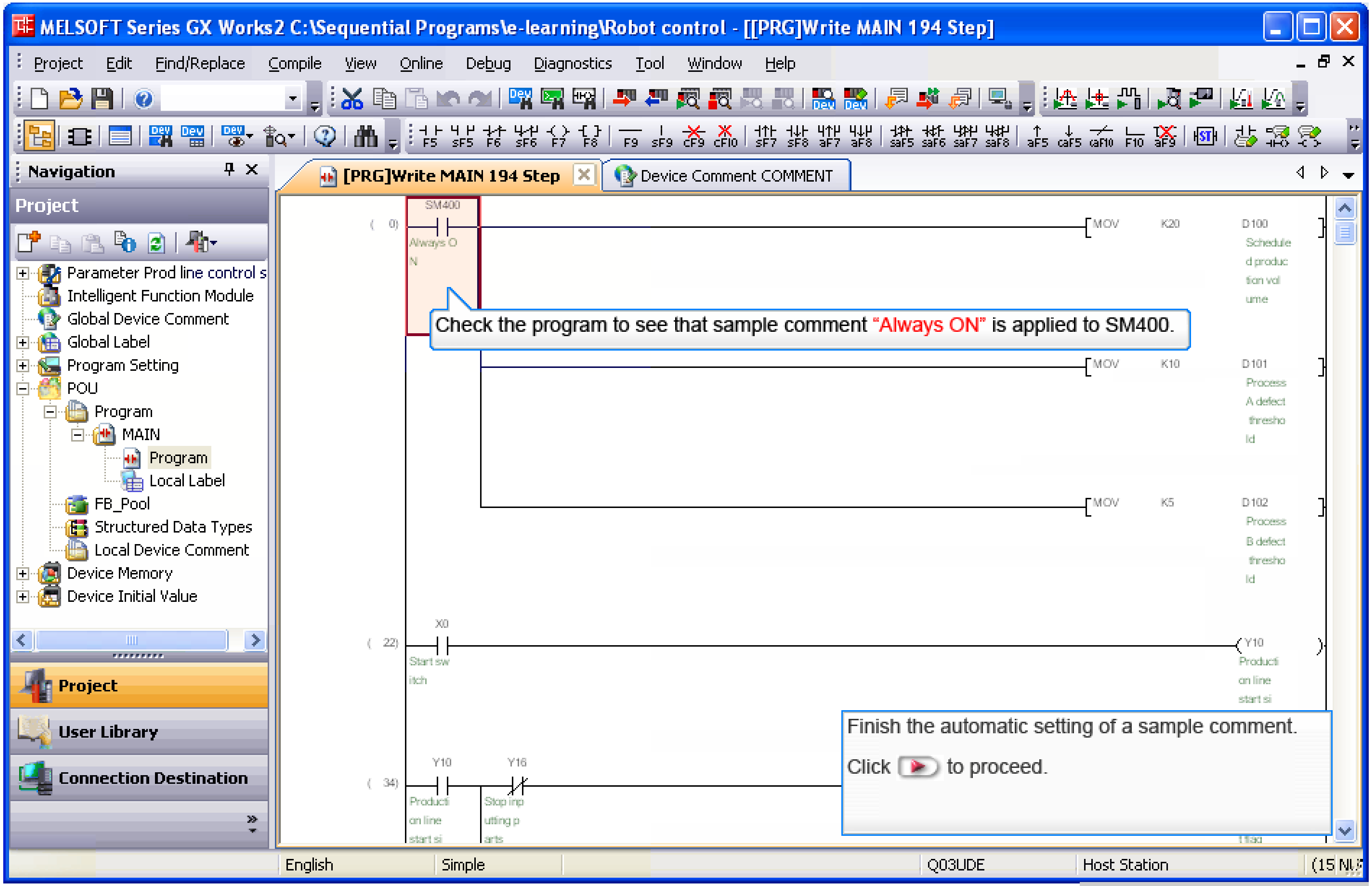
2.6 Making it Easy to Read Programs Containing Special Relays/Registers

If special relays, special registers, and/or intelligent function module devices are used in a program, it can be difficult to understand all of the applications and functions of these devices. You need to read the program with the manual in your hand. Although the program would be easier to read if a comment is attached to each device, considerable time and effort would be required for attaching comments if many devices are used.

Use **"Sample Comment"** to solve these problems.
GX Works2 provides sample comments describing the applications and functions of special relays, special registers, and intelligent function module devices.
Using these sample comments makes it much easier to attach comments to devices to make the program easy to read.
The sample comments can be modified as necessary.



2.6 Making it Easy to Read Programs Containing Special Relays/Registers



```
( 0)  SM400 (Always ON)          [MOV K20 D100  Scheduled production volume]
                                 [MOV K10 D101  Process A defect threshold]
                                 [MOV K5  D102  Process B defect threshold]
( 22) X0 (Start switch)          (Y10 Production line start si...)
( 34) Y10 (Production line start si...)  Y16 (Stop inputting parts)
```

Check the program to see that sample comment "Always ON" is applied to SM400.

Finish the automatic setting of a sample comment.

Click ▶ to proceed.

Chapter 3 Debugging

Learning steps in Chapter 3

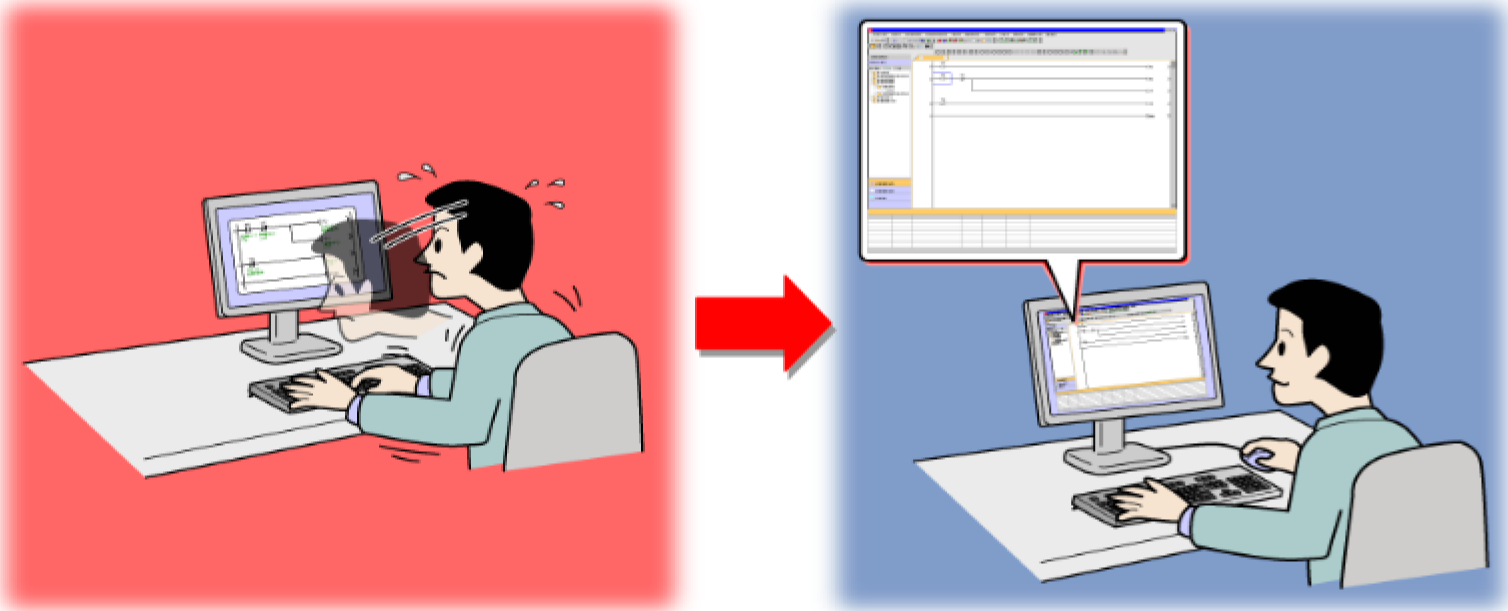
In Chapter 3, you will learn about the functions used for debugging.
GX Works2 provides many different monitoring and debugging tools to correct errors (bugs).
Create error-free, quality programs using these monitoring and debugging tools.

3.1 Monitoring Only Target Devices and Labels

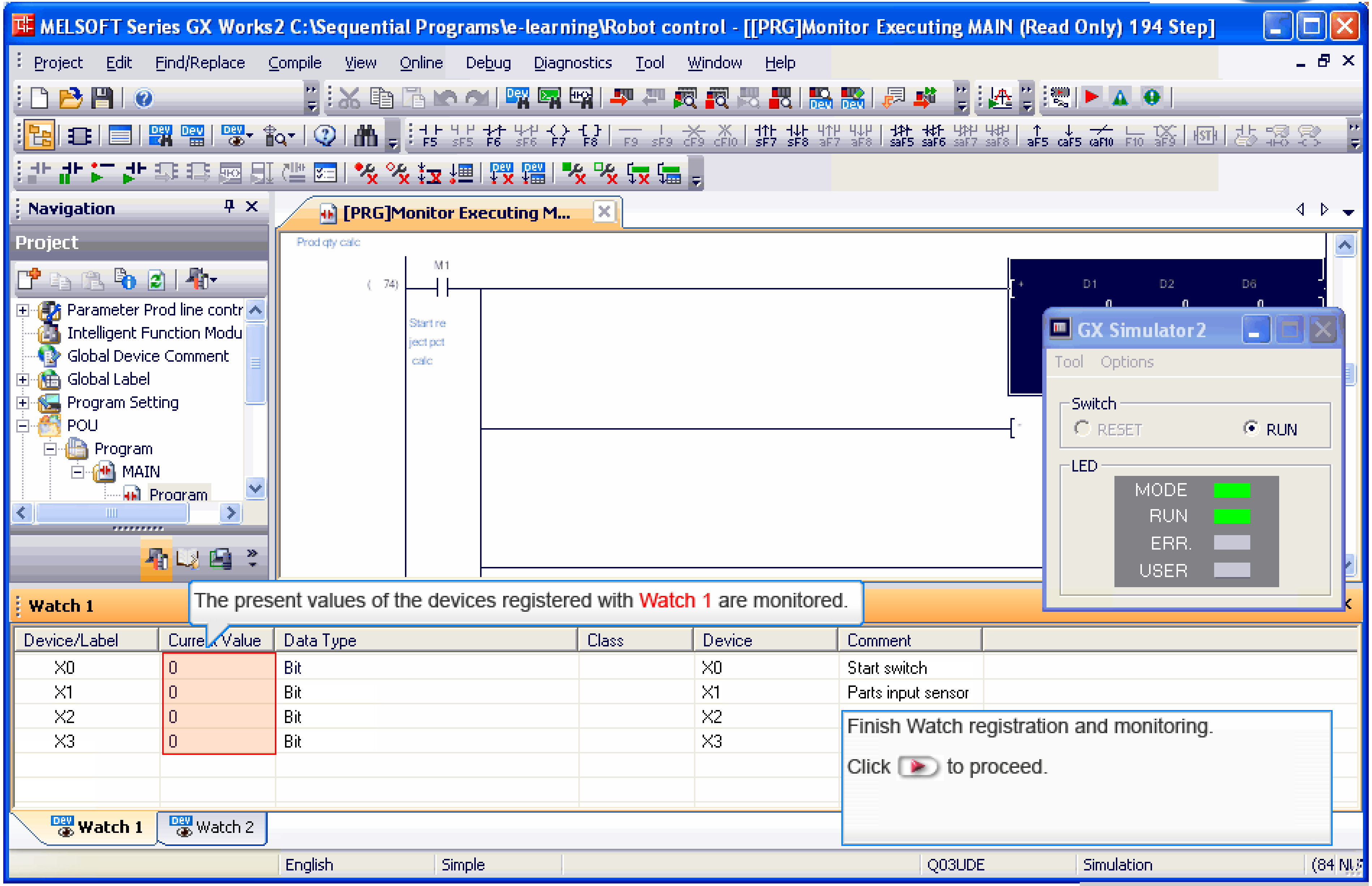
Quality

A program uses many instructions and devices.
In addition, the long vertical length of a program means that only part of it is displayed at a time on the PC monitor screen.
Therefore, the ladder monitor alone cannot monitor the entire operation.

Use **"Watch"** to solve these problems.
This function can be used to monitor only the devices and labels that you have specified in advance.
Two or more Watches can be created to ensure that each range is monitored.



3.1 Monitoring Only Target Devices and Labels


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Monitor Executing MAIN (Read Only) 194 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
[PRG]Monitor Executing M...
Prod qty calc
M1
Start re ject pct calc
GX Simulator2
Tool Options
Switch
RESET
RUN
LED
MODE
RUN
ERR.
USER
Watch 1
Watch 2
English
Simple
Q03UDE
Simulation

The present values of the devices registered with Watch 1 are monitored.

Device/Label	Current Value	Data Type	Class	Device	Comment
X0	0	Bit		X0	Start switch
X1	0	Bit		X1	Parts input sensor
X2	0	Bit		X2	
X3	0	Bit		X3	

Finish Watch registration and monitoring.

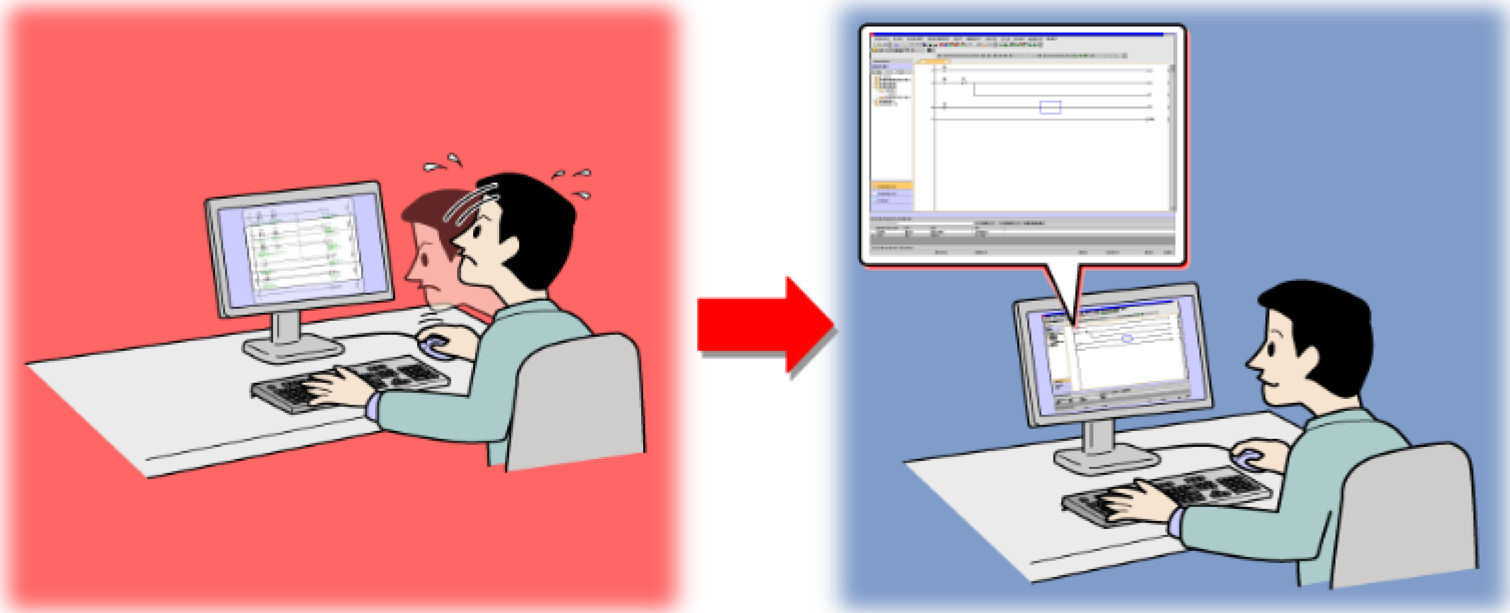
Click ▶ to proceed.

3.2 Checking Use Status of Devices and Labels

Quality

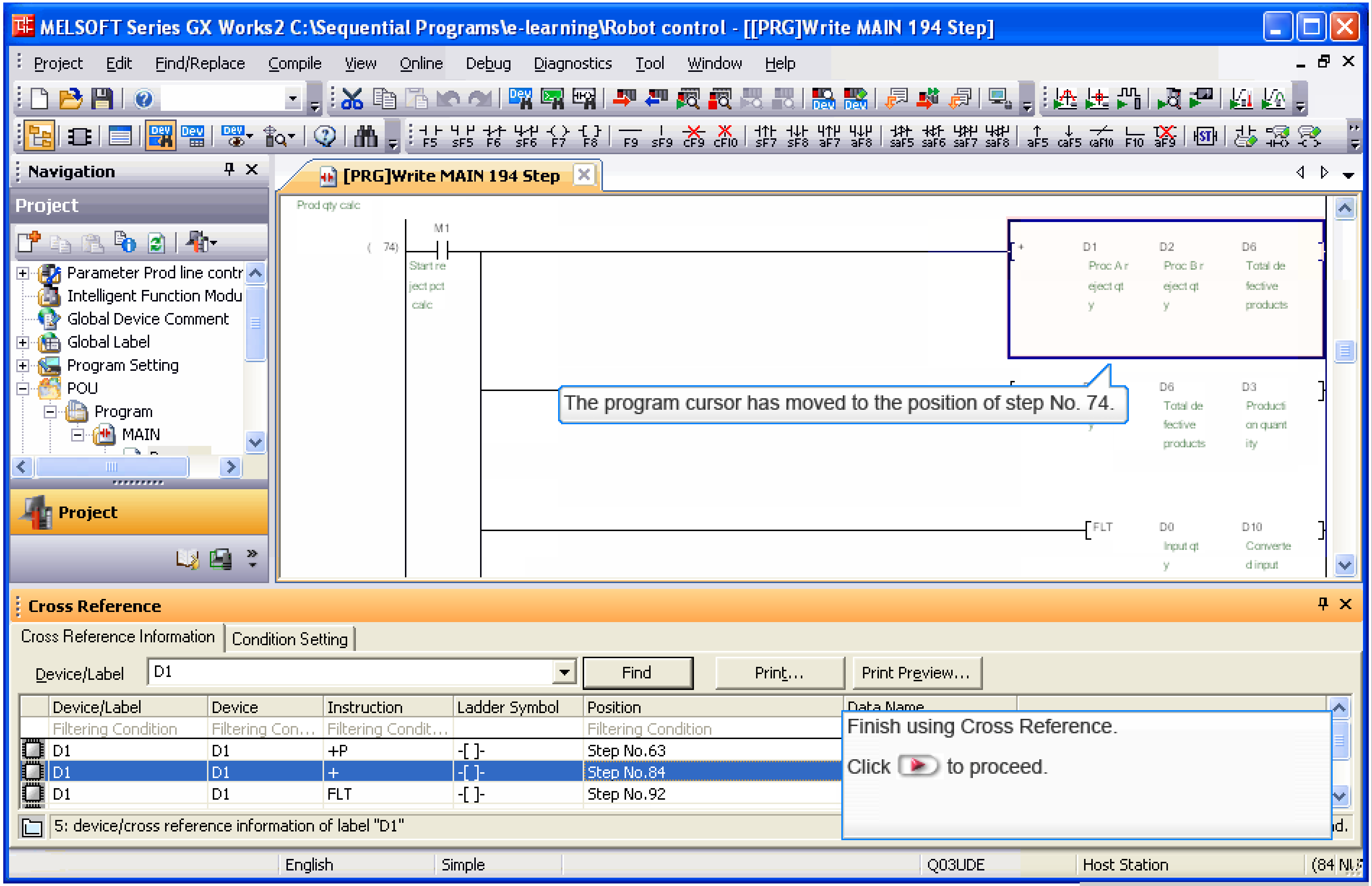
A program uses the same devices and labels at different locations.
You may want to check the use status of these devices and labels by comparing them between locations.

Use **"Cross reference"** to do this.
This function lists the locations of the devices and labels matching the search conditions so that you can compare them and check for incorrect usage.



PLC_GX_Works2_Advanced_ENG

3.2 Checking Use Status of Devices and Labels



The program cursor has moved to the position of step No. 74.

Device/Label: D1

Device/Label	Device	Instruction	Ladder Symbol	Position	Data Name
Filtering Condition	Filtering Con...	Filtering Condit...		Filtering Condition	
D1	D1	+P	-[ ]-	Step No.63	
D1	D1	+	-[ ]-	Step No.84	
D1	D1	FLT	-[ ]-	Step No.92	

5: device/cross reference information of label "D1"

Finish using Cross Reference.

Click ▶ to proceed.

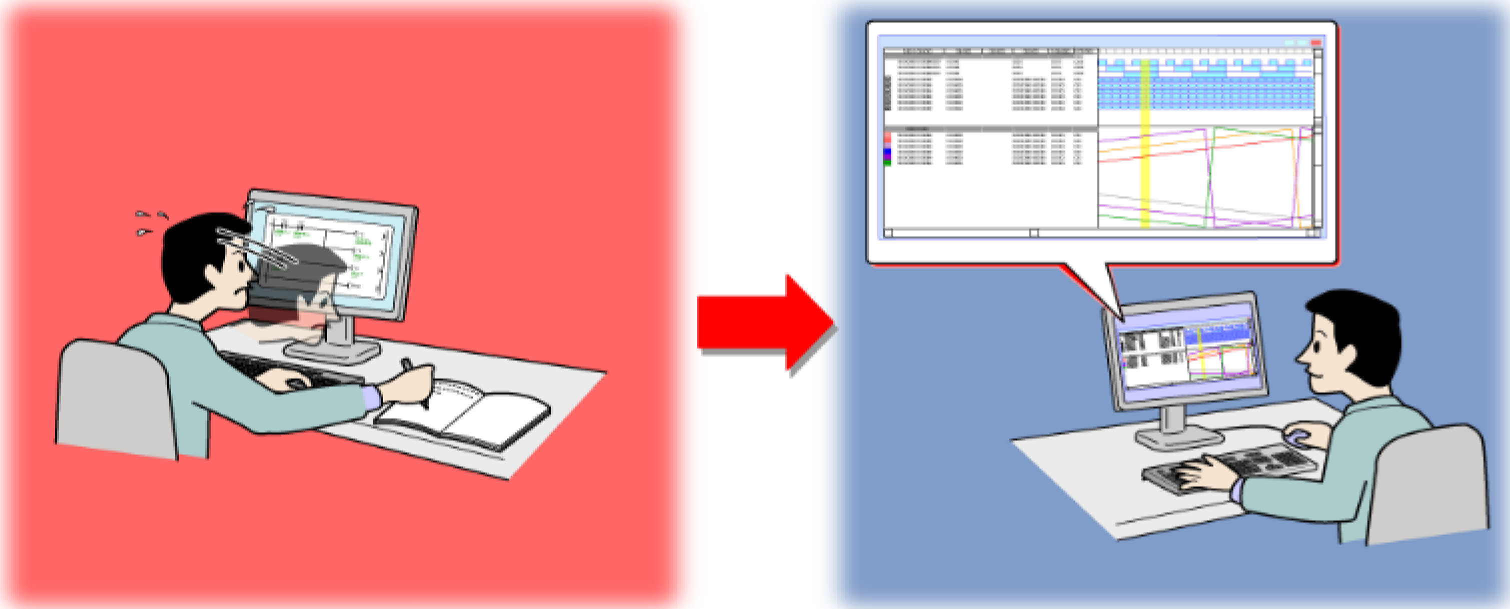
1-GX_Works2_Advanced_na00115_eng-A

3.3 Collecting Information on Device Value Change over Time

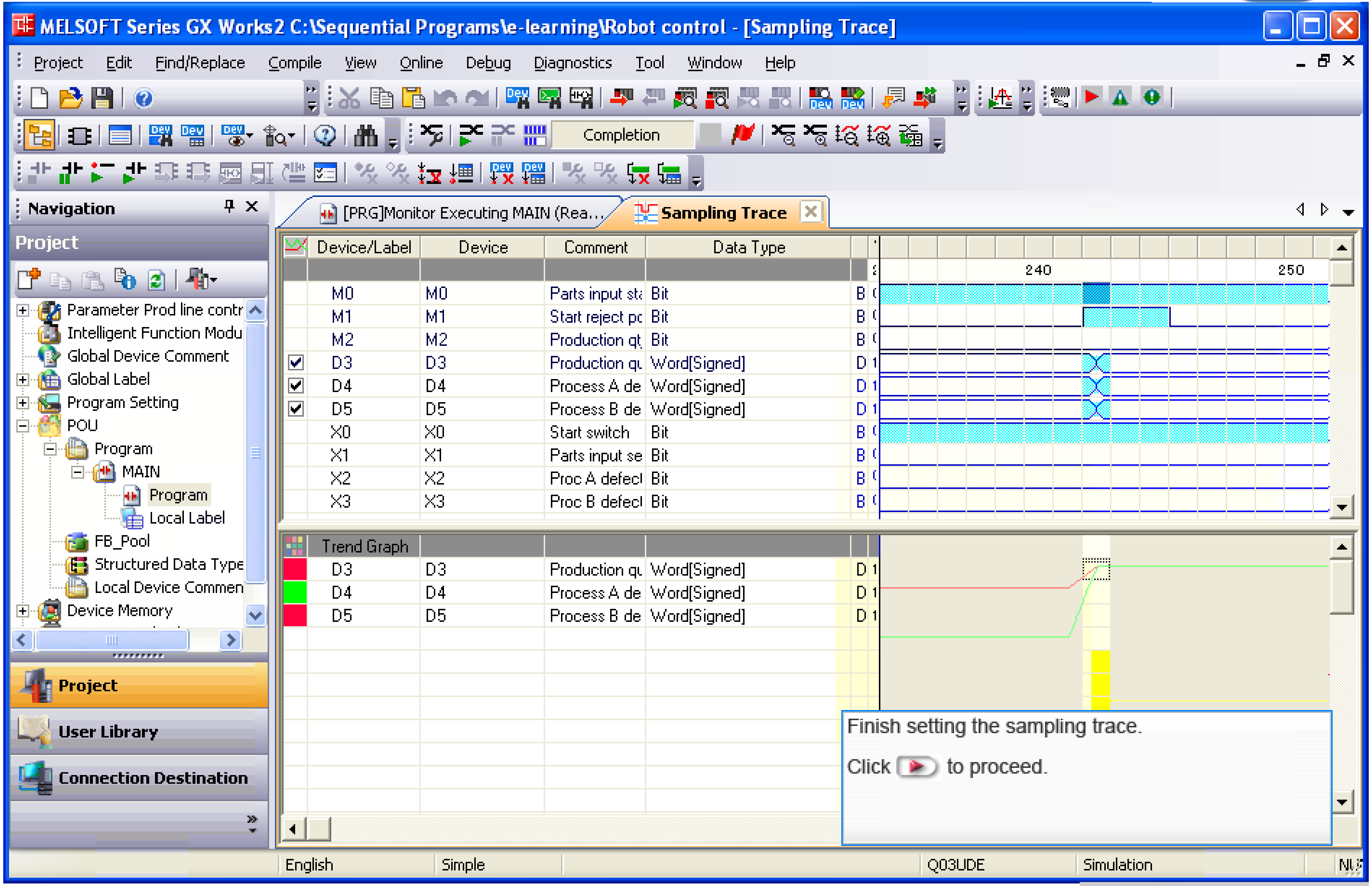
Quality

You may want to confirm that the changes in device and label values are within the design range or perform troubleshooting by checking the changes that have occurred in the event of a failure.

Use **"Sampling Trace"** in this case. (Only for MELSEC-Q and MELSEC-L series)
This function can be used to monitor and record the changes in device and label values over time. The record can be saved to files.



3.3 Collecting Information on Device Value Change over Time


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [Sampling Trace]

Device/Label	Device	Comment	Data Type
M0	M0	Parts input st	Bit
M1	M1	Start reject pc	Bit
M2	M2	Production qt	Bit
✓ D3	D3	Production qu	Word[Signed]
✓ D4	D4	Process A de	Word[Signed]
✓ D5	D5	Process B de	Word[Signed]
X0	X0	Start switch	Bit
X1	X1	Parts input se	Bit
X2	X2	Proc A defect	Bit
X3	X3	Proc B defect	Bit

Trend Graph			
🟥 D3	D3	Production qu	Word[Signed]
🟩 D4	D4	Process A de	Word[Signed]
🟥 D5	D5	Process B de	Word[Signed]

Finish setting the sampling trace.

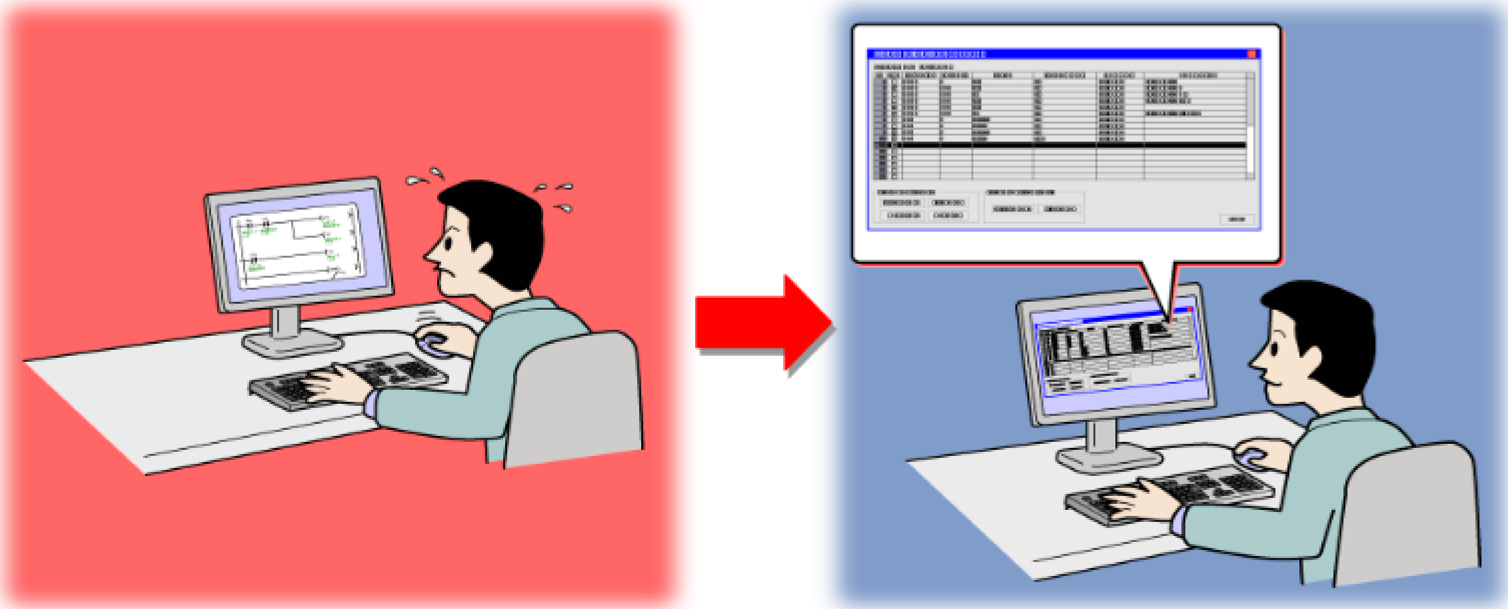
Click ▶ to proceed.

3.4 Changing Device Values without Correcting the Program

Quality

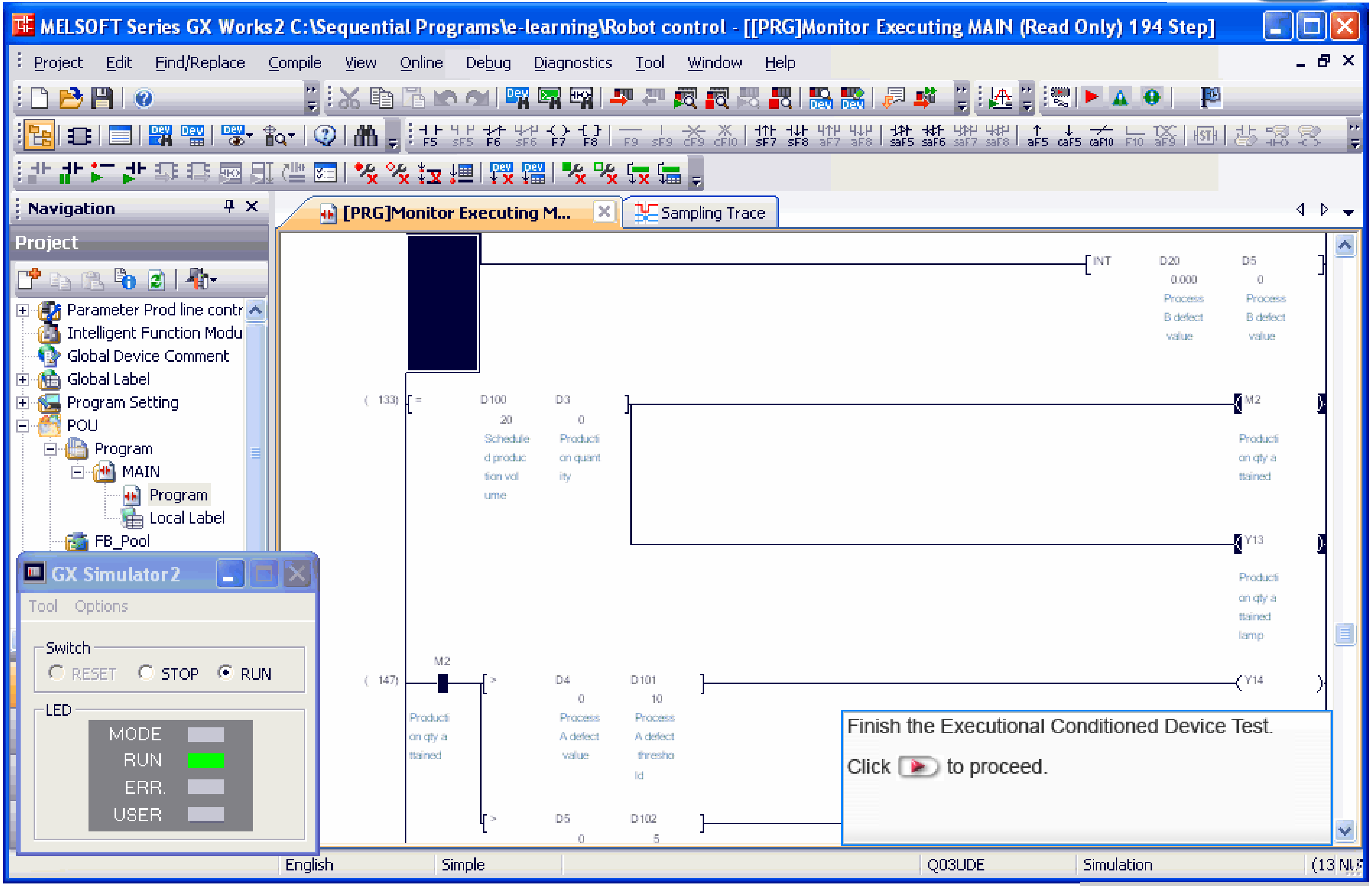
When performing debugging, you may want to forcibly change device values to check the difference in program operation.
However, this means that you must modify the program each time a device value is changed, which takes considerable time and effort.
In addition, if you forget to set the modified program back to the original settings, it may cause another failure.

Use **"Executional Conditioned Device Test"** in this case. (Only for MELSEC-Q and MELSEC-L series)
This function changes the device values upon execution of the step number specified in advance, without your having to modify the program.



PLC_GX_Works2_Advanced_ENG

3.4 Changing Device Values without Correcting the Program


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Monitor Executing MAIN (Read Only) 194 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
[PRG]Monitor Executing M...
Sampling Trace
GX Simulator2
Tool Options
Switch
RESET STOP RUN
LED
MODE
RUN
ERR.
USER
English
Simple
Q03UDE
Simulation

```
                                              [INT  D20      D5   ]
                                                    0.000    0
                                                    Process  Process
                                                    B defect B defect
                                                    value    value

(133)[=  D100      D3       ]-----------------------( M2 )
         20        0                                Producti
         Schedule  Producti                         on qty a
         d produc  on quant                         ttained
         tion vol  ity
         ume                  |---------------------( Y13 )
                                                    Producti
                                                    on qty a
                                                    ttained
                                                    lamp
      M2
(147)-| |--[>  D4      D101  ]----------------------( Y14 )
   Producti    0       10
   on qty a    Process Process
   ttained     A defect A defect
               value   thresho
                       ld
           [>  D5      D102  ]
               0       5
```

Switch: RESET STOP [✓ RUN]

Finish the Executional Conditioned Device Test.

Click ▶ to proceed.

3.5 Debugging Program Operation Step by Step

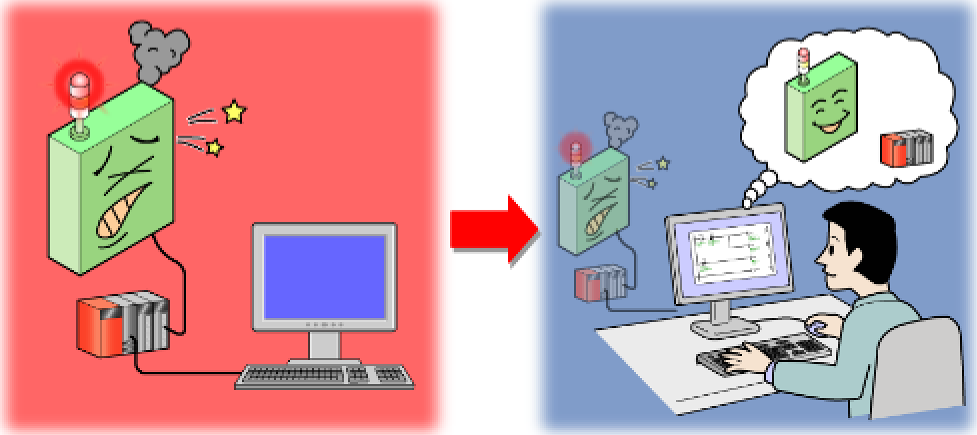
Quality

During debugging, you may want to confirm the instruction execution in each step or check for changes in a device value.
However, step-by-step debugging can be difficult due to the fast program execution speed (scan time).

Use the **"Step Execution Function"** to solve this problem. (Only for MELSEC-Q and MELSEC-L series)
This function enables program execution one step at a time and thus implements step-by-step debugging.
Use the step execution function together with the **simulation function**. (The step execution function cannot be used for debugging on an actual PLC.)

The following functions can be used by the step execution function.

Function	Description
Break execution	This function executes the program until the specified break conditions are satisfied. Program execution is stopped when the break conditions are satisfied. Use a break point and break device to specify the break conditions.
Step execution	This function executes the program step by step.
Partial execution	This function executes the program only from the specified location.



3.5.1 Using the Step Execution Function

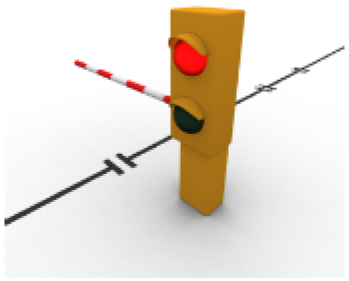
Quality

Specify a **break point** and **break device** as the debugging start location and start condition, respectively.
You can also specify a **skip range** within which you want to temporarily avoid program execution. (Only for MELSEC-Q and MELSEC-L series)

When the break conditions are satisfied after **break execution** is started, program execution is interrupted.
Thereafter, while executing program operation step by step with the **step execution function**, check for device value changes to locate a fault.

<Break point>
Set a break point where you want to interrupt program execution.
Specify this in units of steps.
Up to 64 break points can be set in the entire project.

<Break device>
Set a break device based on which program execution is interrupted when the device or label value satisfies the specified condition.
Up to 16 bit and/or word devices can be set.

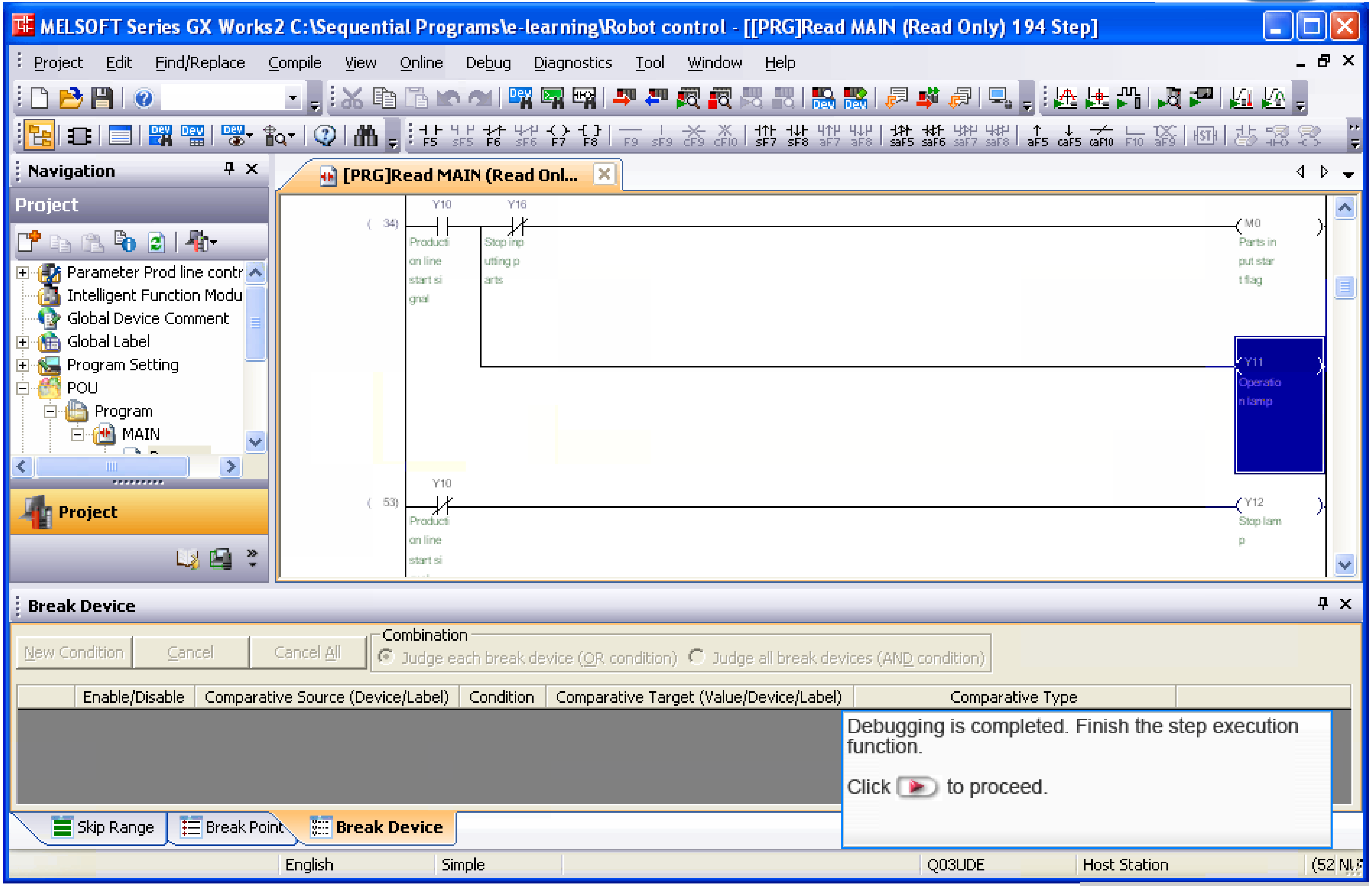


<Skip range>
Set a range within which the program should not be executed, in units of ladder blocks, during step execution.
Up to 16 ranges can be specified in the entire project.

<Skip range application>
A failure point can be narrowed down by using the **skip range**.
Perform break execution with and without a skip range set.
If a failure occurs only when the skip range setting is released, this indicates that the range includes a fault.

PLC_GX_Works2_Advanced_ENG

3.5.1 Using the Step Execution Function


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Read MAIN (Read Only) 194 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
[PRG]Read MAIN (Read Onl...
(34) Y10 Producti on line start si gnal
Y16 Stop inp utting p arts
M0 Parts in put star t flag
Y11 Operatio n lamp
(53) Y10 Producti on line start si
Y12 Stop lam p
Break Device
New Condition
Cancel
Cancel All
Combination
Judge each break device (OR condition)
Judge all break devices (AND condition)
Enable/Disable
Comparative Source (Device/Label)
Condition
Comparative Target (Value/Device/Label)
Comparative Type
Skip Range
Break Point
Break Device
English
Simple
Q03UDE
Host Station

Debugging is completed. Finish the step execution function.

Click ▶ to proceed.

3.6 Simulating the Operation of an External Device

Quality

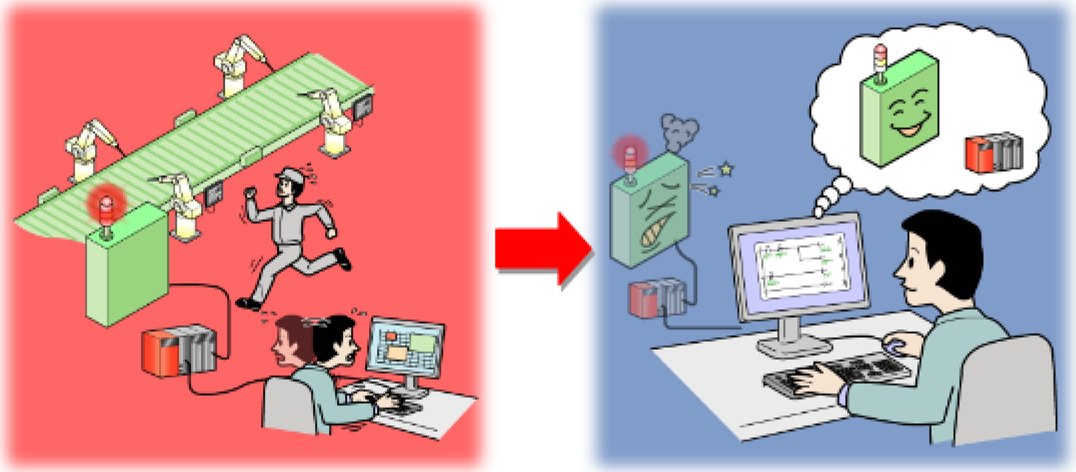
Debugging in conjunction with the operation of an external device is not possible in a development environment in which an external device cannot be used, such as the simulation function.
To solve this problem, a debugging program that simulates the operation of an external device is conventionally added.
However, not only does it take considerable time and effort to create a simulation program, but it is also necessary to modify the program when changing the operation.

Use **"I/O System Setting"** to solve this problem.
This function can simulate the operation of an external device without using a debugging program.
The operation of an external device can be easily set or changed in the setting window. Therefore, conventional program creation/modification is not necessary.

The operation of an external device can be set in the following two ways:

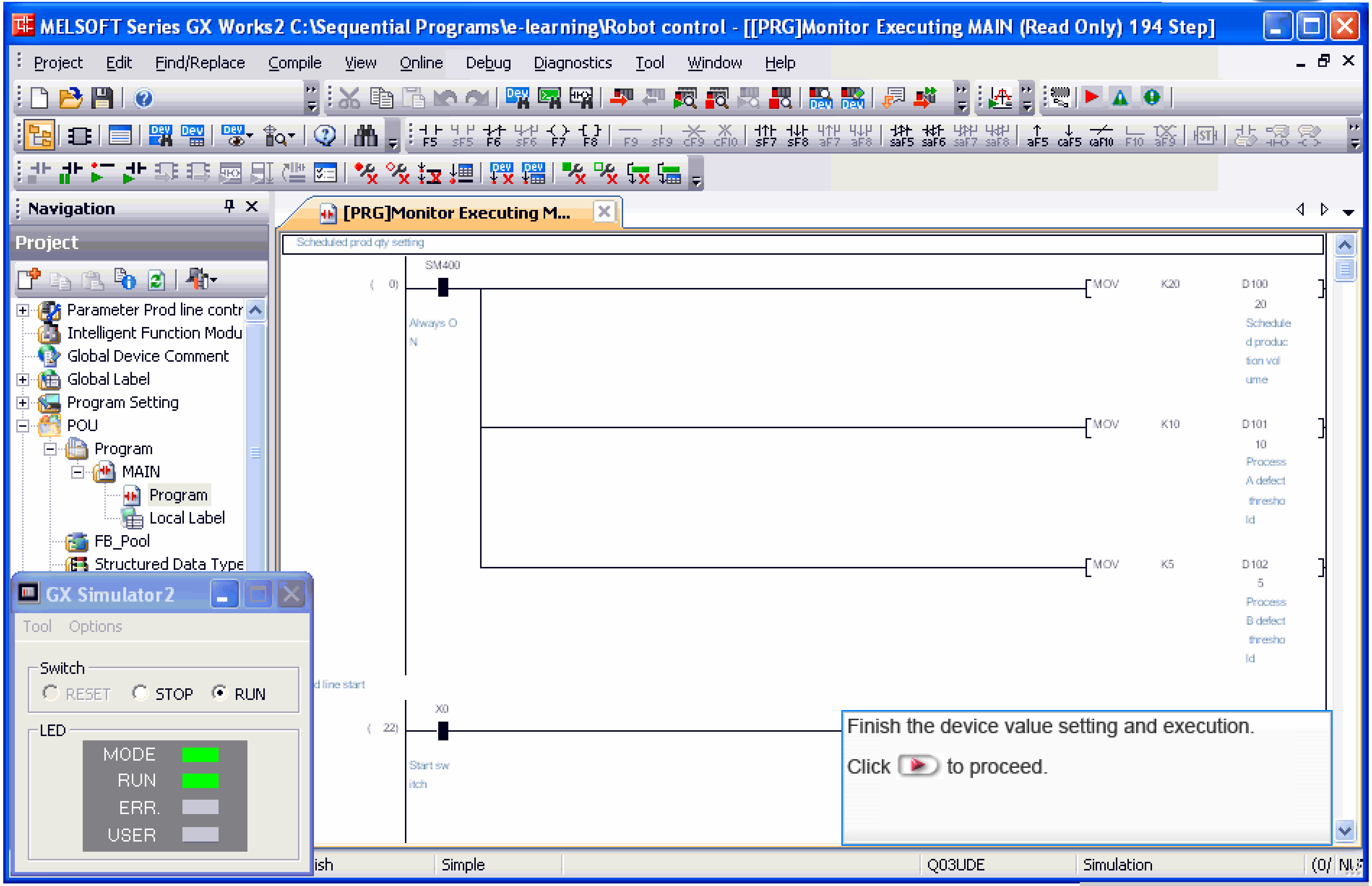
<Setting device values>
The specified device value can be changed at the timer-set time after the specified conditions are satisfied.

<Setting in timing chart format>
A device change that has been set in the specified timing chart format can be driven when the specified conditions are satisfied.



PLC_GX_Works2_Advanced_ENG

3.6.1 Inputting device values for setting the I/O system


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Monitor Executing MAIN (Read Only) 194 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line contr
Intelligent Function Modu
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Type
[PRG]Monitor Executing M...
GX Simulator2
Tool Options
Switch
RESET STOP RUN
LED
MODE
RUN
ERR.
USER
Simple
Q03UDE
Simulation

```
Scheduled prod qty setting
( 0)  SM400                                   [MOV  K20  D100 ]
      Always ON                                               20
                                                     Scheduled production volume
                                              [MOV  K10  D101 ]
                                                              10
                                                     Process A defect threshold
                                              [MOV  K5   D102 ]
                                                               5
                                                     Process B defect threshold
...d line start
( 22) X0
      Start switch
```

Switch: ( ) RESET ( ) STOP [✓ RUN]

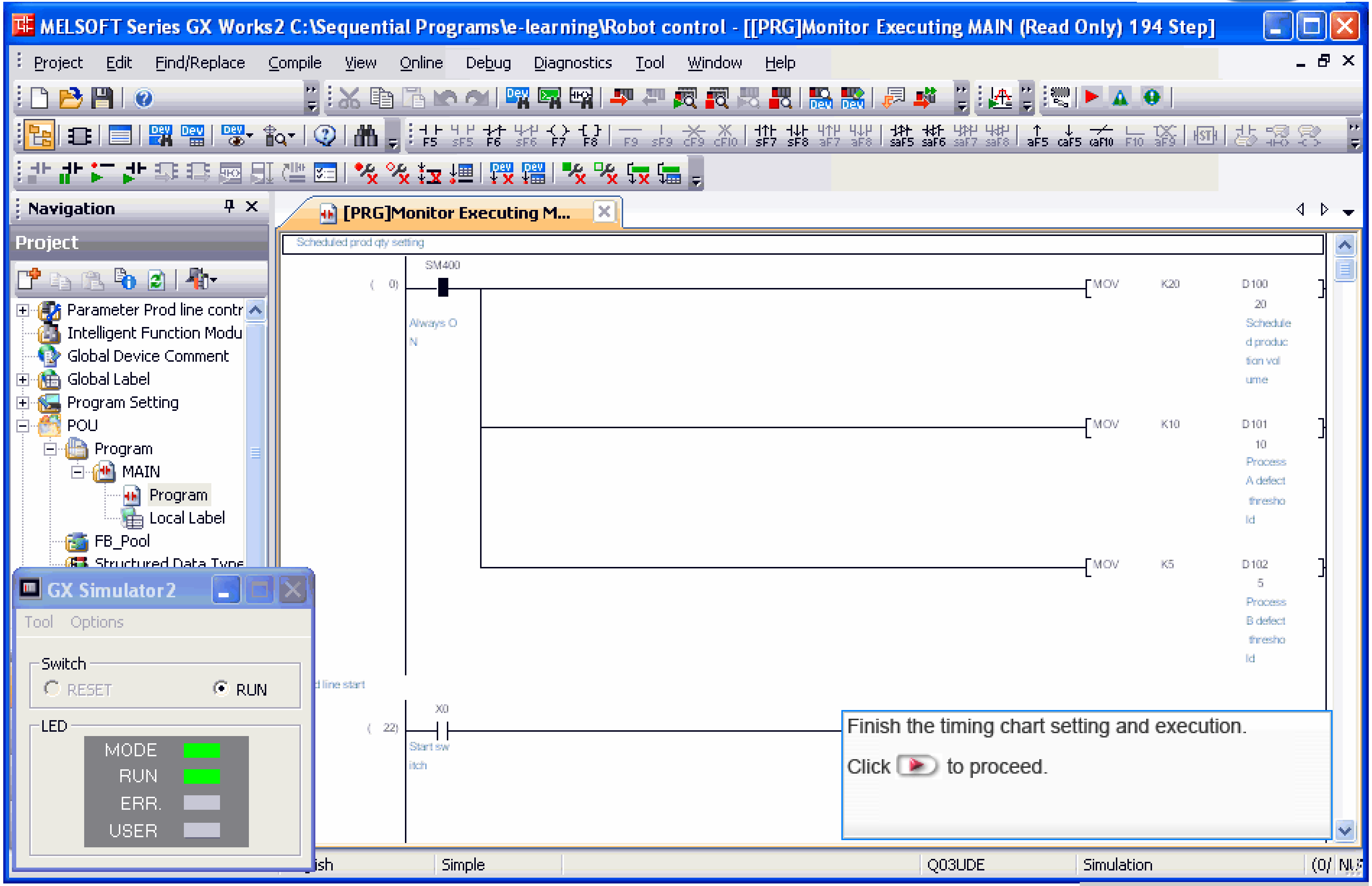
LED	
MODE	🟩
RUN	🟩
ERR.	⬜
USER	⬜

Finish the device value setting and execution.

Click ▶ to proceed.

PLC_GX_Works2_Advanced_ENG

3.6.2 Using timing chart format for setting I/O system



MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Monitor Executing MAIN (Read Only) 194 Step]

```
Scheduled prod qty setting
        SM400
(   0)──┤■├──┬──────────────────────[MOV   K20   D100 ]
        Always O  │                                20
        N         │                                Scheduled production volume
                  ├──────────────────────[MOV   K10   D101 ]
                  │                                10
                  │                                Process A defect threshold
                  └──────────────────────[MOV   K5    D102 ]
                                                   5
                                                   Process B defect threshold
d line start
        X0
(  22)──┤ ├──
        Start switch
```

Finish the timing chart setting and execution.

Click ▶ to proceed.

PLC_GX_Works2_Advanced_ENG

Chapter 4 Project Management and Security Measures

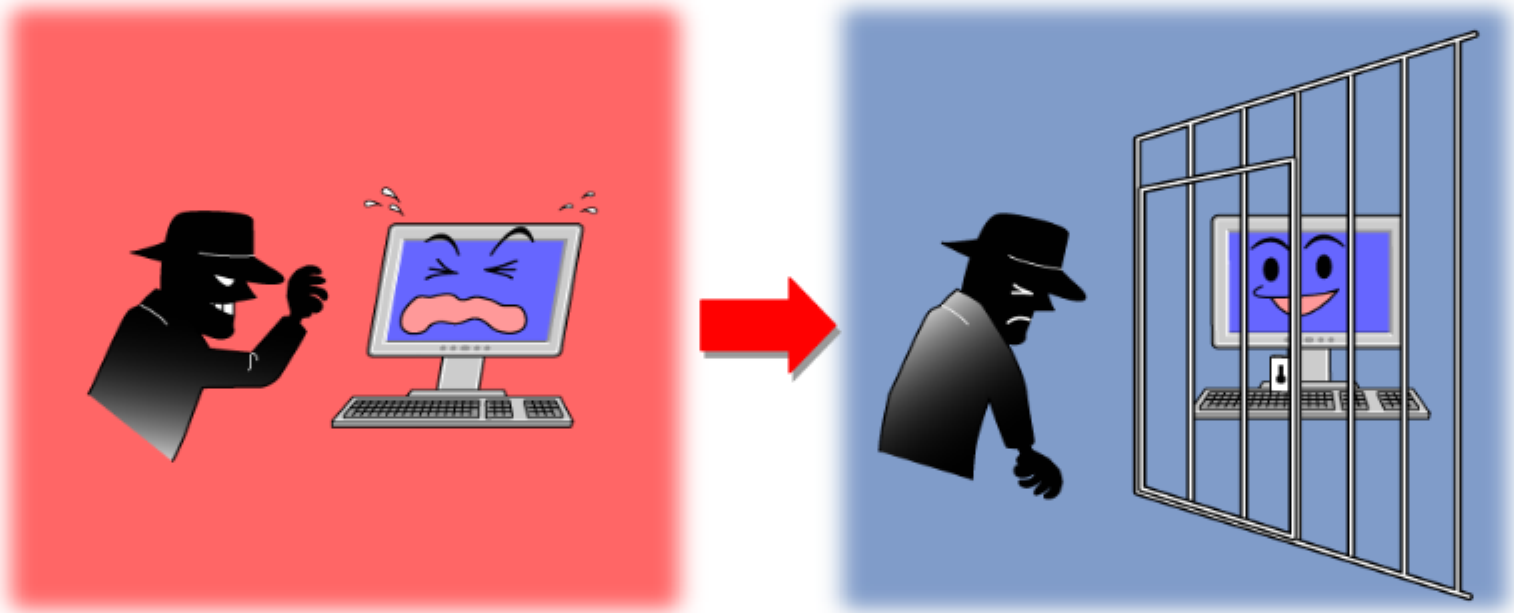
Learning steps in Chapter 4

In Chapter 4, you will learn about the functions used for project management and security measures.

4.1 Preventing Leakage of Know-How and Unauthorized Modification of Programs

The sequence program includes strategically important know-how and data.
The leakage of know-how and data from the program to the outside could have a devastating effect on business.
Unauthorized modification of the program could lead to production problems such as by stopping the system.

Use **"Security"** to solve these problems.
This function limits the users who can access each project under protection by password.
It can also limit the range of data or functions that each user can access or operate.
The function thus prevents unauthorized users from browsing or editing programs.



4.1.1 Limiting accessible data by each user

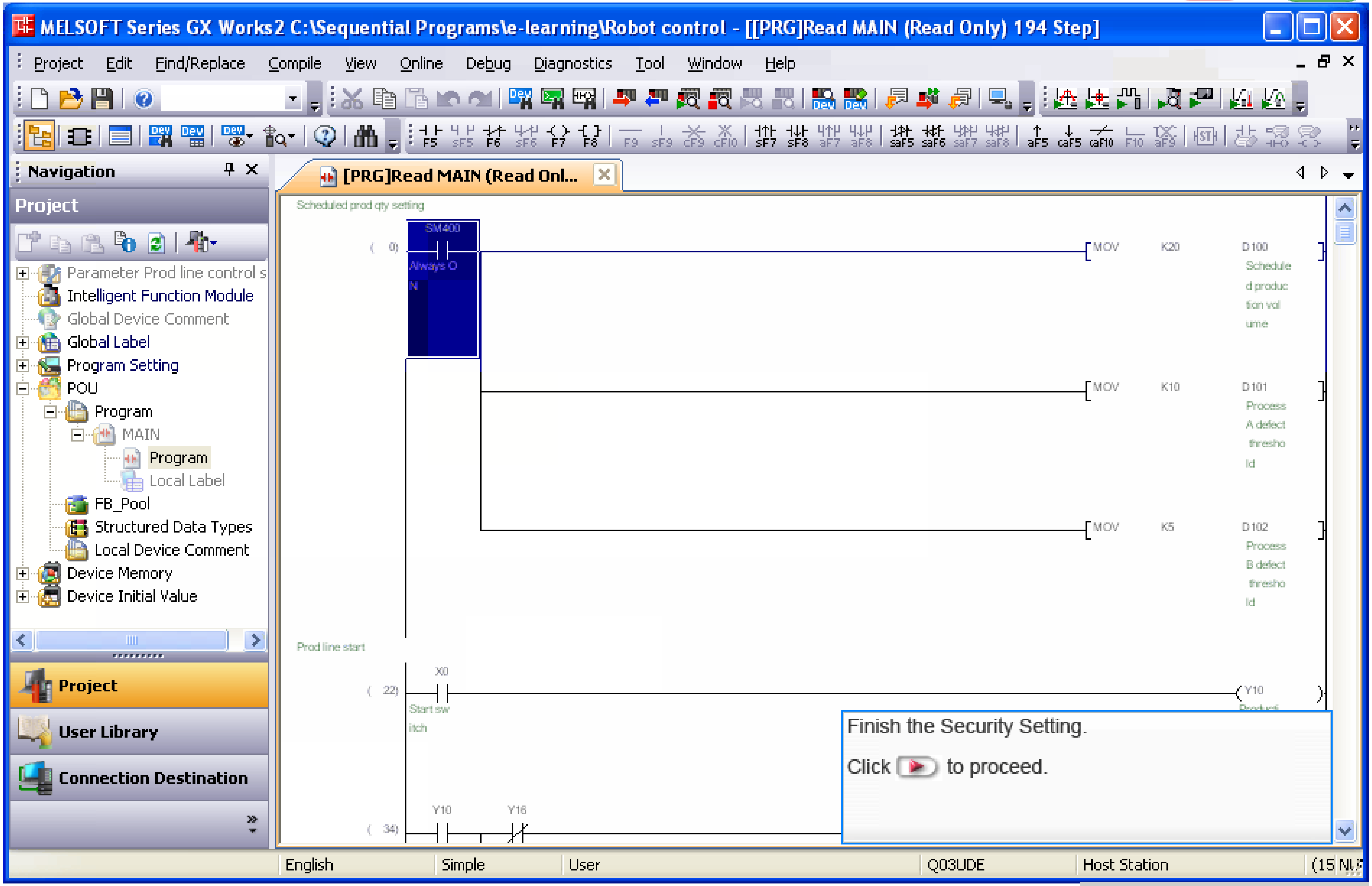
A large-scale sequence program is often developed by two or more programmers sharing the work.
In a case of team development, the range of accessible data and available functions must be properly managed according to the work range and skills level of each programmer and the confidentiality of the data handled by each programmer.
This access management can be implemented by setting security **access levels**.

<Access level>
Operation privileges for the data included in the project can be set for each user.
The following five access levels can be set.

Access level			Operation authority
High	Administrators	Administrator level	Authorized to use all functions.
↕	Developers (Level3)	Developer level	Security settings, data accessing and some operations are restricted.
	Developers (Level2)		
	Developers (Level1)		
Low	Users	Operator level	Only project data browsing is available.Unable to read from PLC CPU.

PLC_GX_Works2_Advanced_ENG

4.1.1 Limiting accessible data by each user



MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Read MAIN (Read Only) 194 Step]

Finish the Security Setting.

Click ▶ to proceed.

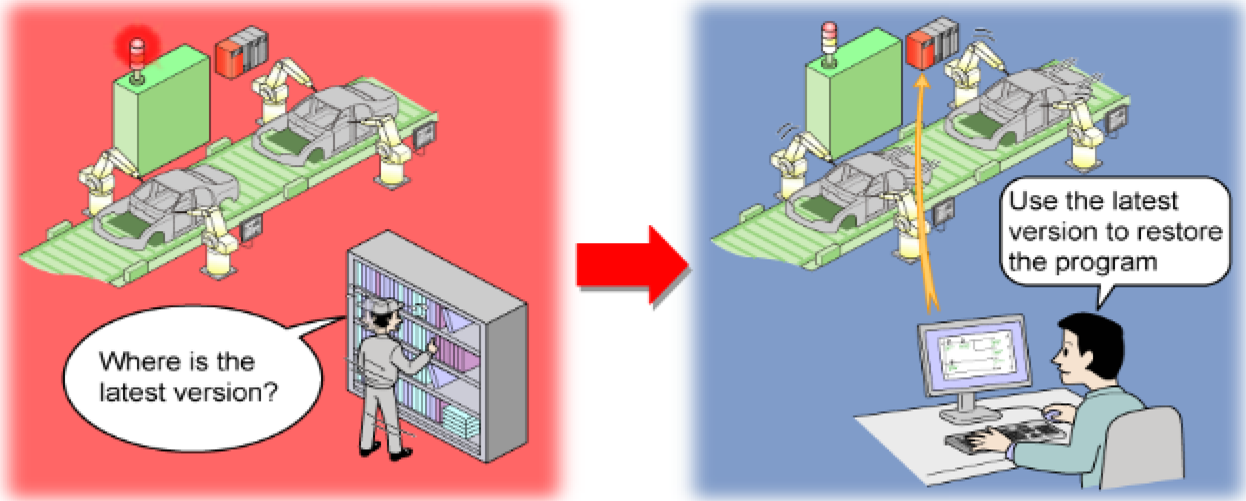
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line control s
Intelligent Function Module
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Types
Local Device Comment
Device Memory
Device Initial Value
Project
User Library
Connection Destination
[PRG]Read MAIN (Read Onl...
Scheduled prod qty setting
SM400 Always ON
MOV K20 D100 Scheduled production volume
MOV K10 D101 Process A defect threshold
MOV K5 D102 Process B defect threshold
Prod line start
X0 Start switch
Y10 Product
Y10 Y16
English Simple User Q03UDE Host Station

4.2 Project Backup and Version Management

An important program could be lost due to programmable controller CPU failure.
If a backup program is not available, there is little hope of a quick recovery.
Even if a backup program is available, it would be difficult to determine if the version is the same as the lost program unless version management is secured.
To be prepared for unexpected circumstances, it is necessary to conduct periodic backup and ensure version management.

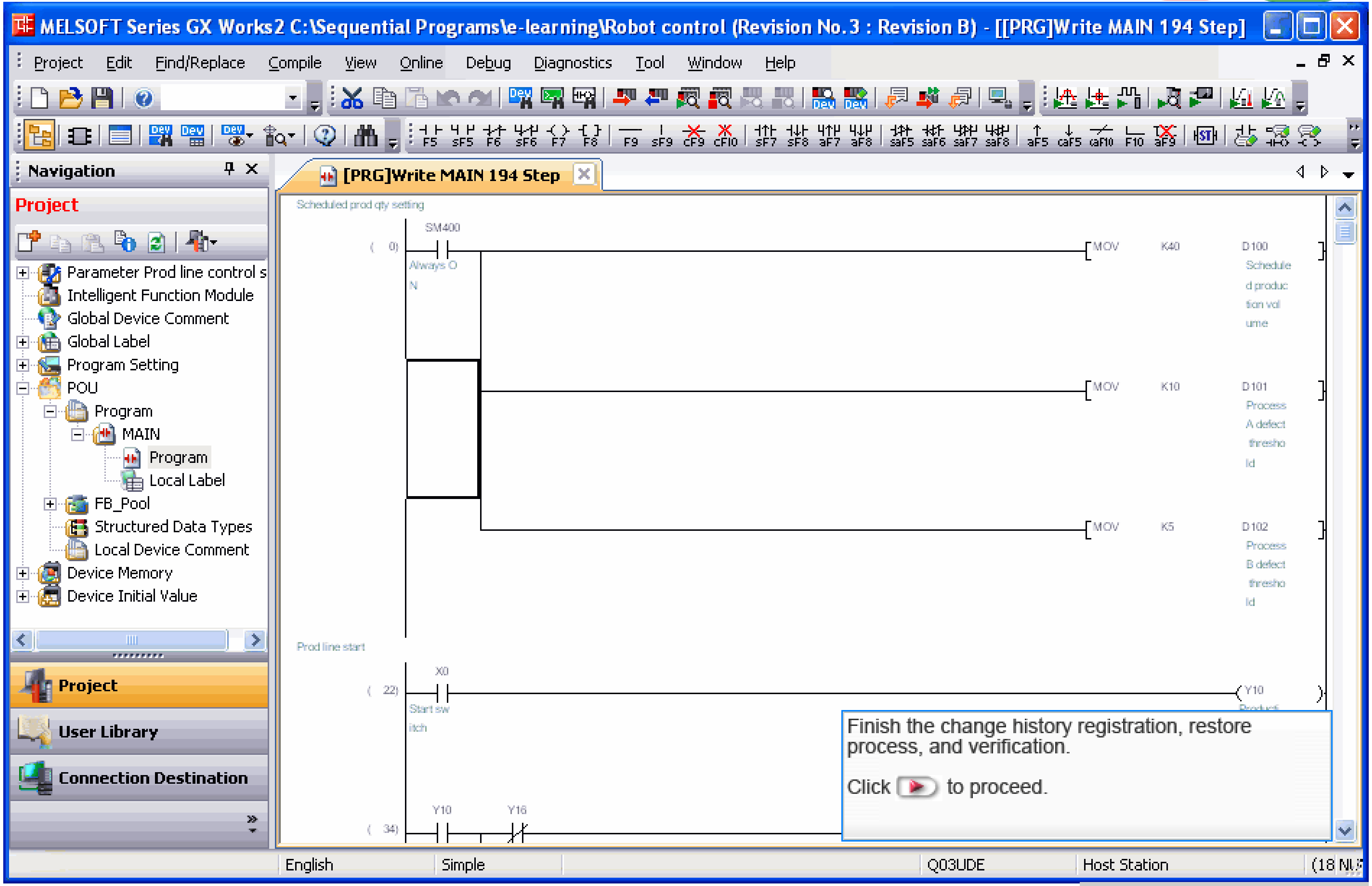
Use **"Change history"** to solve these problems.
This function can record up to 100 sets of change histories (history number, date/time, user, title, comment) of the project.
The project data at the time of recording is also backed up at the same time.

Version management ensured by the change history function allows you to restore lost programs, verify program versions, and thus implement quick recovery in the case of unexpected circumstances.


Where is the latest version?
Use the latest version to restore the program

PLC_GX_Works2_Advanced_ENG

4.2 Project Backup and Version Management



```
Scheduled prod qty setting
(   0)  SM400                              [MOV  K40  D100 ]
        Always ON                                 Scheduled production volume
                                           [MOV  K10  D101 ]
                                                  Process A defect threshold
                                           [MOV  K5   D102 ]
                                                  Process B defect threshold
Prod line start
(  22)  X0                                        (Y10 )
        Start switch
(  34)  Y10   Y16
```

Finish the change history registration, restore process, and verification.

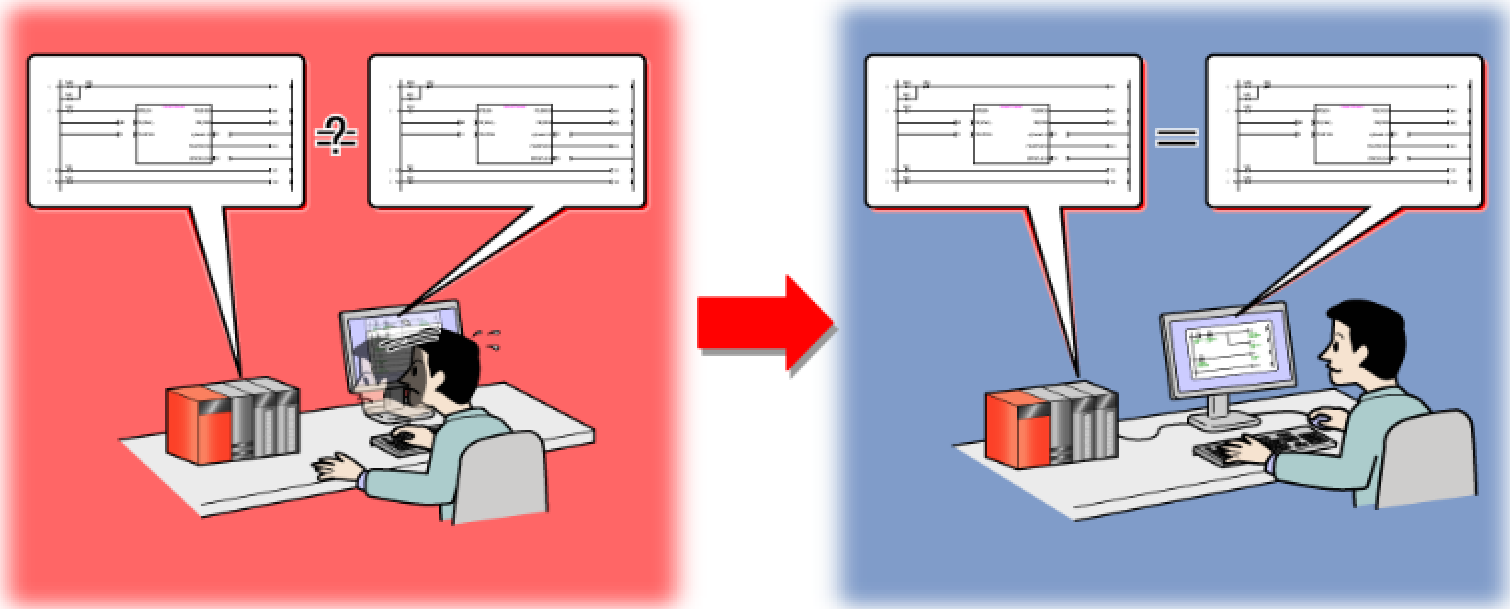
Click ▶ to proceed.

4.3 Comparing Programs Saved to Programmable Controller and Personal Computer

Management

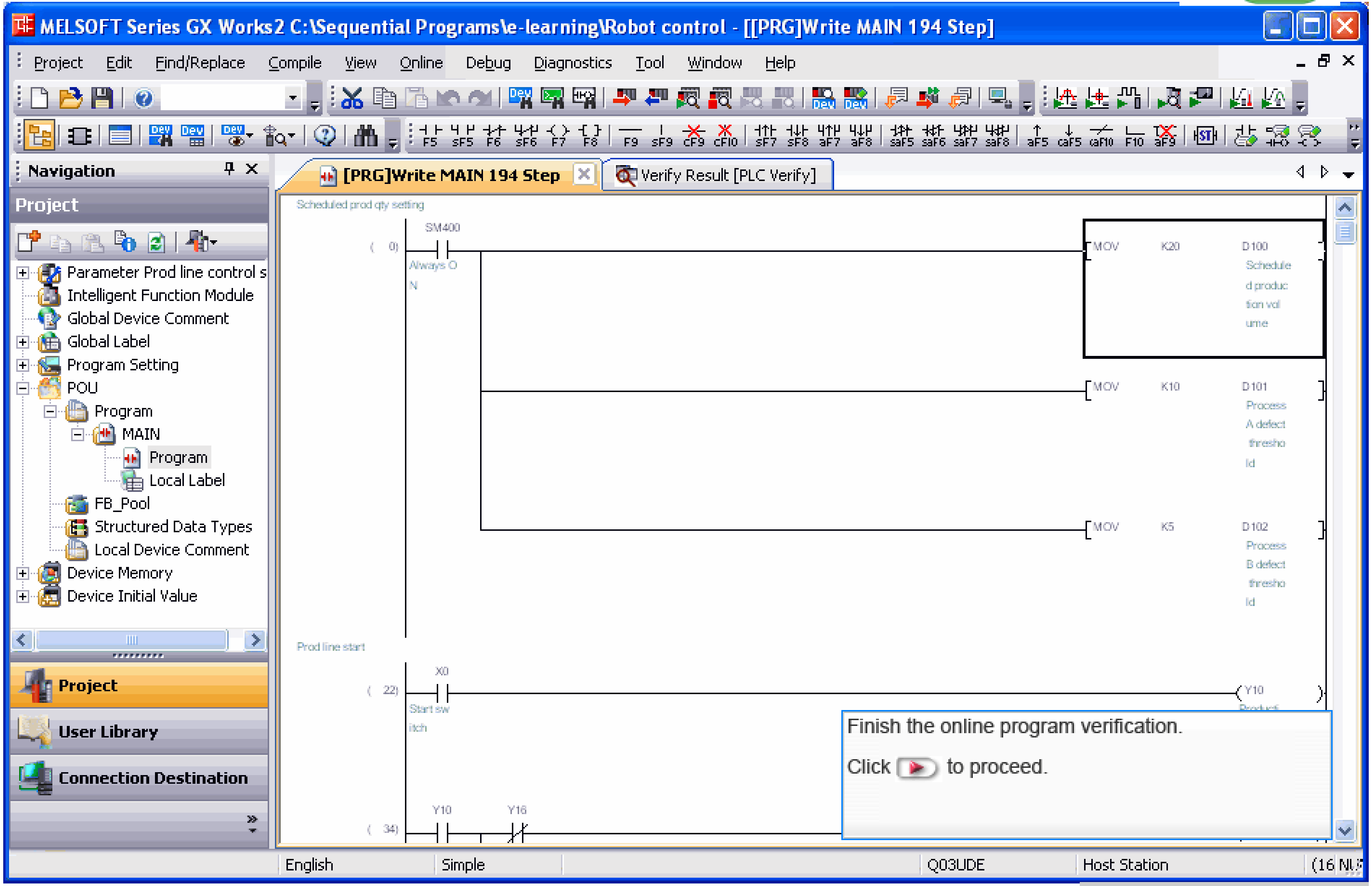
Normally, programs are saved to a PC in the development environment and they are also written to the PLC.
These two programs are not always the same.
Performing only a visual check to see if they are the same could result in errors.

Use **"Verify with PLC"** to solve this problem.
This function can verify that the program opened by GX Works2 matches the program written to the PLC.



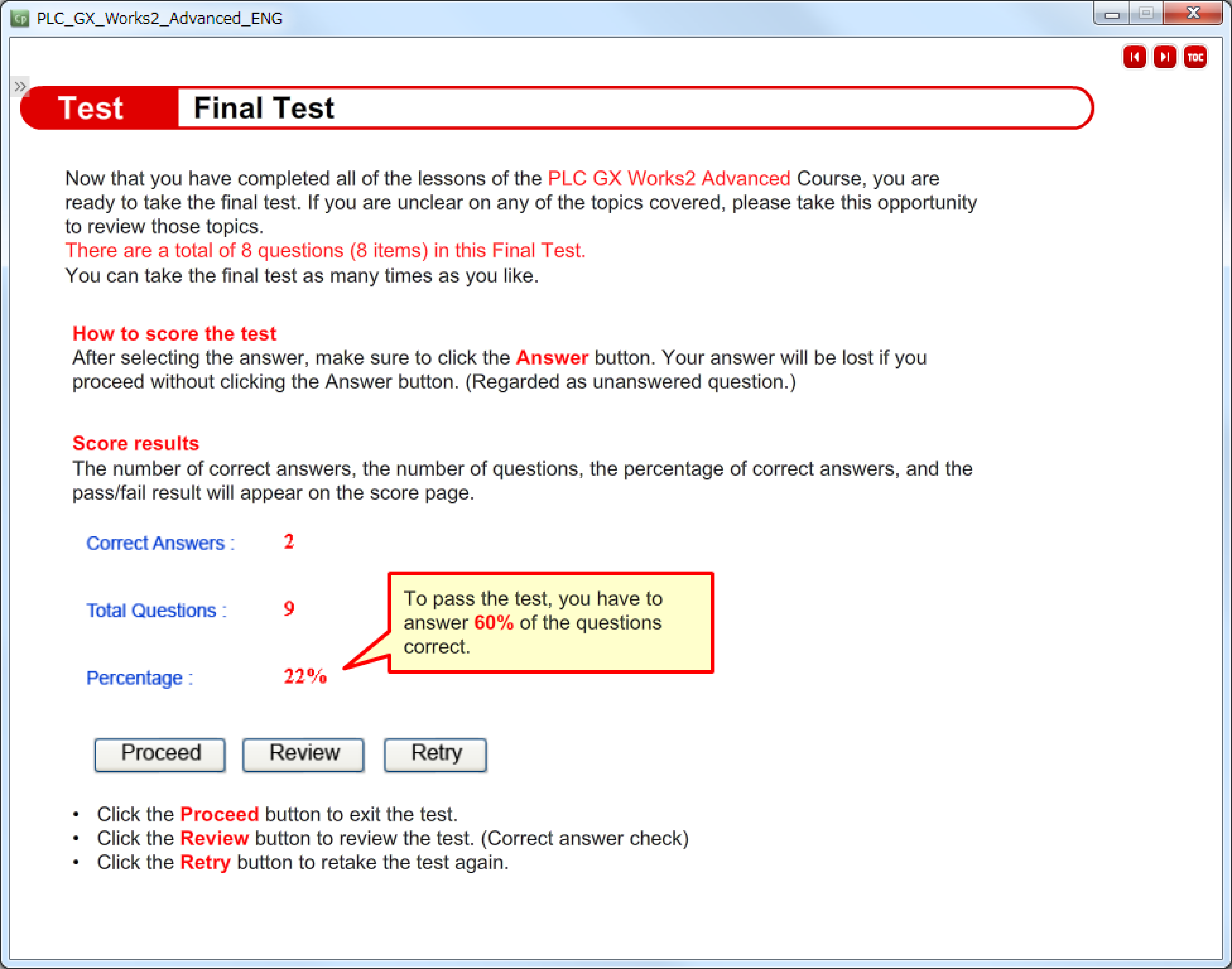
PLC_GX_Works2_Advanced_ENG

4.3 Comparing Programs Saved to Programmable Controller and Personal Computer


MELSOFT Series GX Works2 C:\Sequential Programs\e-learning\Robot control - [[PRG]Write MAIN 194 Step]
Project Edit Find/Replace Compile View Online Debug Diagnostics Tool Window Help
Navigation
Project
Parameter Prod line control s
Intelligent Function Module
Global Device Comment
Global Label
Program Setting
POU
Program
MAIN
Program
Local Label
FB_Pool
Structured Data Types
Local Device Comment
Device Memory
Device Initial Value
Project
User Library
Connection Destination
[PRG]Write MAIN 194 Step
Verify Result [PLC Verify]
Scheduled prod qty setting
SM400
Always ON
MOV K20 D100 Scheduled production volume
MOV K10 D101 Process A defect threshold
MOV K5 D102 Process B defect threshold
Prod line start
X0 Start switch
Y10
Y10 Y16
English
Simple
Q03UDE
Host Station

Finish the online program verification.

Click ▶ to proceed.



Test Final Test

Now that you have completed all of the lessons of the PLC GX Works2 Advanced Course, you are ready to take the final test. If you are unclear on any of the topics covered, please take this opportunity to review those topics.
There are a total of 8 questions (8 items) in this Final Test.
You can take the final test as many times as you like.

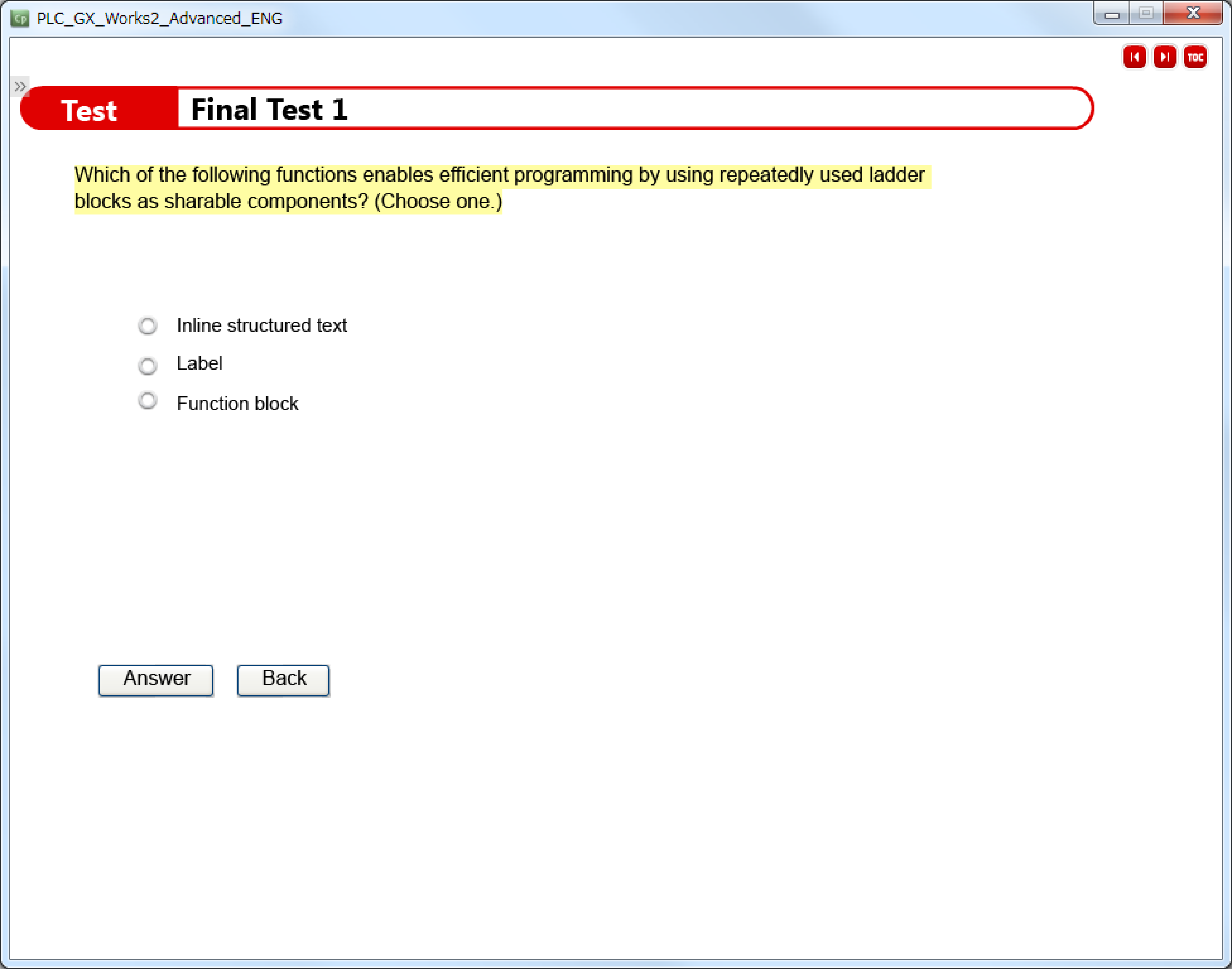
How to score the test
After selecting the answer, make sure to click the **Answer** button. Your answer will be lost if you proceed without clicking the Answer button. (Regarded as unanswered question.)

Score results
The number of correct answers, the number of questions, the percentage of correct answers, and the pass/fail result will appear on the score page.

Correct Answers :	2
Total Questions :	9
Percentage :	22%

To pass the test, you have to answer **60%** of the questions correct.

- Click the **Proceed** button to exit the test.
- Click the **Review** button to review the test. (Correct answer check)
- Click the **Retry** button to retake the test again.



PLC_GX_Works2_Advanced_ENG

TOC

Test

Final Test 1

Which of the following functions enables efficient programming by using repeatedly used ladder blocks as sharable components? (Choose one.)

- ○ Inline structured text
- ○ Label
- ○ Function block

Answer Back

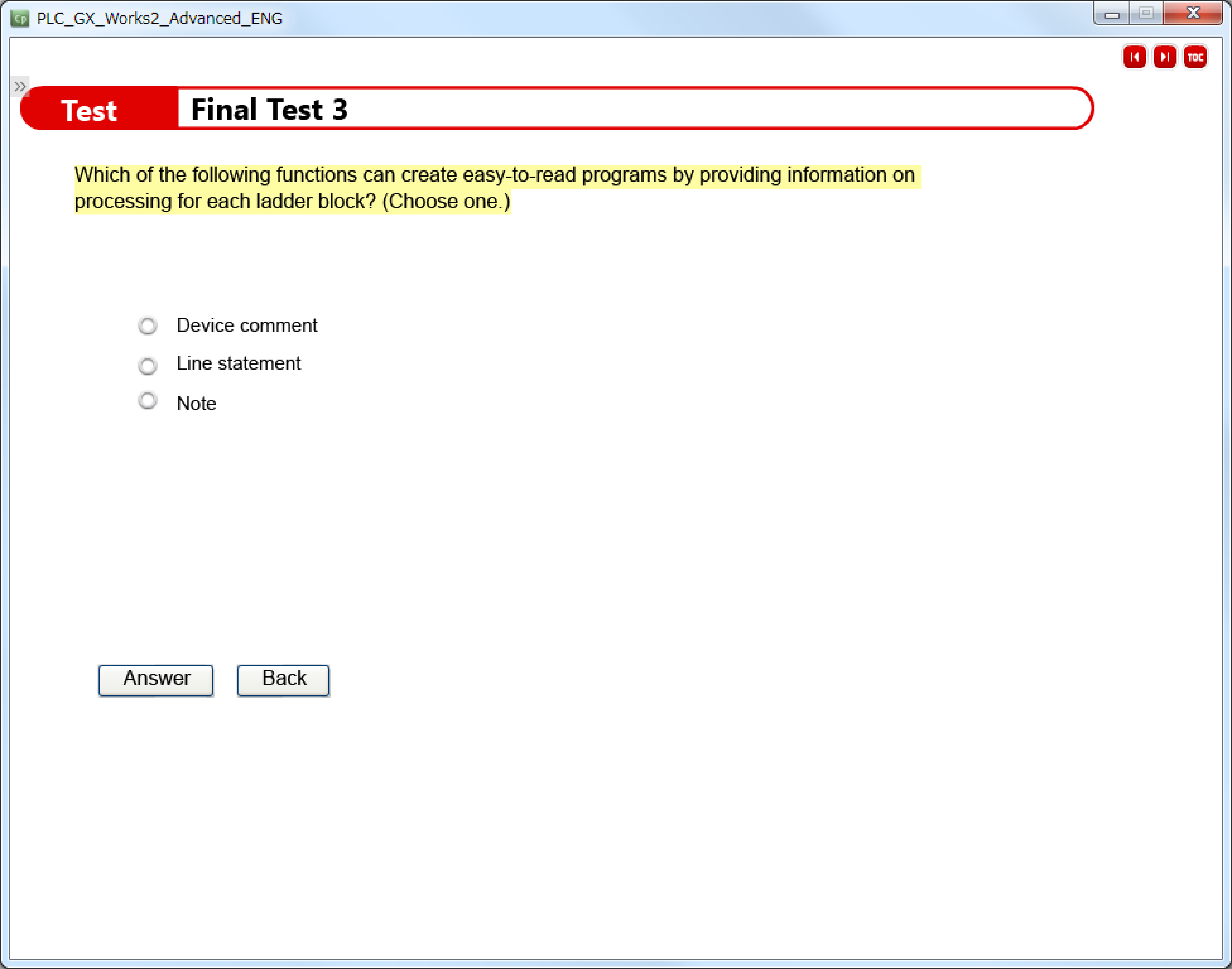
PLC_GX_Works2_Advanced_ENG

Test Final Test 2

Which of the following functions can create easy-to-read programs by changing the device names to names associated with their application? (Choose one.)

- Device comment
- Label
- Note

Answer Back

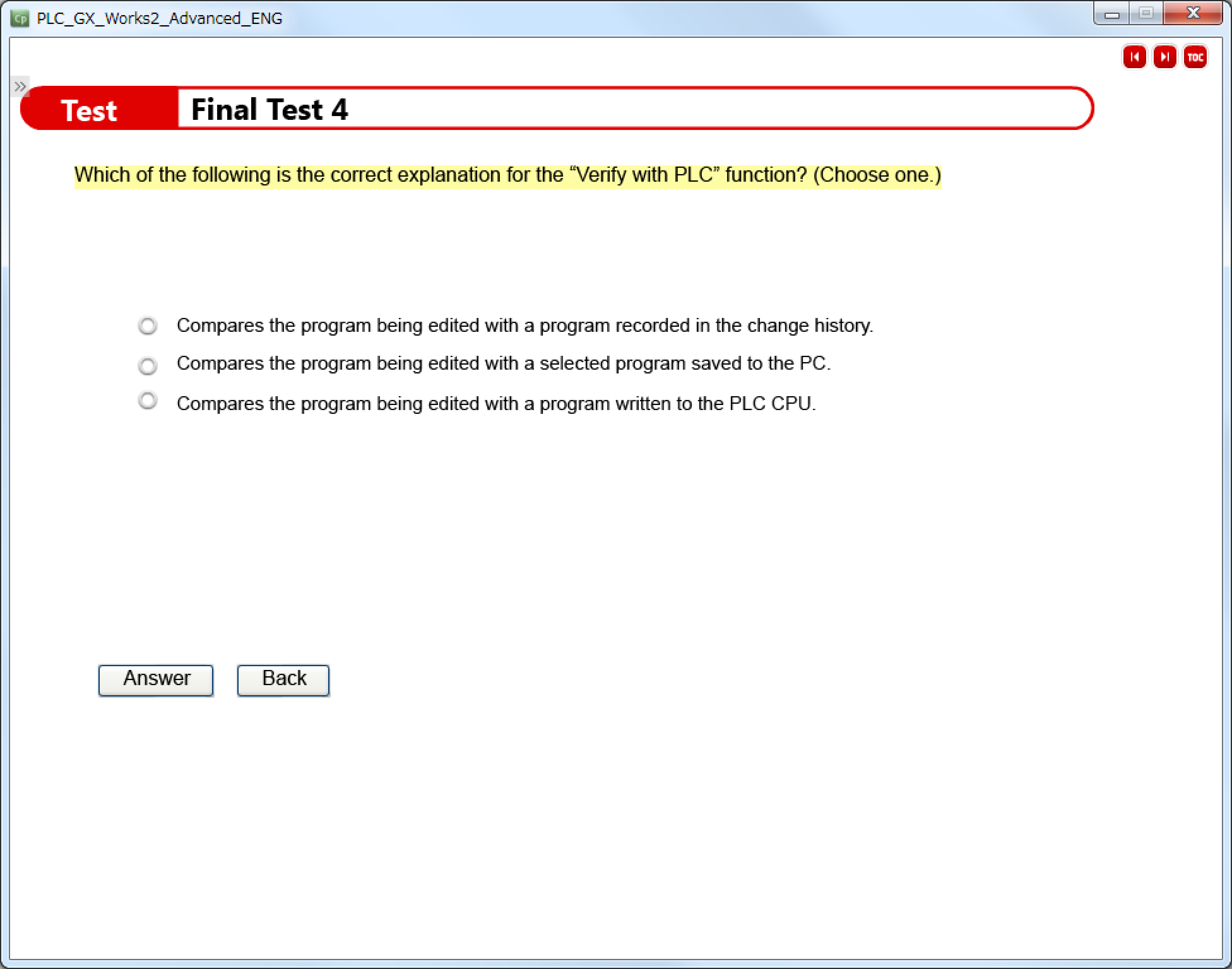


Test

Final Test 3

Which of the following functions can create easy-to-read programs by providing information on processing for each ladder block? (Choose one.)

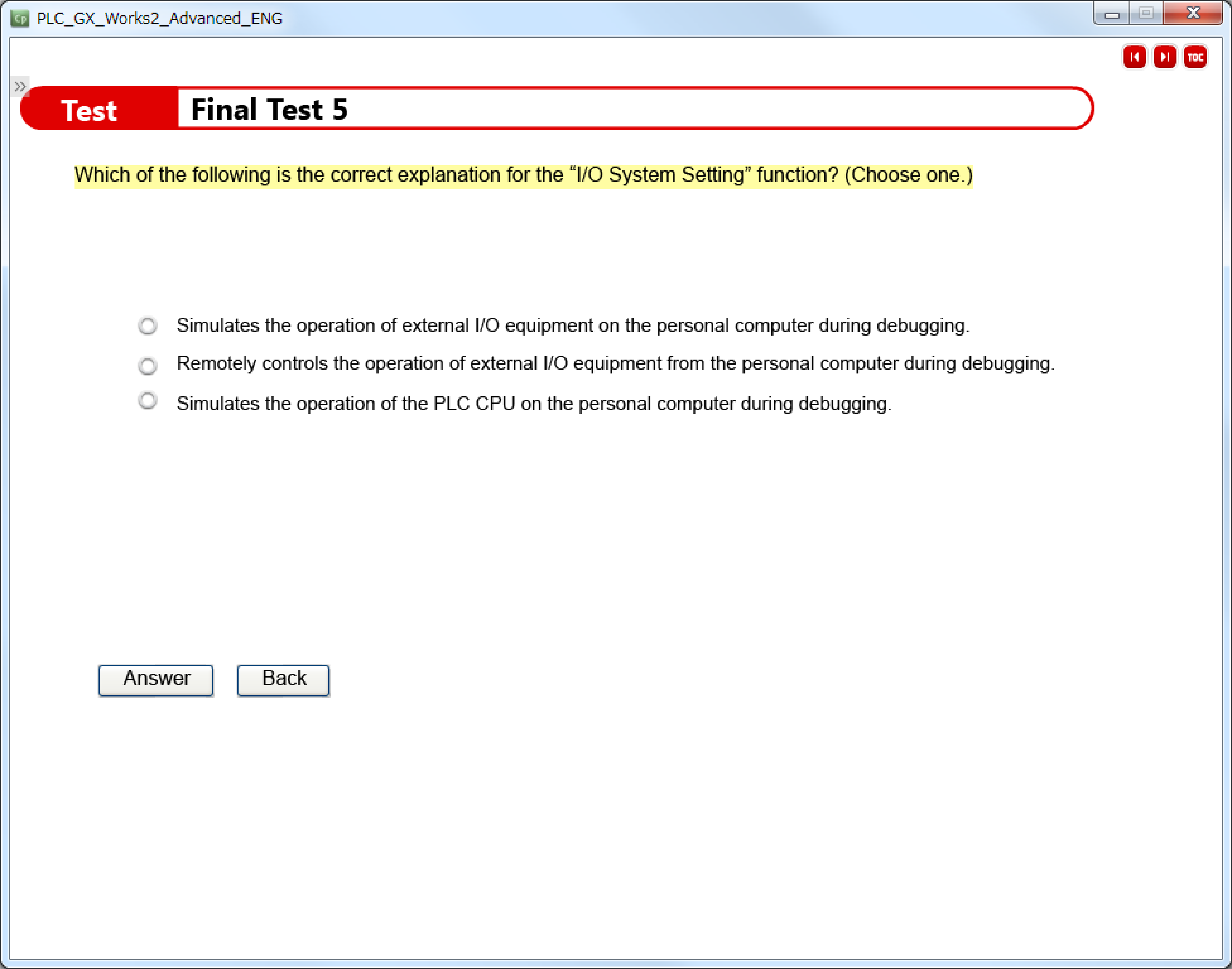
- Device comment
- Line statement
- Note



Test Final Test 4

Which of the following is the correct explanation for the “Verify with PLC” function? (Choose one.)

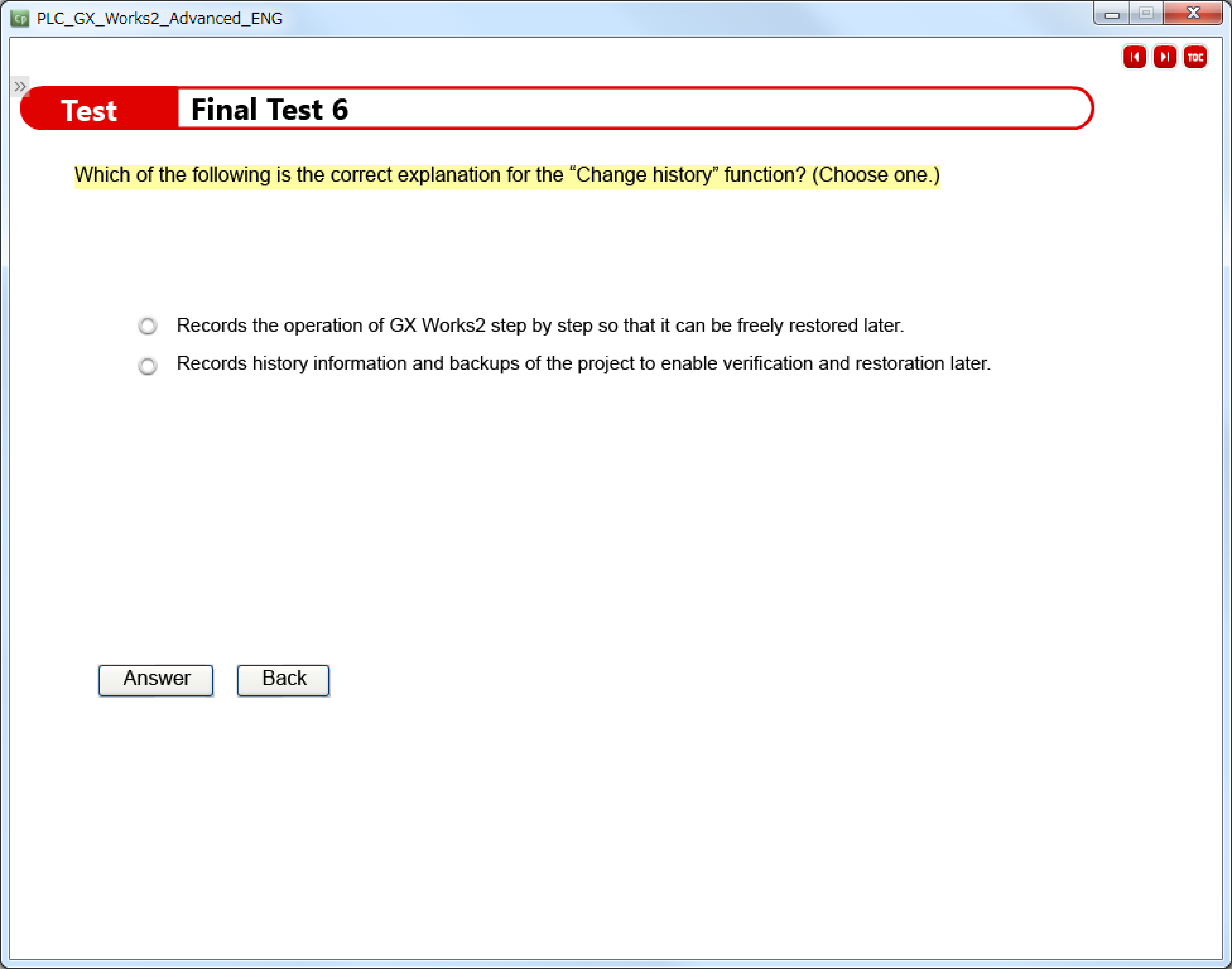
- ○ Compares the program being edited with a program recorded in the change history.
- ○ Compares the program being edited with a selected program saved to the PC.
- ○ Compares the program being edited with a program written to the PLC CPU.



Test Final Test 5

Which of the following is the correct explanation for the “I/O System Setting” function? (Choose one.)

- ○ Simulates the operation of external I/O equipment on the personal computer during debugging.
- ○ Remotely controls the operation of external I/O equipment from the personal computer during debugging.
- ○ Simulates the operation of the PLC CPU on the personal computer during debugging.

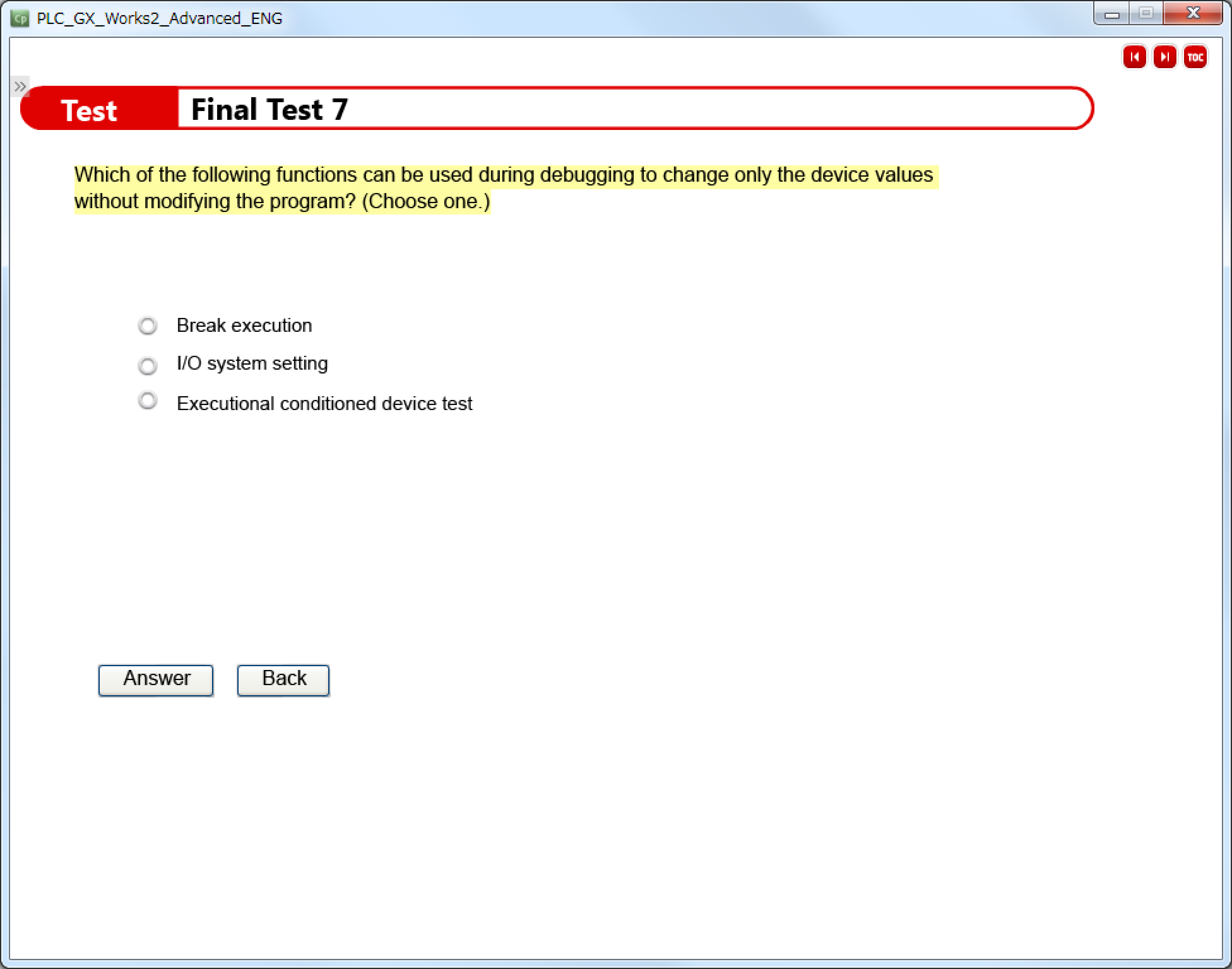


PLC_GX_Works2_Advanced_ENG

Test Final Test 6

Which of the following is the correct explanation for the “Change history” function? (Choose one.)

- ○ Records the operation of GX Works2 step by step so that it can be freely restored later.
- ○ Records history information and backups of the project to enable verification and restoration later.



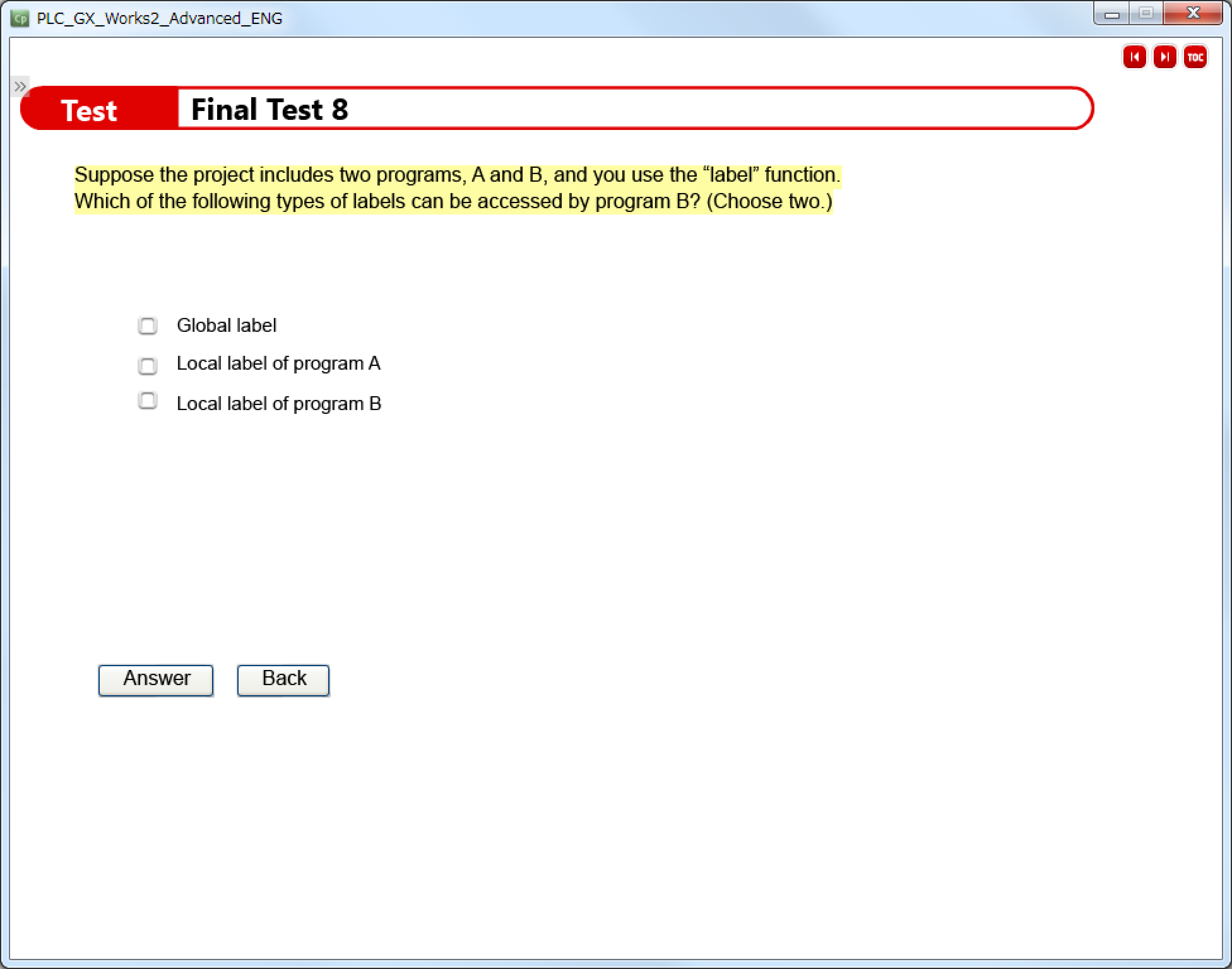
PLC_GX_Works2_Advanced_ENG

Test

Final Test 7

Which of the following functions can be used during debugging to change only the device values without modifying the program? (Choose one.)

- ○ Break execution
- ○ I/O system setting
- ○ Executional conditioned device test

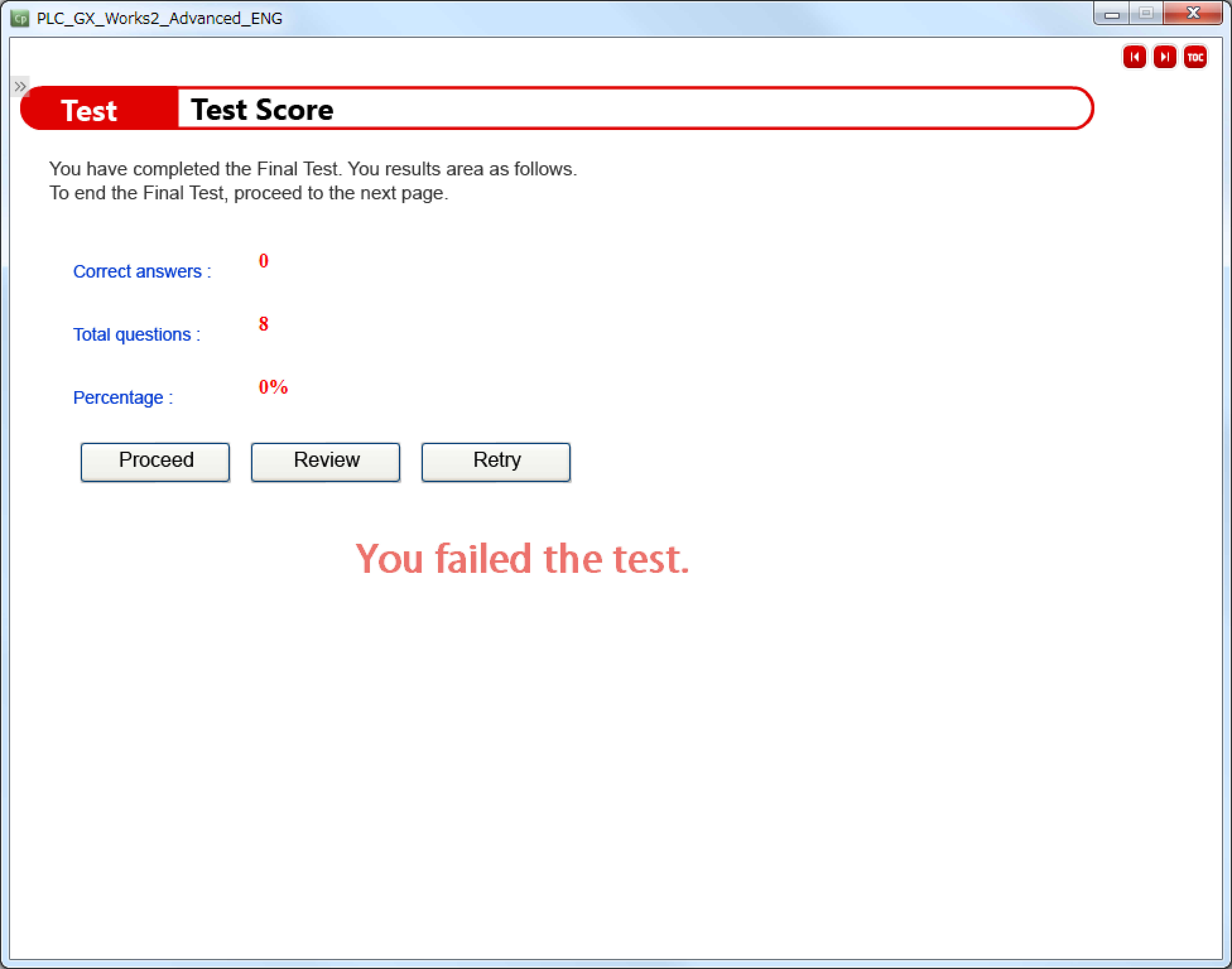

PLC_GX_Works2_Advanced_ENG

Test

Final Test 8

Suppose the project includes two programs, A and B, and you use the "label" function.
Which of the following types of labels can be accessed by program B? (Choose two.)

- ☐ Global label
- ☐ Local label of program A
- ☐ Local label of program B

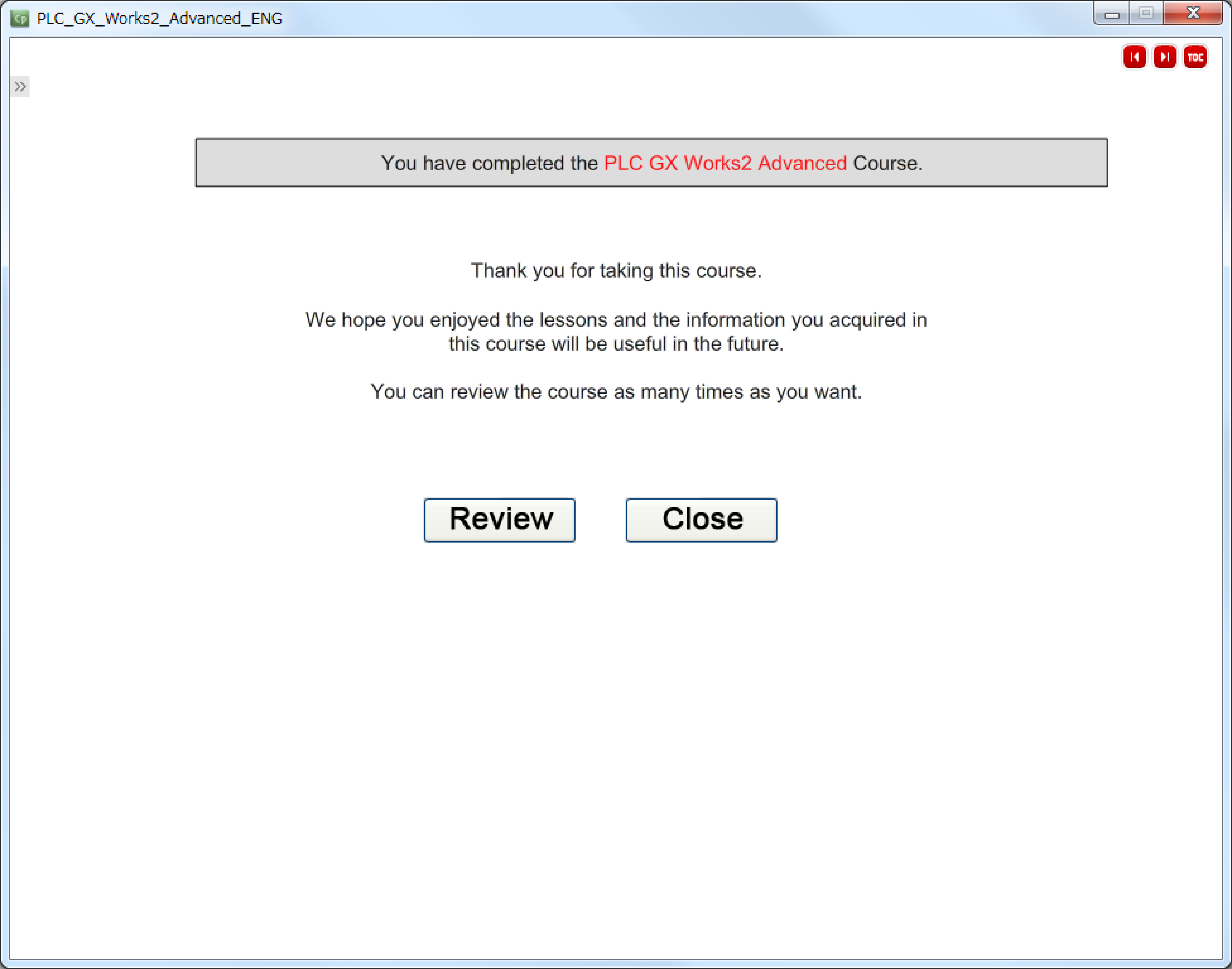

PLC_GX_Works2_Advanced_ENG

Test | Test Score

You have completed the Final Test. You results area as follows.
To end the Final Test, proceed to the next page.

Correct answers :	0
Total questions :	8
Percentage :	0%

You failed the test.



You have completed the PLC GX Works2 Advanced Course.

Thank you for taking this course.

We hope you enjoyed the lessons and the information you acquired in this course will be useful in the future.

You can review the course as many times as you want.