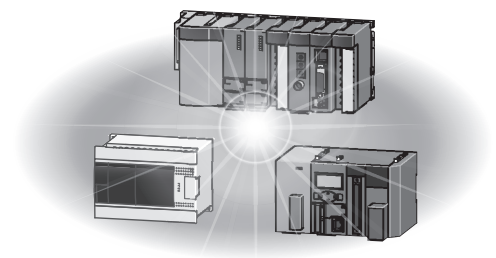


Programmable Controller

MELSEC **Q**series MELSEC *L*series

Structured Text (ST)  
Programming Guide Book

---





## • SAFETY PRECAUTIONS •

(Read these precautions before using this product.)

Before using this product, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly. If products are used in a different way from that specified by manufacturers, the protection function of the products may not work properly.

The precautions given in this manual are concerned with this product only. For the safety precautions for the programmable controller system, refer to the user's manual for the CPU module.

In this manual, the safety precautions are classified into two levels: "⚠ WARNING" and "⚠ CAUTION".



Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.



Indicates that incorrect handling may cause hazardous conditions, resulting in minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠ CAUTION" may lead to serious consequences.

Observe the precautions of both levels because they are important for personal and system safety.

Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

### [Design Precautions]

#### ⚠ WARNING

- When data change, program change, or status control is performed from a personal computer to a running programmable controller, create an interlock circuit outside the programmable controller to ensure that the whole system always operates safely.  
Furthermore, for the online operations performed from a personal computer to a programmable controller CPU, the corrective actions against a communication error due to such as a cable connection fault should be predetermined as a system.

### [Security Precautions]

#### ⚠ WARNING

- To maintain the security (confidentiality, integrity, and availability) of the programmable controller and the system against unauthorized access, denial-of-service (DoS) attacks, computer viruses, and other cyberattacks from external devices via the network, take appropriate measures such as firewalls, virtual private networks (VPNs), and antivirus solutions.

## [Startup/Maintenance Precautions]

### CAUTION

- The online operations performed from a personal computer to a running programmable controller CPU (program change when a programmable controller CPU is RUN, forced I/O operation, operating status change such as RUN-STOP switching, and remote control operation) have to be executed after the manual has been carefully read and the safety has been ensured.

When changing a program while a programmable controller CPU is RUN (Online program change), it may cause a program corruption in some operating conditions. Fully understand the precautions described in Section 16.9 before use.

- Note that exchangeable modules online (while the power is on) are restricted and each of the modules has its predetermined replacement procedure. For details, refer to the online module change section in the manual of the module supporting the function.

## • CONDITIONS OF USE FOR THE PRODUCT •

- (1) MELSEC programmable controller ("the PRODUCT") shall be used in conditions;
  - i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and
  - ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

- (2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI ELECTRIC SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI ELECTRIC USER'S, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above restrictions, Mitsubishi Electric may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi Electric and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTS are required. For details, please contact the Mitsubishi Electric representative in your region.

- (3) Mitsubishi Electric shall have no responsibility or liability for any problems involving programmable controller trouble and system trouble caused by DoS attacks, unauthorized access, computer viruses, and other cyberattacks.

REVISIONS

\* The manual number is given on the bottom left of the back cover.

Print Date	* Manual Number	Revision
Feb., 2003	SH (NA)-080368E-A	First printing
Jul., 2003	SH (NA)-080368E-B	Correction Section 5.3
Oct., 2003	SH (NA)-080368E-C	Correction Section 5.2
Jun., 2004	SH (NA)-080368E-D	Correction Abbreviations and Generic Terms in This Manual, Chapter 3
May, 2008	SH (NA)-080368E-E	Correction Abbreviations and Generic Terms in This Manual, Chapter 3
Dec., 2008	SH (NA)-080368E-F	Correction About Manuals, Abbreviations and Generic Terms in This Manual, Chapter 3
Jan., 2010	SH (NA)-080368E-G	Addition CONDITIONS OF USE FOR THE PRODUCT Correction SAFETY PRECAUTIONS, INTRODUCTION, About Manuals, How to Use This Manual, Abbreviations and Generic Terms in This Manual, Chapter 1, Chapter 3
Oct., 2014	SH (NA) 080368E-H	Correction About Manuals, How to Use This Manual
Nov., 2022	SH (NA) 080368E-I	Correction SAFETY PRECAUTIONS, CONDITIONS OF USE FOR THE PRODUCT, INTRODUCTION

Japanese Manual Version SH-080365-K

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

## INTRODUCTION

Thank you for purchasing the Mitsubishi Electric MELSEC-Q/L series programmable controllers. Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC-Q/L series programmable controllers to handle the product correctly.

When applying the program examples provided in this manual to an actual system, ensure the applicability and confirm that it will not cause system control problems.

## CONTENTS

SAFETY PRECAUTIONS.....	A- 1
CONDITIONS OF USE FOR THE PRODUCT .....	A- 3
REVISIONS .....	A- 4
INTRODUCTION.....	A- 5
CONTENTS.....	A- 5
MANUALS .....	A- 7
HOW TO USE THIS MANUAL .....	A- 8
GENERIC TERMS AND ABBREVIATIONS .....	A- 9
<b>1 OVERVIEW</b>	<b>1- 1 to 1- 2</b>
<b>2 ST PROGRAM CREATION PROCEDURE</b>	<b>2- 1 to 2- 2</b>
<b>3 ST PROGRAMMING</b>	<b>3- 1 to 3- 16</b>
Creating a new ST project .....	3- 1
Defining the labels.....	3- 3
Entering a program .....	3- 8
Converting (compiling) the ST program .....	3-14
<b>4 READ/WRITE FROM/TO PLC CPU</b>	<b>4- 1 to 4- 2</b>
<b>5 DEBUGGING THE PROGRAM</b>	<b>5- 1 to 5- 7</b>
5.1 Monitoring the Sequence Program.....	5- 1
5.2 Device Test .....	5- 2
5.3 Online Change .....	5- 4
<b>6 SAVING THE PROGRAM INTO THE PERSONAL COMPUTER</b>	<b>6- 1 to 6- 2</b>
<b>7 INTRODUCTION TO USEFUL FUNCTIONS FOR ST PROGRAM EDITING</b>	<b>7- 1 to 7- 2</b>

8 ST PROGRAMMING APPLICATION (PASTING FB TO LADDER PROGRAM)	8- 1 to 8- 20
---	---------------

8.1 Creating an FB .....	8- 1
Creating a new project .....	8- 1
Adding a new FB .....	8- 2
Defining FB variables .....	8- 5
Creating an FB in ST language .....	8- 9
8.2 Pasting the FB to a Main Program .....	8-11
Defining the local variables .....	8-11
Creating a main program .....	8-13
8.3 Online .....	8-16
Writing to programmable controller CPU.....	8-16
Monitoring the sequence program.....	8-17
Confirming the program behavior .....	8-18

INDEX	Index- 1 to Index- 2
-------	----------------------



## MANUALS

The manuals related to this product are shown below.  
Refer to the following table when ordering required manuals.

### Related Manuals

Manual Name	Manual Number (Model Code)
GX Developer Version8 Operating Manual (Startup) Explains the system configuration, installation method and startup procedure of GX Developer. (Sold separately)	SH-080372E (13JU40)
GX Developer Version8 Operating Manual Explains operation methods such as creating, printing, monitoring, and debugging programs using GX Developer. (Sold separately)	SH-080373E (13JU41)
GX Developer Version8 Operating Manual (Structured Text) Explains operation methods such as creating and printing structured text (ST) programs using GX Developer. (Sold separately)	SH-080367E (13JU37)
GX Developer Version8 Operating Manual (Function Block) Explains the editing and monitoring operations of the function using GX Developer. (Sold separately)	SH-080376E (13JU44)
MELSEC-Q/L Programming Manual (Structured Text) Explains the programming methods in structured text language. (Sold separately)	SH-080366E (13JF68)
MELSEC-Q/L Programming Manual (Common Instructions) Explains the methods of using the sequence instructions, basic instructions and application instructions. (Sold separately)	SH-080809ENG (13JW10)
GX Works2 Beginner's Manual (Structured Project) Explains the structured text (ST) programming methods using GX Works2. (Sold separately)	SH-080788ENG (13JZ23)

### REMARK

The Operating Manuals are included on the software package in a PDF file format.  
Manuals in printed form are sold separately for single purchase. Order a manual by quoting the manual number (model code) listed in the table above.

## HOW TO USE THIS MANUAL

### *This Guidebook ...*

This guidebook is a commentary written for those who will use the GX Developer Version 8 software package (hereafter abbreviated to GX Developer) to create structured text (hereafter abbreviated to ST) programs for the first time.

"Chapter 1" introduces the overview of the ST language and the features of the ST language in the MELSEC-Q/L series.

"Chapter 2 to Chapter 6" introduce a series of basic operation methods, such as the methods of creating, debugging and saving programs in ST language through sample programs.

"Chapter 7" introduces useful functions available from GX Developer.

"Chapter 8" introduces the method of creating a program, which uses an ST-written function block (FB) in a ladder program from the main program created in ladder form, as an application program.

"Chapters 4, 5 and 8" use the programmable controller CPU for explanation.

### *Programming Manual ...*

Use the "MELSEC-Q/L Programming Manual (Structured Text)" to perform structured text (ST) programming with GX Developer. It is suitable for the users who have the knowledge and programming experience of programmable controller ladder programs and for the users who have the knowledge and programming experience of high-level languages such as the C language.

### *Operating Manual ...*

The "GX Developer Version 8 Operating Manual (Structured Text)" is a commentary that gives in-depth explanation of the operation methods for creating structured text programs using GX Developer. Refer to the manual when information on operations details is necessary.

### *When information on other than structured text programming is necessary ...*

Refer to the "GX Developer Version 8 Operating Manual" or "GX Developer Version 8 Operating Manual (Startup)".

### *When using GX Works2 ...*

Refer to "GX Works2 Beginner's Manual (Structured Project)". It describes the procedures for creating ST programs with GX Works2, checking the operation with a programmable controller CPU module, and others.



## GENERIC TERMS AND ABBREVIATIONS

This guidebook uses the generic terms and abbreviations listed in the following table to discuss the software packages and programmable controller CPUs.

Corresponding module models are also listed if needed.

Generic terms and abbreviations	Description
ST	Stands for structured text.
FB	Stands for function block.
GX Developer	Generic product name for model names SW8D5C-GPPW-E, SW8D5C-GPPW-EA, SW8D5C-GPPW-EV and SW8D5C-GPPW-EVA.
Basic model QCPU	Generic term for Q00JCPU, Q00CPU and Q01CPU of function version B or later.
High Performance model QCPU	Generic term for Q02(H)CPU, Q06CPU, Q12HCPU and Q25HCPU.
Universal model QCPU	Generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDECPU, Q04UDHCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU and Q26UDEHCPU.
Process CPU	Generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU.
Redundant CPU	Generic term for Q12PRHCPU and Q25PRHCPU.
QCPU (Q mode)	Generic term for Q00J, Q00UJ, Q00, Q00U, Q01, Q01U, Q02(H), Q02PH, Q02U, Q03UD, Q03UDE, Q04UDH, Q04UDEH, Q06H, Q06PH, Q06UDH, Q06UDEH, Q10UDH, Q10UDEH, Q12H, Q12PH, Q12PRH, Q13UDH, Q13UDEH, Q20UDH, Q20UDEH, Q25H, Q25PH, Q25PRH, Q26UDH and Q26UDEHCPU.
LCPU	Generic term for L02CPU and L26CPU-BT.

The following explains the symbols and information used in this guidebook.

Symbol	Description	Example
Point	Gives the section-related knowledge and useful information.	
[     ]	Menu name of menu bar	[ Project ]
(     )	Icon of toolbar	
<<     >>	Tab name of dialog box	<<Select file>>
	Command button of dialog box	<input type="button" value="Jump"/> Button

## What is the ST language ... ?





The ST language is defined in the International Standard IEC61131-3 that stipulates the logic description system in open controllers.

The ST language supports operators, control syntaxes and functions to permit the following descriptions.

- Control syntaxes such as conditional sentence-dependent selective branch and repetitive sentence-based repetition
- Expressions using operators (\*, /, +, -, <, >, =, etc.)
- Call of user-defined function blocks (FB)
- Call of functions (MELSEC functions, IEC functions)
- Description of comments

## What are the features ... ?

The following introduces the main features of ST programs in the QCPU (Q mode)/LCPU.

-  **Design efficiency improved by defining processings as parts.**  
With often used processings defined as parts in the form of function blocks (FB) in ST language, they can be used in necessary areas of each program.  
This not only enhances the efficiency of program development but also reduces program mistakes, improving program quality.
-  **Program change during system operation (online change).**  
Part of a running program can be changed without the programmable controller CPU being stopped.
-  **Connection with other language programs.**  
Since other languages than the ST are also supported, the language adequate for processing can be used to increase the efficiency of program development.  
For example, write sequence control in a ladder program, and operation processing in ST language.  
Multiple languages support widespread application under optimum control.
-  **A wealth of functions available.**  
The MELSEC functions compatible with various common instructions for the QCPU (Q mode)/LCPU and the IEC functions defined in IEC61131-3 are available for ST programs.

# MEMO

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

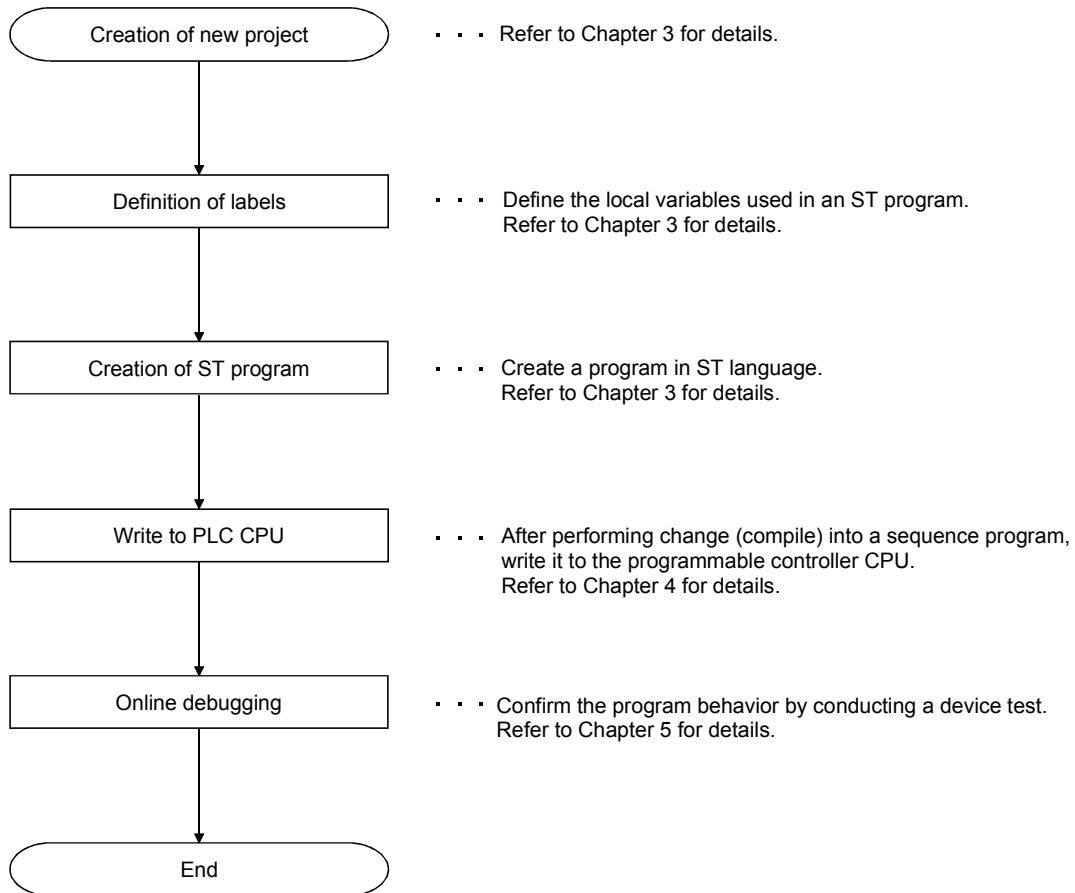
---

---

---

The following flowchart indicates the basic procedure from ST program creation to online debugging.

In the following example, programming was performed using only an ST program.



### REMARK

For details of each operation, refer to the "GX Developer Operating Manual" given in Related Manuals.

MEMO

2

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

Chapter 3 explains general basic operations from the input to convert (compile) of an ST program. The following items will be explained in this chapter.

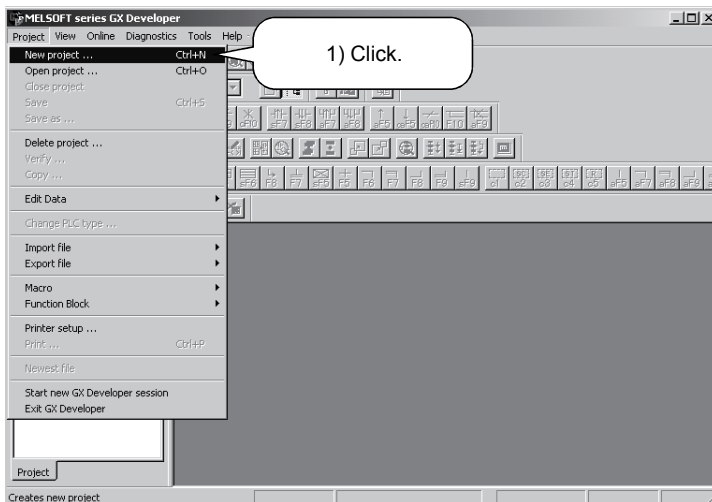
- ☞ Creating a new ST project.
- ☞ Defining the labels to be used in an ST program.
- ☞ Creating an ST program.
- ☞ Converting (compiling) the created ST program into an executable sequence program.
- ☞ Correcting the program if a convert (compile) error occurs.

## Creating a new ST project.

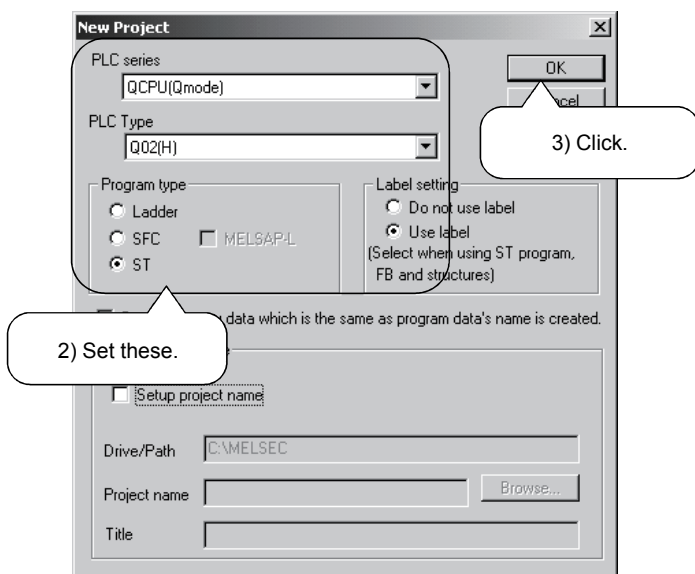
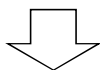
3

### ■ Creating a new ST project

The operation method to create a new project will be explained.



1) Click [Project] → [New project] in the menu.



2) Enter as follows.

- PLC series : QCPU (Q mode)
- PLC Type : Q02 (H)
- Label setting : Use label
- Program type : ST

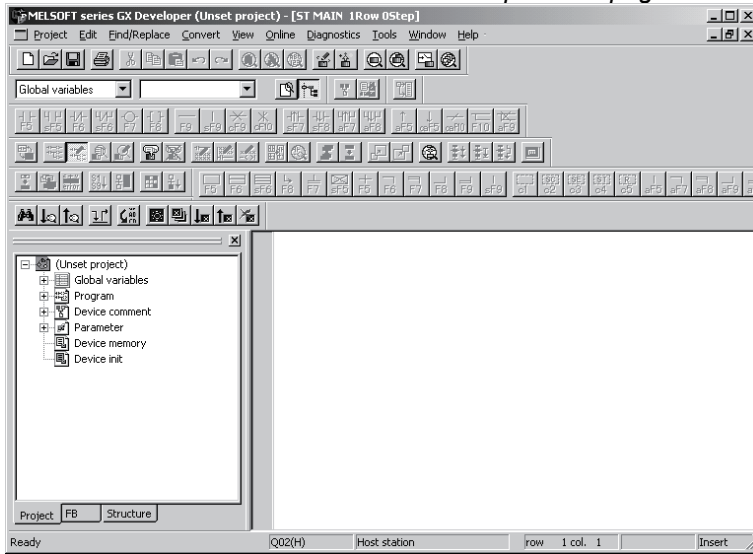
3) Click the **OK** button



To next page



↓  
From previous page



- 4) A new ST project is created.
- \* The ST edit screen opens and an ST program can be entered.

**REMARK**

Here, "Q02(H)" is set as the PLC type.  
There are the following programmable controller CPU types that are applicable to ST programs.

Basic model QCPU	High Performance model QCPU	Universal model QCPU	Process CPU	Redundant CPU	LCPU
Q00JCPU	Q02CPU	Q00UJCPU	Q02PHCPU	Q12PRHCPU	L02CPU
Q00CPU	Q02HCPU	Q00UCPU	Q06PHCPU	Q25PRHCPU	L26CPU-BT
Q01CPU	Q06HCPU	Q01UCPU	Q12PHCPU		
	Q12HCPU	Q02UCPU	Q25PHCPU		
	Q25HCPU	Q03UDCPU			
		Q03UDECPU			
		Q04UDHCPU			
		Q04UDEHCPU			
		Q06UDHCPU			
		Q06UDEHCPU			
		Q10UDHCPU			
		Q10UDEHCPU			
		Q13UDHCPU			
		Q13UDEHCPU			
		Q20UDHCPU			
		Q20UDEHCPU			
		Q26UDHCPU			
		Q26UDEHCPU			

## Defining the labels

What does "defining the labels" mean?

To use labels, variables used as labels must be clarified. This is called "defining the labels". If a program that uses undefined labels is converted (compiled), an error occurs and a sequence program cannot be created.

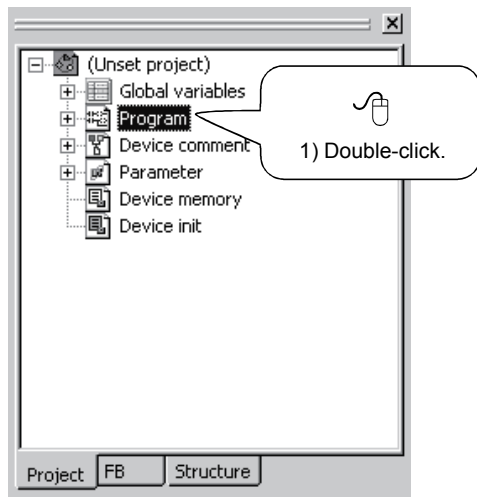
There are two different labels; global variables and local variables. The global variables can be used in the whole project. The local variables can be used in only the program where the labels have been defined.

Here, the local variables used in the program example that will be entered later will be actually defined.

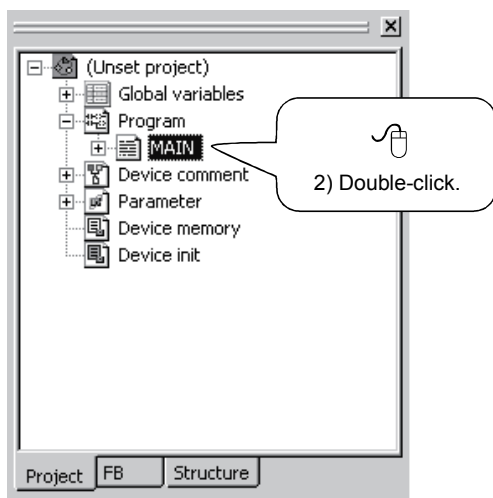
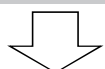
### ■ Displaying the Local variables setting screen

The operation method to define local variables will be explained.

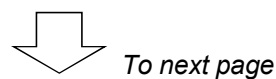
For the global variables, refer to the "GX Developer Operating Manual".

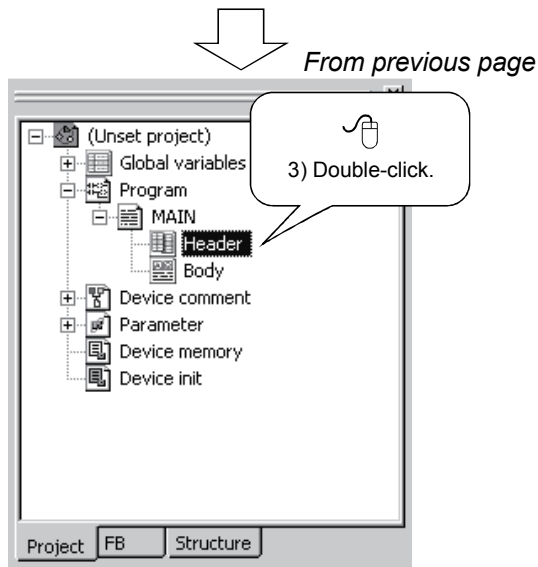


1) Double-click "Program" in the <<Project>> tab.

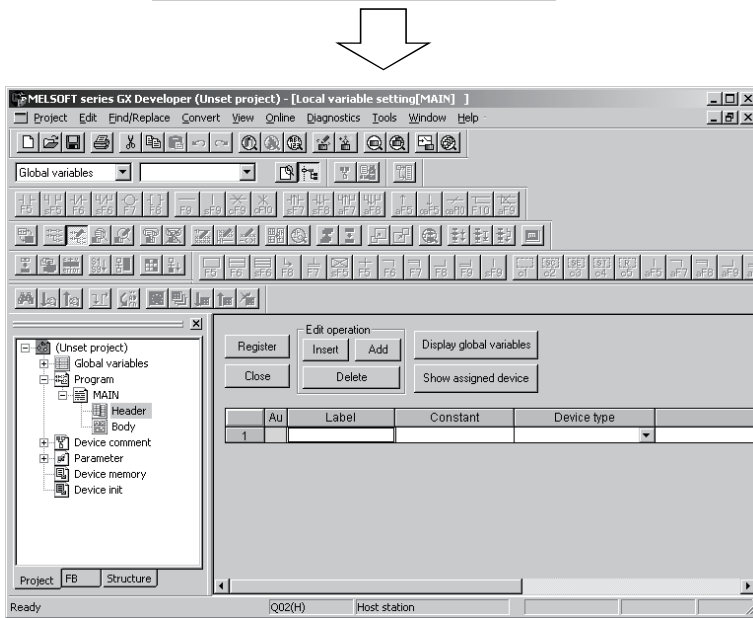


2) Double-click "MAIN".





3) Double-click "Header".



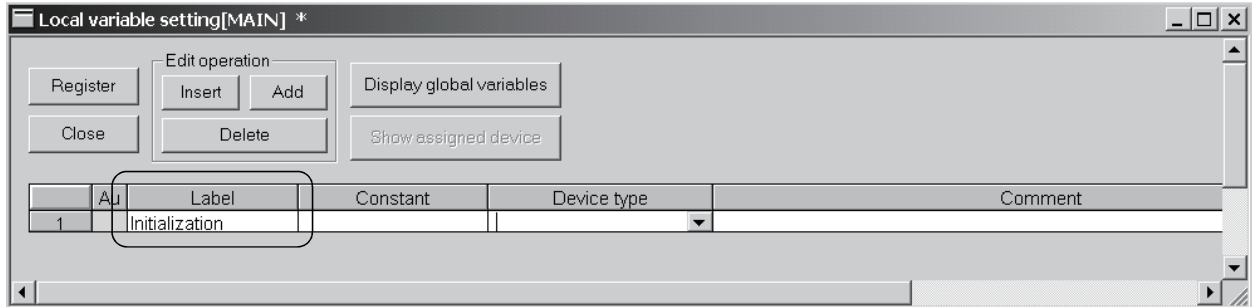
4) The Local variables setting screen is displayed.

## ■ Setting the local variables (Header)

1) Enter a label name.

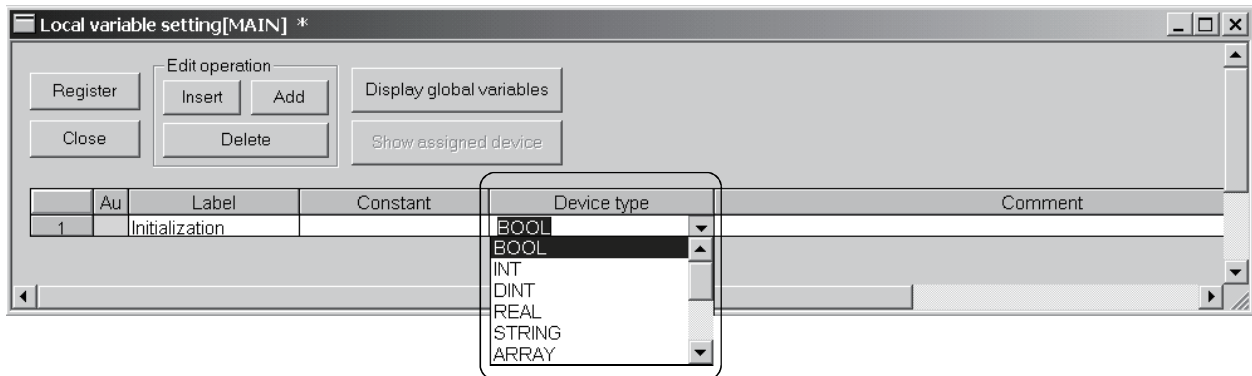
Enter a label name within 16 characters. The character strings that cannot be used as a label name are reserved words and actual devices. Enter other labels.

\* For the reserved words, refer to the "GX Developer Operating Manual".



2) Enter a device type.

Enter it directly or make selection from the list box.

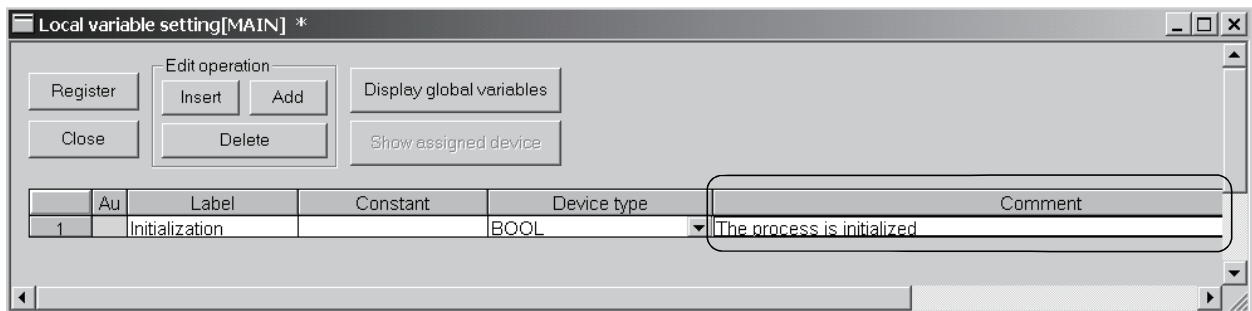


3) Enter a comment into the label.

Enter it within 64 characters.

The comment can be displayed in the tool tip format of the label information.

\* For the label information, refer to "CHAPTER 7 INTRODUCTION TO USEFUL FUNCTIONS FOR ST PROGRAM EDITING" or "GX Developer Operating Manual (Structured Text)".



To next page

↓  
From previous page

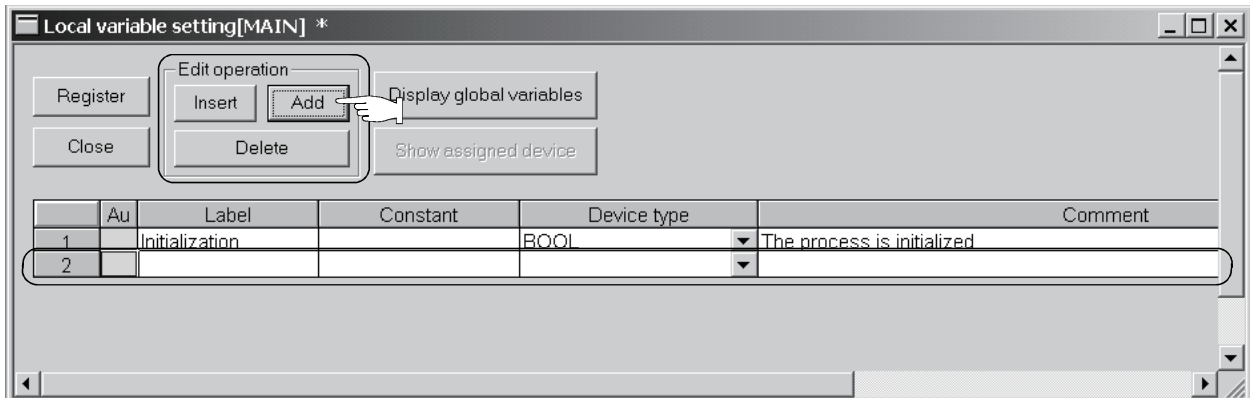
4) When entering labels continuously, click the **Insert** or **Add** button under Edit operation to add a line.

The buttons under Edit operation have the following functions.

**Insert** button ... Inserts a blank line into the current cell position.

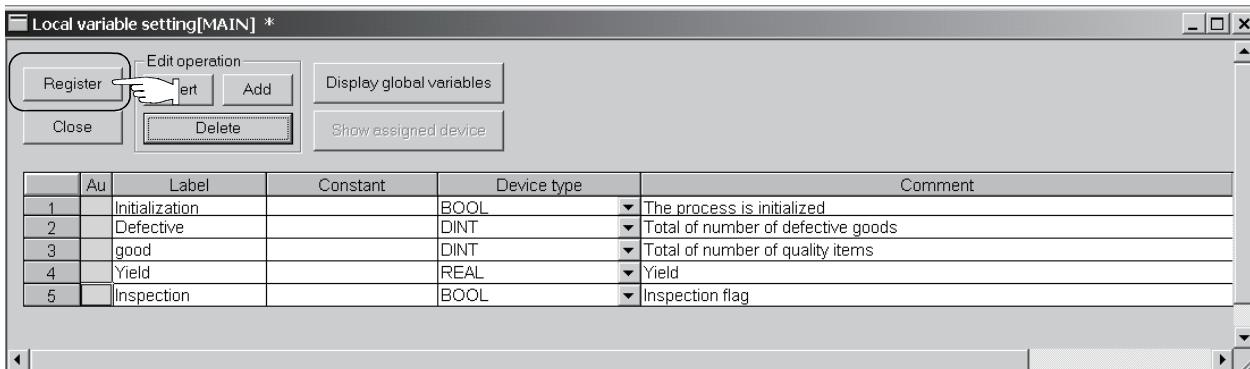
**Add** button ..... Inserts a blank line into the place one line below the current cell position.

**Delete** button ... Deletes one line in the current cell position.

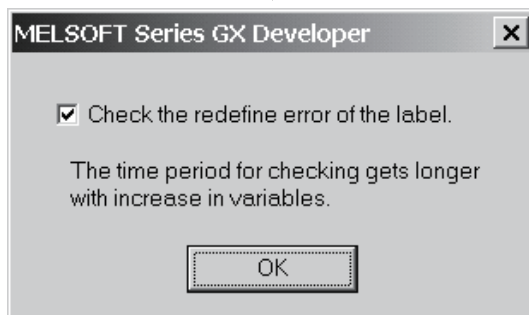


↓

5) After input is complete, click the **Register** button.



↓



Click the **OK** button.

↓  
To next page

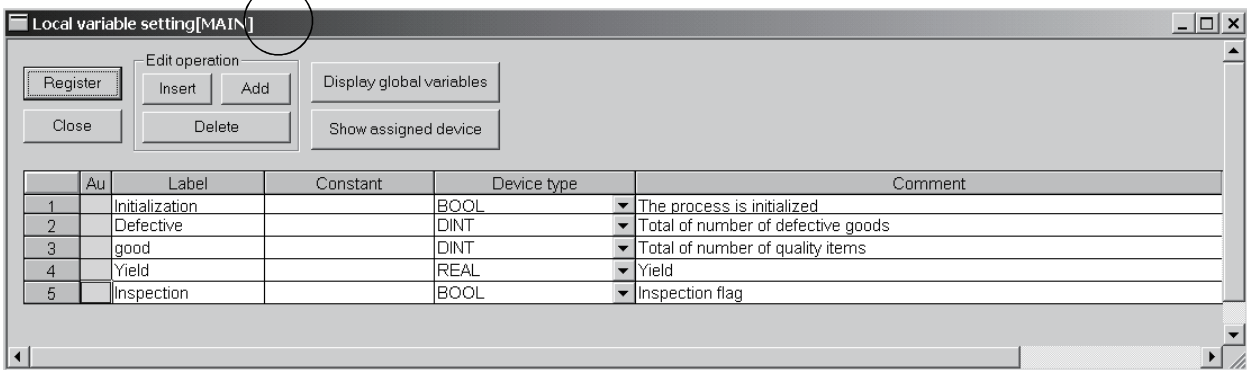
From previous page



Registration is completed.

Click the **OK** button.

When registration is made, "\*" displayed on the title bar disappears.



**REMARK**

For details of the local variables, refer to the "GX Developer Operating Manual" given in Related Manuals.

## Entering a program

*What should be noted during input?*

A program can be input freely in text format using the ST edit screen. Note the following points during input.

- Use a space, `Tab` key or `Enter` key to enter a blank.
- When the defined label, control syntax or comment is input, the character color changes.  
If it does not change, the possible cause is an input mistake or undefined label.

Now, actually input a program in List-1.

List-1.

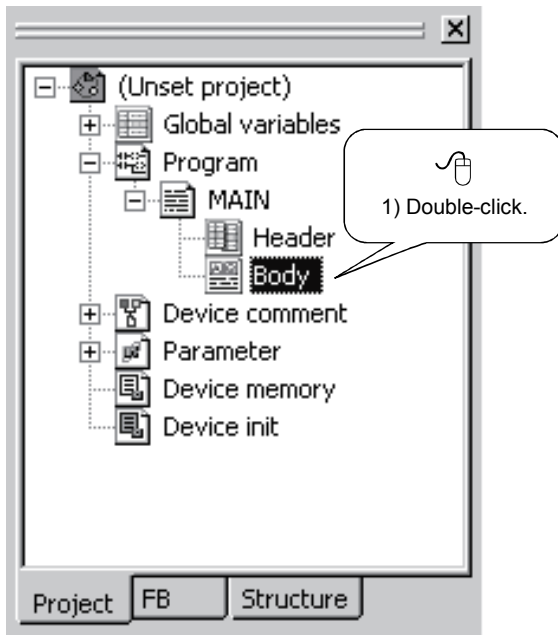
```
IF      Initialization THEN
      good := 0; Defective := 0; Yield := 0.0;

ELSE
      IF Inspection THEN
            good := good + 1;

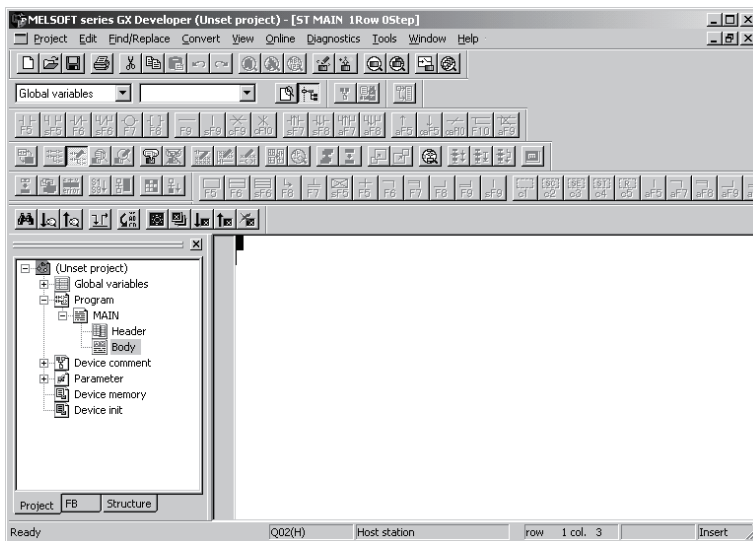
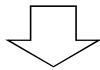
      ELSE
            Defective := Defective + 1;
      END_IF;

      Yield := DINT_TO_REAL(good)/DINT_TO_REAL(good + Defective);
END_IF;
```

## ■ Displaying the ST edit screen

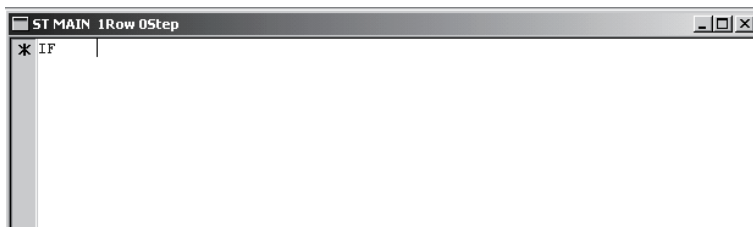


1) Double-click "Body" in the <<Project>> tab.



2) The ST edit screen is displayed.

## ■ Entering the characters



Enter "IF".

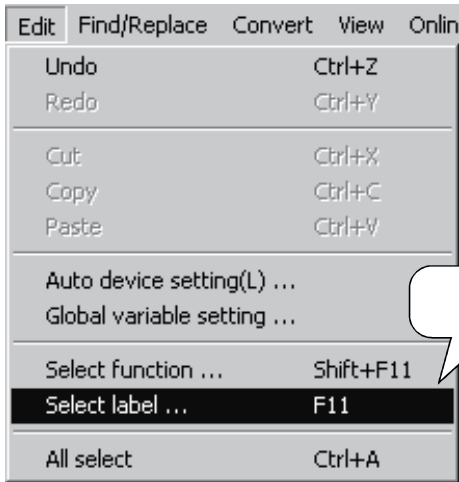
\* If entered in lower case, a control syntax is converted automatically into upper case.



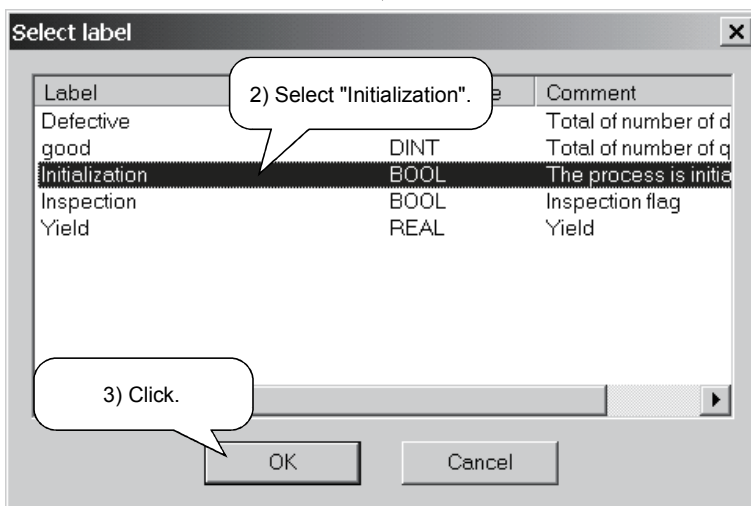
## ■ Entering a label

To enter a label, a label name may be entered directly or the label selection function be used.  
 To use the label selection function, labels must have been entered in advance.

Here, the input method using the label selection function will be explained.

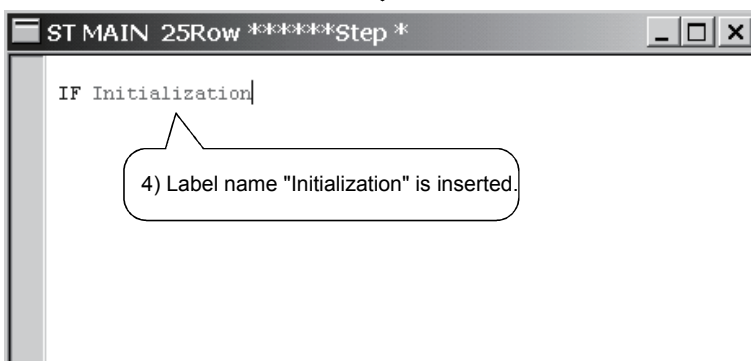


1) Click [Edit] → [Select label] in the menu.



2) Select the label to be entered.

3) Click the **OK** button.

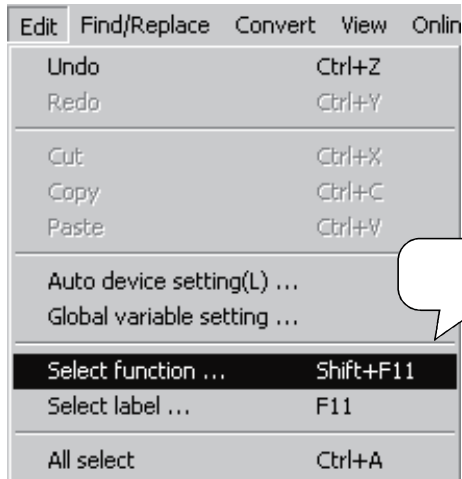


4) The label is inserted.

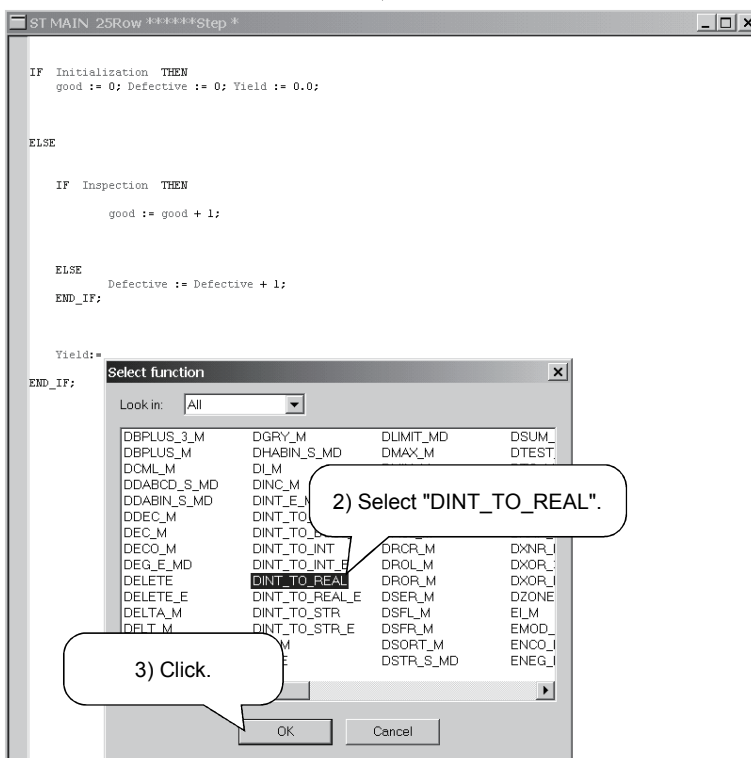
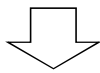
## ■ Entering a function

Enter a function in upper case. To enter, a function may be entered directly or the function selection function be used.

Here, the input method using the function selection function will be explained.

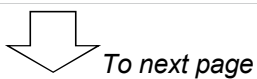


1) Click [Edit]→[Select function] in the menu.

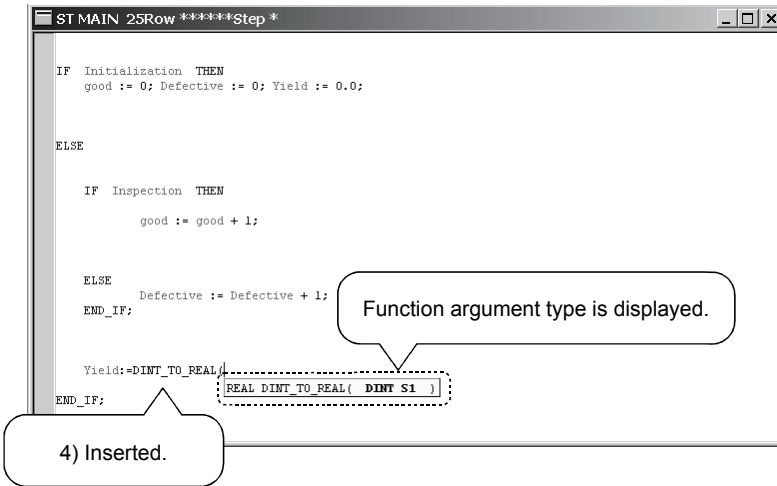


2) Select the label to be entered.

3) Click the **OK** button.



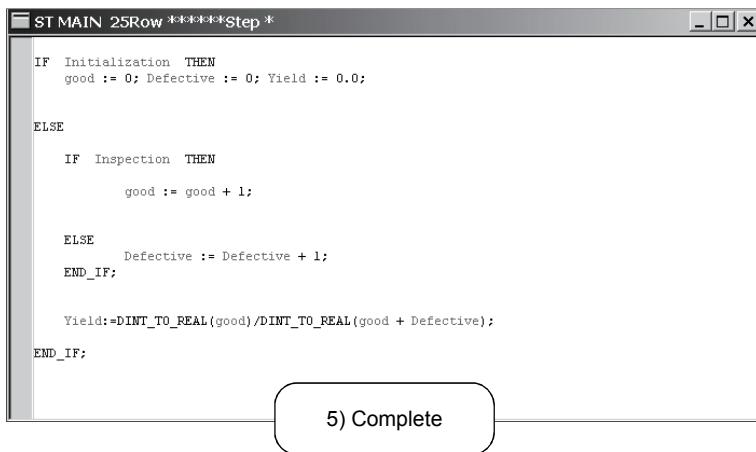
From previous page



4) The function name is inserted.

\* The function argument type is displayed in the tool tip format.

↓

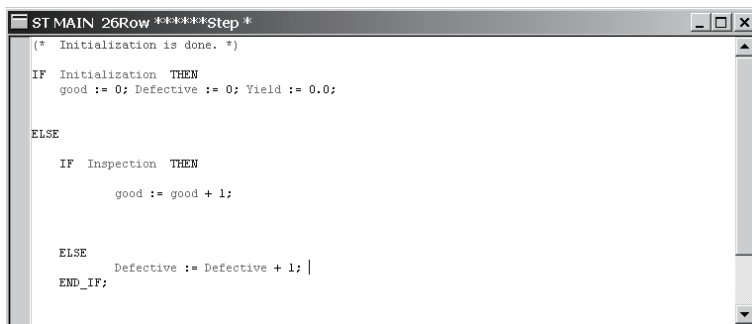


5) Refer to the function argument type displayed in the tool tip format, and enter the argument to complete the entry.

## ■ Entering a comment

Comments do not affect the program behavior at all. When program processings are described as comments, they give the at-a-glance pictures of the processings.

First, enter a comment on Line 1 of the program.



\* Enter a comment by enclosing it with "(\*" that represents the beginning of the comment and "\*)" that represents an end.

To next page

↓  
From previous page

```

ST MAIN 25Row *****Step *
(* Initialization is done. *)
IF Initialization THEN
  good := 0; Defective := 0; Yield := 0.0;
(* The normal operation is processed. *)
ELSE
  (* Is the inspection passing? *)
  IF Inspection THEN
    good := good + 1; (* The number of non-defective articles is added. *)
    (* The inspection is failing. *)
  ELSE
    Defective := Defective + 1; (* The number of defective goods is added. *)
  END_IF;
  (* The yield is calculated. *)
  Yield:=DINT_TO_REAL(good)/DINT_TO_REAL(good + Defective);
END_IF;
    
```

Refer to the example given on the left and enter comments. (List-2)

This completes program input.

### Point

- Display of label information  
The label information can be displayed in the tool tip format.  
Operation : Place the mouse pointer in the label position.  
Display : Label name -> Label type -> Label comment -> Device \*  
\*: The device is displayed after convert (compile) is performed.

```

ST MAIN 25Row 143Step
(* Initialization is done. *)
IF Initialization THEN
  good := 0; Defective := 0; Yield := 0.0;
(* The normal operation is processed. *)|
ELSE
  (* Is the inspection passing? *)
  IF Inspection THEN
    good := good + 1; (* The number of non-defective articles is
    good->LOCAL->Total of number of quality items->D12284
    (* The inspection is failing. *)
  ELSE
    Defective := Defective + 1; (* The number of defective goods is add
    
```

Label information is displayed.

- Change of display color  
Control syntax, comment and label character string colors, ST edit screen background color, etc. can be changed.  
Operation: Choose [Tools] → [Change display color] in the menu.
- Setting of auto indent  
Indentation at the time when the **Enter** key is pressed and the Tab width at the time when the **Tab** key is pressed can be set.  
Operation: Choose [Tools] → [ST editor settings] in the menu.  
For details, refer to the "GX Developer Operating Manual (Structured Text)".

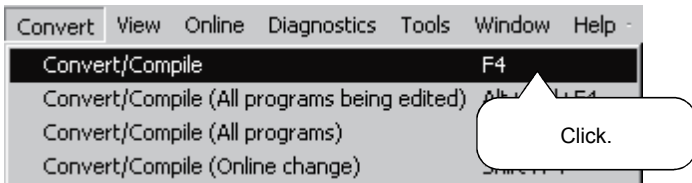
## Converting (compiling) the ST program

*What is convert (compile)?*

Changing the program created on the ST edit screen into a sequence program that can be executed by the programmable controller CPU is called convert (compile).

### ■ Performing convert (compile)

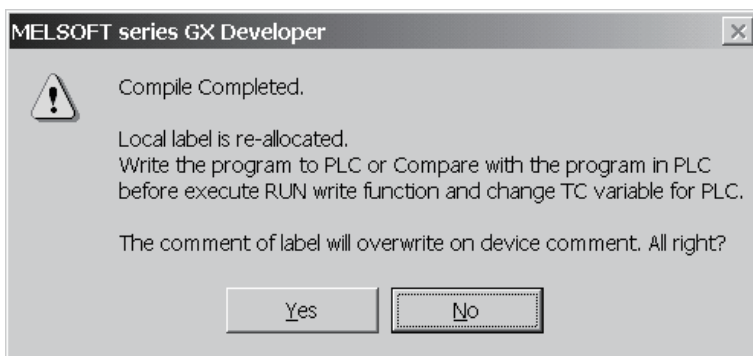
The convert (compile) operation method will be explained using the created program.



1) Click [Convert] → [Convert/Compile] in the menu.

#### (1) At normal completion

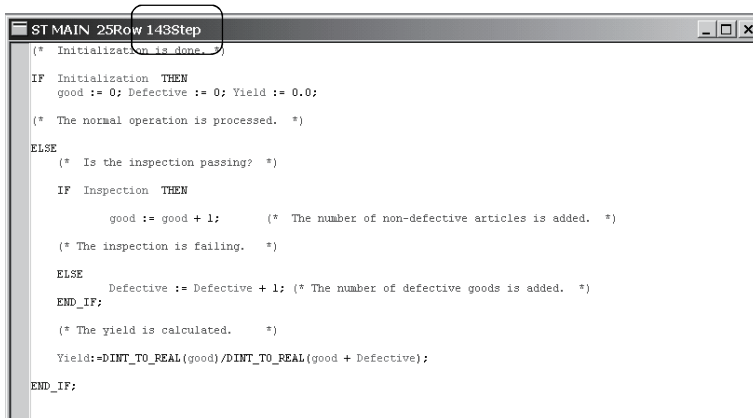
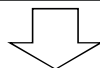
The following message is displayed.



Convert (compile) is completed.

Since the confirmation screen shown on the left is displayed, click the **No** button.

If the **Yes** button is clicked, the "Comment data to be referred to (comment by program) does not exist" message may be displayed.



\* At normal completion of convert (compile), the number of steps is displayed on the title bar.

(2) When error occurs

The Compile error (Detail) dialog is displayed.

Now, actually see the debugging operation at occurrence of a compile error.

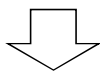
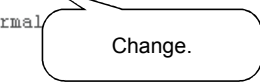
1) Change the program so that a compile error will occur.

```

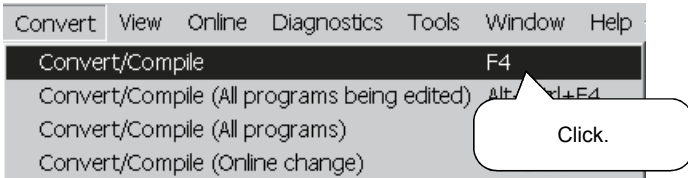
ST MAIN 25Row 143Step
(* Initialization is done. *)
IF Initialization THEN
* good := 0.0; Defective :=
(* The normal
ELSE
/* To the inspection part
    
```

Change Line 3 in List-2.

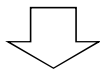
RYOUHIN := 0; → RYOUHIN := 0. 0;



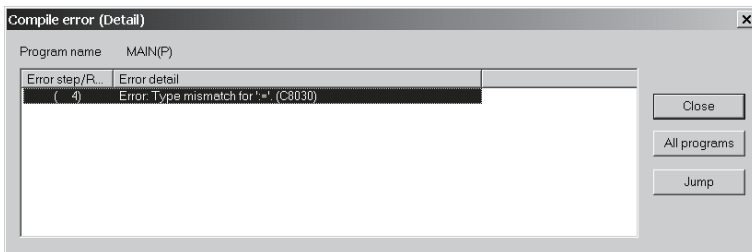
2) Perform convert (compile).



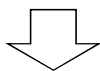
Click [Convert] → [Convert/Compile] in the menu.



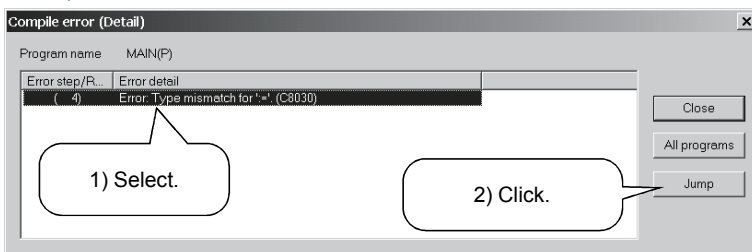
3) A compile error occurs and the dialog is displayed.



Confirm the error step/line and error definition.

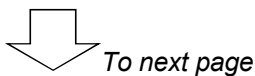


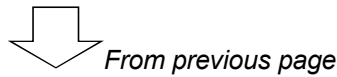
4) Confirm the line on which the error has occurred.



1) Select the error definition with the mouse.

2) Click the **Jump** button.





5) Track down the cause and correct the faulty part.

The error indication mark is displayed on the ST edit screen.

Confirm the error definition and program contents, and correct the program.

```

ST MAIN 25Row *****Step *
(* Initialization is done. *)
IF Initialization THEN
good := 0.0; Defective := 0; Yield := 0.0;
The normal operation is processed. *)
Error indication mark *)
    good := good + 1; (* The number of non-defective articles is added. *)
(* The inspection is failing. *)
ELSE
Defective := Defective + 1; (* The number of defective goods is added. *)
END_IF;
(* The yield is calculated. *)
Yield:=DINT_TO_REAL(good)/DINT_TO_REAL(good + Defective);
END_IF;
    
```

Correct Line 3 in List-2.

good := 0.0; → good := 0;

Click [Convert] → [Convert/Compile] in the menu.



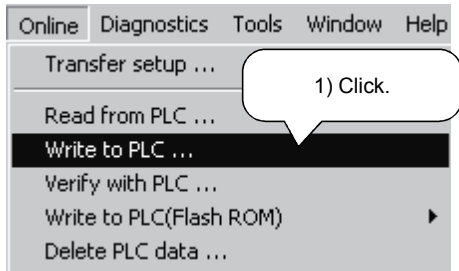
The error location and actually corrected part may be different. Identify the faulty part from the error definition displayed in the "Compile error (Detail)" dialog and the program contents of the line where the error indication mark is displayed.

Chapter 4 explains the procedure to write the converted (compiled) sequence program to the programmable controller CPU and the procedure to read the sequence program from the programmable controller CPU.

## ■ Performing write to PLC

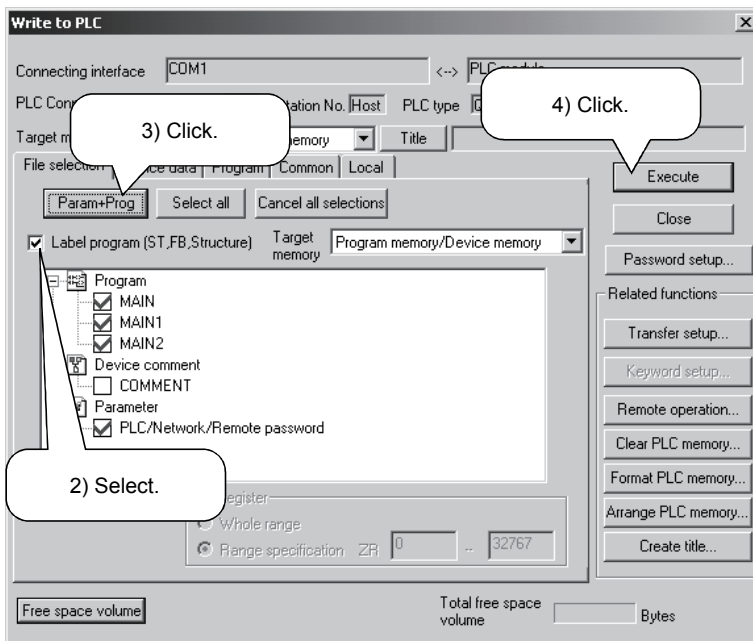
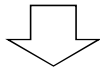
The operation method for write to PLC will be explained.

Display the Write to PLC dialog and write the program and parameters to the programmable controller CPU.



\* When performing write to PLC, put the programmable controller CPU in a STOP status.

1) Click [Online] → [Write to PLC] in the menu.

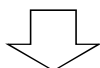


2) Select the "Label program (ST, FB, structure)" check button in the <<File selection>> tab.

\* When the check button is not selected, only the actual program is written.

3) Click "Param + Prog".

4) Click the **Execute** button.



\* Reset the programmable controller CPU and put it in a RUN status.

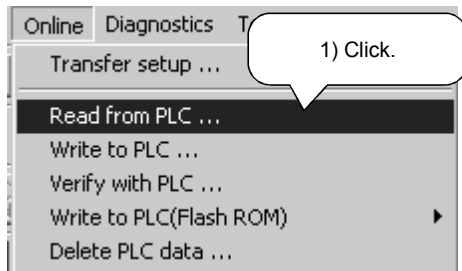
If an error occurs, choose [Diagnostics] → [PLC diagnostics] in the menu of GX Developer, and confirm the error definition.



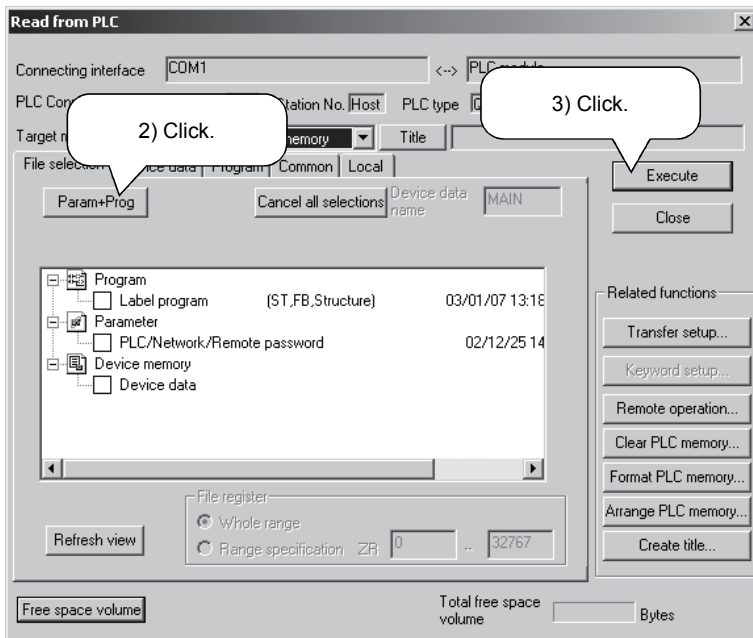
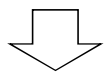
## ■ Performing read from PLC

The operation method for read from PLC will be explained.

Display the Read from PLC dialog and read the program and parameters from the programmable controller CPU.

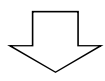


1) Click [Online] - [Read from PLC] in the menu.



2) Click "Param + Prog" in the <<File selection>> tab.

3) Click the [Execute] button.



If an error occurs, choose [Diagnostics] - [PLC diagnostics] in the menu of GX Developer, and confirm the error definition.



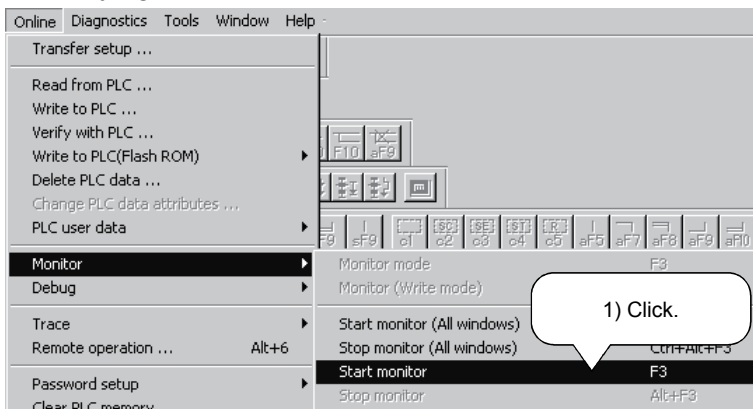
Chapter 5 explains the online debugging operation of the sequence program written to the programmable controller CPU using the monitor function and device test function.

- ☞ Monitoring the sequence program.
- ☞ Changing the bit device value and conducting a device test.
- ☞ Changing part of the sequence program and writing it to the programmable controller CPU in RUN status.

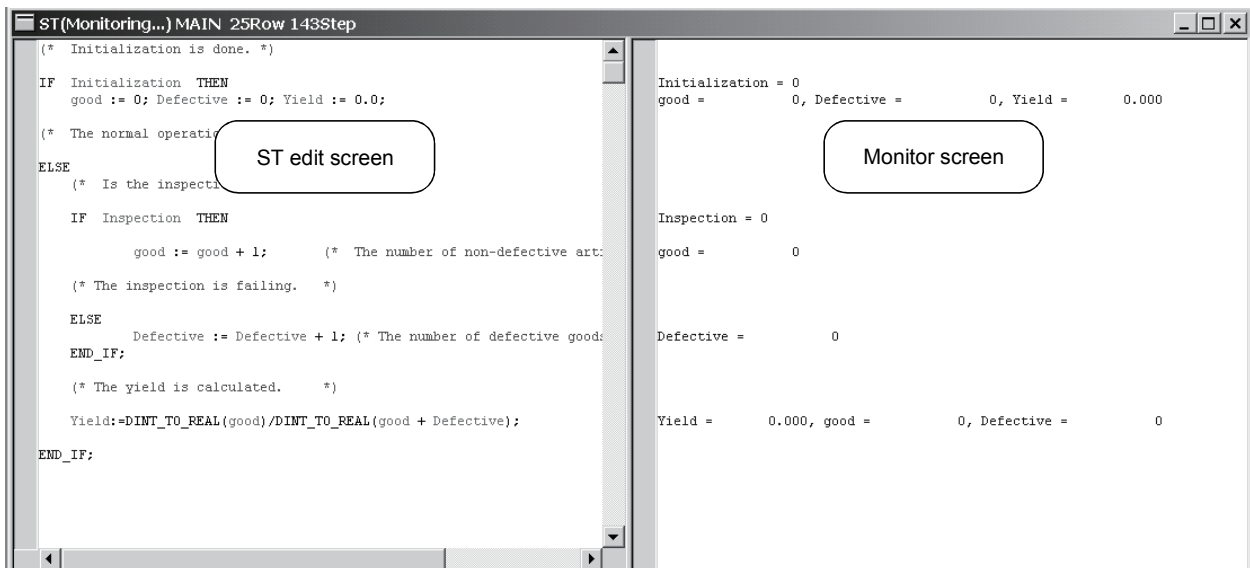
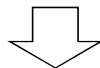
## 5.1 Monitoring the Sequence Program

This section explains the operation method to monitor the sequence program.

### ■ Displaying the monitor screen



1) Click [Online] → [Monitor] → [Start monitor] in the menu.



The labels displayed on the ST edit screen are displayed on the same lines of the monitor screen.

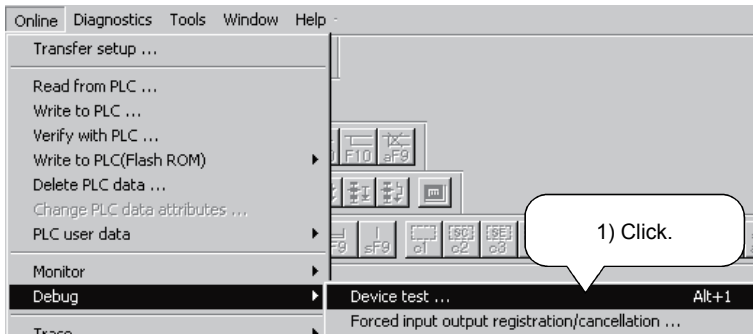
## 5.2 Device Test

The value of the label (bit device/word device) in the programmable controller CPU can be changed directly.

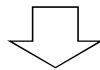
Here, the bit device value is changed to confirm the program behavior.

### ■ Confirming the program behavior

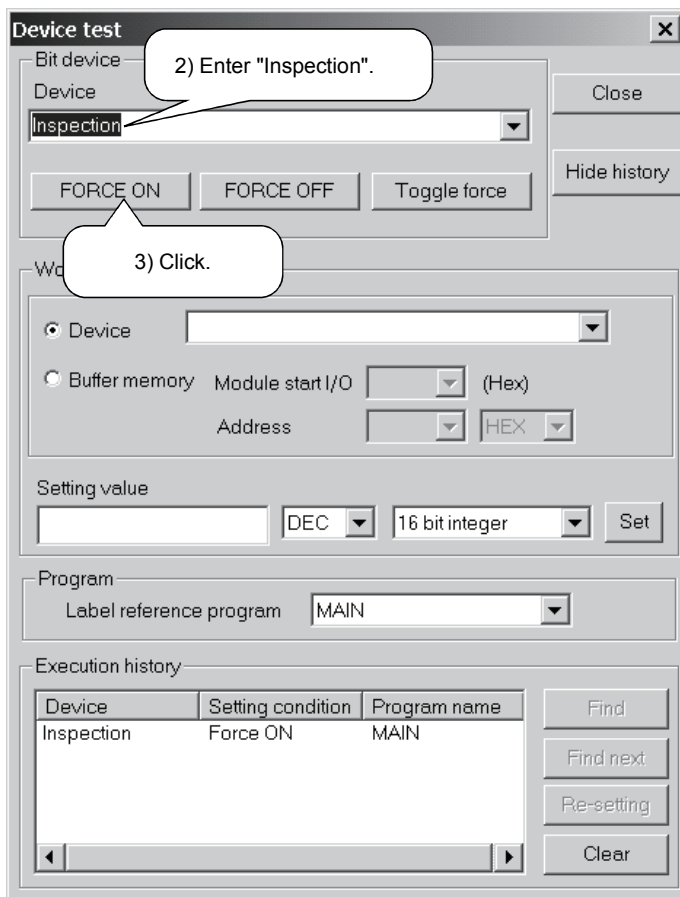
The operation to change the bit device value will be explained.



1) Click [Online] → [Debug] → [Device test] in the menu.

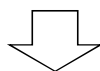


Forcibly turn ON the label "Inspection" that represents the bit device.

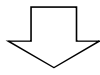


2) Input "Inspection" into the bit device.

3) Click the **FORCE ON** button.

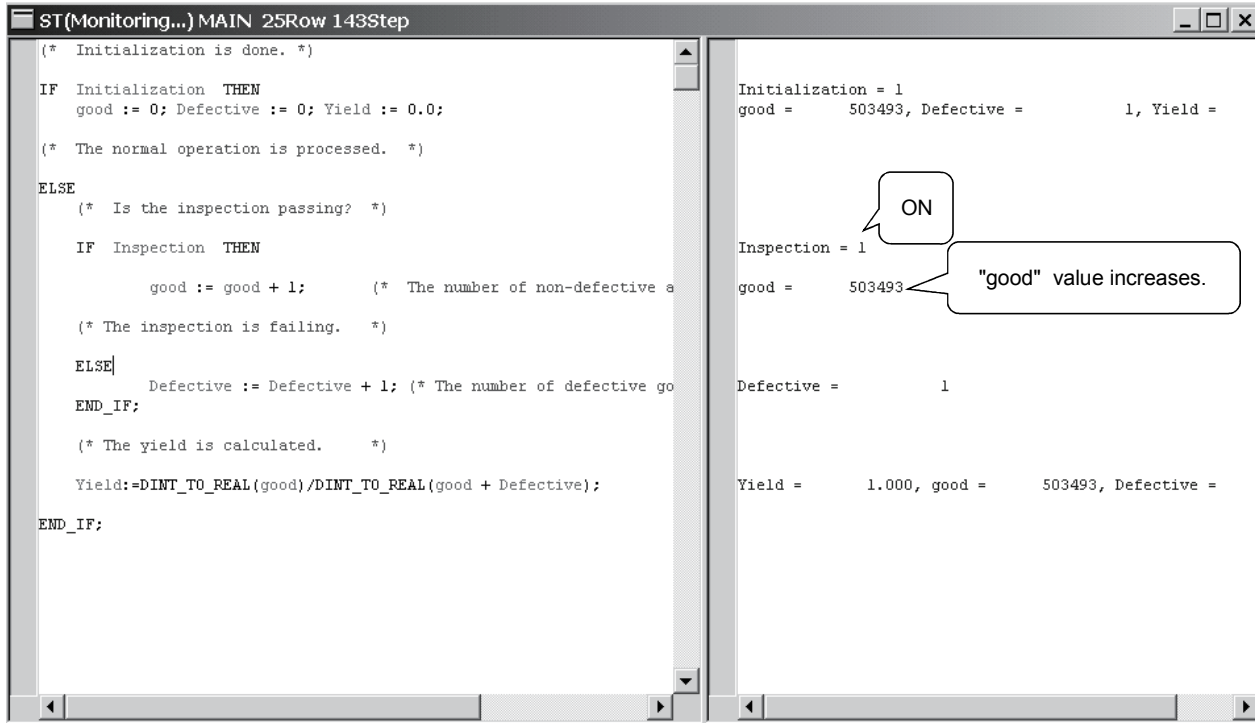


To next page



From previous page

Make confirmation on the monitor screen.



Also change the other label values and confirm the program behavior.



GX Developer supports the following debug functions for the programs created in ST language.

- Break execution : debugs programs by halting the program execution at the location specified by break point.
- 1 line execution : debugs programs by halting the program execution line-by-line.

For details, refer to the relevant section in GX Developer Operating Manual (Structured Text).

## 5.3 Online Change

When the programmable controller CPU is in a RUN status, part of the sequence program can be changed.

This is called online change.

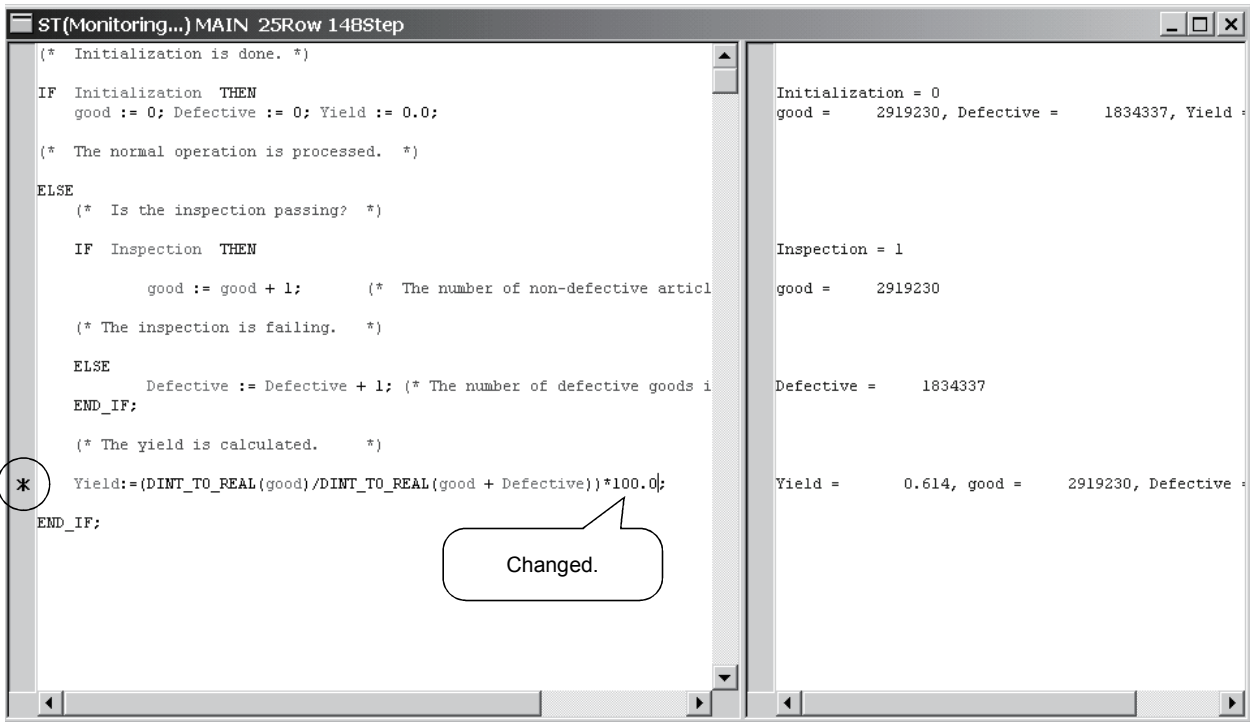
Actually change part of the sequence program and perform online change.

### ■ Changing part of the program and performing online change

Change the calculation expression of "Yield" and perform online change.

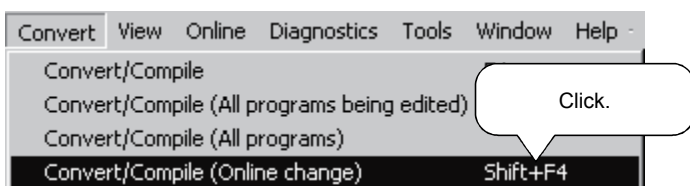
1) Change part of the program.

```
Yield := DINT_TO_REAL (good)/DINT_TO_REAL (good + Defective);
                                     ↓
Yield := (DINT_TO_REAL (good)/DINT_TO_REAL (good + Defective))*100.0;
```



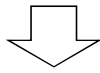
"\*" indicating the line to be online changed is displayed on the indicator bar of the target line.

2) Execute online change.



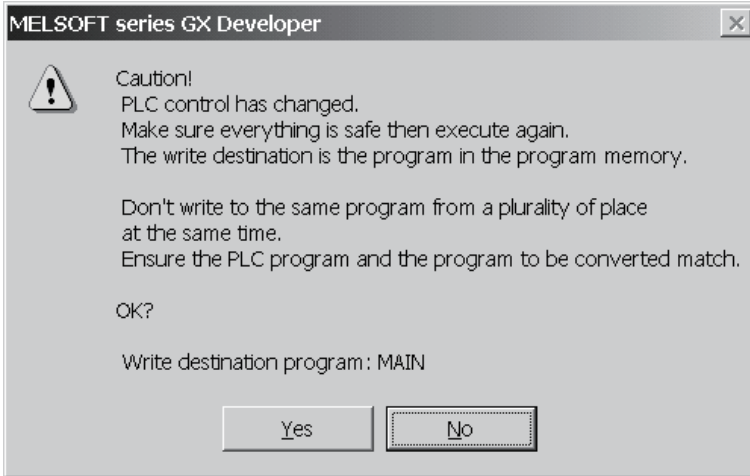
Click [Convert] → [Convert/Compile (Online change)] in the menu.



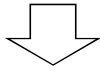


*From previous page*

3) The confirmation message is displayed.



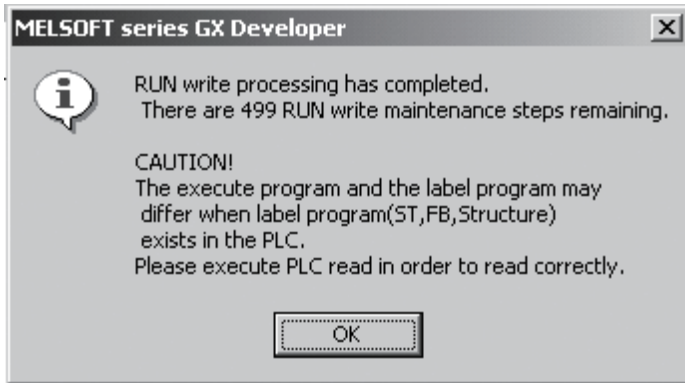
Click the **Yes** button.



*To next page*

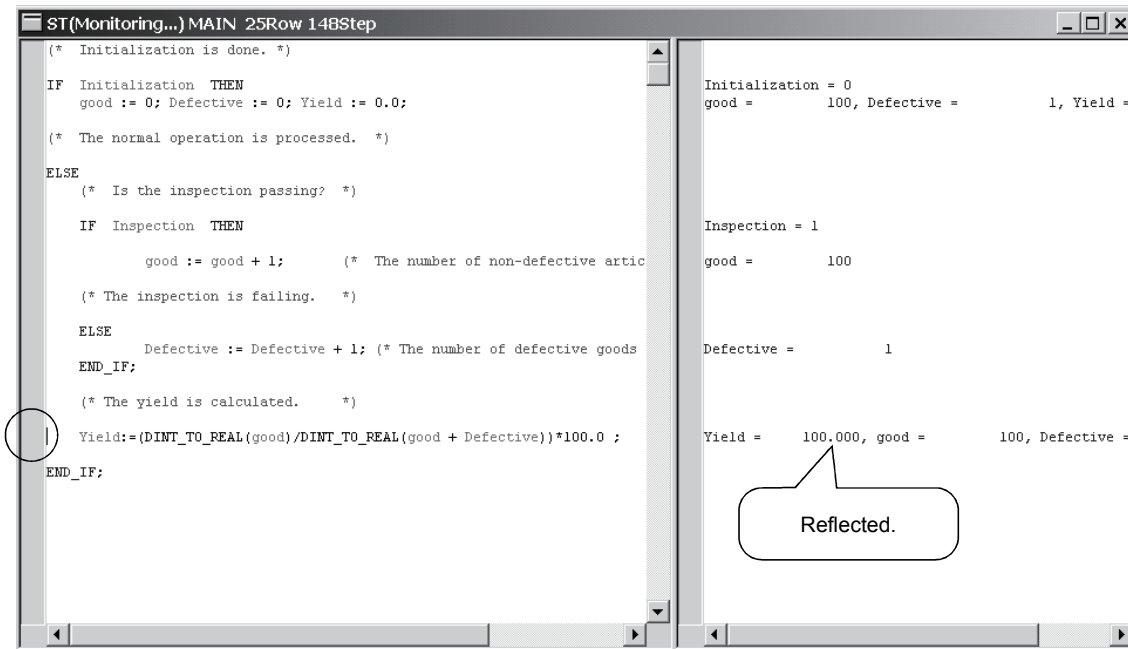
↓  
From previous page

4) Online change is completed.



Click the **OK** button.

↓



"\*" that indicates the online change target line disappears.

On the monitor screen, confirm that the present value of "Yield" has changed.





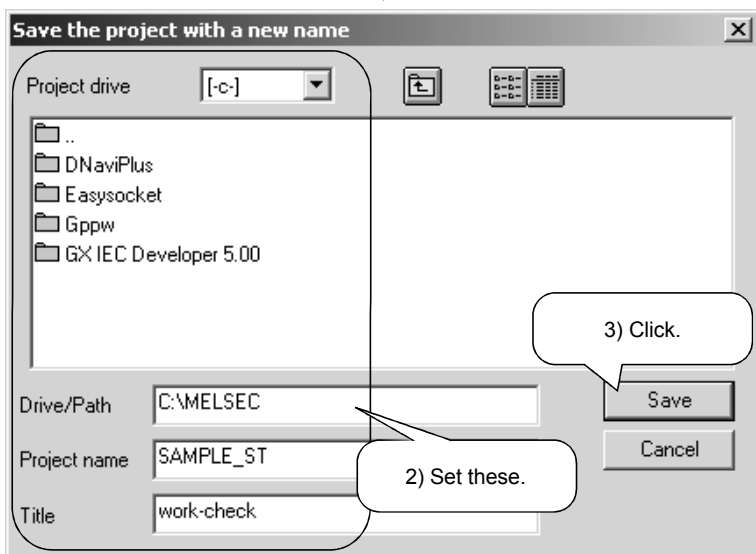
In Chapter 6, the completed project is saved with a name.

## ■ Saving the project

Save the created project with a name.



1) Click [Project] → [Save as] in the menu.



2) Input as follows.  
• Drive/Path : C:\MELSEC  
• Project name : SAMPLE\_ST  
• Title : work-check

3) Click the **Save** button.

The program created this time was saved as described below.

Drive/Path : C:\MELSEC  
Project name : SAMPLE\_ST  
Title :work-check

This completes a series of operations from the creation of the new ST project to the input and online debugging of the program to the storage of the project. Fully understand the operations performed until now, and proceed to Chapter 8.



This chapter introduces useful functions for editing ST programs.

For more information, refer to the "GX Developer Operating Manual (Structured Text)".

## (1) Window division

Every time it is desired to confirm the contents midway through editing of a large program, it is troublesome to scroll the screen to see the program ...

At such a time, use "Window division".

Choosing [Window] → [Divide into two] in the menu displays the screen in vertically divided windows. The divided windows can be scrolled/edited individually.

## (2) Bookmark

When it is desired to jump to a specific line, it is troublesome to search the program from the beginning ...

At such a time, use "Bookmark".

Preset the bookmark by choosing [Find/Replace] → [Bookmark setting/release] or [Find/Replace] → [Find] → "Set bookmark" in the menu.

By choosing [Find/Replace] → [Bookmark list] in the menu, any line can be selected from the Bookmark list dialog to make a jump to that line.

## (3) Display of label information

It is desired to know the device assigned to the label ...

At such a time, use "Label information".

When the mouse pointer is placed on the label,

Label name -> Label type -> Label comment -> Device
---

is displayed in the tool tip format for at-a-glance confirmation of the contents.

Note: · Applicable to converted (compiled) programs.

- Confirmation can also be made by activating "Show assigned device" on the Local variables setting screen.

## (4) Select function

It is desired to input the function whose name has been forgotten ...

At such a time, use "Select function".

Choosing [Edit] → [Select function] displays the Select function dialog to allow the function name to be selected. Also, since the function argument type is displayed in the tool tip format when the function is inserted, the argument can be input with reference to that argument type.

## (5) Change of display color and font

It is difficult to differentiate between the character strings on the edit screen since they have the same color, or it is desired to change the character size ...

At such times, use "Change display color" or "Font".






Choosing [Tools] → [Change display color] in the menu displays the Change display color dialog to allow the comment, control syntax, character, label and background colors to be selected. Changing the display color improves readability.

Choosing [Tools] → [Font] in the menu displays the Font dialog to allow the font type, style and size to be selected. Making easy-to-use setting improves operability.



Chapters 1 to 7 explained the basic operation methods and functions for creating ST programs. In Chapter 8, create a function block (FB) in ST language and paste it to a ladder program to create a program to be used.

The main items to be explained in this chapter are as follows.

-  Adding a new FB.
-  Defining FB variables.
-  Creating an ST program.
-  Creating an FB in ST language.
-  Using the ST-written FB in a main program.

## 8.1 Creating an FB

*What is a function block (FB)?*

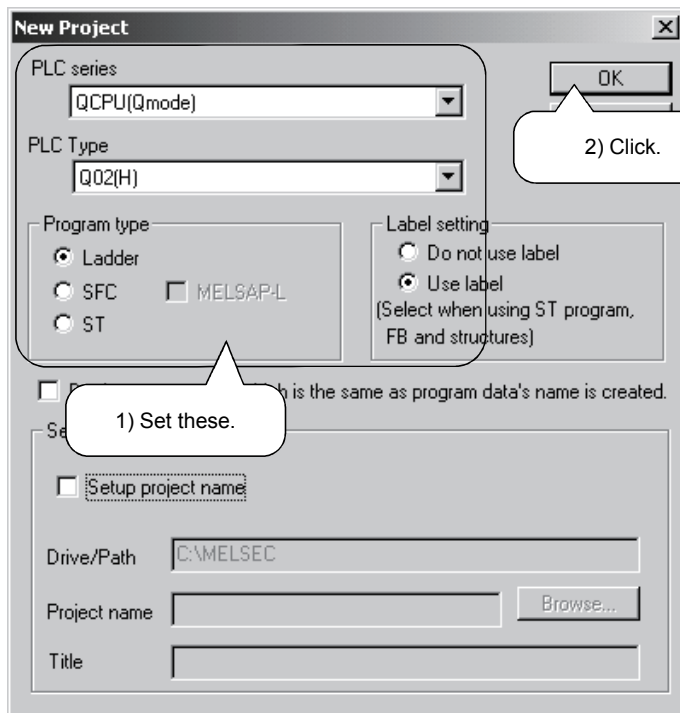
An FB, where often used processings are predefined as parts, can be used in the necessary area of each program.

### Creating a new project

#### ■ Creating a new project

The creation method of a new project to create a main program in ladder format will be explained.

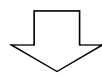
Click [Project] → [New project] in the menu to display the New project dialog.



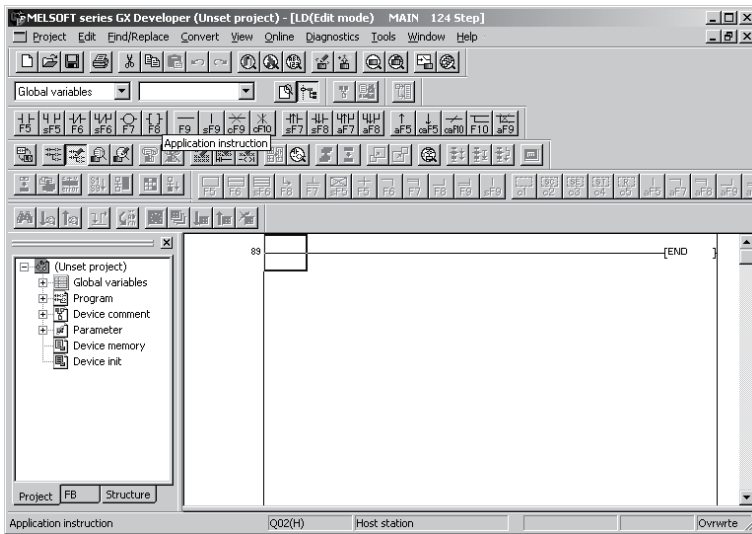
1) Enter as follows.

- PLC series :QCPU (Q mode)
- PLC type :Q02(H)
- Label setting :Use label
- Program type :Ladder

2) Click the **OK** button.

 To next page

↓  
From previous page

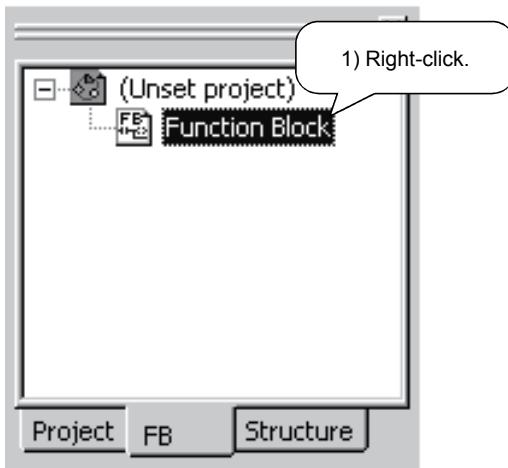


3) A new project is created.

## Adding a new FB

### ■ Adding an FB

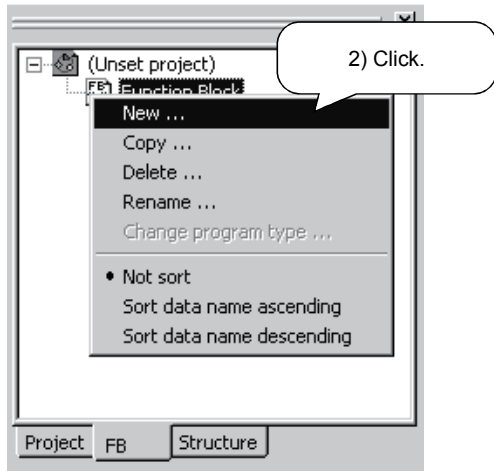
The operation method to add a new ST-written FB will be explained.



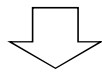
1) Right-click "Function Block" in the <<FB>> tab to display the menu.

↓  
To next page

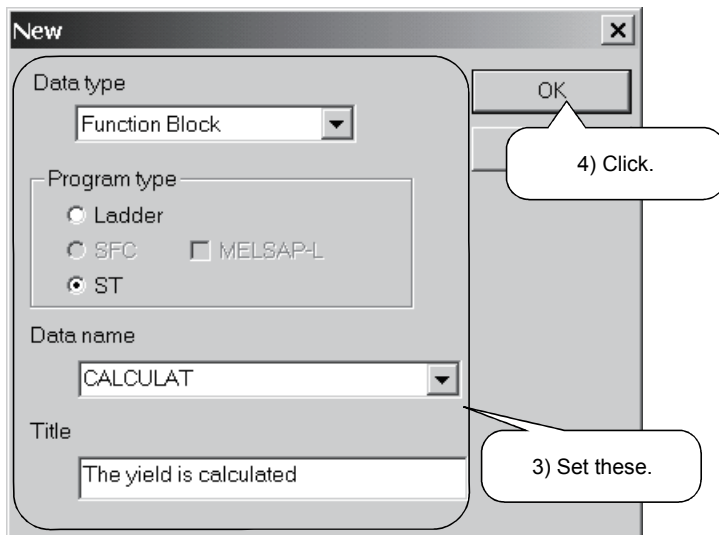
From previous page



2) Click "New".



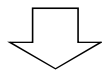
Set the New dialog.



3) Enter as follows.

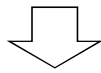
- Data type : Function Block
- Program type : ST
- Data name : CALCULAT
- Title : The yield is calculated

4) Click the **OK** button.

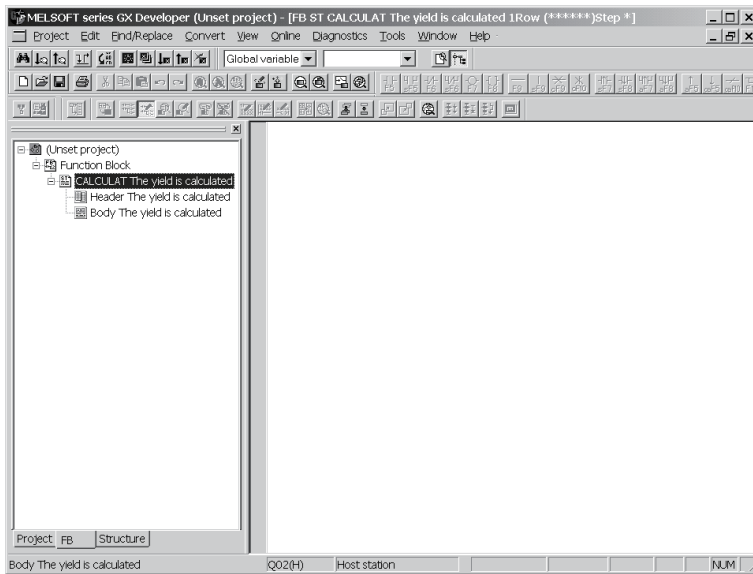


5) Click the **Yes** button.

To next page



From previous page



6) A new data name: CALCULAT is added.

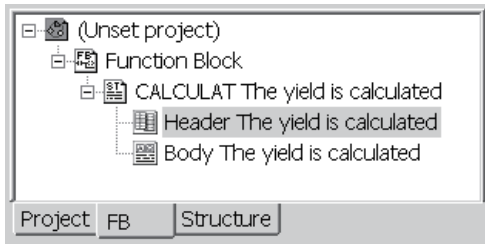


Defining FB variables

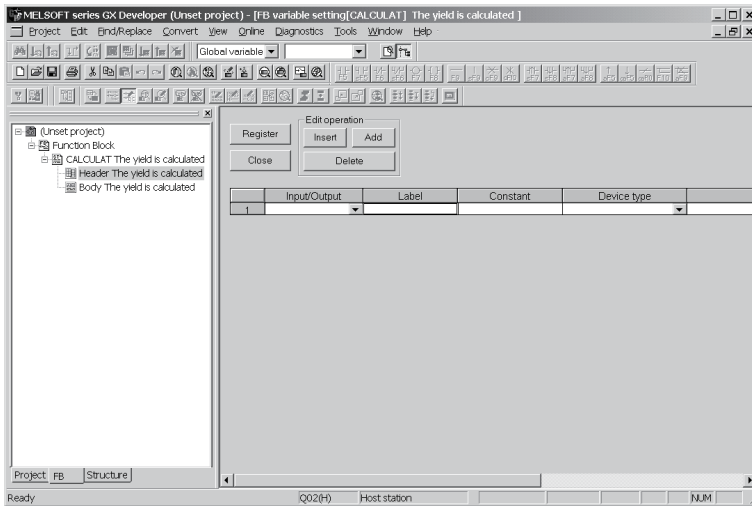
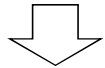
The labels used in the FB are called FB variables (FB labels).

■ Displaying the FB variable (FB labels) setting screen

Here, the operation method to define FB variables (FB labels) will be explained.



1) Double-click "Header" in the <<FB>> tab.



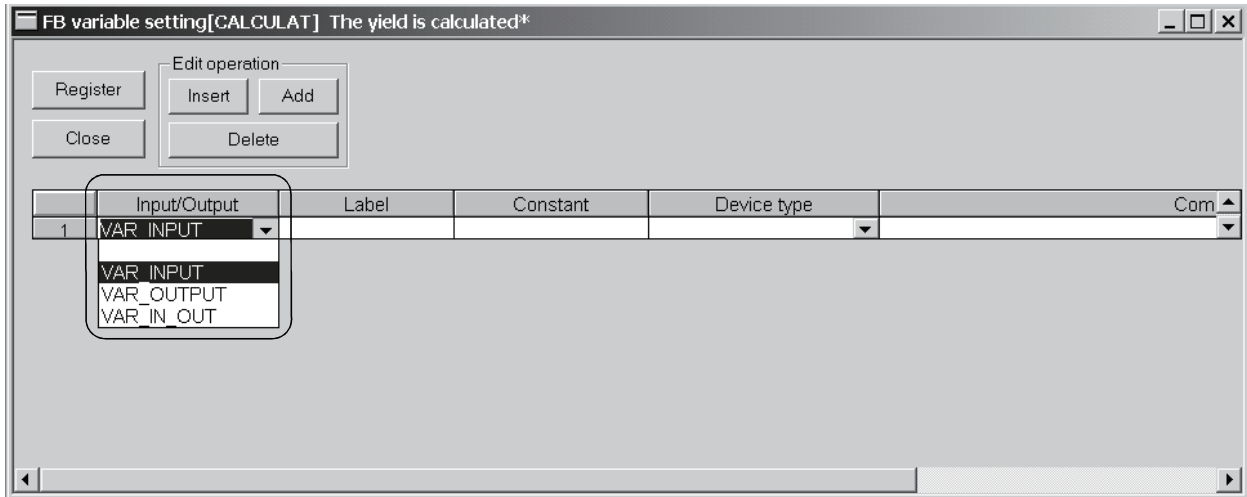
2) The FB variable (FB labels) setting screen is displayed.

■ Setting the FB variables (FB labels)

1) Select the Input/Output type.

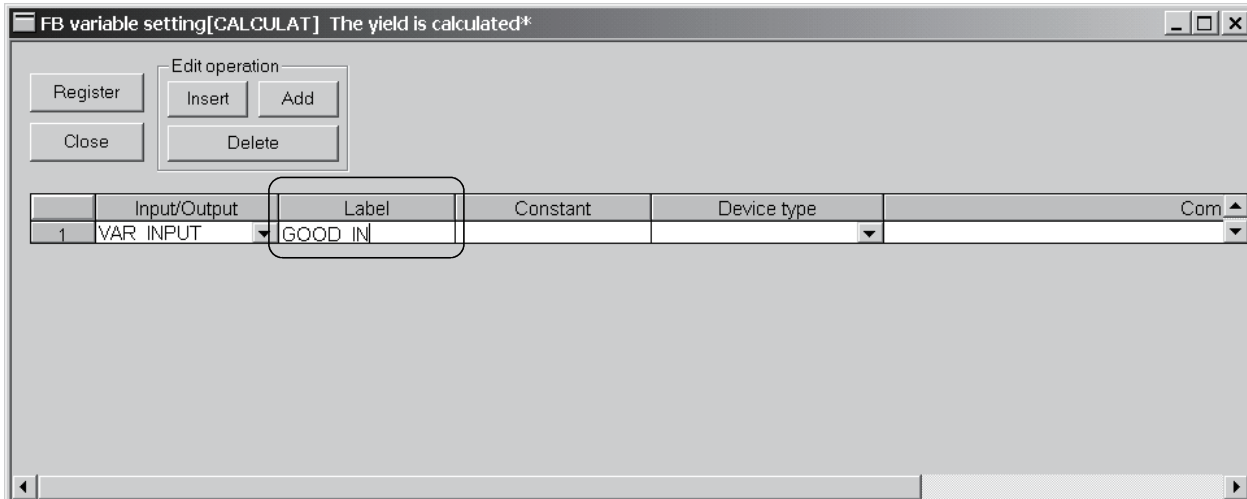
Select the label type. There are the following four types.

- VAR\_INPUT ..... Variable input from FB outside
- VAR\_OUTPUT ..... Variable output to FB outside
- VAR\_IN\_OUT ..... Variable having the input and output functions
- "Blank" ..... Variable used in FB inside



2) Enter the label name.

Enter the label name within 16 characters.

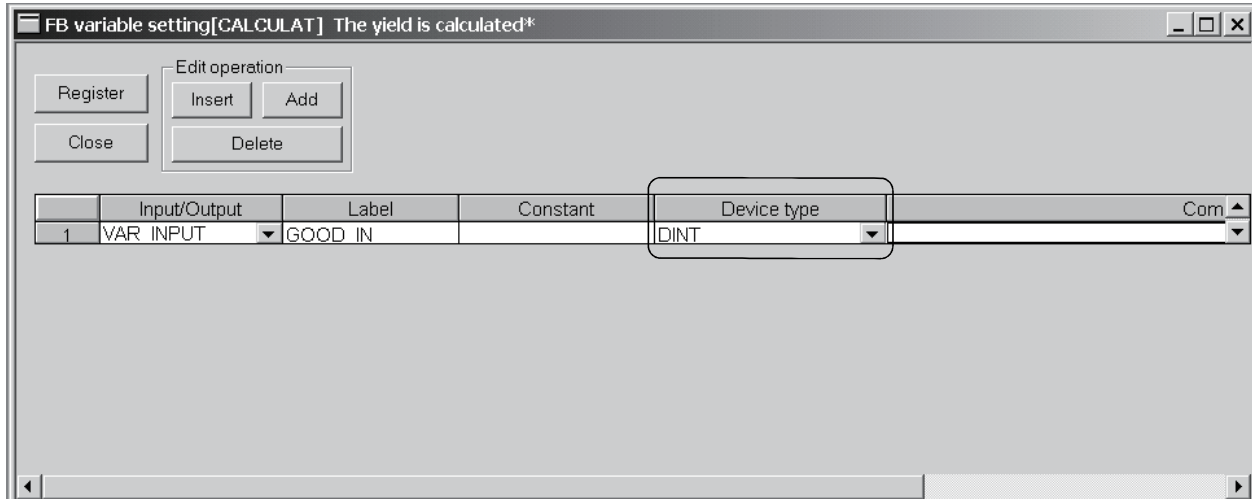


To next page

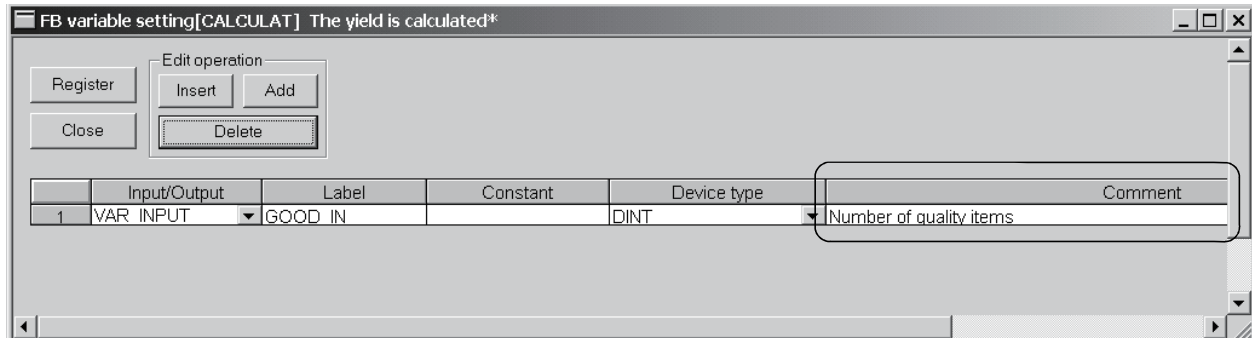


From previous page

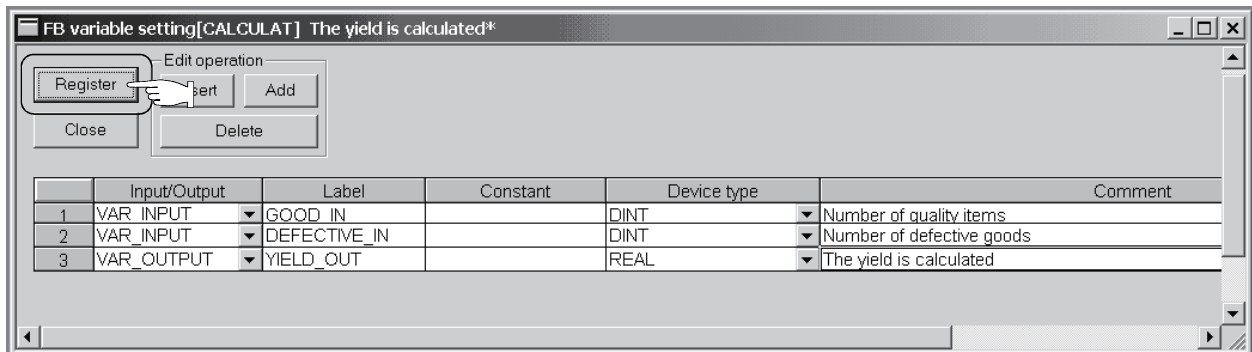
- 3) Enter the device type.  
Enter it directly or make selection from the list box.



- 4) Enter a comment into the label.  
Set it within 64 characters.

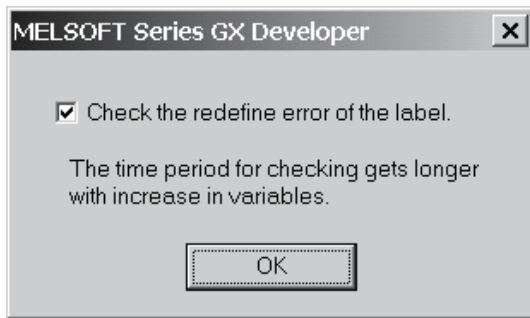


- 5) After input is complete, click the **Register** button.

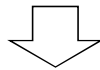


To next page

From previous page



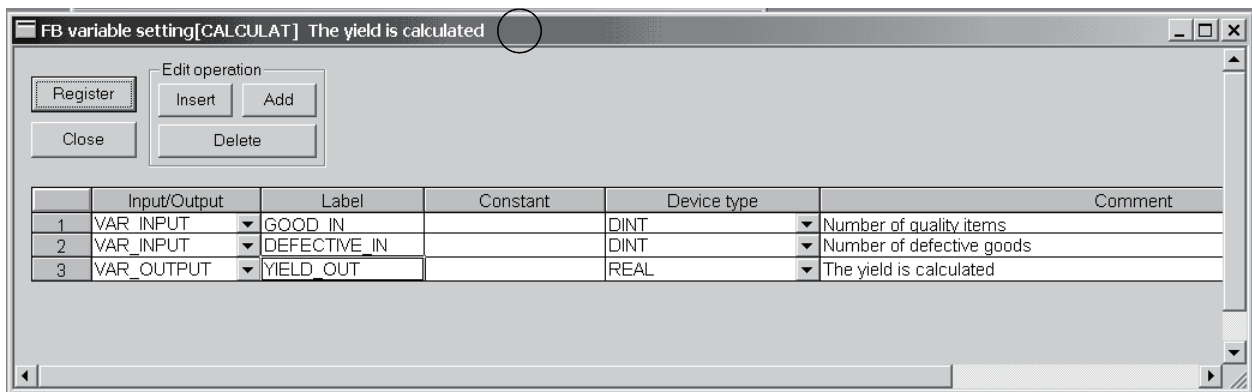
Click the **OK** button.



Registration is completed.  
Click the **OK** button.



When registration is made, "\*" displayed on the title bar disappears.



**REMARK**

For details, refer to the "GX Developer Operating Manual (Function Block)" given in Related Manuals.

Creating an FB in ST language

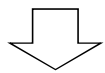
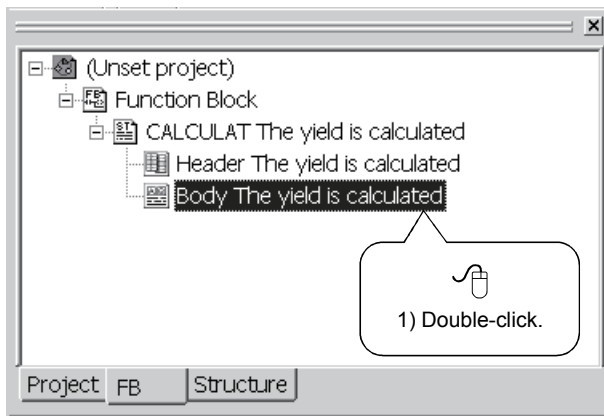
The operation to input the program in List-3 will be explained.

List-3

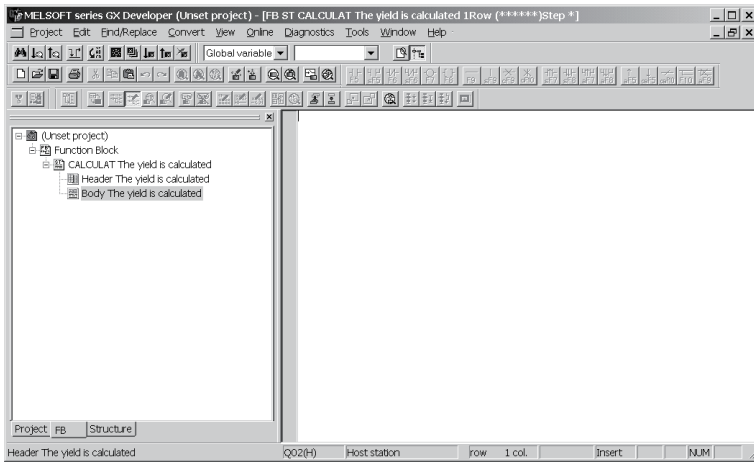
```
YIELD_OUT := DINT_TO_REAL(GOOD_IN)/DINT_TO_REAL(GOON_IN + DEFECTIVE_IN);
```

■ Displaying the FB definition screen

1) Double-click "Body".



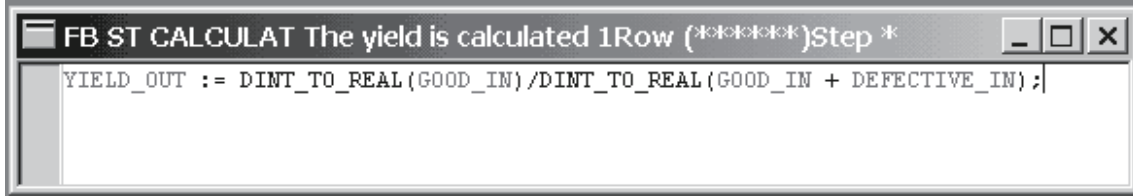
2) The FB definition screen is displayed.



## ■ Inputting the program

Input the FB program body as in the method of inputting the main program described in Chapter 3.

Input the program in List-3.

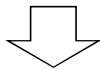


## ■ Converting (compiling) the FB

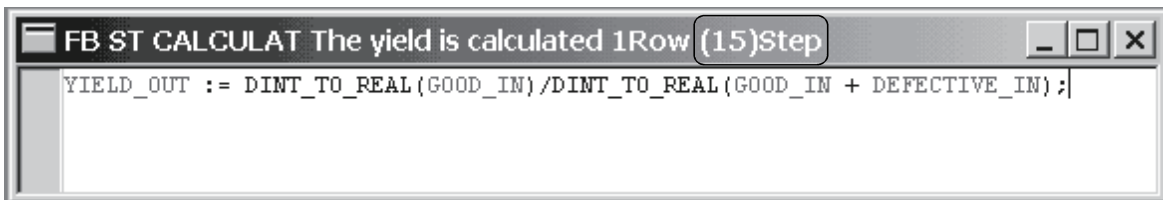
Click [Convert] → [Convert/Compile] in the menu to perform convert (compile).



Compile processing is completed.  
Click the **[OK]** button.



At normal completion of convert (compile), the number of steps is displayed on the title bar.



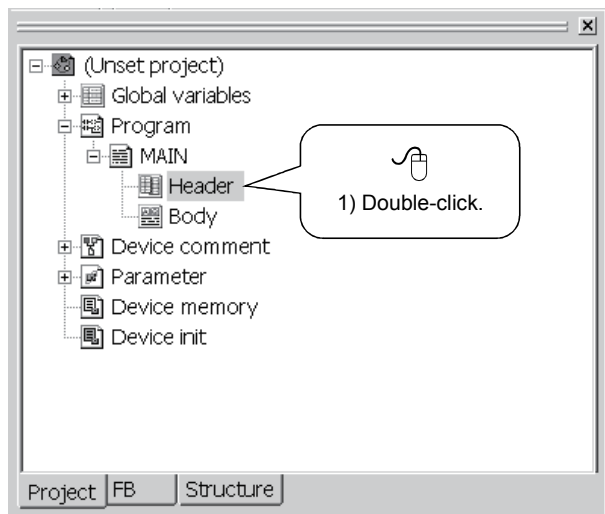
## 8.2 Pasting the FB to a Main Program

Create a main program (ladder) using the FB created in Section 8.1.

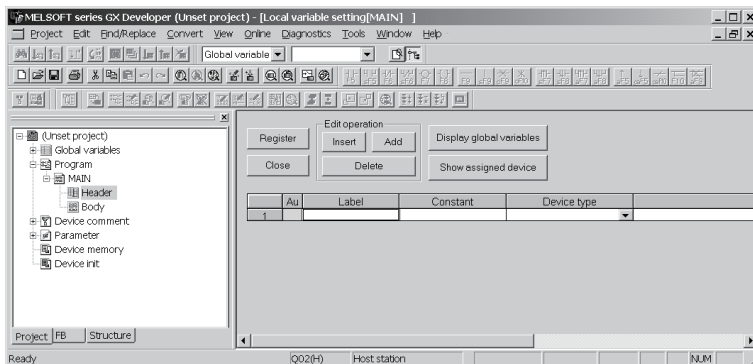
### Defining the local variables

Define the labels used in the main program.

#### ■ Displaying the Local variables setting screen



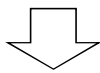
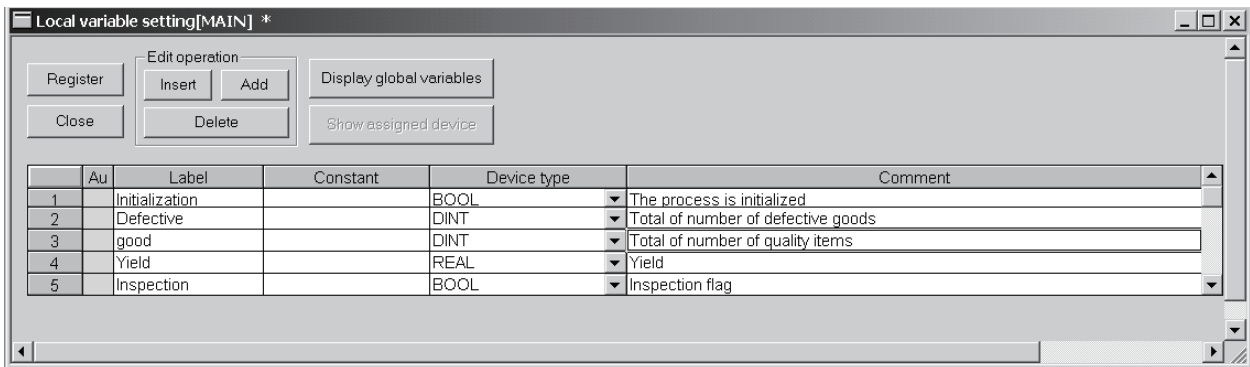
1) Double-click "Header".



2) Local variable setting screen is displayed.

## ■ Setting the local variables (headers)

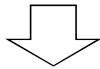
Refer to Chapter 3 and make setting as follows.



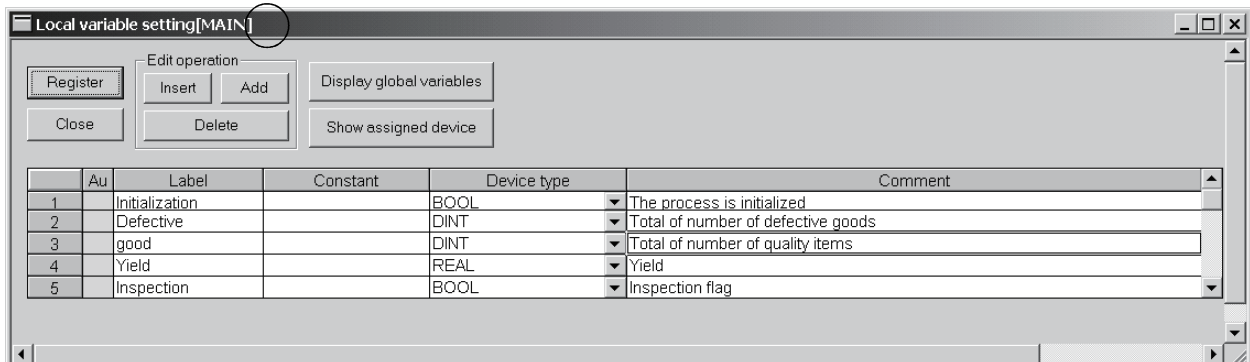
After input is complete, click the **Register** button.

The registration of the local variables is completed.

Click the **OK** button.



When registration is made, "\*" displayed on the title bar disappears.

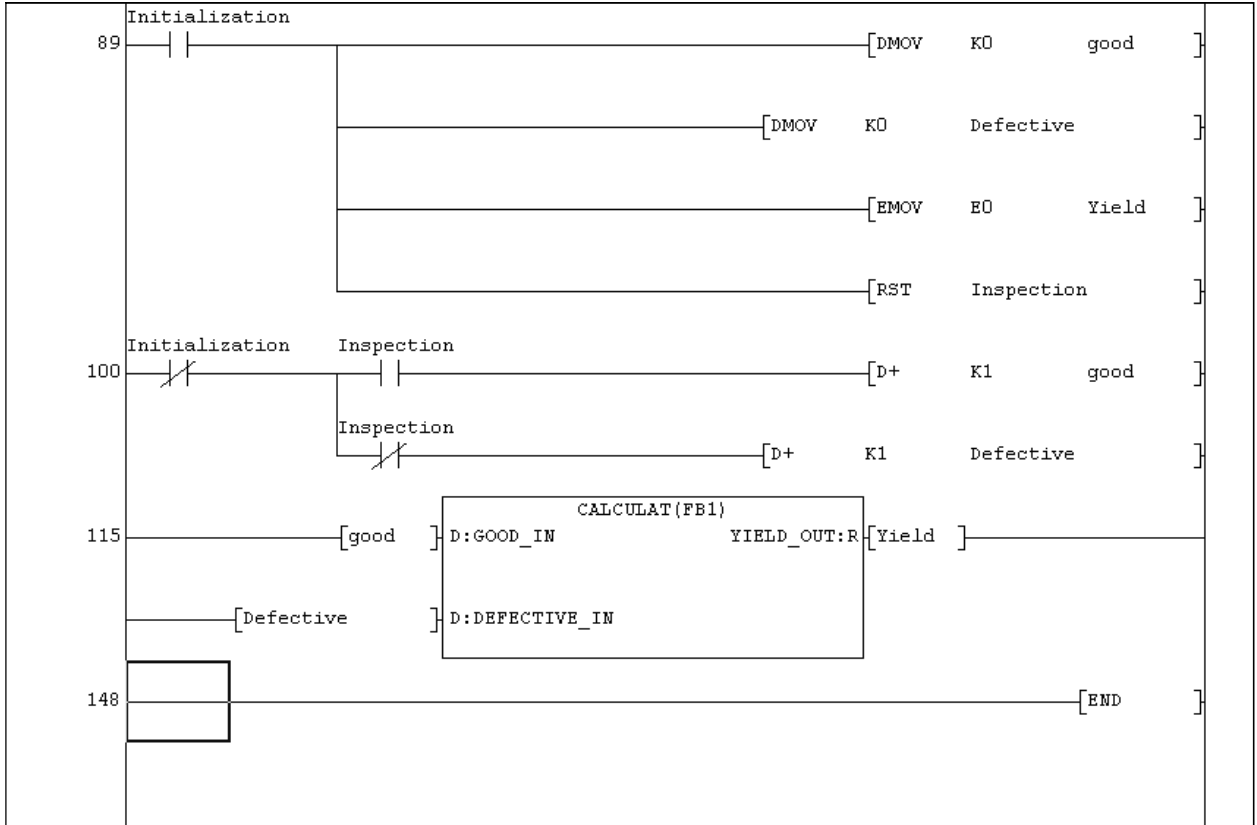




Creating a main program

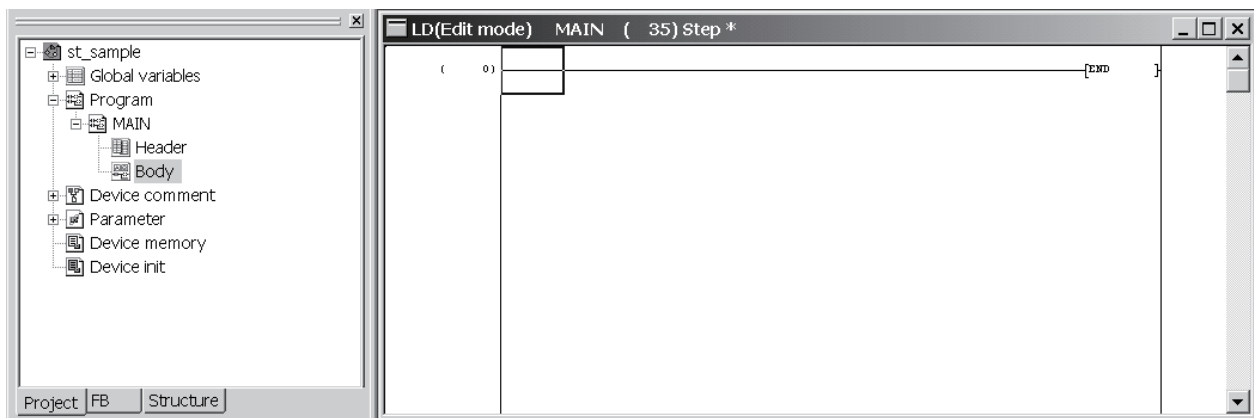
Display the main program edit screen and input the following program (List-4).

List-4



■ Displaying the edit screen

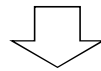
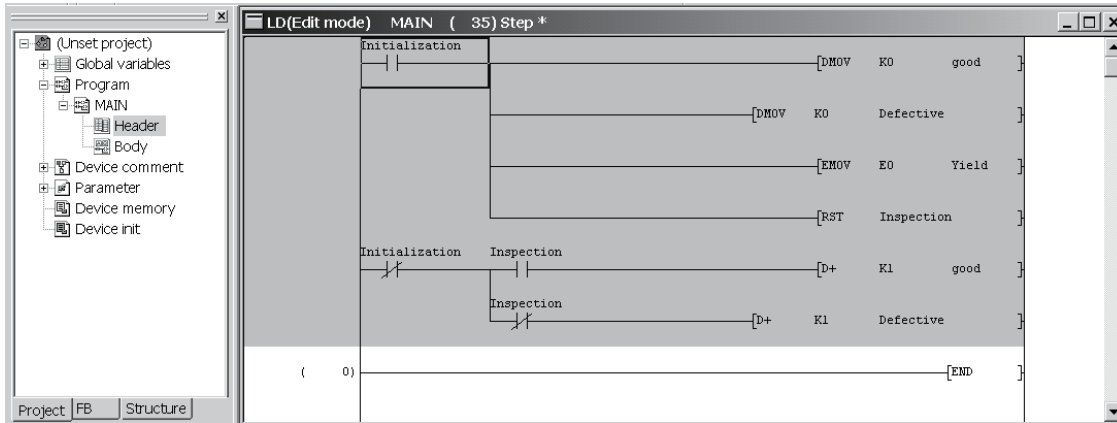
Double-click "Body" in the <<Project>> tab to display the edit screen.



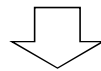
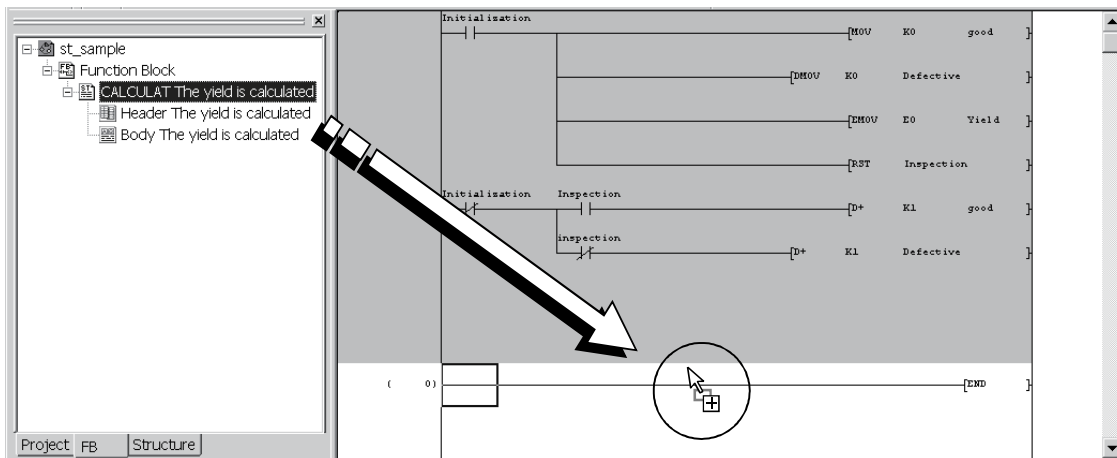
↓ To next page

From previous page

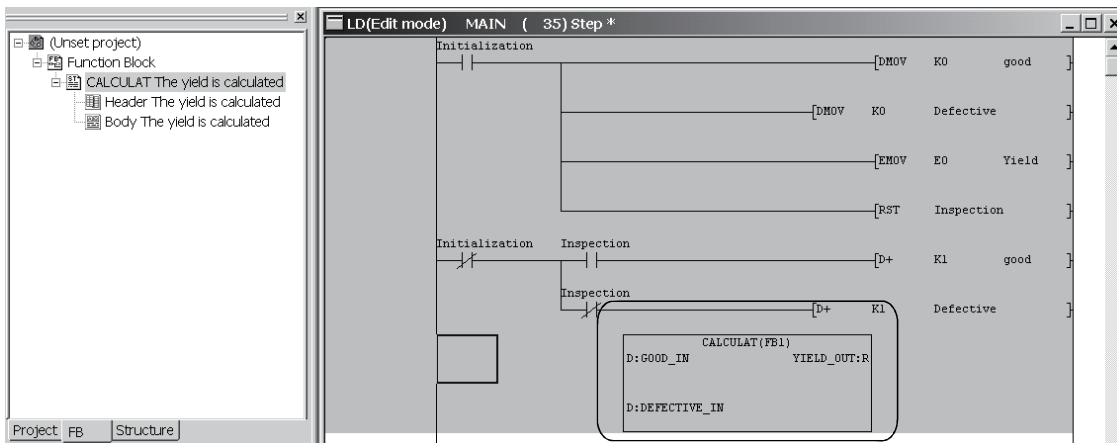
- Inputting the program in ladder format  
Refer to the following diagram and input the program.



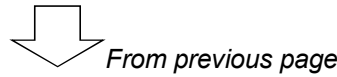
- Pasting the FB  
Switch to the <<FB>> tab, and drag and drop the FB program to the target place.



The FB is inserted into the main program.

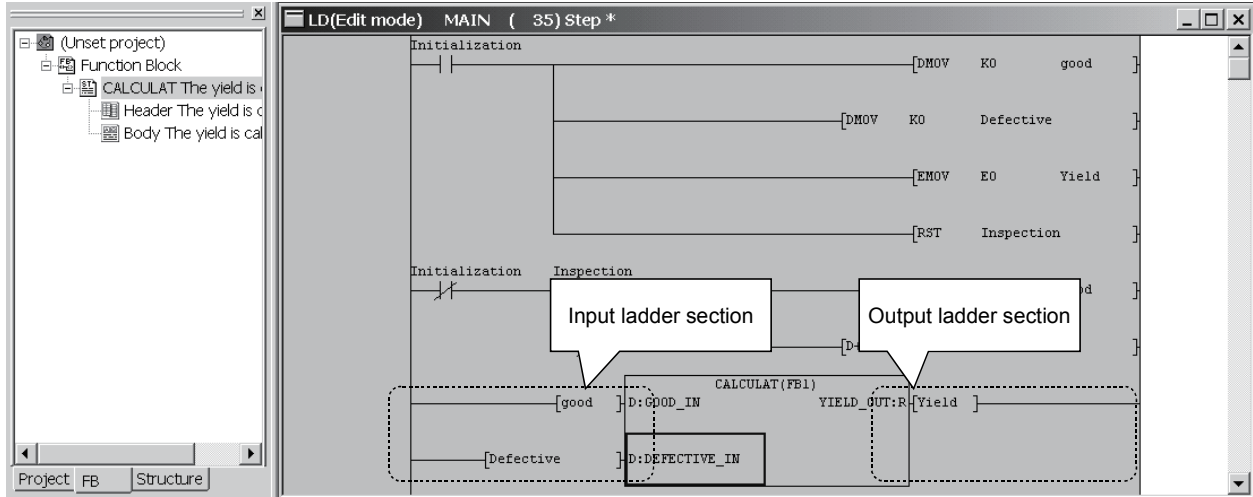


To next page



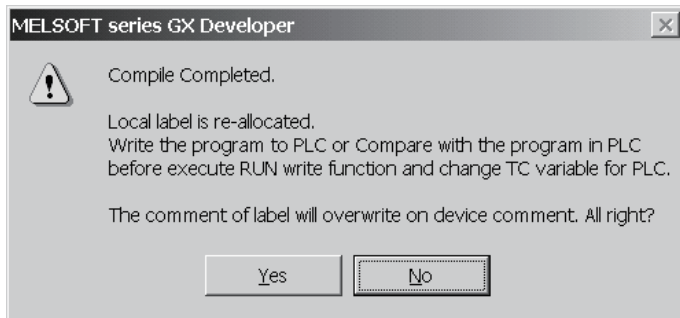
■ Inputting the input ladder section and output ladder section

Refer to the following diagram, and input the input ladder section and output ladder section.

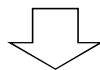


■ Performing convert (compile)

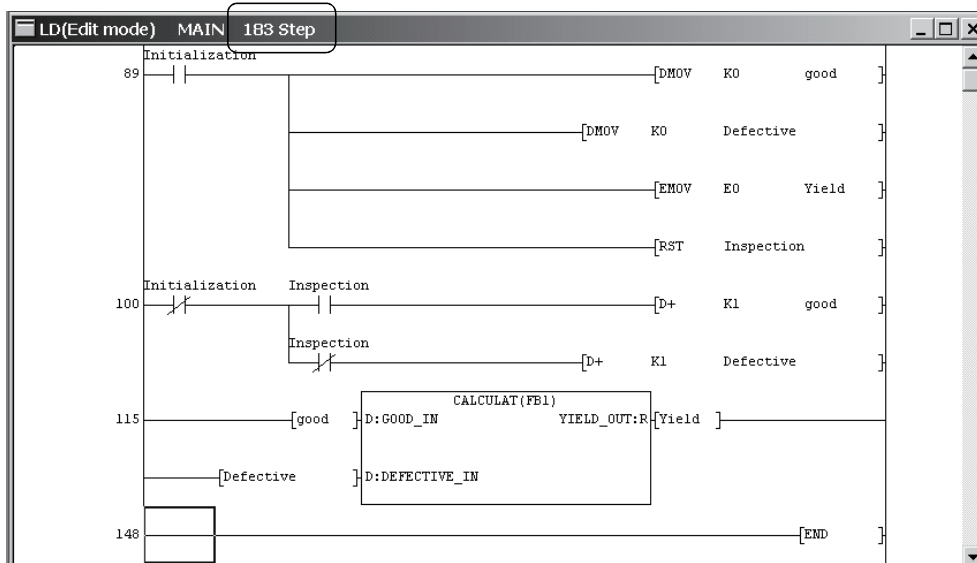
Click [Convert] → [Convert/Compile] in the menu to perform convert (compile).



Convert (compile) is completed. Click the **No** button.



When convert (compile) is completed, the number of steps is displayed on the title bar.



## 8.3 Online

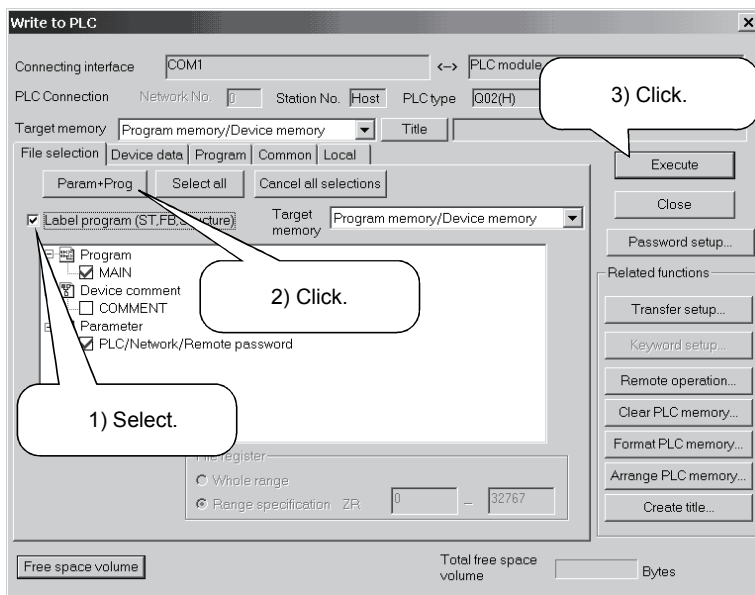
Write the sequence program to the programmable controller CPU, and confirm the program behavior using the monitor function and device test function.

### Writing to programmable controller CPU

#### ■ Performing write to PLC

Refer to Chapter 4 and perform write to PLC.

Choose [Online] → [Write to PLC] in the menu to display the Write to PLC dialog.



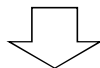
\* When performing write to PLC, put the programmable controller CPU in a STOP status.

1) Choose the "Label program (ST, FB, structure)" check button in the <<File selection>> tab.

\* If the check button is not chosen, only the actual program is written.

2) Click "Param + Prog".

3) Click the **Execute** button.



3) Write to PLC is completed.

\* Reset the programmable controller CPU and put it in a RUN status.

If an error occurs, choose [Diagnostics] → [PLC diagnostics] in the menu of GX Developer, and confirm the error definition.

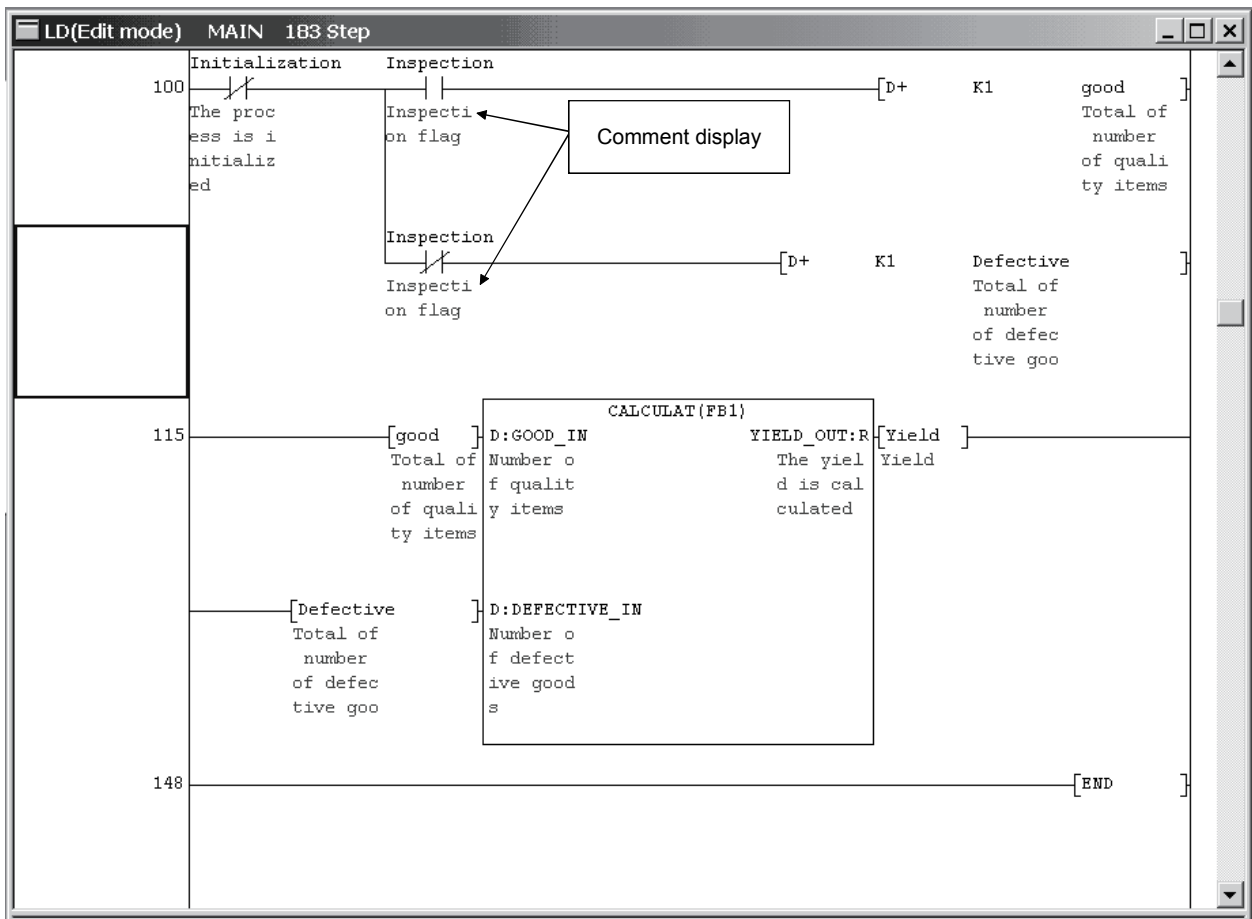
## Monitoring the sequence program

Monitor and confirm the sequence program.

The monitor start/stop operation is as follows.

- When starting monitor  
[Online] → [Monitor] → [Monitor mode]
- When stopping monitor  
[Online] → [Monitor] → [Stop monitor]
- When resuming monitor  
[Online] → [Monitor] → [Start monitor]

### Monitor display



The comments set on the Local variables setting screen can be displayed by choosing [View] → [Comment] in the menu.

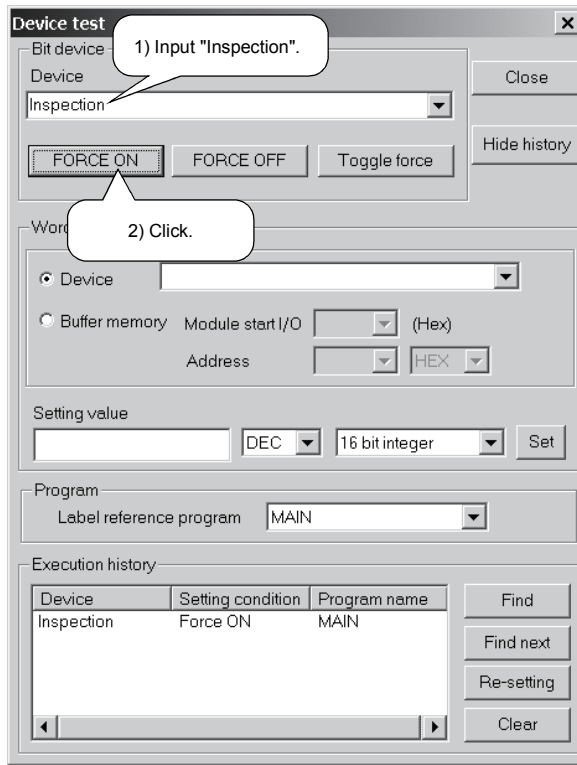
## Confirming the program behavior

Change the value of the bit device in the programmable controller CPU and confirm the program behavior.

### Conducting a device test

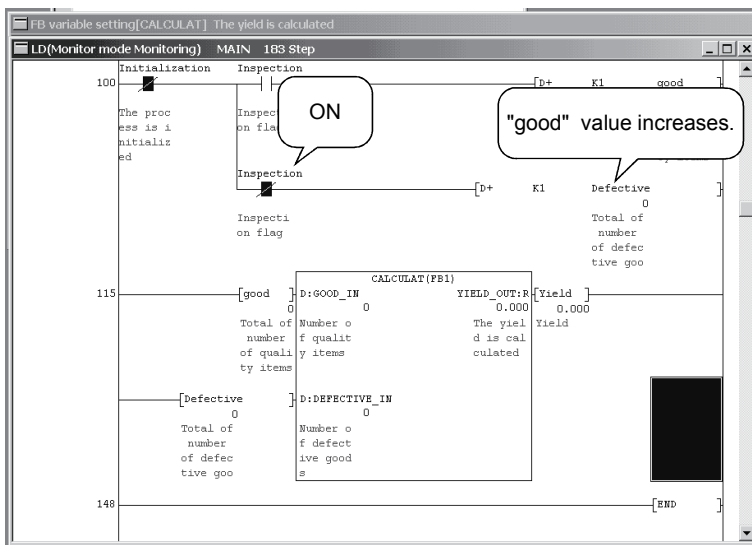
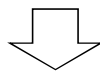
Refer to Section 5.2 and change the value of the bit device.

Choose [Online] → [Debug] → [Device test] in the menu to display the Device test dialog.



1) Input "Inspection" into the bit device.

2) Click the **FORCE ON** button.



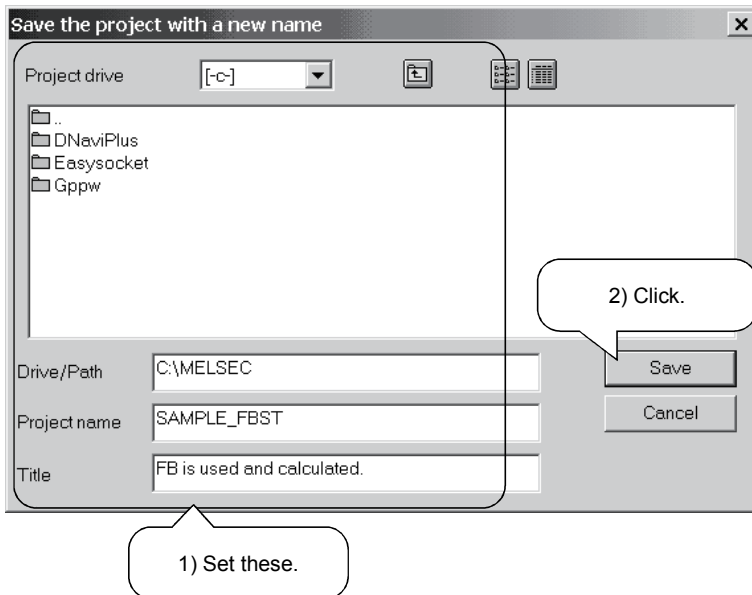
3) Check that the program is running correctly.

\* Also change the other label values and confirm the program behavior.

## ■ Saving the project

Refer to Section 6 and save the created project with a name.

Click [Project] → [Save as] in the menu to display the Save the project with a new name dialog.



1) Enter as follows.

- Drive/Path : C:\MELSEC
- Project name : SAMPLE\_FBST
- Title : FB is used and calculated.

2) Click the **Save** button.

The program created this time was saved as described below.

Drive/Path : C:\MELSEC  
 Project name : SAMPLE\_FBST  
 Title : FB is used and calculated

This ends the explanation of a series of operation methods for program creation.

To further proceed to the next step, it is recommended to refer to the manuals given in the section "Related Manuals".





## INDEX

## [A]

- Adding an FB ..... 8- 2
- Alarm history display function ..... 6-16
- Alarm list display function
  - System alarm ..... 6-19
  - User alarm ..... 6-18

## [C]

- Convert (compile)
  - Converting (compiling) the FB ..... 8-10
  - Error indication mark ..... 3-16
  - Performing convert (compile) ..... 3-14
  - The compile error (Detail) ..... 3-15
  - What is convert (compile) ..... 3-14

## [D]

- Device Test
  - Confirming the program behavior ..... 5- 2, 8-18
- Displaying the FB definition screen ..... 8- 9
  - Inputting the program ..... 8-10
- Displaying the FB variable screen ..... 8- 5
- Displaying the ST edit screen ..... 3- 9

## [E]

- Editing
  - Bookmark ..... 7- 1
  - Change of font ..... 7- 1
  - Change of display font ..... 7- 1
  - Display of label information ..... 7- 1
  - Select function ..... 7- 1
  - Window division ..... 7- 1

## [L]

- Label
  - What does defining the labels mean ..... 3- 3
- Local variable
  - Displaying the setting screen ..... 3- 3, 8-11
  - Enter a comment into the label ..... 3- 5
  - Enter a device type ..... 3- 5
  - Enter a label name ..... 3- 5
  - Setting the local variables ..... 3- 5, 8-12

## [M]

- Monitor
  - Monitoring the Sequence Program
    - ..... 5- 1, 8-17

## [O]

- Online change ..... 5- 4

## [P]

- Pasting the FB ..... 8-14
- Performing read from PLC ..... 4- 2
- Performing write to PLC ..... 4- 1, 8-16
- Program
  - Entering a comment ..... 3-12
  - Entering a function ..... 3-11
  - Entering a label ..... 3-10
  - Entering the characters ..... 3- 9
  - Function argument ..... 3-12
  - Function selection function ..... 3- 11, 7- 1
  - Inputting the input ladder section and output ladder section ..... 8-15
  - Inputting the program in ladder format ..... 8-14
  - What should be noted during input ..... 3- 8
- Project
  - Creating a new project ..... 8- 1
  - Creating a new ST project ..... 3- 1
  - Saving the project ..... 6- 1, 8-19

## [S]

- Enter a comment into the label ..... 8- 7
- Enter the device type ..... 8- 7
- Enter the label name ..... 8- 6
- Select the Input/Output type ..... 8- 6
- Setting the FB variables ..... 8- 6

## [W]

- What is a function block (FB)
  - What is a function block (FB) ..... 8- 1
- What is the ST language ..... 1- 1



# TRADEMARKS

---

The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.

In some cases, trademark symbols such as '™' or '®' are not specified in this manual.





SH(NA)-080368E-I(2211)KWIX

MODEL: ST-GUIDE-E

MODEL CODE: 13JF69

## **MITSUBISHI ELECTRIC CORPORATION**

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN  
NAGOYA WORKS : 1-14, YADA-MINAMI 5-CHOME, HIGASHI-KU, NAGOYA, JAPAN

When exported from Japan, this manual does not require application to the  
Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.