User's
Manual

YS1000 Series

YSS1000 Setting Software for YS1000 Series/YS1700 Programmable Function User's Manual

IM 01B08K01-02E

vigilantplant®

YOKOGAWA
Yokogawa Electric Corporation

IM 01B08K01-02E
5th Edition

Product Registration

Thank you for purchasing YOKOGAWA products.
YOKOGAWA provides registered users with a variety of information and services.
Please allow us to serve you best by completing the product registration form accessible from our homepage.

http://www.yokogawa.com/ns/reg/

Introduction

Thank you for purchasing the YS1000 Series single-loop controller (hereinafter referred to as "YS1000") and the YSS1000 Setting Software for YS1000 Series.
This manual describes how to use the YSS1000 Setting Software for YS1000 Series and the user program functions of the YS1700. Please read through this user's manual carefully before using the product.
The following ten manuals are provided as references for the YSS1000 Setting Software for YS1000 Series.

- **Electronic manuals contained in the provided CD-ROM**

Manual Name	Manual Number	Description
YS1500 Indicating Controller/YS1700 Programmable Indicating Controller Operation Guide	IM 01B08B01-01E	This manual describes the basic operation methods, installation and wiring.
YS1500 Indicating Controller/YS1700 Programmable Indicating Controller User's Manual	IM 01B08B01-02E	This manual describes the detailed functions and setting items. It does not contain user programs and communication functions.
YSS1000 Setting Software for YS1000 Series Installation Manual	IM 01B08K01-01E	This manual describes the operating environment of the YSS1000 Setting Software for YS1000 Series and how to install/uninstall the software.
YSS1000 Setting Software for YS1000 Series/YS1700 Programmable Function User's Manual	IM 01B08K01-02E	This manual describes how to use YSS1000 Setting Software for YS1000 Series, and the programmable functions and peer-to-peer communications of the YS1700. This manual
YS1310 Indicator with Alarm Operation Guide	IM 01B08D01-01E	This manual describes the basic operation methods, installation and wiring.
YS1310 Indicator with Alarm User's Manual	IM 01B08D01-02E	This manual describes the detailed functions and setting items. This manual does not describe the communication functions.
YS1350 Manual Setter for SV Setting/ YS1360 Manual Setter for MV Setting Operation Guide	IM 01B08E01-01E	This manual describes the basic operation methods, installation and wiring.
YS1350 Manual Setter for SV Setting/ YS1360 Manual Setter for MV Setting User's Manual	IM 01B08E01-02E	This manual describes the detailed functions and setting items. This manual does not describe the communication functions.
YS1000 Series Communication Interface User's Manual	IM 01B08J01-01E	This manual describes how to use YS1000 in Ethernet, serial and DCS-LCS communication, and YS1000 internal registers.
YS1000 Series Replacement Manual	IM 01B08H01-01E	This manual describes the compatibility of installation and wiring with YS100, YS80, EBS, I, EK, HOMAC, and 100 Line.

The provided CD-ROM also contains the YS110 manual.

Notice

- The contents of this manual are subject to change without notice as a result of continuing improvements to the instrument's performance and functions.
- Every effort has been made to ensure accuracy in the preparation of this manual. Should any errors or omissions come to your attention, however, please inform YOKOGAWA Electric's sales office or sales representative.
- Under no circumstances may the contents of this manual, in part or in whole, be transcribed or copied without our permission.

Trademarks

- Our product names or brand names mentioned in this manual are the trademarks or registered trademarks of YOKOGAWA Electric Corporation (hereinafter referred to as YOKOGAWA).
- Microsoft, Windows, Windows 2000, Windows XP, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- Adobe, Acrobat, and Postscript are either registered trademarks or trademarks of Adobe Systems Incorporated.
- Ethernet is a registered trademark of XEROX Corporation.
- We do not use the TM or ® mark to indicate these trademarks or registered trademarks in this user's manual.
- All other product names mentioned in this user's manual are trademarks or registered trademarks of their respective companies.

5th Edition : Aug. 2009 (YK)

How to Use this Manual

Usage

First read through the Operation Guide to understand the basic operations and then read this manual. For details on control and communication functions, see the respective manuals.
This User's Manual is organized into Chapters 1 to 12.

Chapter	Title and Description	YS1700	YS1500/ YS1310/ YS1350/ YS1360
1	Overview Briefly describes the functions of the YSS1000 Setting Software for YS1000 Series.	✓	✓
2	YSS1000 Operation Guide Describes how to set parameters, set up events, and perform upload/download operations, monitoring, file management operations, and printing on the YS1700, YS1500, YS1310, YS1350 and YS1360.	✓	✓
3	User Program Creation Guide Describes how to create user programs.	✓	–
4	Operation of Computation and Control Programs Describes the basic content of user programs.	✓	–
5	Basic Usage of Control Modules Describes basic usage of user program control (basic control, cascade control and selector control) modules.	✓	–
6	Applied Usage of Control Modules Describes applications using the extended functions of control modules in user programs.	✓	–
7	Operations and Application of Computing Module (Instructions) Describes computing modules and instructions used in user programs.	✓	–
8	Using Peer-to-peer Communication Describes peer-to-peer communication. To use peer-to-peer communication, a user program must be built.	✓	–
9	Maintenance Describes how to maintain user programs, and the Check Support Function of YSS1000.	✓	✓ “Maintenance of user programs” is not applicable.
10	Sample Program Presents an example of flow control with temperature-pressure control of an ideal gas flow.	✓	–
11	Worksheets/Program Sheets/Parameter Sheets Provides worksheets and program sheets that are used when designing programs.	✓	✓
12	List of Text Program Instructions Presents a list as a useful reference for looking up details on the operation of instructions when programming by the text method. This list describes operations before and after execution of instructions.	✓	–

✓: Applicable
–: Not applicable

Symbols Used in This Manual

This symbol is used on the instrument. It indicates the possibility of injury to the user or damage to the instrument, and signifies that the user must refer to the user's manual for special instructions. The same symbol is used in the user's manual on pages that the user needs to refer to, together with the term "WARNING" or "CAUTION."

WARNING

Calls attention to actions or conditions that could cause serious or fatal injury to the user, and indicates precautions that should be taken to prevent such occurrences.

CAUTION

Calls attention to actions or conditions that could cause injury to the user or damage to the instrument or property and indicates precautions that should be taken to prevent such occurrences.

Note

Identifies important information required to operate the instrument.

▶

Indicates related operations or explanations for the user's reference.

[ ]

Indicates a character string displayed on the display.

Setting Display

Indicates a setting display and describes the keystrokes required to display the relevant setting display.

Setting Details

Provides the descriptions of settings.

Description

Describes restrictions, etc. regarding a relevant operation.

Operation

Describes operation procedures.

Blank Page

Contents

Chapter 3 User Program Creation Guide

Chapter 4 Operation of Computation and Control Programs

Chapter 7 Operations and Application of Computing Module (Instructions)

Chapter 8 Using Peer-to-peer Communication

Chapter 9 Maintenance

Blank Page

1.1 Overview of Function

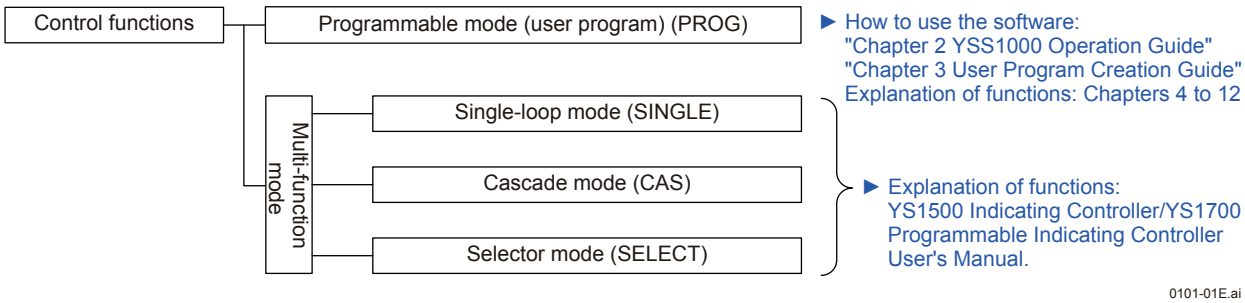
1.1.1 YS1000 Setup Functions

The YSS1000 Setting Software for YS1000 Series runs on a PC, and is for making and setting up parameters and user programs for the YS1000 Series.
The PC can also communicate with the YS1000 to perform tuning or monitor user programs.

YS1000 Setup Function		Description	YS1700 Programmable Mode		YS1700/ YS1500 Multi-function Mode	YS1310/ YS1350/ YS1360
			Function block programming	Text programming		
Parameter setup	Data sheet	File information, I/O comments, and parameter comments can be made.	✓	✓	✓	✓
	Parameter data	Function settings, PID parameters, tables, scales, constants, etc. can be set.	✓	✓	✓	✓
Event indicating setup		When an event occurs on the YS1000, popup window messages can be displayed on the Operation Display. (This cannot be set from the YS1000.)	✓	✓	✓	✓
Creation of user program	Function block programming	With function block programming, user programs can be easily made. Up to 400 function blocks can be registered.	✓	–	–	–
	Text programming	User programs can be made by conventional text programming (reverse Polish notation). Up to 1000 steps can be programmed in text programs.	–	✓	–	–
Communications	Download Upload Compare	Programmed parameters and user programs can be written to the YS1000, and parameters held on the YS1000 can be read. (Writing to the YS1000 is possible when its operation status is STOP.)	✓	✓	✓	✓
	Tuning	PID parameters can be adjusted (tuned) while viewing PV/SV/MV trends. The operation mode can also be changed.	✓	✓	✓	✓
	Register monitor	The registers used in the user program can be monitored.	✓	✓	–	–
	Function block monitor	A function block diagram of control modules written in the user program can be monitored.	✓	✓	–	–
	Module monitor	Input/output values of modules can be monitored.	✓	–	–	–
	Simulated event display	The content set to the event indicating function can be displayed on the YS1000 for simulation.	✓	✓	✓	✓
File	Save Open Compare	Parameters and user programs that were made on YSS1000 or read from YS1000 can be saved to disk on the PC.	✓	✓	✓	✓
Convert data of old models		Parameter data and user programs of YS80 (SLPC), YS100, and YSS20 R1.04 and later can be converted to enable editing with YSS1000.	–	✓	✓	✓
Save user program as component		Whole created user programs, individual program sheets or program pages can be saved as components. This makes them easy to reuse.	✓	–	–	–
Printing		Parameters and user programs can be printed.	✓	✓	✓	✓

✓: Available
–: Not available

Two modes are available for YS1700 control functions, the programmable mode and the multi-function mode.



YS1700 Control Functions

The YS1700 incorporates a programmable mode (PROG). This mode allows the user to make computation and control functions as desired. In this mode, the control period can be selected from 50, 100 and 200 ms.
The YS1700 also incorporates the same multi-function mode (single-loop mode (SINGLE), cascade mode (CAS), selector mode (SELECT)) as the YS1500. In these modes, the control period is fixed at 100 ms.

► Multi-function mode: YS1500 Indicating Controller/YS1700 Programmable Indicating Controller User's Manual (CD-ROM).
► Controller mode (CTL): YS1500 Indicating Controller/YS1700 Programmable Indicating Controller Operation Guide.

1.1.2 Check Support Functions

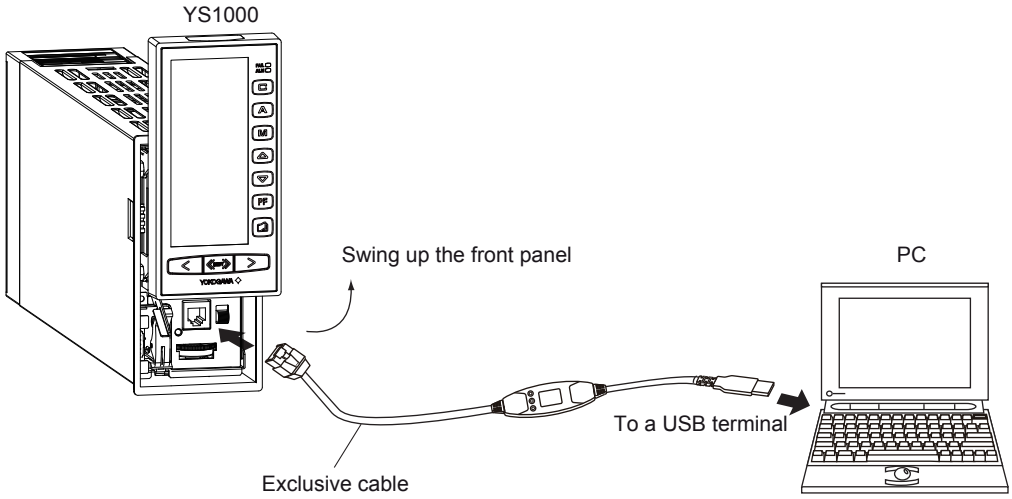
The Check Support Function displays the Operation Guide to check the YS1000 in order to support the check operation.

YS1000 Check Support Function		Description
Check Items		Key, LCD, LED lamp, Transmitter power supply, Digital input, Digital output, Analog input, Analog output, Hard manual (H MAN), FAIL, Power failure time detection, External appearance, Terminal, Custom check (Number of check items to which user can register: Max. 10 items)
Communications	Download Upload Delete	Up to 40 records of check result data can be saved to the YS1000. Check result data saved to the YS1000 can be loaded as check history data. (Saving to the YS1000 is enabled only in the operation STOP state)
File	Save Open	Check result data generated with the YSS1000 and check history data loaded from the YS1000 can be saved on the PC disk.
Printing		A Check Result List, check result data file, and check history data file can be printed.

1.2 Connecting the YS1000 to a PC and Setup Parameters

■ Connecting by the YS1000 exclusive cable

Connect the exclusive USB cable to the panel connector inside the YS1000 swing-up front panel. With this connection, only one YS1000 can be connected to a PC.



Setting Details

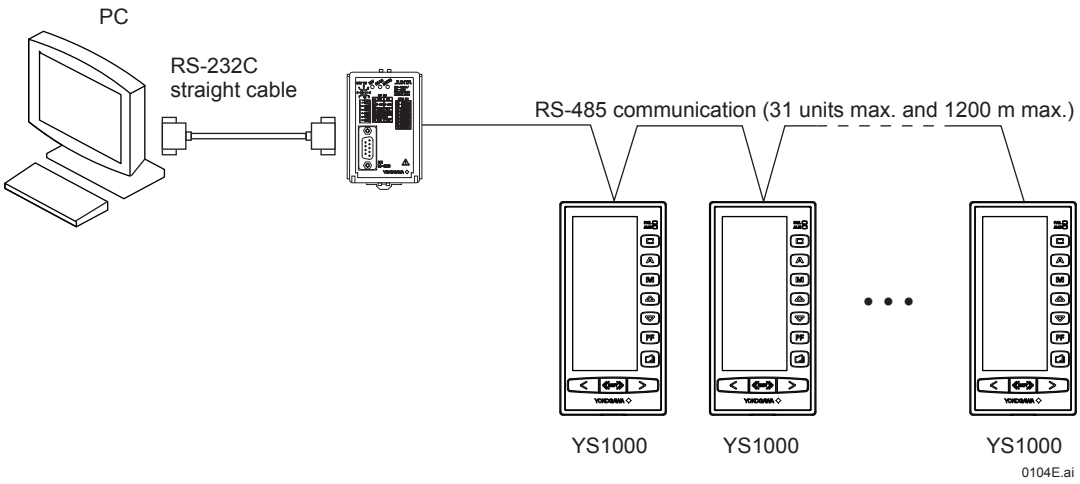
The communication parameters on the YS1000 need not be set.

■ Connecting by the communications terminal (optional code: /A31)

Various data can be read from or written to a PC via the communications terminal on the YS1000 rear.

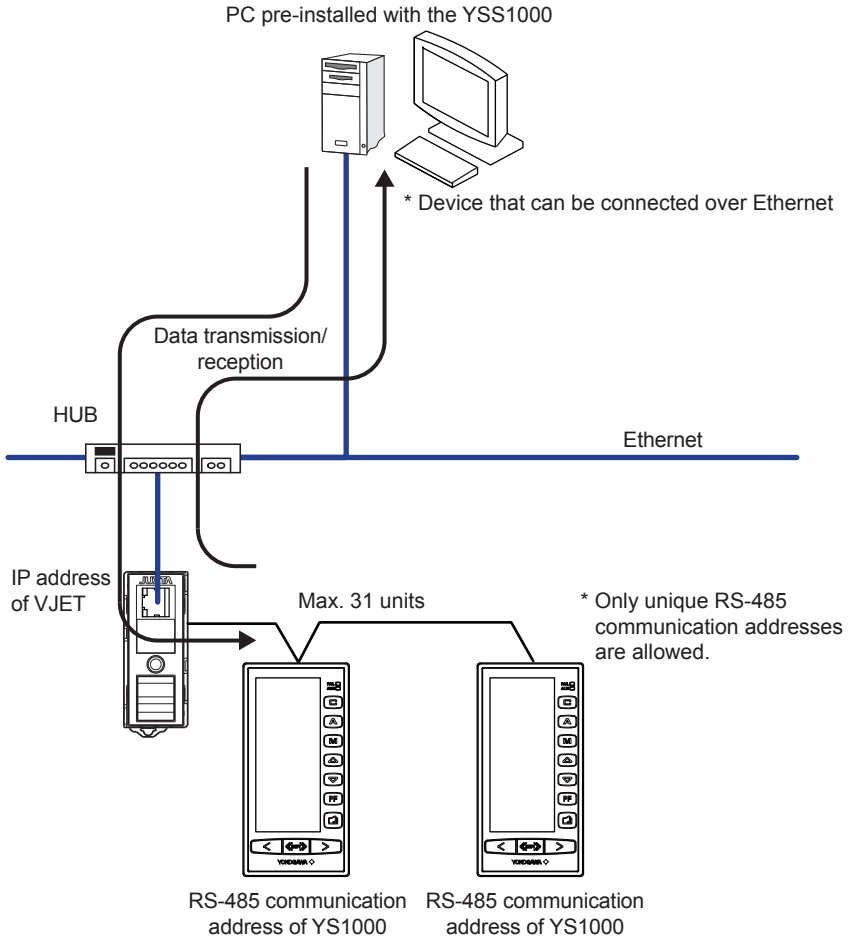
With this connection, an RS-485/RS-232C converter (*1) is required. For details on wiring, refer to the respective Operation Guides.

*1: Model ML2 (made by Yokogawa Electric Corporation) is recommended. (Set active control of a driver to Auto.)



The figure below is an example of communication through Ethernet/RS-485 converter (*2).

*2: Model VJET (made by Yokogawa Electric corporation) is recommended. For the wiring and setting of VJET, refer to the user's manual of VJET.



Setting Details

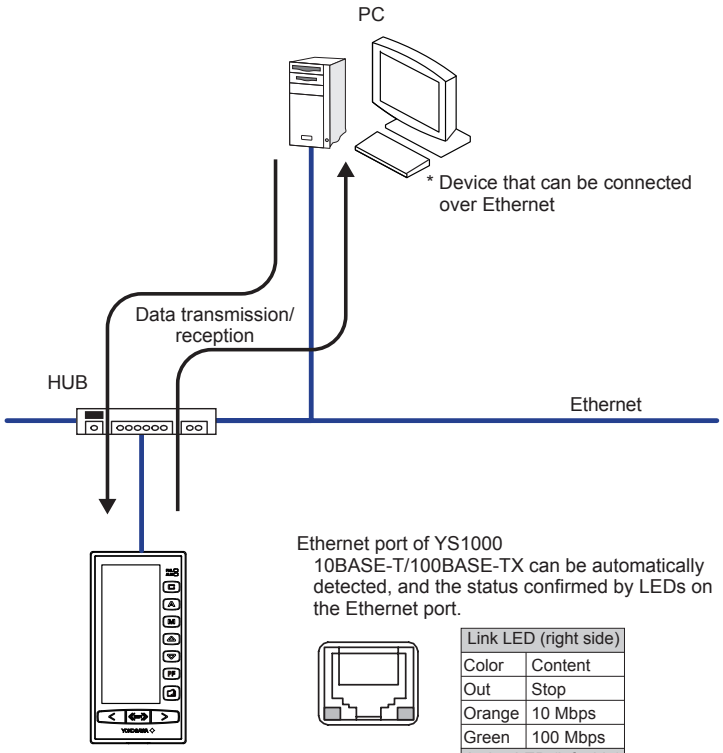
The communication parameters on the YS1000 must be set as follows.

Parameter	Name	Setting Range	Setting
DREG1	RS-485 communication D register setting for High/Low level	H-L: High-Low, L-H: Low-High	Set to H-L.
PSL	RS-485 protocol selection	PCL, PCLSUM, MODASC, MODRTU, YS, P-to-P	Set to MODRTU.
ADR	RS-485 communication address	1 to 99	Any value (only unique addresses are allowed)
STBIT	RS-485 stop bit	1,2 (bit)	Use the same communication setup as that set on YSS1000. (When the VJET is used, set to "1.")
PAR	RS-485 parity	NONE, ODD, EVEN	Use the same communication setup as that set on YSS1000. (When the VJET is used, use the same communication setup as that set on VJET.)
DLEN	RS-485 data length	7, 8 (bit)	Set to "8."
BPS	RS-485 baud rate	1200, 2400, 4800, 9600, 19200, 38400 (bps)	Use the same communication setup as that set on YSS1000. (When the VJET is used, set to "9600.")

► Parameter settings: set the above parameters referring to “Operating the Engineering Displays” in the respective Operation Guides.

■ Connecting by the Ethernet communications terminal (optional code: /A34)

Connect the YS1000 to a network that is capable of communications by a 10BASE-T/100BASE-TX standard-compliant cable, and read/write the various data to the YS1000 connected to this network. For details on wiring, refer to the respective Operation Guides.



Link LED (right side)	
Color	Content
Out	Stop
Orange	10 Mbps
Green	100 Mbps
Active LED (left side)	
Color	Content
Out	Stop
Orange	Half-duplex
Green	Full-duplex

0105E.ai

Setting Details

The communication parameters on the YS1000 must be set as follows.

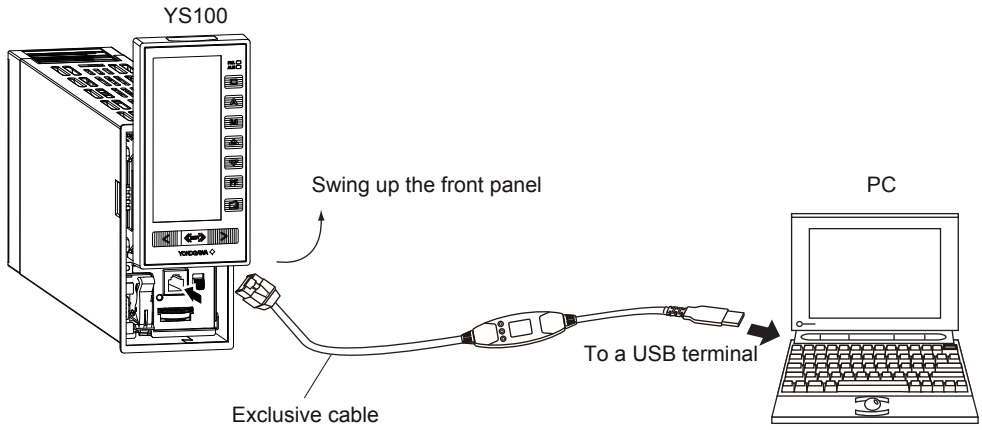
Parameter	Name	Setting Range	Setting
DREG2	Ethernet communication D register setting for High/Low level	H-L: High-Low, L-H: Low-High	Set to H-L.
IPAD1 to IPAD4	IP address 1 to 4	0 to 255	Default: 192.168.1.1 Setting position: IPAD1.IPAD2.IPAD3.IPAD4 (Note)
SM1 to SM4	Subnet mask 1 to 4	0 to 255	Default: 255.255.255.0 Setting position: SM1.SM2.SM3.SM4 (Note)
DG1 to DG4	Default gateway 1 to 4	0 to 255	Default: 0.0.0.0 Setting position: DG1.DG2.DG3.DG4 (Note)
PORT	Port No.	502, 1024 to 65535	Any value Default: 502
ESW	Ethernet setting switch	-, ENTRY	After changing the IP address, subnet mask and default gateway, set the switch to ENTRY to update the settings. The settings will be updated in about 20 seconds.

Note: Check the IP address, subnet mask and default gateway settings with the network administrator before setting them.

► Parameter settings: set the above parameters referring to "Operating the Engineering Displays" in the respective Operation Guides.

1.3 Connecting the YS100 to a PC (Old Model Data Conversion Function)

Connect the exclusive USB cable to the panel connector inside the YS100 swing-up front panel. With this connection, only one YS100 can be connected to a PC.



Setting Details

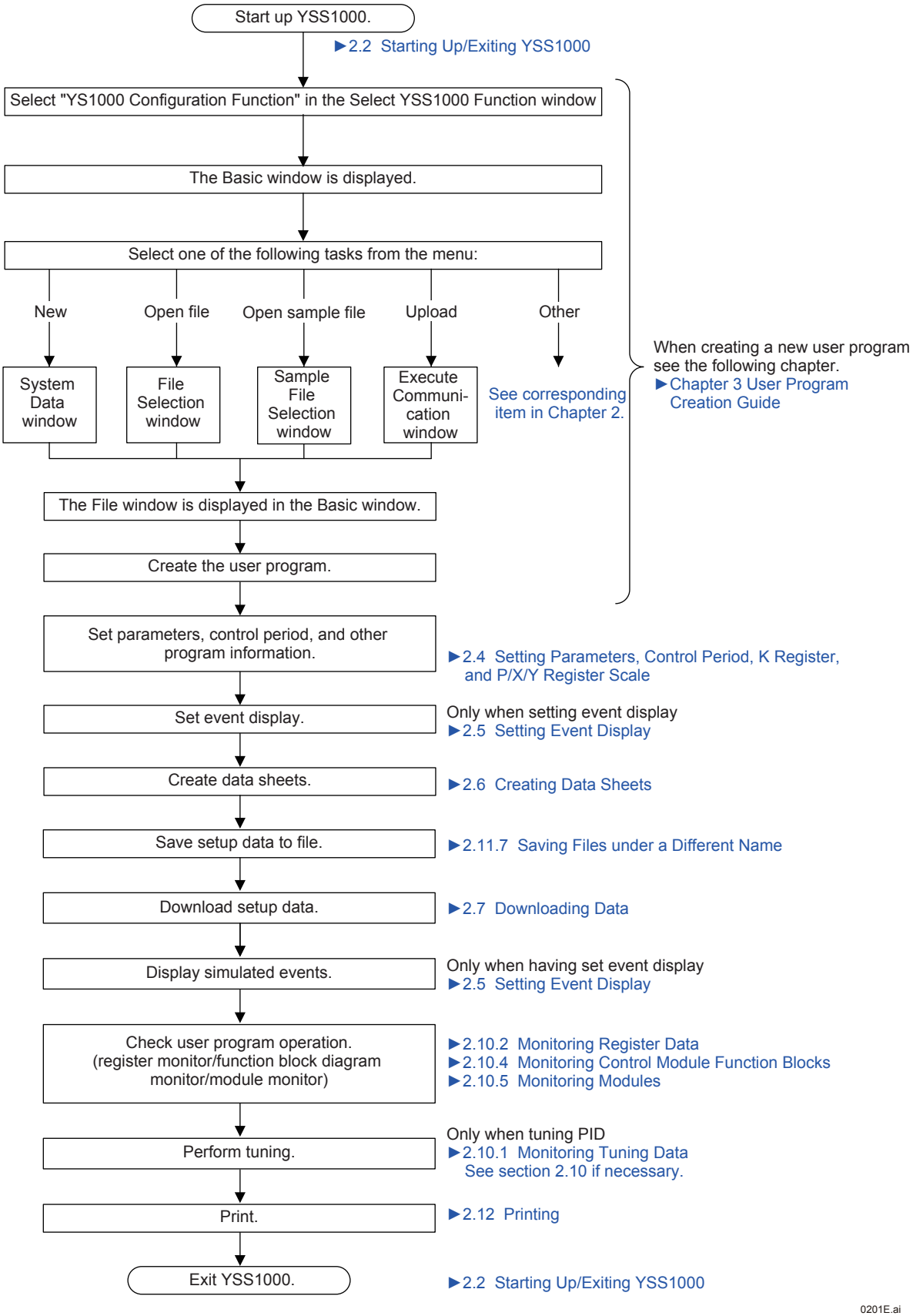
The communication parameters on the YS100 need not be set.

► Operation of YS100: YS100 User's Manual

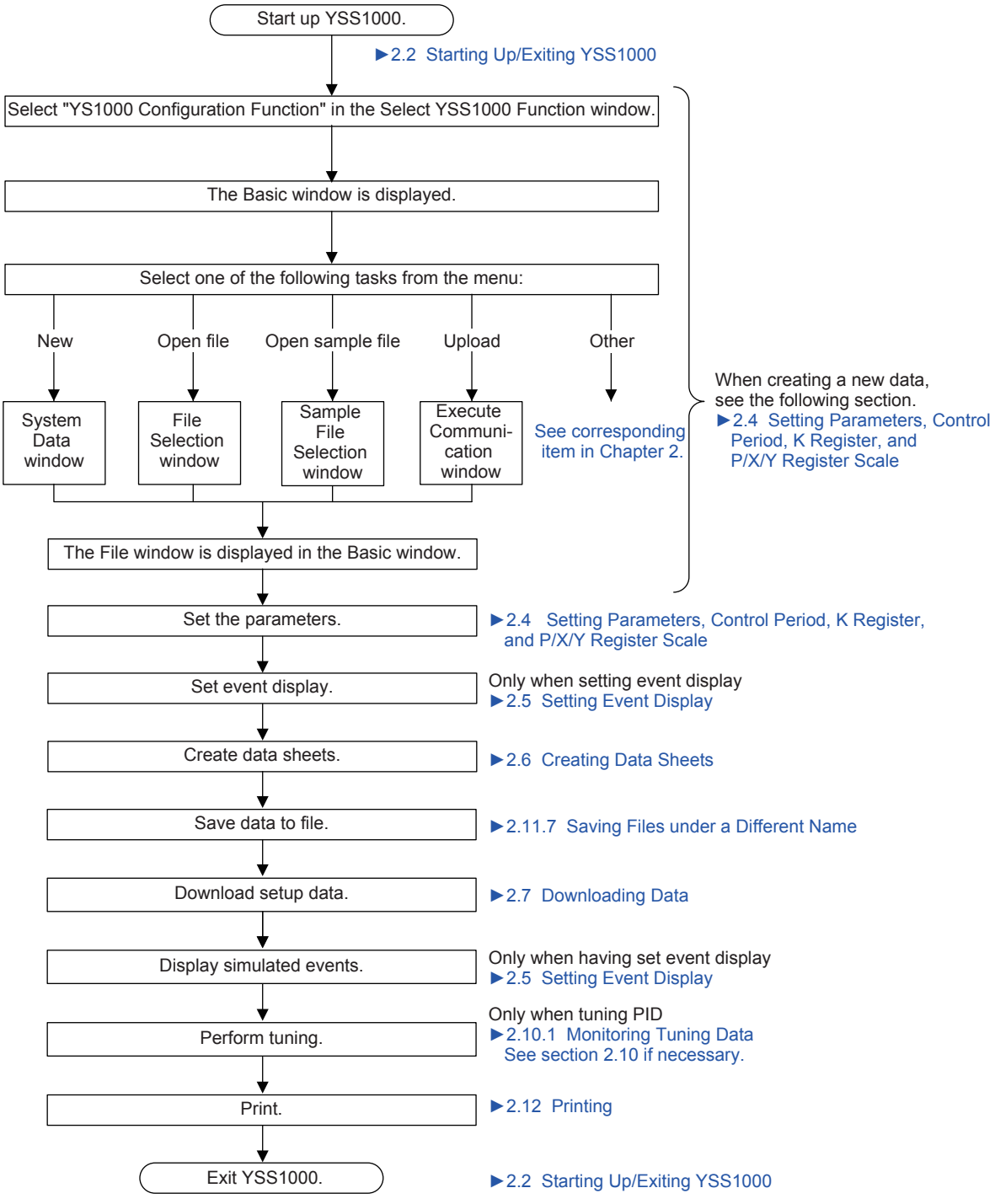
2.1 Setup Flow

The YSS1000 Operation Guide explains how to set up YS1000 parameters, create data sheets, set events, monitor, download, upload, manage files, print, and perform other operations. For details on writing a user program for the YS1700, see "Chapter 3 User Program Creation Guide."

YS1700 Programmable mode setup flow



YS1700/YS1500 Multi-function mode, YS1310, YS1350 and YS1360 setup flow



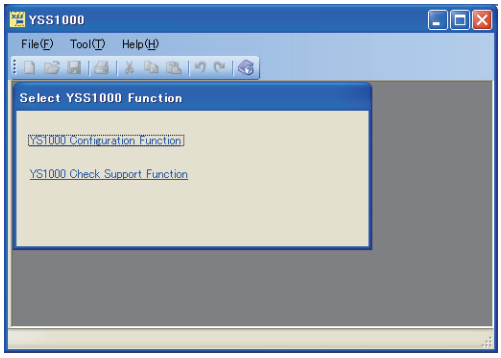
0202E.ai

2.2 Starting Up/Exiting YSS1000

Start up YSS1000.

Operation

1. Click Windows **Start**>**All Programs**>**YS1000 Series**>**YSS1000**.



0203E.ai

Exit YSS1000.

Operation

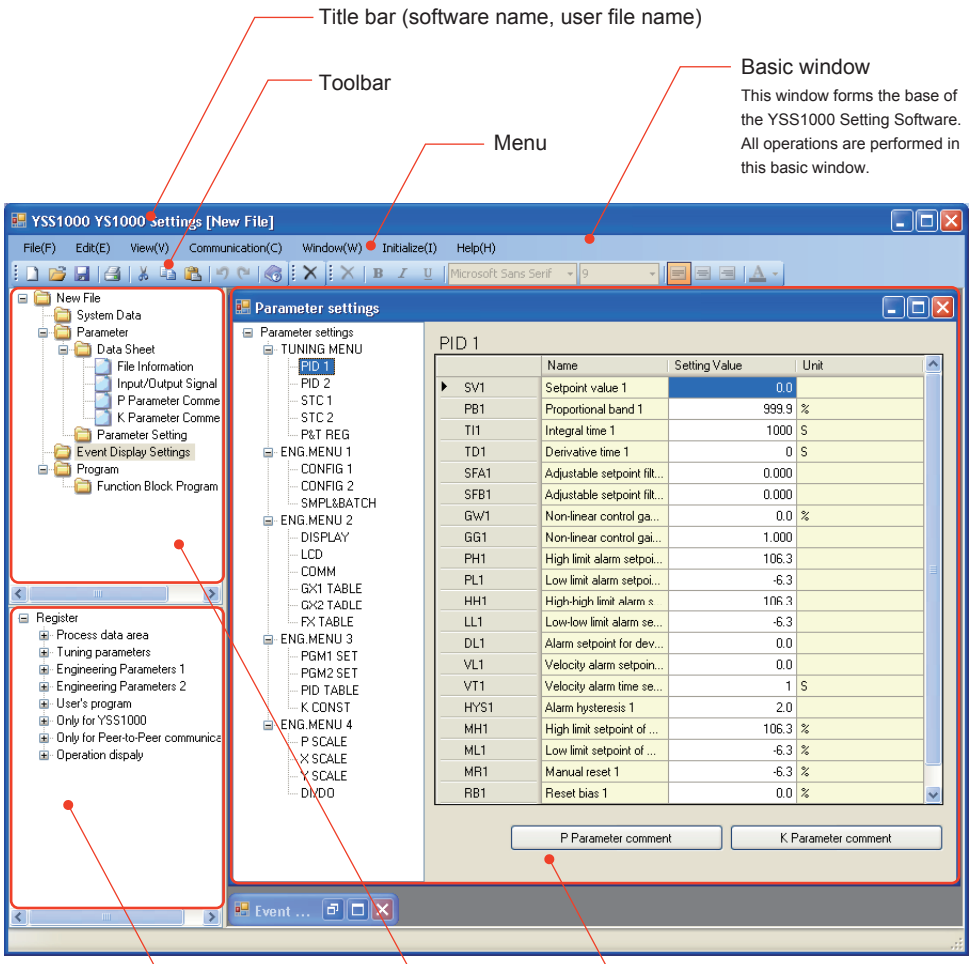
1. Click **File**>**Exit** from the menu or at the top right of the window.

Note

Save the data you are currently working on as necessary.

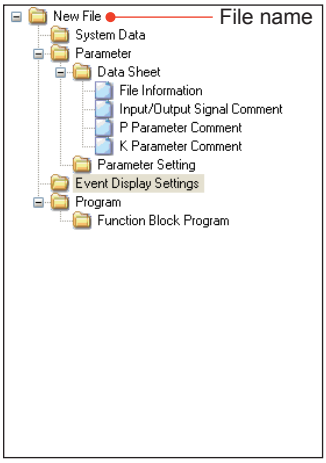
► Saving data: "2.11 File Management Operations"

2.3 Names of Parts in Windows



0204E.ai

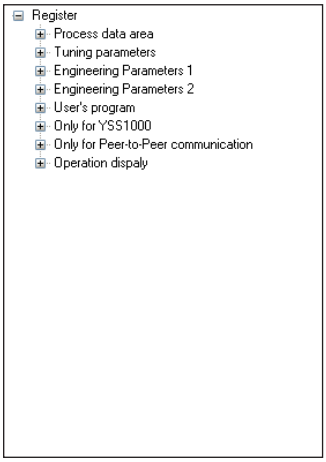
File window



0205E.ai

Item	Operation
Window display/hide method	Click **View**>**File Window** from the menu, and check the check mark to display the window. To hide a window, cancel the check mark.
Open page	Click the folder of the window you want to display. The corresponding window is displayed. To open the Parameter Settings window, click "Parameter Settings."
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder.

Register window



0206E.ai

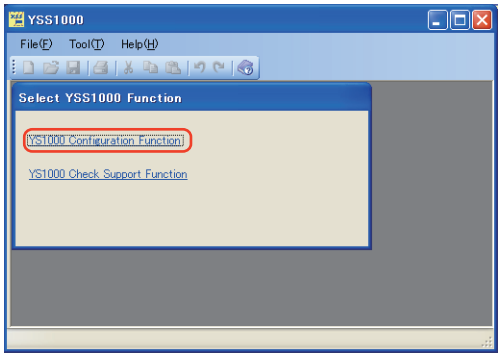
Item	Operation
Window display/hide method	Click **View**>**Register Window** from the menu, and check the check mark to display the window. To hide a window, cancel the check mark.
Set registers	Drag-and-drop registers to areas where they can be registered. The registers that can be registered differ according to the Setting window or the Monitor window. The registers that cannot be registered cannot be registered to the registration area.
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder. Right-click selects [Expand] or [Collapse] in the shortcut menu. When the tree is expanded, the register seach function can be used by input from the keyboard.

2.4 Setting Parameters, Control Period, K Register, and P/X/Y Register Scale

The following procedure is for performing new settings.

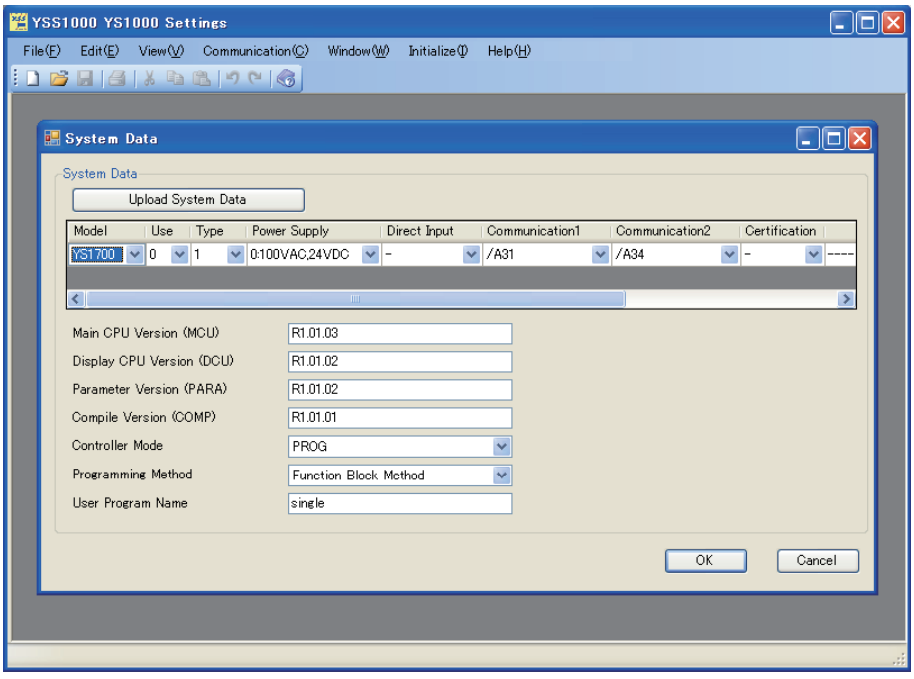
Operation

1. Start up YSS1000, and click "**YS1000 Configuration Function**" in the Select YSS1000 Function window.



0207E.ai

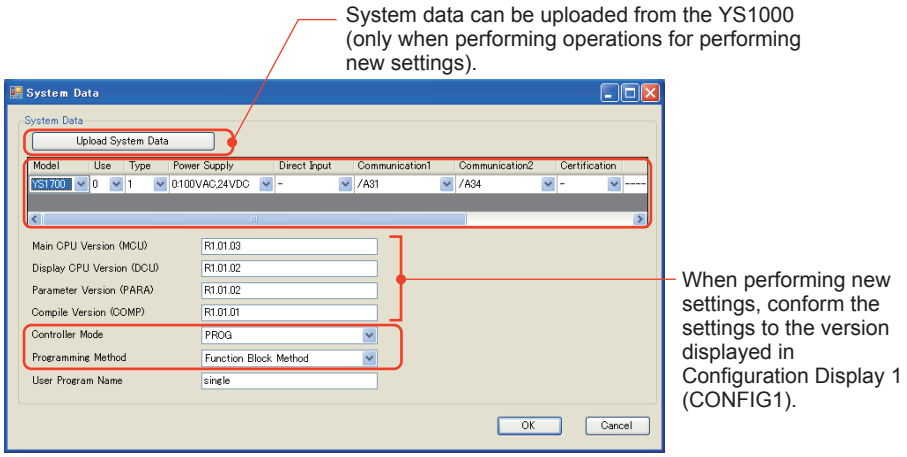
2. Click **File>New** from the menu in the Basic window or on the toolbar to display the **System Data** window.



0208E.ai

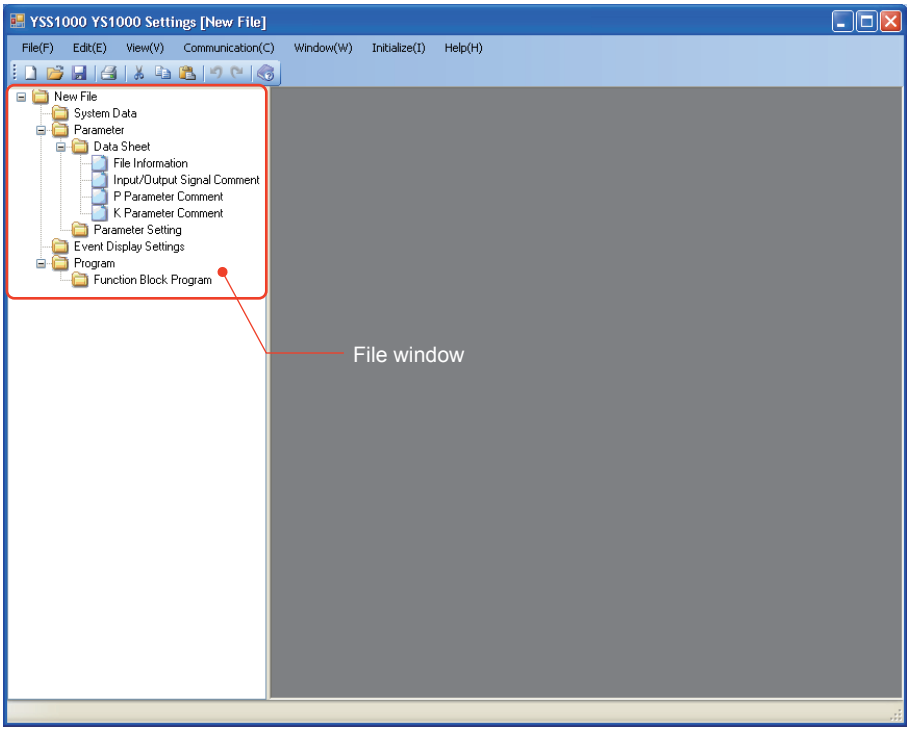
3. Select the model name, suffix code and optional code of the controller to be set, then make the following selections, and click **OK**.

In YS1700 Programmable mode:
Set the controller mode to "PROG."
Select either "Function Block Method" or "Text Method" as the programming method.
In YS1700 Multi-function mode or in case of YS1500:
Select the controller mode "SINGLE", "CAS" or "SELECT."
In case of YS1310, YS1350 or YS1360:
None



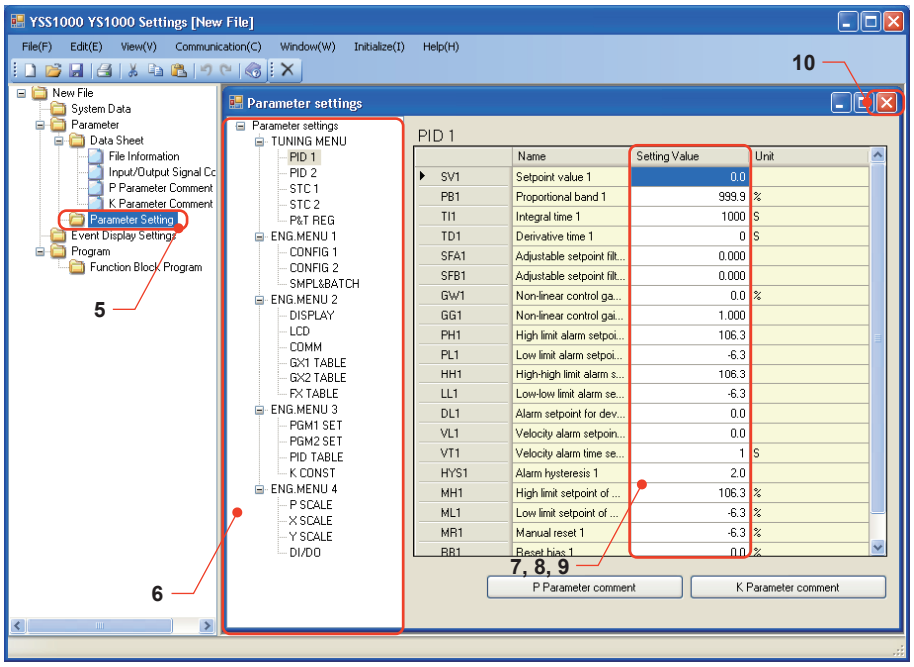
0209E.ai

4. The File window is displayed on the Basic window.



0210E.ai

5. Click "Parameter Settings" in the File window to display the Parameter Settings window on the right side.



0211E.ai

6. Select parameter "MENU" to set up, then the Parameter List is displayed on the Parameter Settings window.

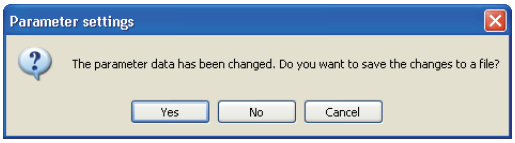
7. Click the setting value cell to enable entry of numerical values.

8. Enter the desired value, and press ENTER.

9. Repeat Steps 6 to 8 of the procedure to set other parameters.

10. To end the parameter setting, click  at the top right of the Parameter Settings window. When the data you are currently working on has not been saved, the following dialog box is displayed:

To save the settings, click **Yes**.
To discard the settings, click **No**.
To return to parameter setup, click **Cancel**.



0212E.ai

► Saving data to file: "2.11 File Management Operations"
► Setting P parameter comments and K parameter comments: "2.6 Creating Data Sheets"
► Explanation of parameters: Operation Guide (printed version) and User's Manual (CD-ROM) of respective controllers

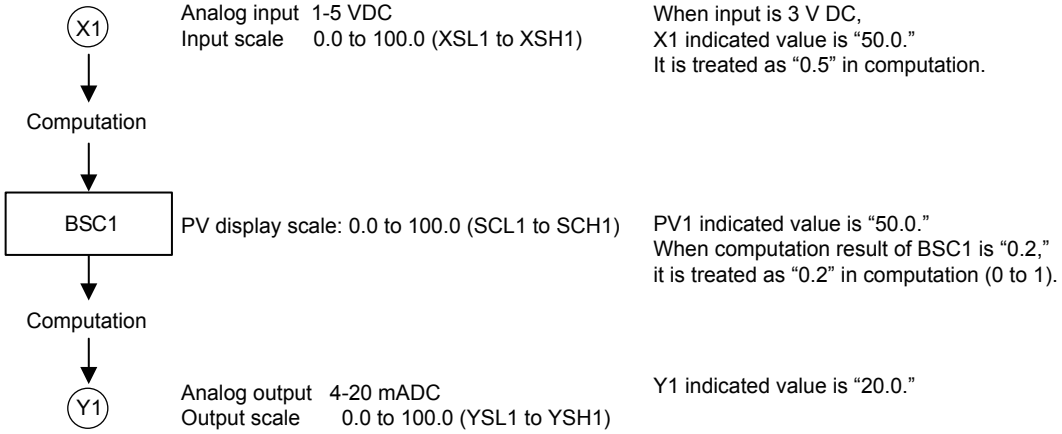
Description

Control period: Can be set in the YSS1000 Setting Software only when the YS1700 is in the programmable mode.

K register: The number of K registers available for user program is 100. However, the number of registers displayed on YS1700 is 60.

P/X/Y scale: P/X/Y scale when creating user programs
P scale is the scale for P registers (P01 to P30).
X scale is the scale for X registers (X1 to X8). (Analog inputs 1 to 8)
Y scale is the scale for Y registers (Y1 to Y6). (Analog outputs 1 to 4, analog output registers 5 and 6)
The factory default of P/X/Y scale is 0.0 to 100.0 (no unit).

Example 1:



0212E-2.ai

P scale

P scale creating user program using P registers (variable registers) with P scale set to factory default, "0.0" is treated as "0" and "100.0" is treated as "1" in the user program. To write "2" to a register, set "200.0."
A desired value can be set as it is if the scale (PSL to PSH) is set to "0 to 1."

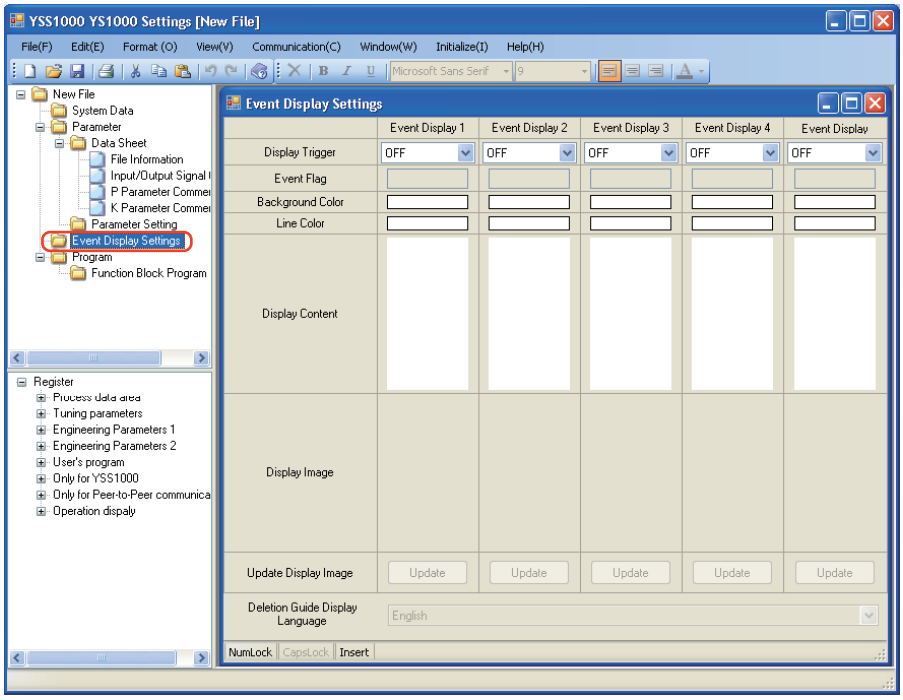
Example 2:
When the PV display scale is set to "0.0 to 500.0 ºC," and P1 register is userd for high limit alarm setpoint parameter, set the same scale (PSL1 = 0.0, PSH1 = 500.0) as PV display scale.

2.5 Setting Event Display

Operation

The procedure up to display of the File window is the same as Steps 1 to 4 of the previous section.

1. Click "**Event Display Settings**" in the File window to display the Event Display Settings window on the right side and the Register window at the same time.

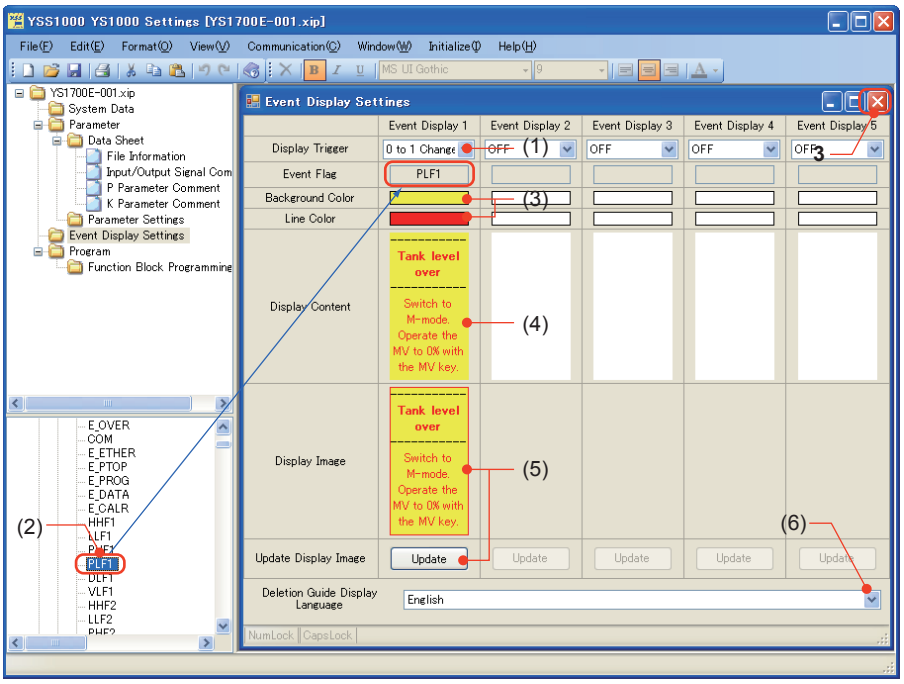


0213E.ai

2. In the Event Display Settings window, perform the following settings. In the figure on next page, an event is displayed when the PV high limit alarm occurs (when setting changes status from 0 to 1).

(1) Set the display trigger to a setting other than OFF.
(2) From the Register window, drag-and-drop the register to be used as the trigger to the event flag ( ).
(3) Select the background color and line color by clicking and selecting in the Color dialog box.
(4) Enter the text to be displayed on the controller's LCD panel in the Display Content. Font size, color and other attributes can be edited.
(5) Click the Update button to display the bitmap data in the Display Image.
(6) Click to select Deletion Guide Display Language.
(Deletion Guide Display Language: When English is selected, the message "Press SHIFT key for 3 seconds to CLOSE the Event Display" is displayed.

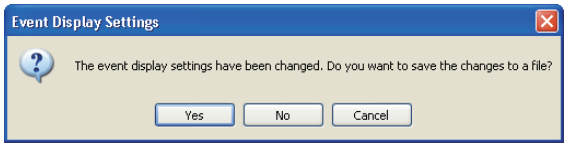
► Deletion Guide Display: Procedure 3 of simulated event display on page 2-15



0214E.ai

3. To end the event display settings, click ☒ in the Event Display Settings window. When the data you are currently working on has not been saved, the following dialog box is displayed:

To save the settings, click **Yes**.
To discard the settings, click **No**.
To return to setting events, click **Cancel**.



0215E.ai

- ► Saving data to file: "2.11 File Management Operations"
- ► Setting P parameter comments and K parameter comments: "2.6 Creating Data Sheets"
- ► Explanation of parameters: Operation Guide (printed version) and User's Manual (CD-ROM) of respective controllers

Description

The "Event Display Settings" function displays a message (English) on the controller's Operation Display as guidance for the operator when an alarm or other event occurs on the controller.
Set the event type, guidance details, guidance colors, and other settings, and download these to the controller.

Display Trigger When 0 to 1 change: Displayed when an event occurs
When 1 to 0 change: Displayed when an event is canceled

If multiple events occur simultaneously, they are displayed in an overlaid condition. The order of priority is (high) event 1 > 2 > 3 > 4 > 5 (low), and the event with the highest priority is displayed on top.

- ► Event Display function: "Using Event Functions" in the User's Manual (CD-ROM) of respective controllers

Note

Place the Event Display Settings window so that it is not hidden in the File, Module or Register window, and click the Update button. If the Update button is pressed with the Event Display Settings window hidden, the display image will not be correct.

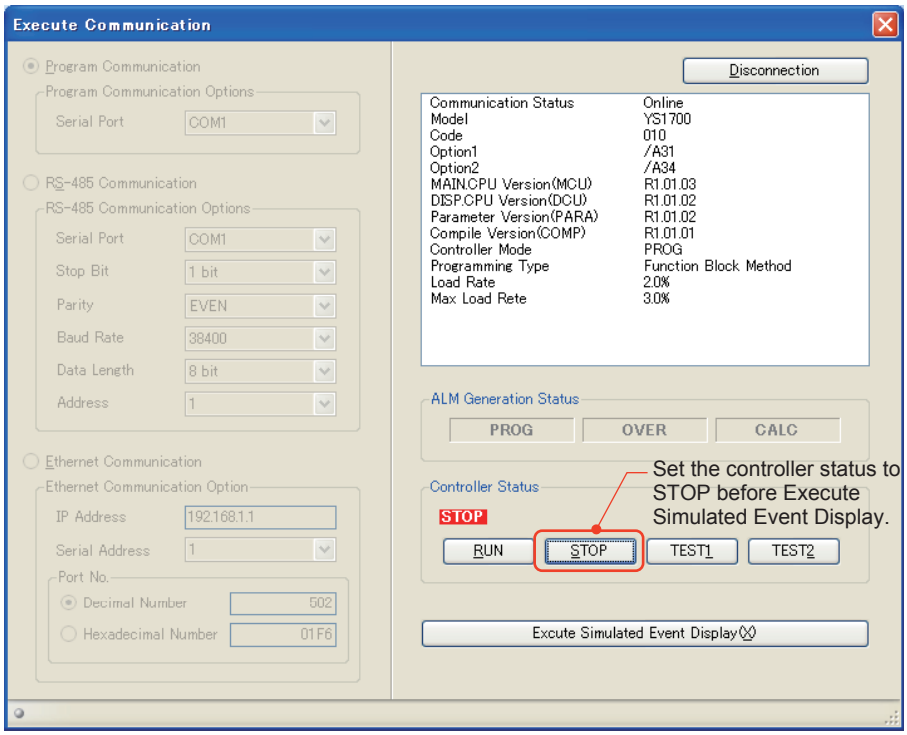
Register search function

When the Register window is active, right-click the window, select [Expand], and enter the target register in alphanumerics on the keyboard.

Simulated event display

Operation

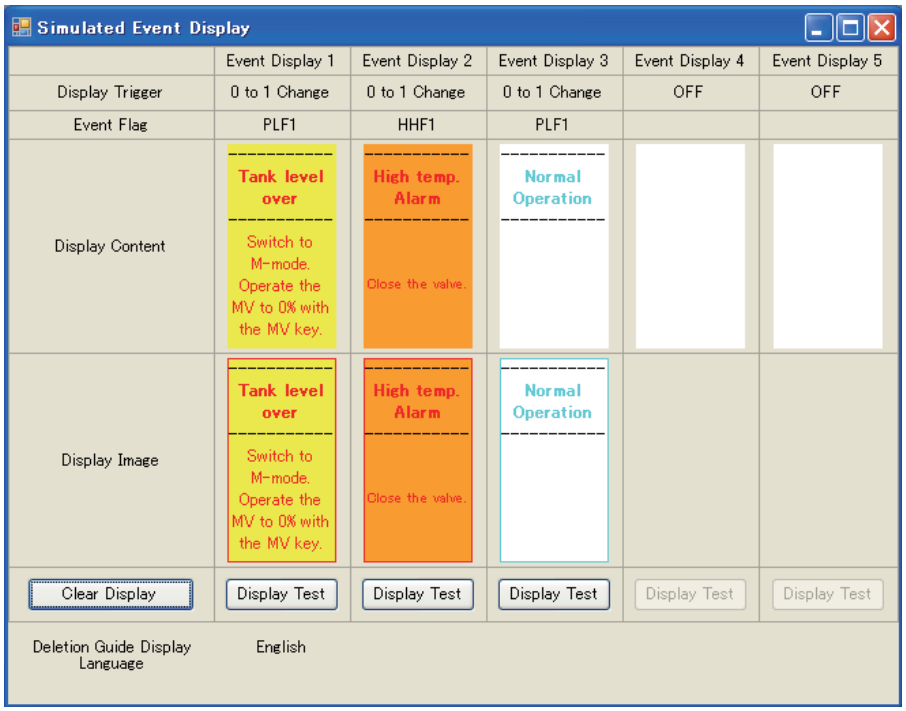
1. Click **Communication>Simulated Event Display** from the menu to display the Execute Communication window.



0216E.ai

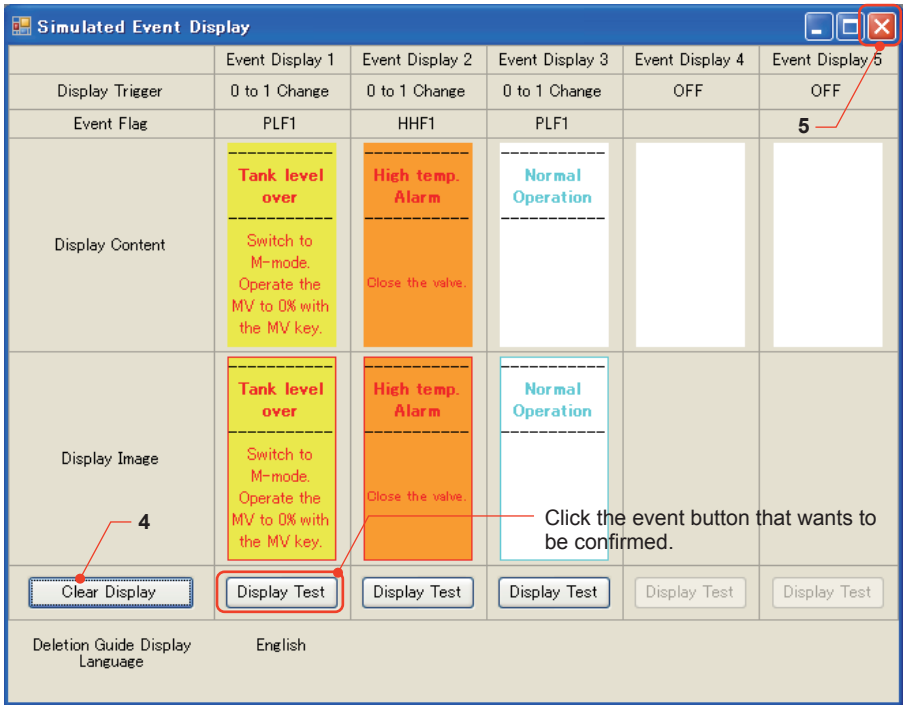
2. Set the communication conditions, and click **Execute Simulated Event Display** to display the Simulated Event Display window.

(The content set to the YS1000 is displayed.)

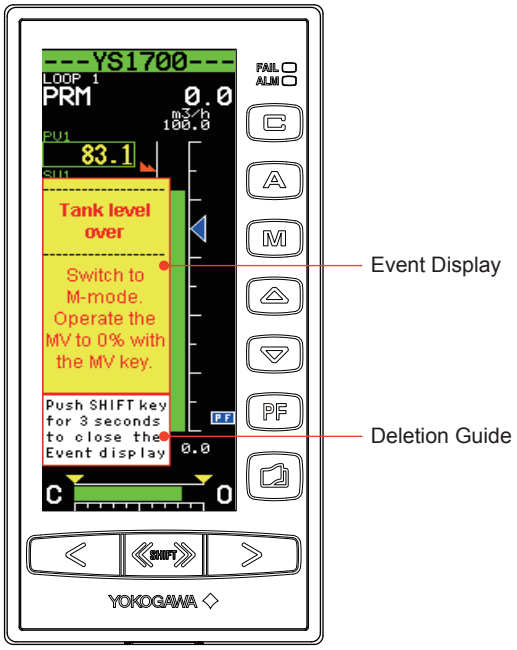


0217E.ai

3. Click the button for the event display to be checked. Events are displayed for simulation on the YS1000's LCD regardless of the status of the event flag.



0218E.ai



0218-2E.ai

4. To clear the Simulated Event Display, click **Clear Display**.

5. To exit the Simulated Event Display, click ☒ at the top right of the Simulated Event Display window.

Description

The "Simulated Event Display" is a function for displaying an event display downloaded to the YS1000 for simulation on the YS1000's LCD regardless of the status of the event flag. When using this function, set the YS1000 to a stop status (STOP).
Before using this function, the setup data containing the event display setting function must be downloaded to the YS1000, and communication must be enabled beforehand.
See "1.2 Connecting the YS1000 to a PC and Setup Parameters."
Settings cannot be changed in the Simulated Event Display window.

► Function description: "Using the Event Function" in the respective User's Manuals.

<Execute Communication window>

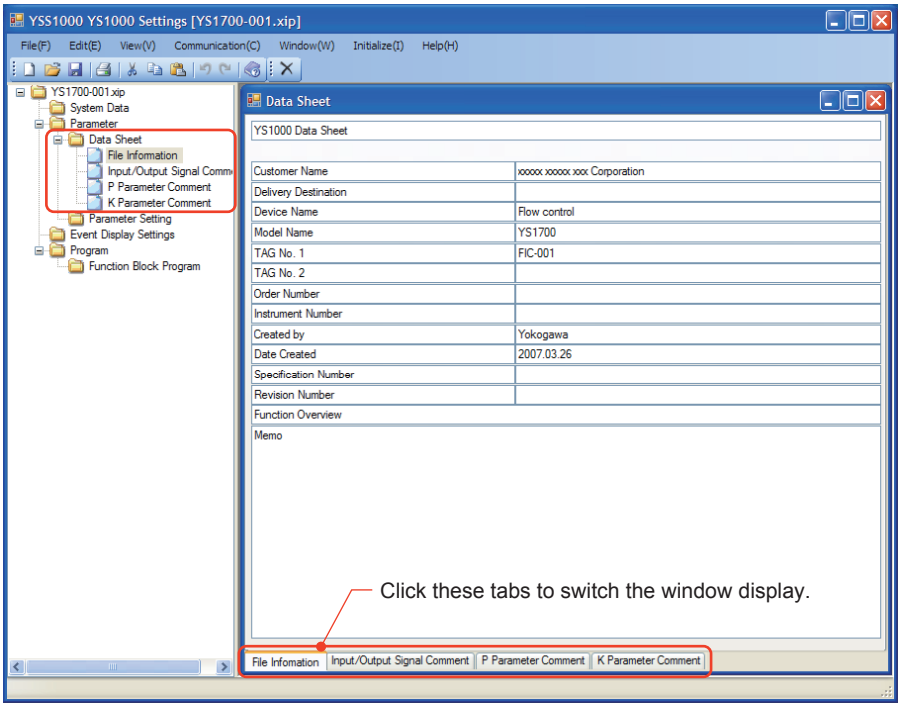
- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

2.6 Creating Data Sheets

Operation

1. Click **Data Sheet>File Information** in the File window to display the Data Sheet window on the right side.

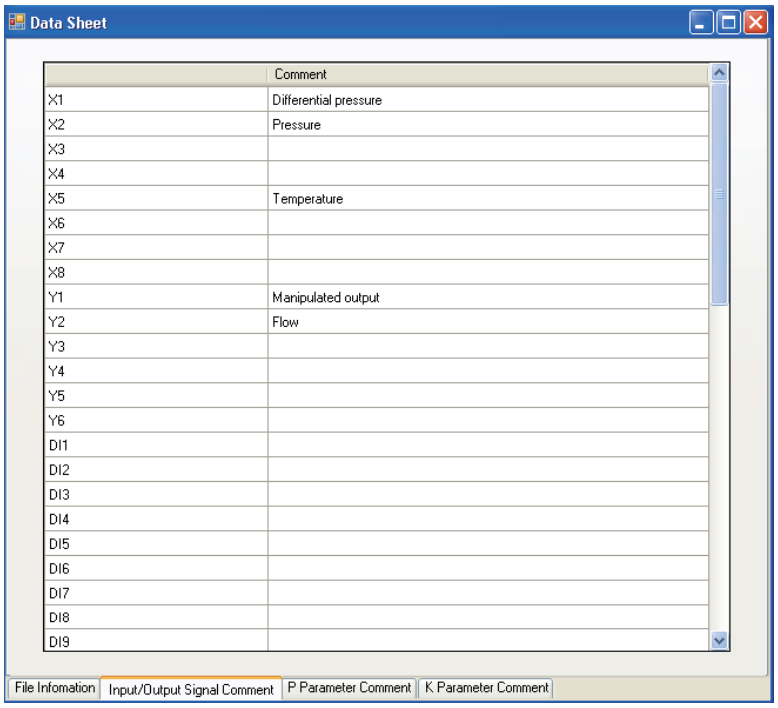
The figure below shows the file information.



0219E.ai

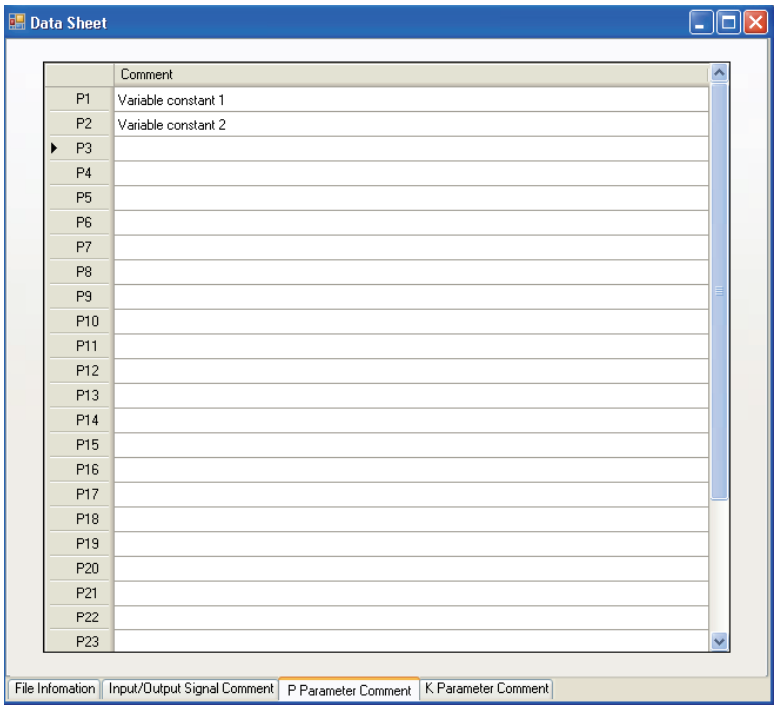
The window display changes as follows when the following are clicked:
Data Sheet>Input/Output Signal Comment in the File window
Data Sheet>P Parameter Comment in the File window
Data Sheet>K Parameter Comment in the File window

Input/Output Signal Comment window



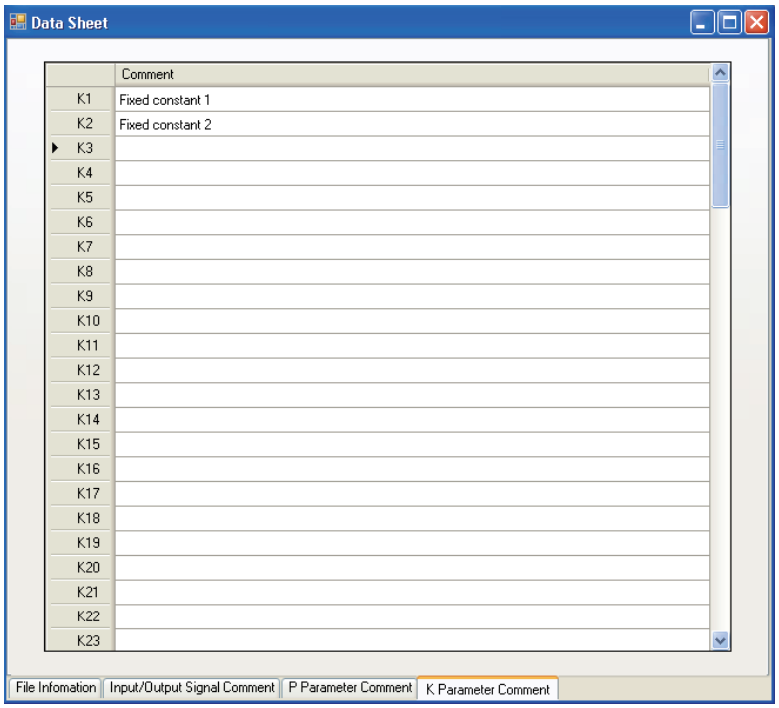
0220E.ai

P Parameter Comment window



0221E.ai

K Parameter Comment window



0222E.ai

Description

"Data sheet creation" is a function for creating data sheets for submission to the customer. Data sheets can also be printed.
When data is downloaded to the YS1000, data sheets also are downloaded at the same time.
The P Parameter Comment and K Parameter Comment windows are displayed when the YS1700 is in the programmable mode.

Window	Max. Number of Entered Characters
File Information	Left cell: 32 characters, right cell: 40 characters, Memo field: 1200 characters
Input/Output Signal Comment	Left cell: 20 characters, right cell: 40 characters
P Parameter Comment	60 characters
K Parameter Comment	60 characters

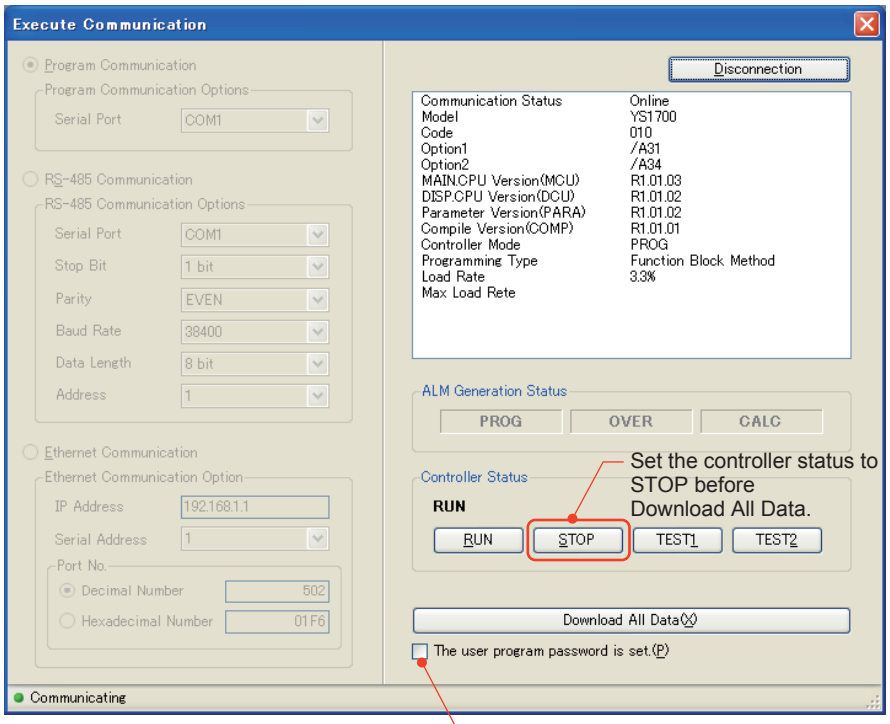
► Printing: "2.12 Printing"

2.7 Downloading Data

Download All

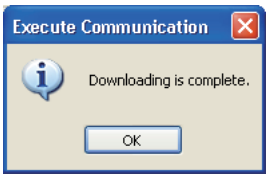
Operation

1. Click **Communication>Download All Data** from the menu to display the Execute Communication window.



0223E.ai

2. Set the communication conditions, and click **Download All Data**. The following message is displayed when the download is completed:



0224E.ai

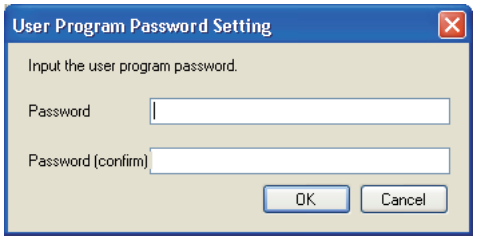
If the following message is displayed after execution of the download is confirmed, follow the instructions in the message:

When the data you are currently working on has not been saved, a dialog box prompting you to save the data is displayed.
- To save the data, click **Yes**.
- To cancel the download, click **Cancel**.

► Saving data: "2.11 File Management Operations"

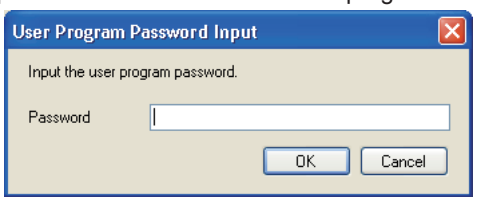
When the controller mode of the data to be downloaded differs from that of the YS1500/YS1700, a dialog box prompting you to change the controller mode is displayed.
- To change the controller mode, click **Yes**. The download is continued.
- To choose not to change the controller mode, click **No**. The download is canceled.

When the download data contains a user program, a dialog box prompting you to set the program password is displayed. Set a password using eight or less alphanumerics. When choosing not to set a password, do not enter anything and click **OK**. Once a password is set, you can no longer choose not to set a password.



0225E.ai

When overwriting a YS1000 that is set with a program password, enter the program password to download new user program.



0226E.ai

Description

Data that is downloaded by "Download All" includes the following: user file name, system data (System Data window display items), data sheet data, parameter data, event display data, and user program data.
User program data is valid when the YS1700 is in the programmable mode.
If model, type, communication 1, and communication 2 of the system data are not corresponding, data cannot be downloaded.

The "user program password" is a function for preventing user programs downloaded to the YS1700 from being accessed or overwritten inadvertently.
Once a password is set, the password must be entered to perform an upload, download, compare, or module monitoring.
The default password is "not set." Passwords are limited to eight alphanumeric characters, and entry of passwords is case-sensitive.

Individual types of data can be downloaded by the following operations:
- Click **Communication**>**Download Parameter Data** from the menu.
- Click **Communication**>**Download Event Display Data** from the menu.
- Click **Communication**>**Download User Program Data** from the menu.

Data sheet data is included in the parameter data.

<Execute Communication window>
- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

Note

During a download, do not disconnect the connector cable, or turn the YS1000 OFF.

Note

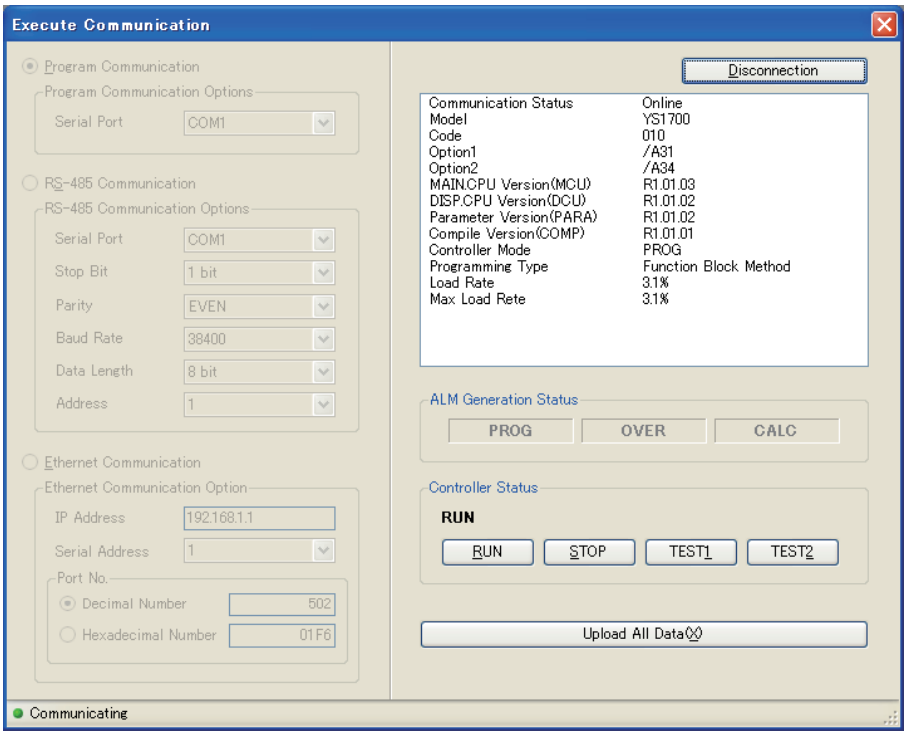
When data is downloaded with the STC mode selection (STC) parameter set to ATSTUP, the mode changes to DISP.

2.8 Uploading Data

Upload All

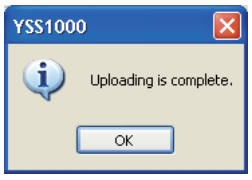
Operation

1. Click **Communication**>**Upload All Data** from the menu to display the Execute Communication window.



0227E.ai

2. Set the communication conditions, and click **Upload All Data**. The following message is displayed when the upload is completed:



0228E.ai

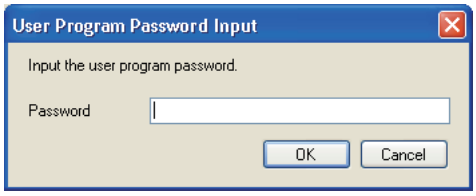
If the following message is displayed after execution of the upload is confirmed, follow the instructions in the message:

When the data you are currently working on has not been saved, a dialog box prompting you to save the data is displayed.

- To save the data, click **Yes**.
- To cancel the upload, click **Cancel**.

► Saving data: "2.11 File Management Operations"

When a program password is set to the user program to be uploaded, a dialog box prompting you to enter the program password is displayed. Enter the password and click **OK**.



0229E.ai

Description

Data that is uploaded by "Upload All" includes the following: user file name, system data (System Data window display items), data sheet data, parameter data, event display data, and user program data.
User program data is valid when the YS1700 is in the programmable mode.

Passwords are limited to eight alphanumeric characters, and entry of passwords is case-sensitive.

Individual types of data can be uploaded by the following operations.
The following operations can be performed only when a file is open.
Also, the data of opened files is overwritten with the uploaded data.

- Click **Communication**>**Upload Parameter Data** from the menu.
- Click **Communication**>**Upload Event Display Data** from the menu.
- Click **Communication**>**Upload User Program Data** from the menu.

Data sheet data is included in the parameter data.

<Execute Communication window>

- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

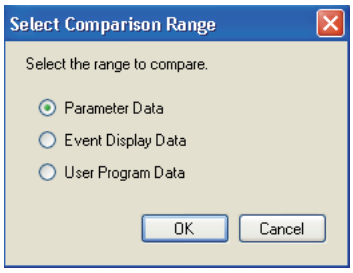
Note

During an upload, do not disconnect the connector cable, or turn the YS1000 OFF.

2.9 Comparing Data with YS1000 Data

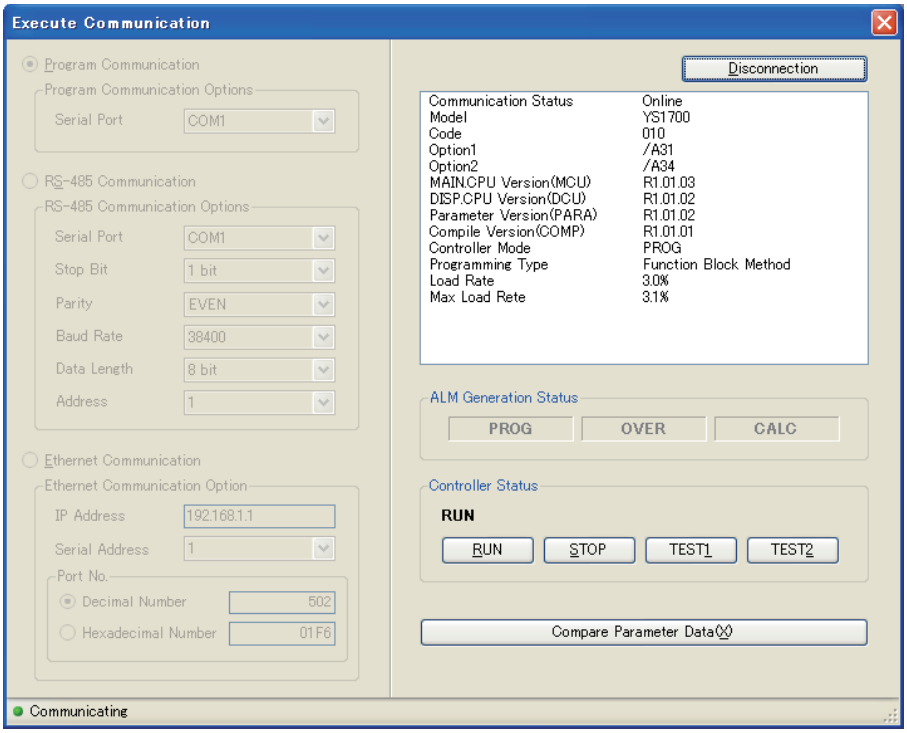
Operation

1. Click **Communication>Compare Communication** from the menu to display the Select Comparison Range window.



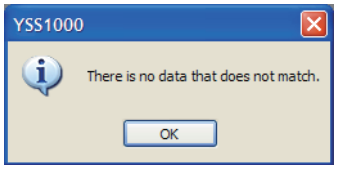
0230E.ai

2. Select the comparison range, and click **OK**. The Execute Communication window is displayed.



0231E.ai

3. Set the communication conditions and click **Compare Parameter Data** to start the comparison. If there is matching data, the following message is displayed. If there is a data mismatch, the mismatching data is displayed.



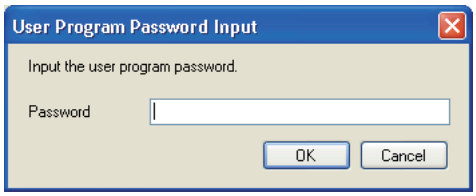
0232E.ai

If the following message is displayed during execution of a comparison, follow the instructions in the message:

"Unable to perform a comparison because the system data does not match. Comparison was stopped."

Probable causes of this error are a different parameter version or compile version.
- To cancel the comparison, click **Yes**.
- To continue the comparison, click **No**.

When a program password is set to the user program to be compared, a dialog box prompting you to enter the program password is displayed. Enter the password and click **OK**.



0233E.ai

Description

Passwords are limited to eight alphanumeric characters, and entry of passwords is case-sensitive.
Input terminals, arrangement information, and program comments are not compared.

<Execute Communication window>
- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Match this parameter setting to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

Note

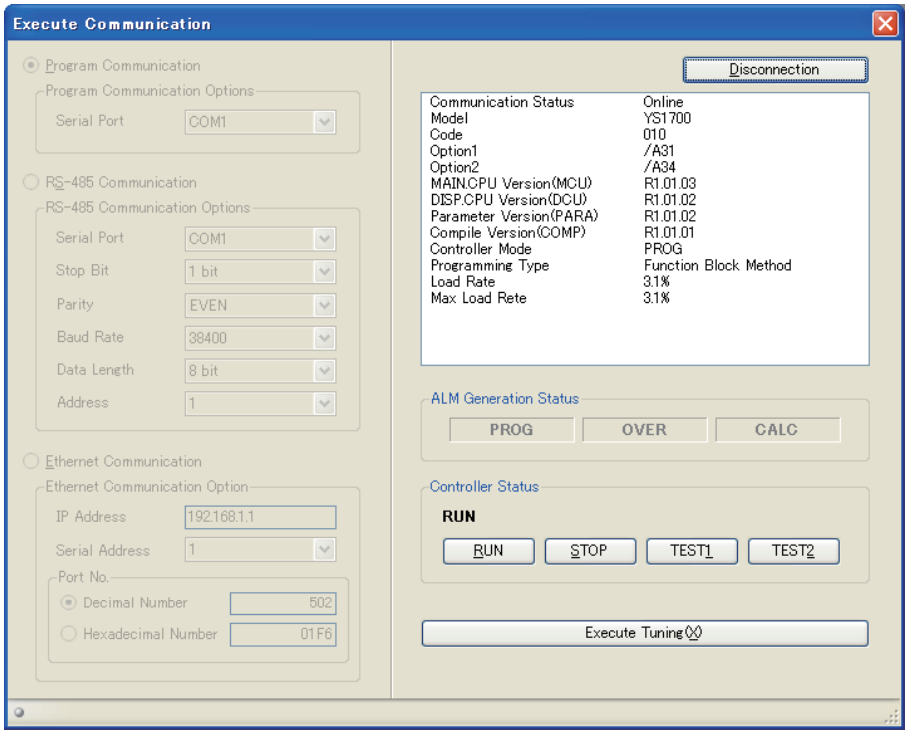
During a comparison, do not disconnect the connector cable, or turn the YS1000 OFF.

2.10 Monitoring Data

2.10.1 Monitoring Tuning Data

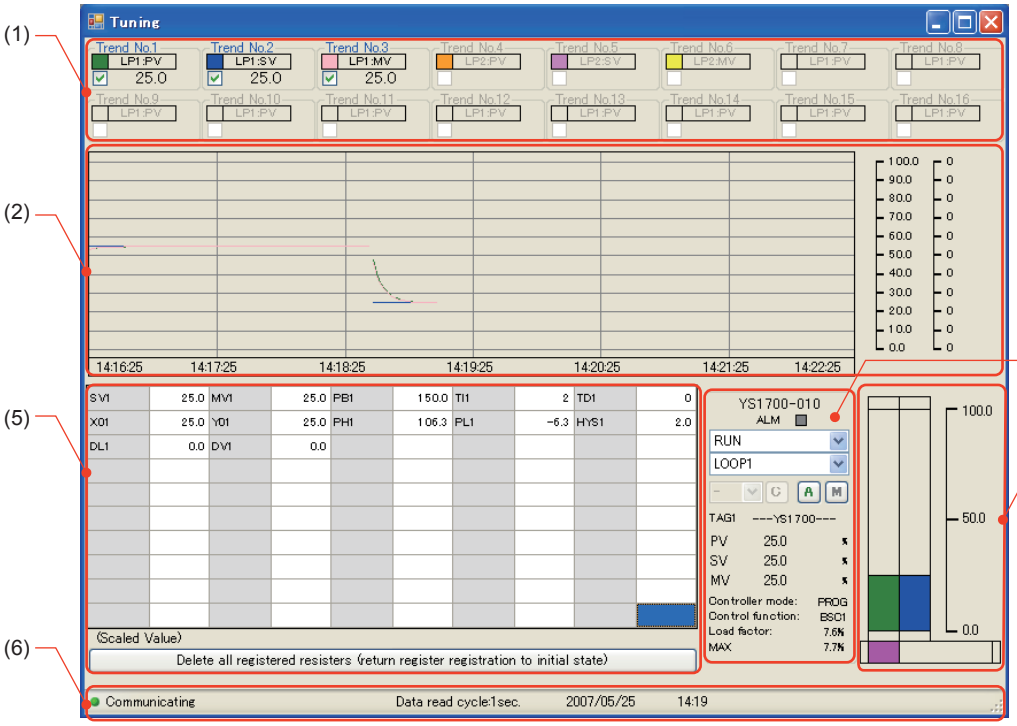
Operation

1. With the Basic window displayed, click **Communication**>**Tuning** from the menu to display the Execute Communication window.



0234E.ai

2. Set the communication conditions and click **Execute Tuning** to display the Tuning window.



0235E.ai

3. Change the proportional band (PB), integral time (TI) and derivative time (TD) according to the register values of the Register Monitor while viewing the trends of process variable (PV), setpoint value (SV) and manipulated output variable (MV).

4. To close the Tuning window, click ☒ at the top right of the window.

When the data you are currently tuning has not been saved, a dialog box prompting you to save the data is displayed.

- To save the data, click **Yes**.
- To discard the data, click **No**.
- To return to the tuning window, click **Cancel**.

► Saving data: "2.11 File Management Operations"

Description

"Tuning" is a function for tuning a YS1000 via a communication link established between the PC and the YS1000.
This function is mainly used when the system is initially started up. The recommended trend sampling period is one day.
The tuning function provides options for sampling and displaying PV, SV and MV values as trend data. The trend data can also be saved to file in CSV format. Trend data for up to 65,000 samples can be saved regardless of the data read cycle. When the number of trend data samples exceeds 65,000, the data is automatically saved to a different file.
Example: When the 65,001st data was sampled in 2007 (year), on 05 (May), 20th, at 21 (hours), 30 (minutes), 50 (seconds), the file name is as follows: 2007_05_20_21_30_50.CSV

When the monitor display is switched (internal values and scaled values) during the monitoring of tuning data, the data at the point in time the display is switched is also saved as trend data.
Example: When 5 minutes is set as the data read cycle and the display is changed at 13:09, trend data at 13:00, 13:05, 13:09, 13:14, and 13:19 is saved as trend data.

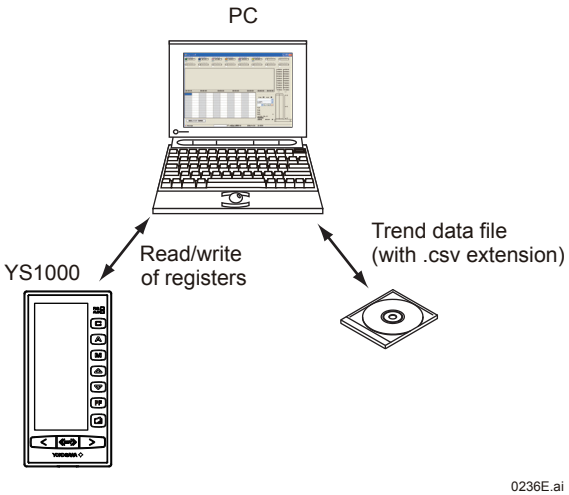
<About switching display of monitoring values>
It may sometimes be difficult to understand how to check the operation of user programs, because parameters with scale (Xn, PVn, SVn, Pn, etc.) are displayed as scaled values and T register and module output registers are displayed as internal values in the scaled value display. However, if you select internal value display, checking the operation of user programs becomes easy because parameters with scale are also displayed as internal values.

Note

When scale parameters (SCHn, SCLn, SCDPn, etc.) are changed during the display of internal values, close and then re-open the Tuning window to update the displayed values to the correct values.

Changes made to register values are reflected on the YS1000.
Multiple Tuning windows cannot be displayed.

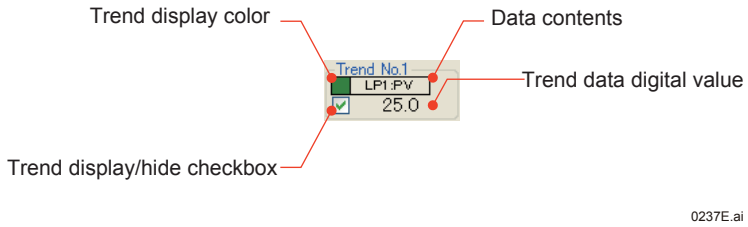
To perform tuning, enable communication with the YS1000. See "1.2 Connecting the YS1000 to a PC and Setup Parameters."



<Execute Communication window>

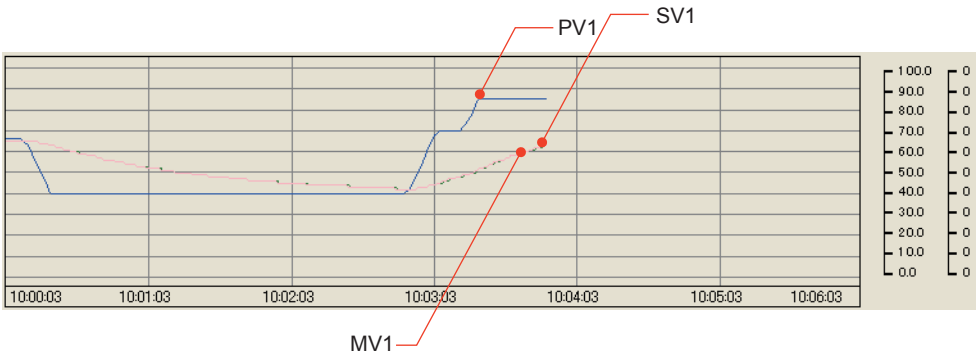
- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

(1)Trend selection data and digital value display area



Indicated Item	Explanation
Number of displayed trends	Max. 6 data
Trend display/hide checkbox	Select whether or not to display trend data as a graph by selecting this checkbox. Even if trend data is hidden, trend data digital values are displayed, and are output as a CSV format file.
Trend display color	PV1: green, SV1: blue, MV1: pink, PV2: orange, SV2: purple, MV2: yellow If you click the trend display color, the Color dialog box appears and you can change the color.
Data contents	LP1: LOOP1, LP2: LOOP2 Content: PV, SV, MV
Trend data digital value	Data (7 digits including sign and decimal point) that is read from the YS1000 is displayed.

(2)Trend display area

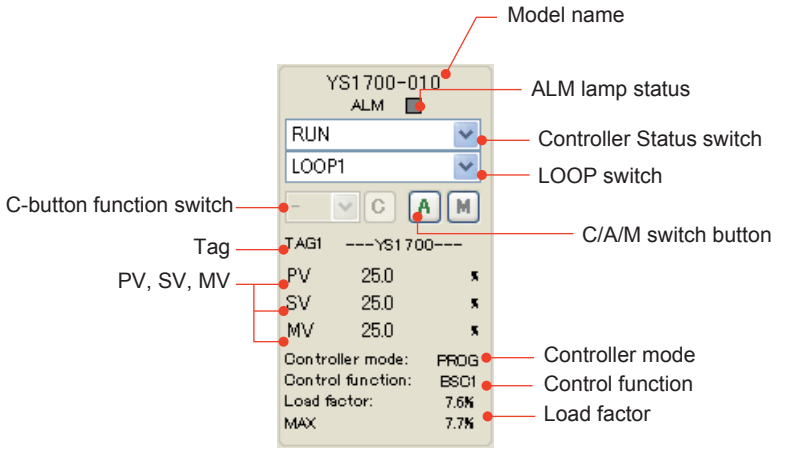


0238E.ai

Indicated Item	Explanation
Scale display	100% value of scale (SCH1, SCH2) to 0% value of scale (SCL1, SCL2) Scale divisions: 11
Trend	Trends are displayed within the range -6.3 to 106.3% (0 to 100% scale). When a trend value exceeds this range, the trend display is limited. Howevzer the actual data that is read is saved to CSV file as it is. Trends are displayed progressively from the left side of the display area.
X-axis (time axis) scale	This is automatically calculated according to the data read cycle.
Background color	If you right-click the trend graph and select Background Color from the shortcut menu, the Color dialog box appears and you can change the color.

(3)LOOP information

The information of the LOOP selected by the LOOP switch is displayed.



0239E.a

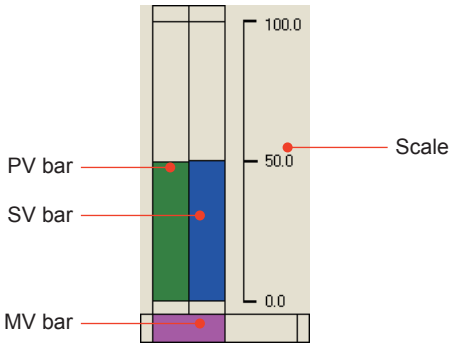
<table>
<tr><th>Indicated Item</th><th>Explanation</th></tr>
<tr><td>Model name, suffix code</td><td>The model name and suffix code that were read when the window is open are displayed.</td></tr>
<tr><td>ALM lamp</td><td>This lamp lights (orange) when an alarm occurs.</td></tr>
<tr><td>Controller Status switch</td><td>The operating status is displayed.</td></tr>
<tr><td>LOOP switch</td><td>Switches between LOOP1 and LOOP2. LOOP2 can be selected only in the following cases:
• YS1310
• (model name: YS1500 or YS1700) and (controller mode is cascade mode or selector mode)
• (model name: YS1700) and (controller mode: programmable mode) and (control function: dual-loop control, cascade control or selector control)</td></tr>
<tr><td>Tag</td><td>The tag (max. 12 characters) of the currently selected LOOP is displayed.</td></tr>
<tr><td>PV, SV, MV</td><td>LOOP1: PV1, SV1, MV1
LOOP2: PV2, SV2, MV2
MV1 is displayed for LOOP2 in the following cases:
• When the controller mode is the cascade mode or selector mode
• When the controller mode is the programmable mode, and the control function is cascade control or selector control</td></tr>
<tr><td>PV, SV, MV units</td><td>The unit of each LOOP is displayed. LOOP1: UNIT1, LOOP2: UNIT2, MV unit: %</td></tr>
<tr><td>C-button function switch</td><td>Switches the function of the C-button. Display and selection of the C-button differs according to the setting of engineering parameters CMOD1 and CMOD2.
<table>
<tr><th>CMOD1 or CMOD2</th><th>Displayed Options</th></tr>
<tr><td>CAS</td><td>CAS</td></tr>
<tr><td>CMP</td><td>SPC, DDC</td></tr>
</table></td></tr>
<tr><td>C/A/M switch button</td><td>Switches between the C, A and M-modes.</td></tr>
<tr><td>Controller mode</td><td>The controller mode is displayed.
PROG: Programmable mode
SINGLE: Single-loop mode
CAS: Cascade mode
SELECT: Selector mode</td></tr>
<tr><td>Control function</td><td>The control function is displayed. (i.e. the control module currently used by the YS1700 user program)
BSC1: Single-loop control
CSC: Cascade control
SSC: Selector control
BSC1, BSC2: Dual-loop control
-: Do not use control function.</td></tr>
<tr><td>Load factor, Maximum load factor (MAX)</td><td>The load factor of the program is displayed. (in YS1700 programmable mode only)
The maximum value since the Tuning window was opened is displayed as the maximum load factor.</td></tr>
</table>

Display Items by Model Name and Controller Mode

Display Content	Display Items by Model Name and Controller Mode ✓: Display –: Invalid (or not displayed)				
	YS1700 (controller mode: PROG)	YS1700 (controller mode: except PROG) YS1500	YS1310	YS1350	YS1360
Display of PV digital value and unit	✓	✓	✓	✓	✓
Display of SV digital value and unit	✓	✓	–	✓	–
Display of MV digital value	✓	✓	–	–	✓
C-button function switch (Note)	✓	✓	–	✓	✓
C-button (Note)	✓	✓	–	✓	✓
A-button	✓	✓	–	–	–
M-button	✓	✓	–	✓	✓
Controller mode	✓	✓	–	–	–
Control function	✓	–	–	–	–
Load factor	✓	–	–	–	–
Maximum load factor (MAX)	✓	–	–	–	–
PV bar	✓	✓	✓	✓	✓
SV bar	✓	✓	–	✓	–
MV bar	✓	✓	–	–	✓

Note: These items are invalid when the setting value of engineering parameters CMOD1 or CMOD2 is "-".

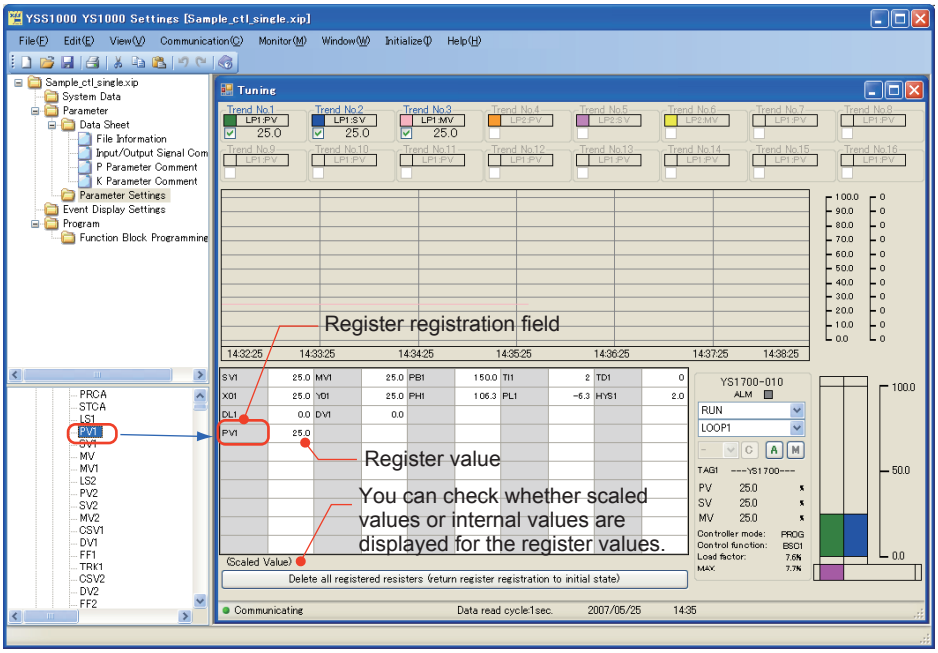
(4)Bar graph display



0240E.ai

Indicated Item	Explanation
PV bar, SV bar, MV bar	PV value (green), SV value (blue) and MV value (pink) are displayed as bar graphs.
Scale	LOOP1: SCL1 (0% value of scale) to SCH1 (100% value of scale) LOOP2: SCL2 (0% value of scale) to SCH2 (100% value of scale) Scale: 3 divisions

(5)Register monitor



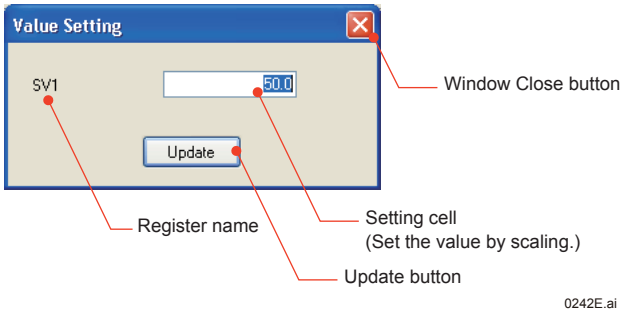
0241E.ai

Indicated Item	Explanation
Register registration field	Register the register to monitor in this field. max. 50 registers To register registers, drag-and-drop the registers to monitor from the Register window. To delete a registered register, click the register and click the Delete key, or right-click the register and execute "Delete" from the shortcut menu.
Register value	The register values are displayed, and can also be changed. To change a register value, double-click the register value and change the value in the Value Setting window that is displayed. The registers that cannot be registered cannot be registered to the registration area. To switch the display of register values, click **Monitor>Display Internal Values** and add a check.
Delete All Registered Registers button	This button deletes all registered registers.

Register search function

When the Register window is active, right-click the window, select [Expand], and enter the target register in alphanumerics on the keyboard.

Value Setting window



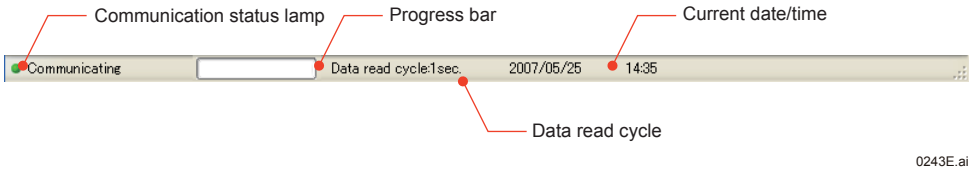
The following defaults are set as the register data. Add or change these value as necessary.

Model Name	Default Setting
YS1500, YS1700	Setpoint value (SV1), manipulated output variable (MV1), proportional band (PB1), integrated time (TI1), and derivative time (TD1) of the displayed LOOP
YS1310	No registers registered
YS1350	Setpoint value (SV) only
YS1360	Manipulated output variable (MV) only

Note

When the YS1700's controller mode is TEST2, simulated input values can be set to Xn and DIn. (Xn: analog inputs, DIn: digital inputs)
However when an Xn or DIn is registered as an output terminal by SUB@SIMPR, the simulated input values are replaced with the values of the registers connected to the output terminals at the next control period.

(6)Status bar

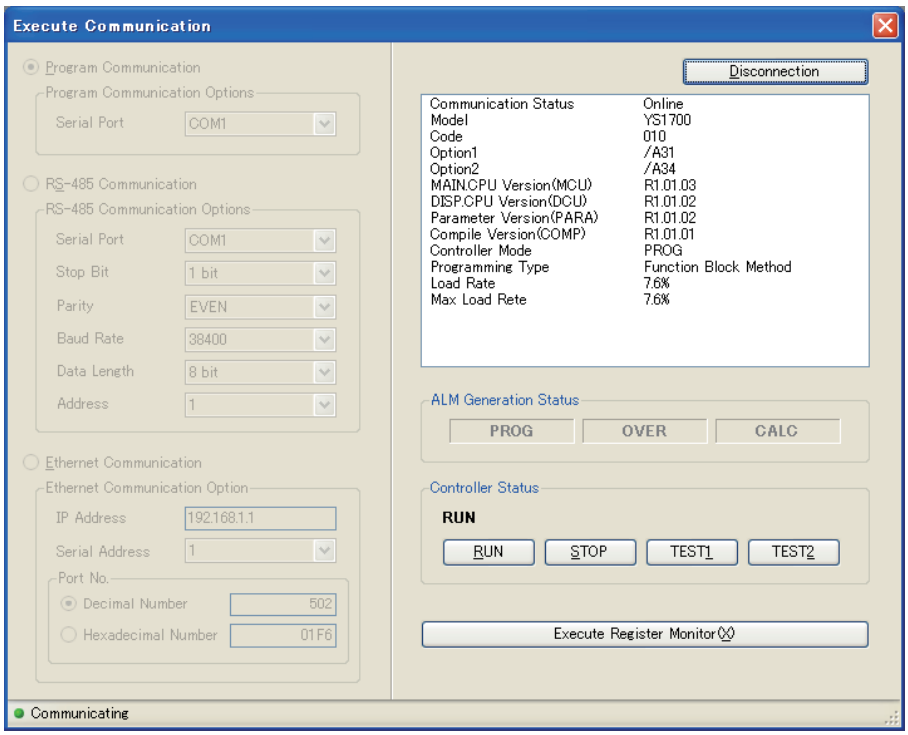


Indicated Item	Explanation
Communication status lamp	Green (blinking): communication in progress, red (lit): communication error (delay) occurring
Progress bar	The progress bar is displayed.
Data read cycle	The data read cycle is displayed. Double-click this item to display the Data Read Cycle Setting window. "2.10.3 Setting the Data Read Cycle"
Current date	The date of the PC's system clock is displayed.
Current time	The time of the PC's system clock is displayed.

2.10.2 Monitoring Register Data (YS1700 Programmable Mode Only)

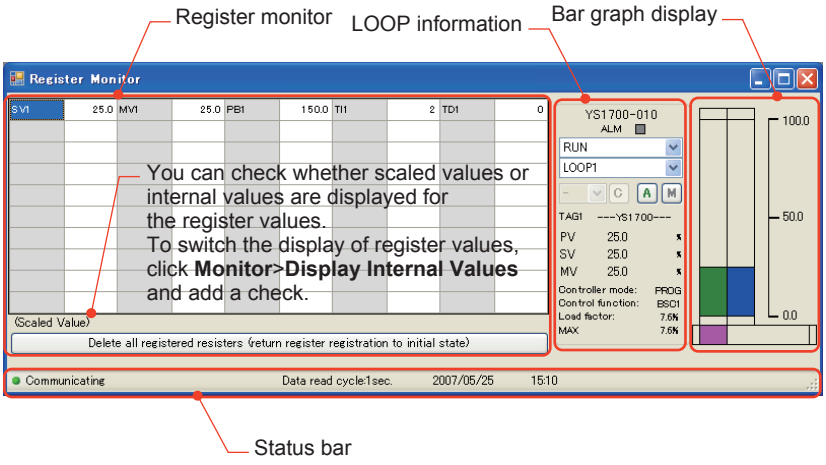
Operation

1. With the Basic window displayed, click **Communication**>**Register Monitor** from the menu to display the Execute Communication window.



0244E.ai

2. Set the communication conditions and click **Execute Register Monitor** to display the Register Monitor window.



0245E.ai

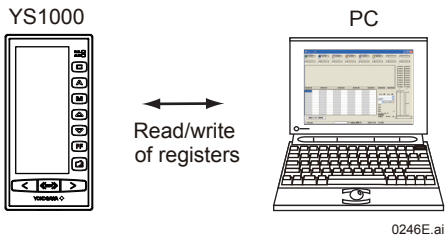
3. Change the register setting values while viewing the trends of process value (PV), setpoint value (SP) and manipulated output variable (MV).

For details on LOOP information, bar graph display, register monitor, and status bar, see "2.10.1."

4. To close the Register Monitor window, click at the top right of the Window.

Description

"Register Monitor" is a function for confirming operation of the user program on the YS1700.
The Register Monitor, Control Module Function Block Monitor, and Module Monitor windows can be displayed at the same time.
The display of the monitored data is refreshed only in the currently active window.
Changes made to register values are reflected on the YS1000.
To monitor registers, enable communication with the YS1000. See "1.2 Connecting the YS1000 to a PC and Setup Parameters."



<About switching display of monitoring values>

It may sometimes be difficult to understand how to check the operation of user programs, because parameters with scale (Xn, PVn, SVn, Pn, etc.) are displayed as scaled values and T register and module output registers are displayed as internal values in the scaled value display. However, if you select internal value display, checking the operation of user programs becomes easy because parameters with scale are also displayed as internal values.

Note

When scale parameters (SCHn, SCLn, SCDPn, etc.) are changed during the display of internal values, close and then re-open the Monitor window to update the displayed values to the correct values.

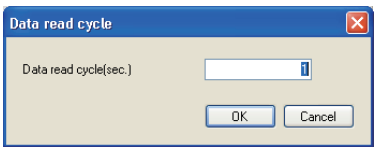
<Execute Communication window>

- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

2.10.3 Setting the Data Read Cycle

Operation

1. Click **Monitor**>**Data Read Cycle** from the menu, or double-click the data read cycle display area on the status bar.



0247E.ai

2. Set the data read cycle, and click **OK**.

Description

When the Tuning, Register Monitor, and Control Module Function Block Monitor windows are displayed, the read cycle at which YS1000 data is displayed on the PC can be set.

If the read cycle is changed when the Tuning window is displayed, the span of the X axis (time axis) of the trend graph becomes as indicated in the table below. The trend graph on the display is deleted and plotting starts from the left side of the X axis (time axis).

Data Read Cycle	Span of X-axis (time axis)
1 second	6 minutes
2 seconds	12 minutes
60 seconds	6 hours

Setting Range	1 to 3600 seconds

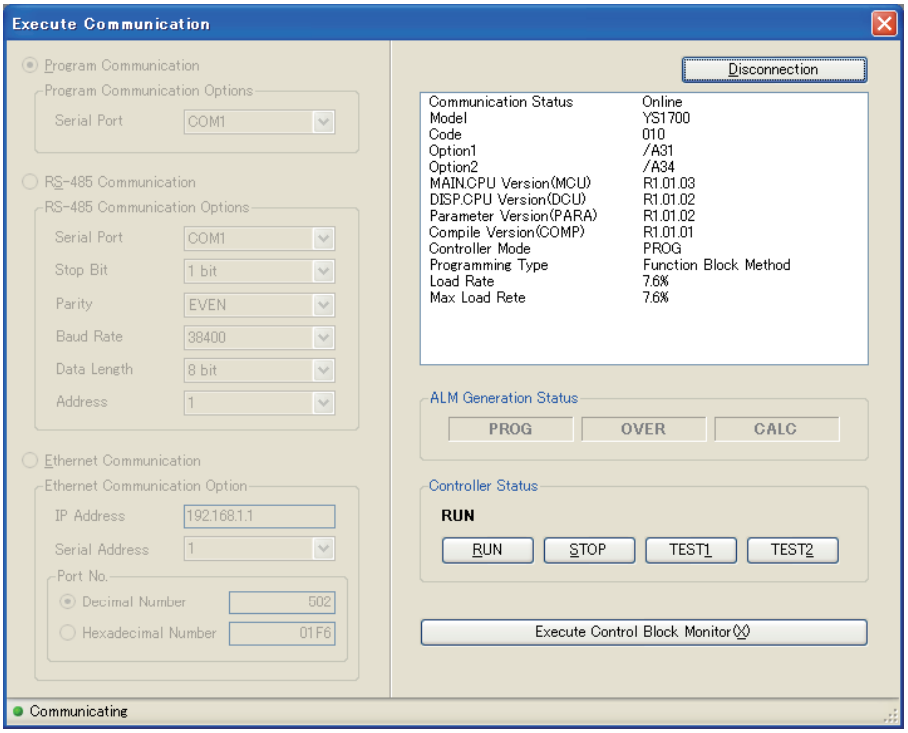
Note

A communication error (delay) will occur if communication cannot keep up with the data read cycle that is already set.

2.10.4 Monitoring Control Module Function Blocks (YS1700 Programmable Mode Only)

Operation

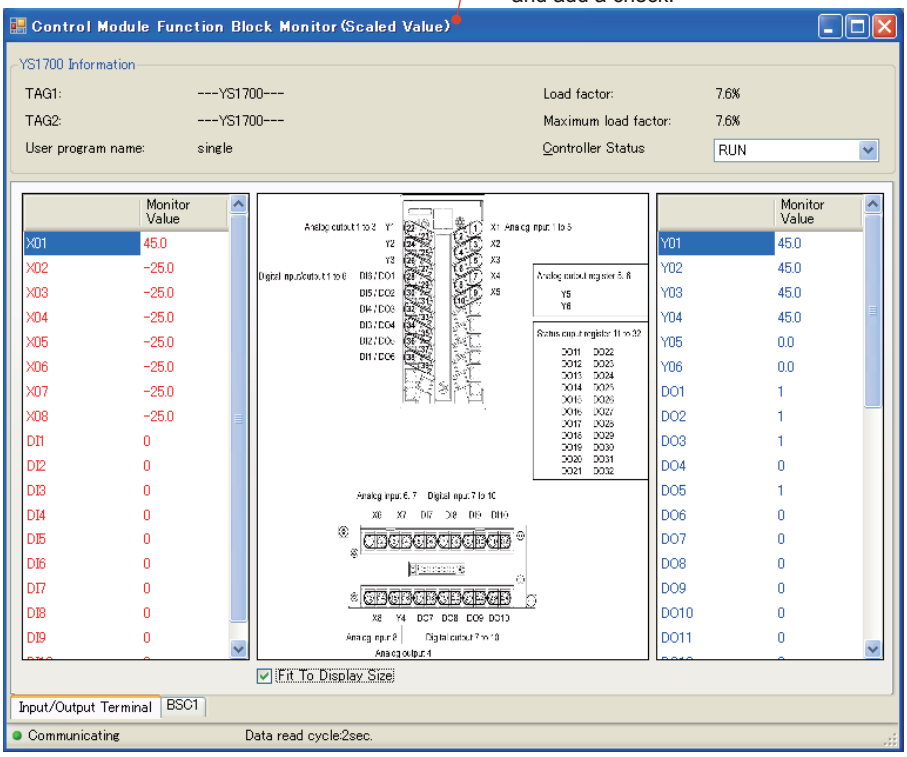
1. Click **Communication>Control Module Function Block Monitor** from the menu to display the Execute Communication window.



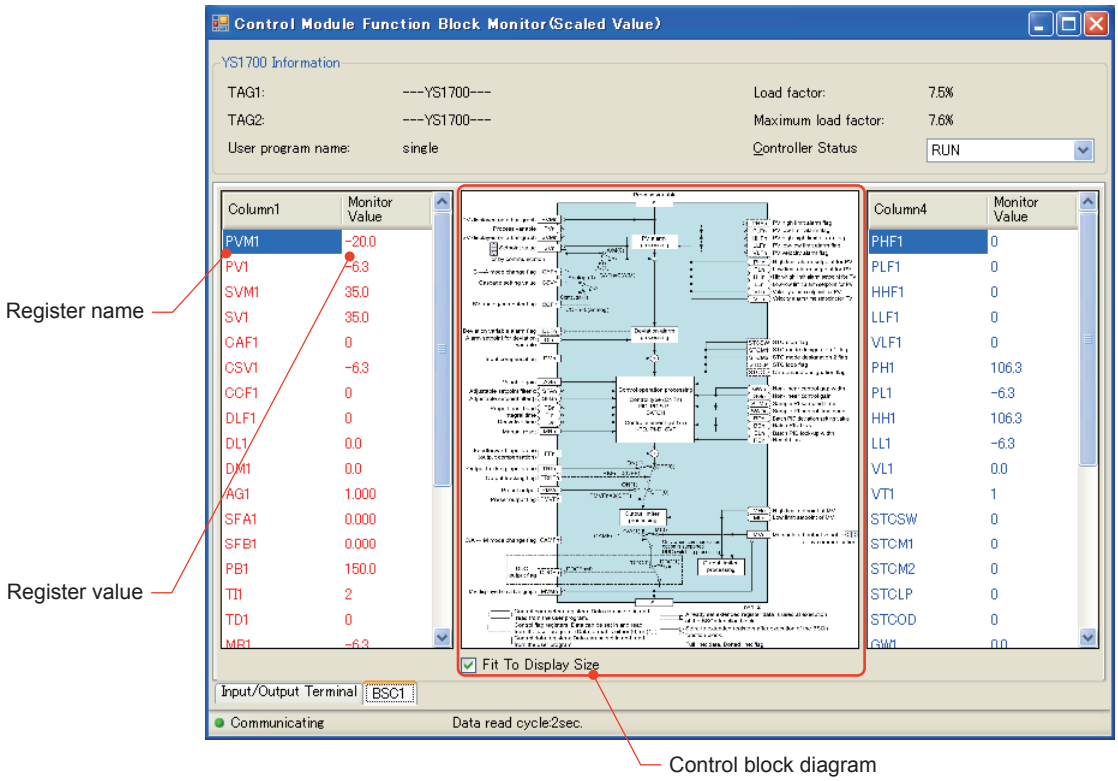
0248E.ai

2. Click **Execute Control Block Monitor** to display the Control Module Function Block Monitor window.

You can check whether scaled values or internal values are displayed for the monitor values.
To switch the display of monitor values, click **Monitor>Display Internal Values** and add a check.

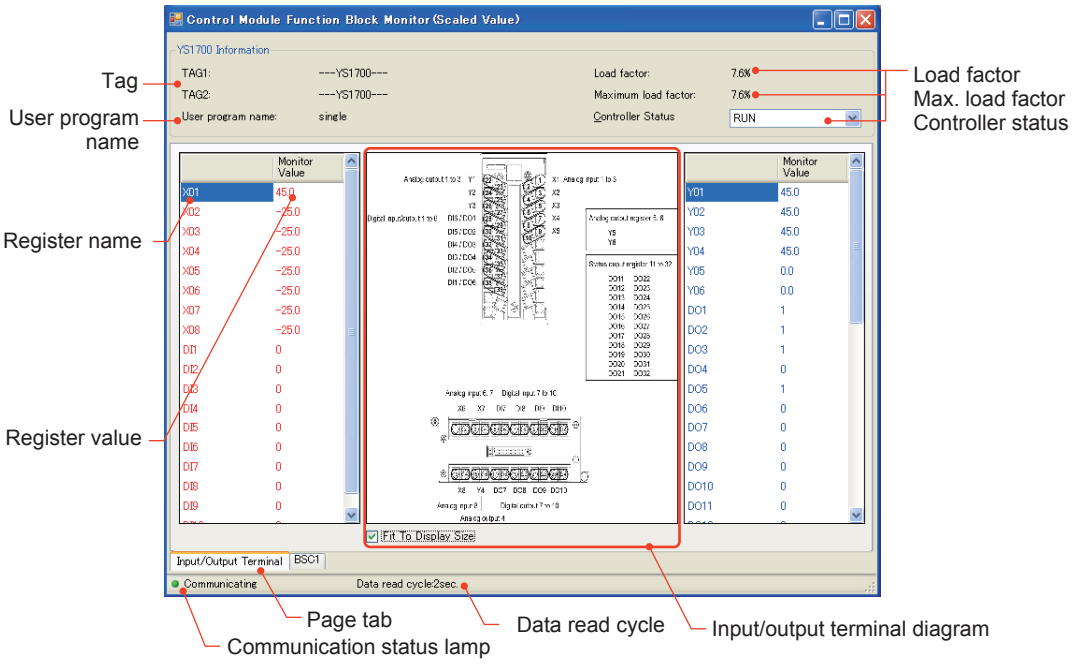


0249E.ai



0250E.ai

3. Click the page tab to switch the page. The figure below shows an example of an input/output terminal diagram.

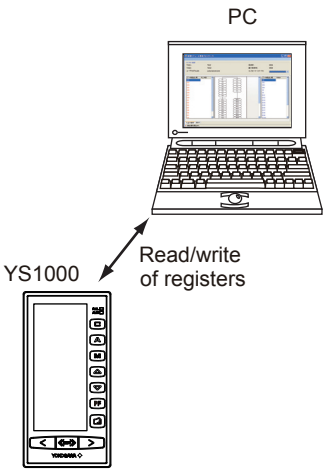


0251E.ai

4. To close the Control Module Function Block Monitor window, click ☒ at the top right of the Window. Change the control status as necessary.

Description

"Control Module Function Block Monitor" is a function for displaying the function block diagrams and for monitoring the register values of the control module function block when the YS1700 is in the programmable mode.
Before performing monitoring, set the YS1700 to a status other than the stop status (STOP).
The Control Module Function Block Monitor, Register Monitor and Module Monitor windows can be displayed at the same time.
The display of the monitored data is refreshed only in the currently active window.



When the control status is "TEST2", simulated inputs (value settings) can be assigned to input terminals.

<About switching display of monitoring values>

It may sometimes be difficult to understand how to check the operation of user programs, because parameters with scale (Xn, PVn, SVn, Pn, etc.) are displayed as scaled values and T register and module output registers are displayed as internal values in the scaled value display. However, if you select internal value display, checking the operation of user programs becomes easy because parameters with scale are also displayed as internal values.

Note

When scale parameters (SCHn, SCLn, SCDPn, etc.) are changed during the display of internal values, close and then re-open the Monitor window to update the displayed values to the correct values.

Note

Set the setting values with scaling while the internal values are displayed.

<Execute Communication window>

- Serial Port: Available ports on the PC are automatically displayed.
- Stop Bit, Parity, Baud Rate, Data Length, and Address: Set the parameters conforming to the YS1000's communication conditions.
- IP Address: Set the parameter conforming to the YS1000's IP address. (for Ethernet communication)
- Serial Address: When communication is performed via an Ethernet/RS-485 converter (e.g. model name: VJET), set the YS1000's RS-485 communication address. (Do not set the same address to two or more controllers.)
- Port No.: Set the port No.

Note

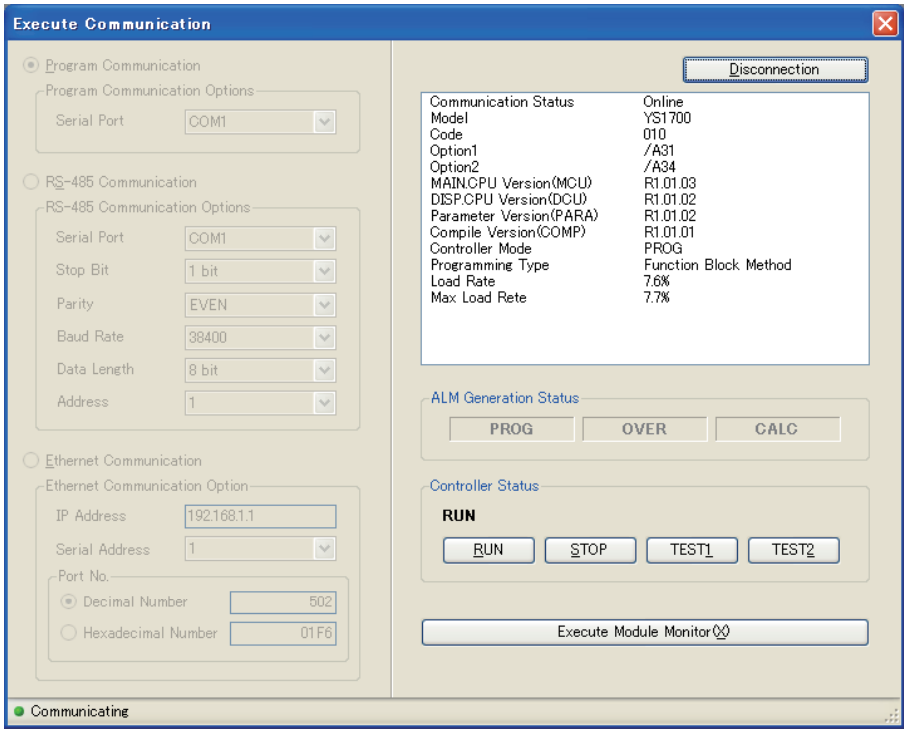
When the YS1700's controller mode is TEST2, simulated input values can be set to Xn and DIn. (Xn: analog inputs, DIn: digital inputs)
However, when an Xn or DIn is registered as an output terminal by SUB@SIMPR, the simulated input values are replaced with the values of the registers connected to the output terminals at the next control period.

Indicated Item	Explanation
Tag	TAG1 or TAG2 of the YS1700 is displayed.
User program name	The user program name of the YS1700 is displayed.
Load factor	The current load factor and the maximum load factor since the Control Block Monitor window was opened are displayed.
Controller status	The control status of the YS1700 is displayed. This status can also be switched.
Page tab	This tab switches the page.
Communication status lamp	Green (blinking): communication in progress, red (lit): communication error (delay) occurring
Data read cycle	See "2.10.3 Setting the Data Read Cycle."
Control block diagram	Register values are displayed in the control block diagram. Register values displayed in control block diagrams are for display only, and cannot be changed.
Input/output terminal diagram	Register values are displayed in the input/output terminal diagram. Input terminal register values can be displayed and changed. Note, however, that these values can be changed only when the control status is TEST2 (test run 2). Output terminal register values are for display only.

2.10.5 Monitoring Modules (YS1700 Programmable Mode Only)

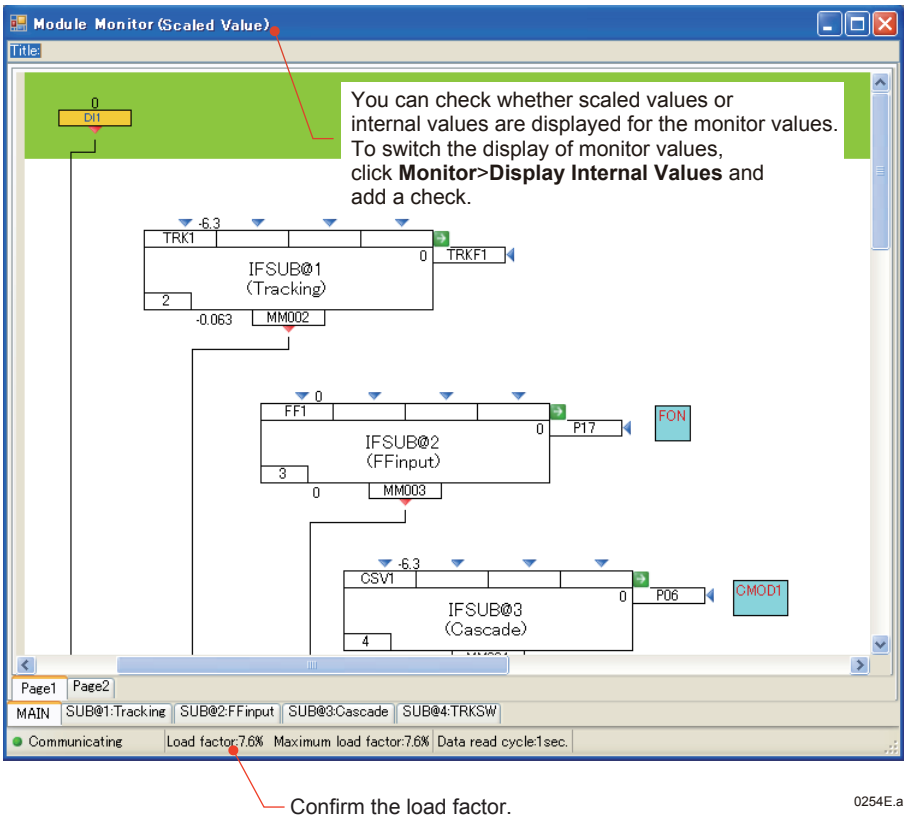
Operation

1. Click **Communication**>**Module Monitor** from the menu to display the Execute Communication window.



0253E.ai

2. Click **Execute Module Monitor** to display the Module Monitor window.

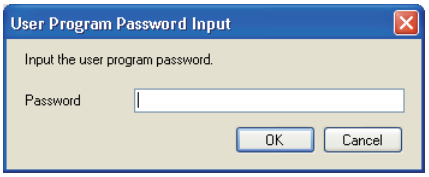


0254E.a

3. To close the Module Monitor window, click ☒ at the top right of the Window.

Note: The register values for subprogram argument registers (SARG01 to SARG04, and SSARG01 to SSARG04) are not displayed.

When a program password is set to the user program to be monitored, a dialog box prompting you to enter the program password is displayed. Enter the password and click **OK**.



0255E.ai

Description

"Module Monitor" is a function for displaying the function block programs and for monitoring register values when the YS1700 is in the programmable mode and programming is performed by the function block programming. Before performing monitoring, set the YS1700 to a status other than the stop status (STOP).
The Module Monitor, Control Module Function Block Monitor, and Register Monitor windows can be displayed at the same time.
The display of the monitored data is refreshed only in the currently active window.
Passwords are limited to eight alphanumeric characters, and entry of passwords is case-sensitive.
When the control status is "TEST2", simulated inputs (value settings) can be assigned to input terminals.

► Load factor: "4.4.4 Program Load Factor"

Note

When the YS1700's controller mode is TEST2, simulated input values can be set to Xn and DIn. (Xn: analog inputs, DIn: digital inputs)
However, when an Xn or DIn is registered as an output terminal by SUB@SIMPR, the simulated input values are replaced with the values of the registers connected to the output terminals at the next control period.

<About switching display of monitoring values>

It may sometimes be difficult to understand how to check the operation of user programs, because parameters with scale (Xn, PVn, SVn, Pn, etc.) are displayed as scaled values and T register and module output registers are displayed as internal values in the scaled value display. However, if you select internal value display, checking the operation of user programs becomes easy because parameters with scale are also displayed as internal values.

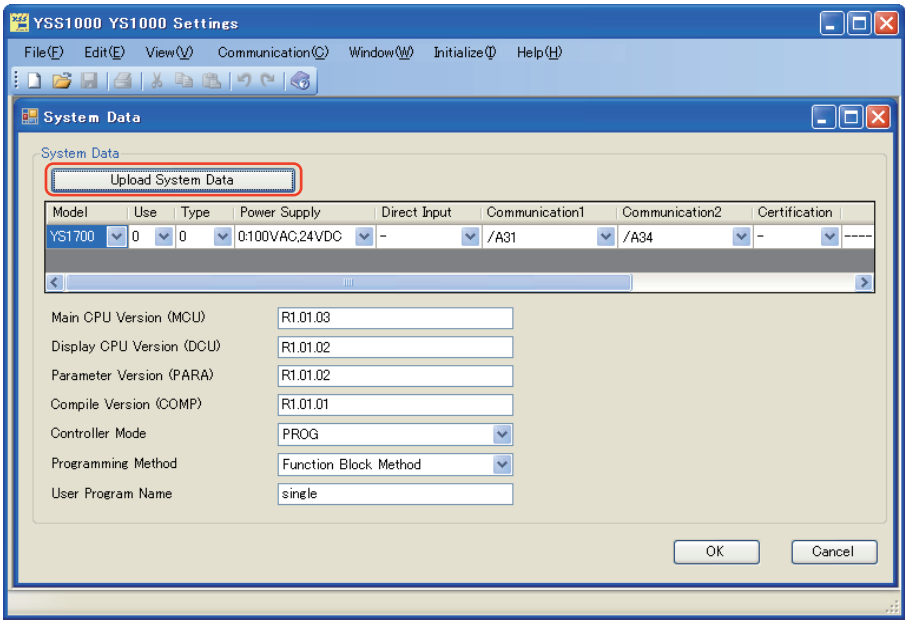
Note

When scale parameters (SCHn, SCLn, SCDPn, etc.) are changed during the display of internal values, close and then re-open the Monitor window to update the displayed values to the correct values.

2.10.6 Viewing System Data

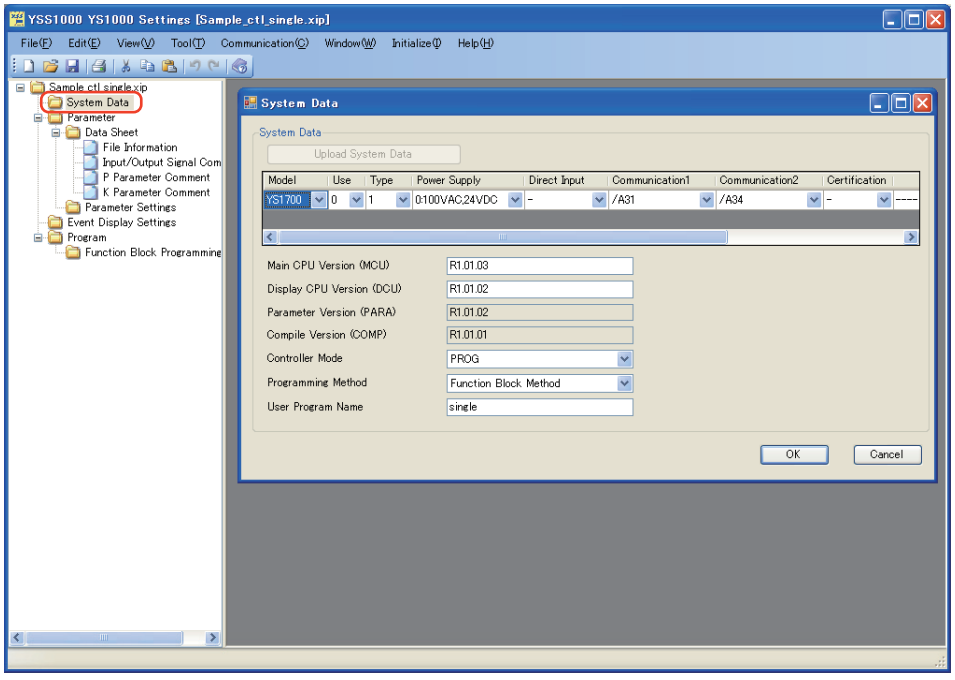
Operation

1. If you see the YS1000 System Data, click **File**>**New** from the menu in the Basic window or on the toolbar to display the System Data window.
2. In the System Data window, click **Upload System Data**.



0256E.ai

If you see the system data you are currently working on, click **System Data** in the file window.



0256-2E.ai

Description

The following items cannot be changed with the file opened, and the Upload System Data button is invalid.

- Model Name
- Parameter Version
- Compile Version

Note

If you change the controller mode with the file opened, the setup data on the software will be initialized.
If you change the programming method with the file opened, the program data on the software will be cleared.

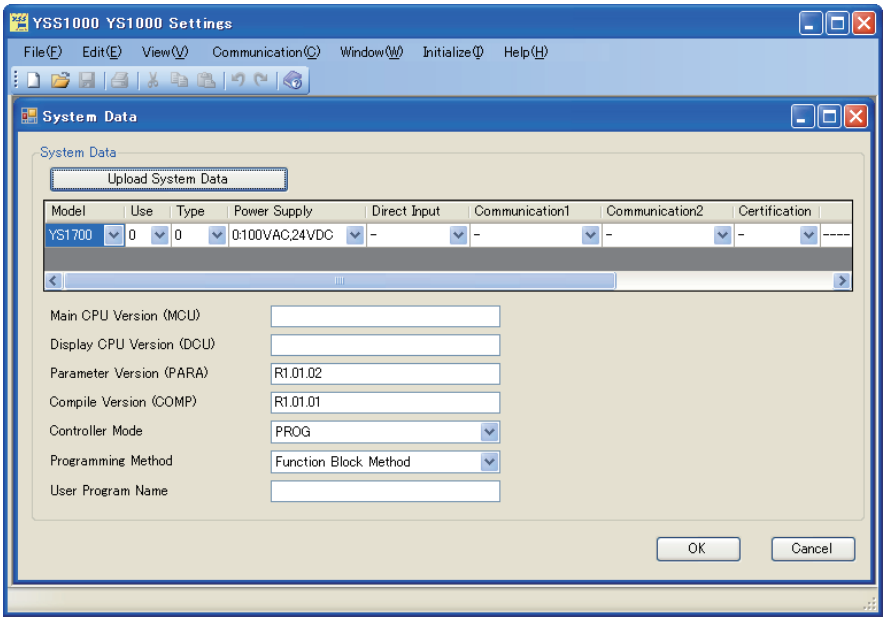
2.11 File Management Operations

Parameter settings, data sheets, event settings, and programs are saved as a single user file appended with the ".xip" file extension. For details on component programs, see “3.2.13 Saving Programs as Components.”

2.11.1 Creating New Files

Operation

1. Click **File**>**New** from the menu in the Basic window or  on the toolbar to display the System Data window.



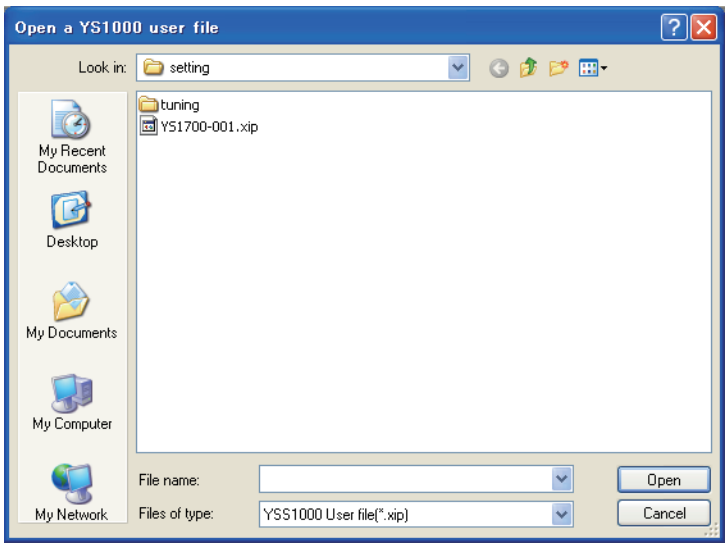
0257E.ai

2. For details on operations such as setting parameters and events, and creating data sheets, see "2.4 Setting Parameters, Control Period, K Register, and P/X/Y Register Scale."

2.11.2 Opening User Files

Operation

1. Click **File**>**Open** from the menu or  on the toolbar to display the Open a YS1000 User File window.



0258E.ai

2. For details on operations such as setting parameters and events, and creating data sheets after opening a user file, see the respective sections in Chapter 2.

When a password is set to the user file, a dialog box prompting you to enter the user file password is displayed. Enter the password and click **OK**.

0258-2E.ai

Description

If there is no parameter table with support for the MCU parameter version of the user file on YSS1000 (the revision of YSS1000 is old), the following message appears when the open user file operation is executed.

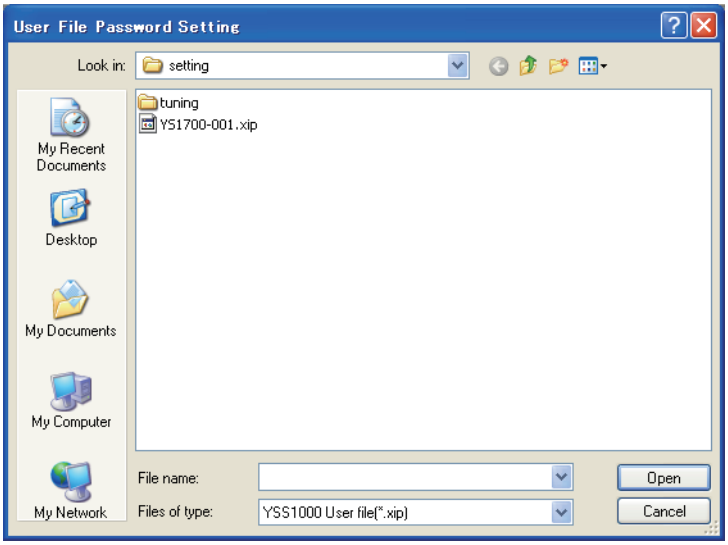
Parameter version of file: ****
In order to open the selected file, you need to update the revision of the software. Since the parameter table version of YSS1000 is old, the parameters that were added cannot be displayed if the process is continued. Do you want to stop the process?

Be careful because the setting values of added parameters will be deleted if you continue the process when this message appears and then overwrite the file by saving after opening the file.

2.11.3 Setting User File Passwords

Operation

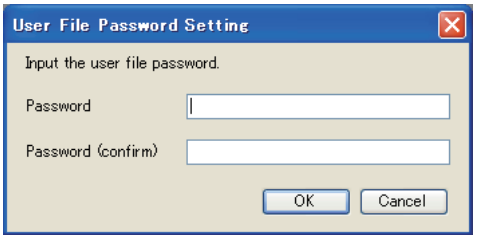
1. When the file is closed, click **File**>**Set User File Password** from the menu to display the User File Password Setting window.



0259E.ai

2. Select the user file to set the password, and click **Open**.

3. Enter the password and click **OK**. You can enter up to eight alphanumerics for the password.



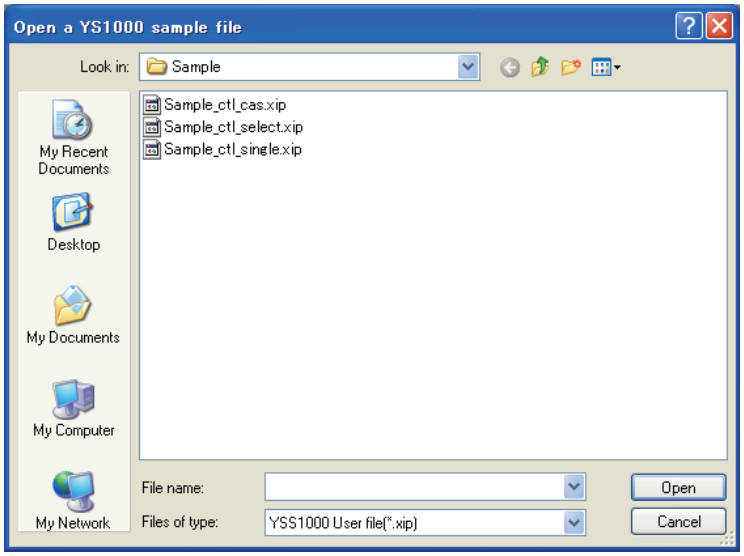
0260E.ai

When setting a user file password, close the file you are working on.

2.11.4 Opening Sample Files

Operation

1. Click **File**>**Sample File** from the menu to display the Open a YS1000 Sample File window.



0262E.ai

2. For details on operations such as setting parameters and events, and creating data sheets after opening a sample file, see the respective sections in Chapter 2.

2.11.5 Closing Files

Operation

1. Click **File**>**Close** from the menu to close the currently active window. To save a file you are working on, use Save As.

2.11.6 Overwriting Files

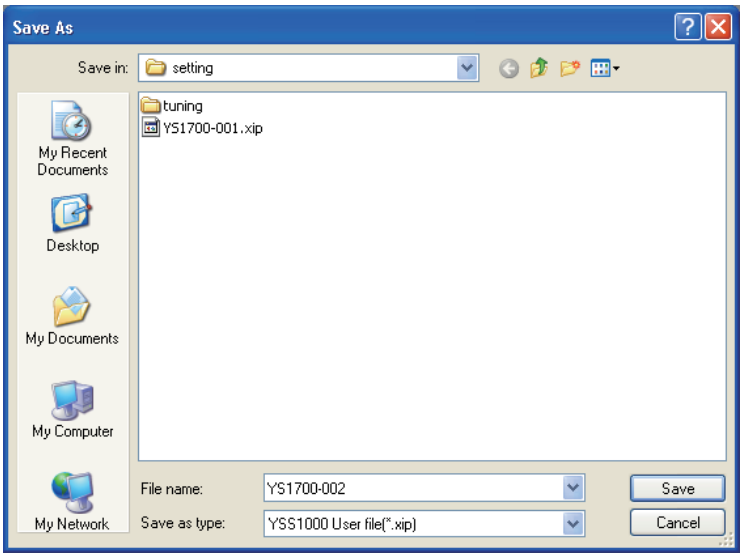
Operation

1. Click **File**>**Save** from the menu or  on the toolbar. The data you are currently working on is saved.

2.11.7 Saving Files under a Different Name

Operation

1. Click **File**>**Save As** from the menu to display the Save As window.



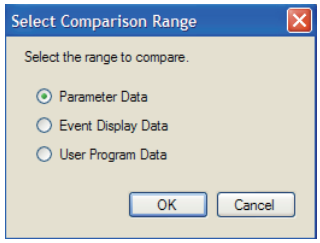
0263E.ai

2. Enter the new name for the file, and click **Save**.

2.11.8 Comparing File Data

Operation

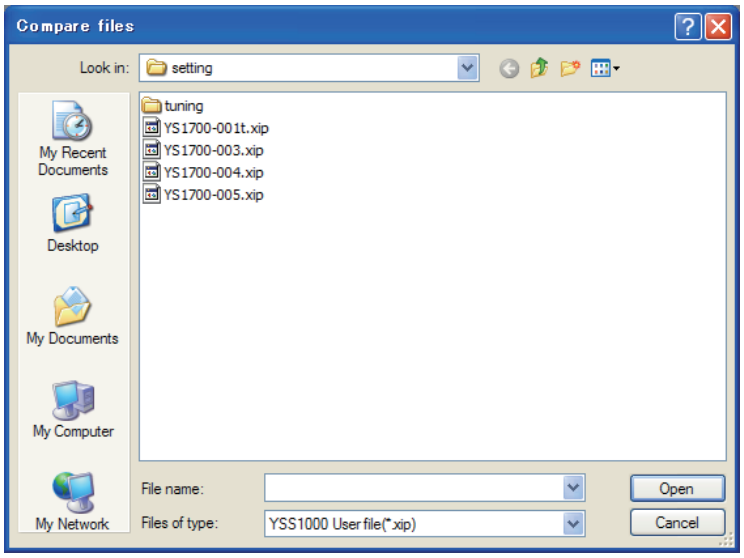
1. Click **File**>**Compare Files** from the menu to display the Select Comparison Range window.



0264E.ai

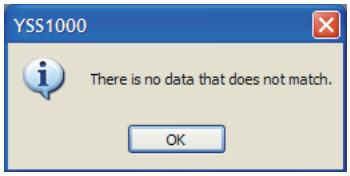
2. Select the range to compare, and click **OK**.

3. Select the files to compare and click **Open**.



0265E.ai

4. Execute the compare. If there is matching data, the following message is displayed. If there is a data mismatch, the mismatching data is displayed.



0266E.ai

If the following message is displayed during execution of a comparison, follow the instructions in the message:

"Unable to perform a comparison because the system data does not match. Comparison was stopped."

Probable causes of this error are a different parameter version or compile version.
• To cancel the comparison, click **Yes**.
• To continue the comparison, click **No**.

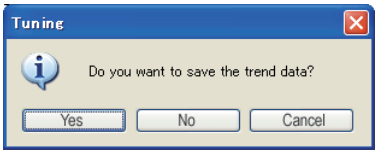
Description

The data you are currently working on is compared with the file data saved on the PC. Input terminals, arrangement information, and program comments are not compared.

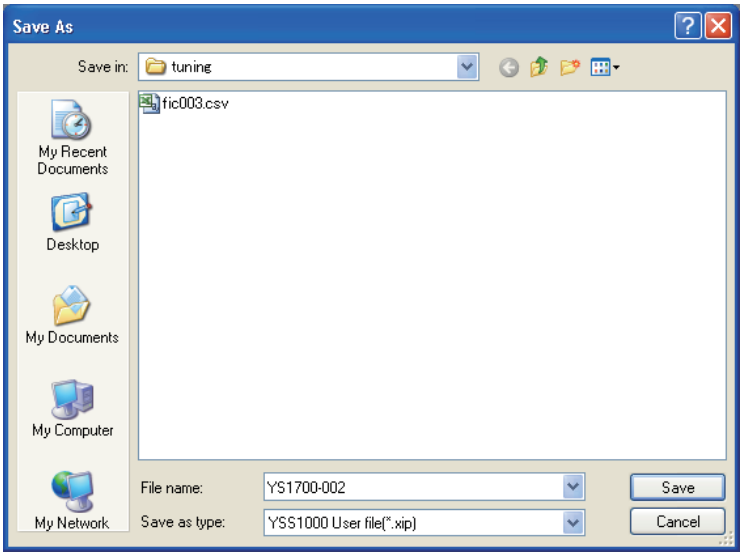
2.11.9 Saving Tuning (Trend) Data

Operation

1. During excution of tuning, click **File**>**Save the trend data** from the menu to display the Save As window.



0267-01E.ai



0267E.ai

2. Enter the new name for the file, and click **Save**.

Description

Tuning (trend) data is saved to file in CSV format.
Trend data for up to 65,000 samples can be saved regardless of the data read cycle.
When the number of trend data samples exceeds 65,000, the data is automatically saved to a different file.
Change the row of Time with EXCEL at time.
Example: When the 65,001st data was sampled in 2007 (year), on 05 (May), 20th, at 21 (hours), 30 (minutes), 50 (seconds), the file name is as follows: 2007_05_20_21_30_50.CSV

Trend data file directory: \Data\Settings\Tuning\ (See "2.11.10 Setting Path for Saving Files.")

YSS1000 Tuning Trend Data

[Auto Scan Information]

No	Model	Address	LOOP	CTL	CNT	TAG	UNIT	SCDP
No.1	YS1700	192.168.200.91	1	PROG	PID	YS1700.LOOP1	%	1
No.2	YS1700	192.168.200.91	2	PROG	PID	YS1700.LOOP2	%	2

[Scale Setting Information]

No.	LOOP	Scale Type	MIN	MAX
A	1	1	0	1000
A	2	2	1000	2000

[Trend Setting Information]

No.	LOOP	Data	Scale	Display Color
Trd_1	1	PV1	1	Green
Trd_2	1	SV1	1	Blue
Trd_3	1	MV1	1	Pink
Trd_4	2	PV2	2	Orange
Trd_5	2	SV2	2	Purple
Trd_6	2	MV2	2	Yellow

[Trend Data]

Date	Time	1.PV	1.SV	1.MV	2.PV	2.SV	2.MV	ALM
2007/1/29	13:45:23	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:24	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:25	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:26	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:27	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:28	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:29	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:30	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:31	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:32	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:33	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:34	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:35	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:36	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:37	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:38	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:39	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:40	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:41	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:42	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:43	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:44	-25	73.8	-6.3	7.5	13.17	-6.3	ON
2007/1/29	13:45:45	-25	73.8	-6.3	7.5	13.17	-6.3	ON

0268.ai

Trend data is always saved as scaled values even if the monitor display (internal values and scaled values) of tuning data is switched.

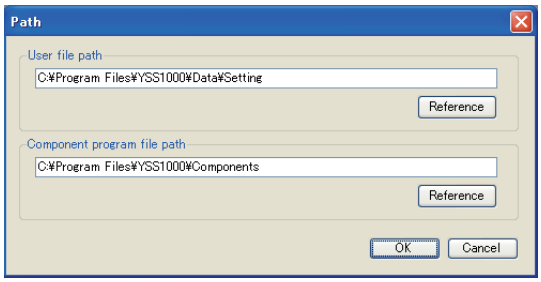
2.11.10 Setting Path for Saving Files

CAUTION

When using Windows Vista, do not set a path including the Program Files folder. If a path including the Program Files folder is set, the YSS1000 will not work correctly.

Operation

1. While the files are closed, click **File**>**Environmental Setting**>**Path** to display the Path window.



0268-01E.ai

2. Set the path for user files or user program files saved as components, and click **OK**.

Description

The path for saving user files and the path for user program files saved as components can be set. The paths for user files and component program files are set as shown in the table below.
(Factory Defaults)

File	Storage Folder	
	Windows 2000/XP	Windows Vista
Common directory	**C:\Program Files\YSS1000**	**C:\Users\<UserName>\Documents\YSS1000**
User file	\Data\Settings\	\Data\Settings\
Component program file	\Components\	\Components\

2.11.11 Creating Communication Logs

Operation

1. While the files are closed, select **File**>**Environmental Setting**>**Communication LOG** and add a check to create communication log files.

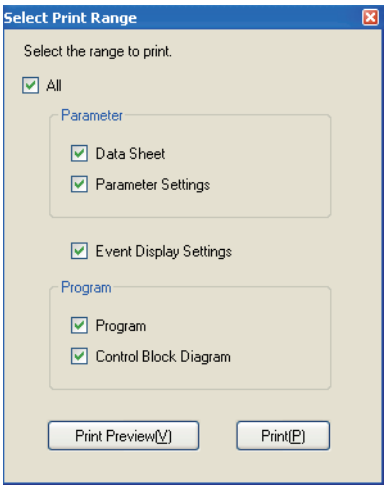
Description

Normally, use without a check added. You can use logs for analysis when, for example, errors are generated during communication. If a check is added, disk space will be used, because a log is created each time communication with YS1000 takes place.

2.12 Printing

Operation

1. Click **File**>**Print** from the menu or  on the toolbar to display the Select Print Range window.



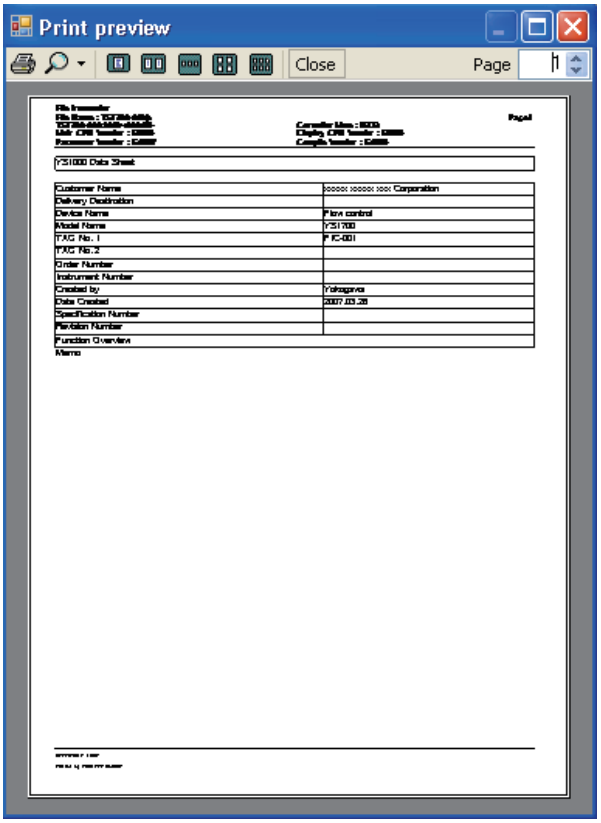
0269E.ai

2. Select the data to print, and click **Print** to display the Print window. Click **Print Preview** to display a print preview on the next page.

3. When printing is completed, click  at the top right of the window.

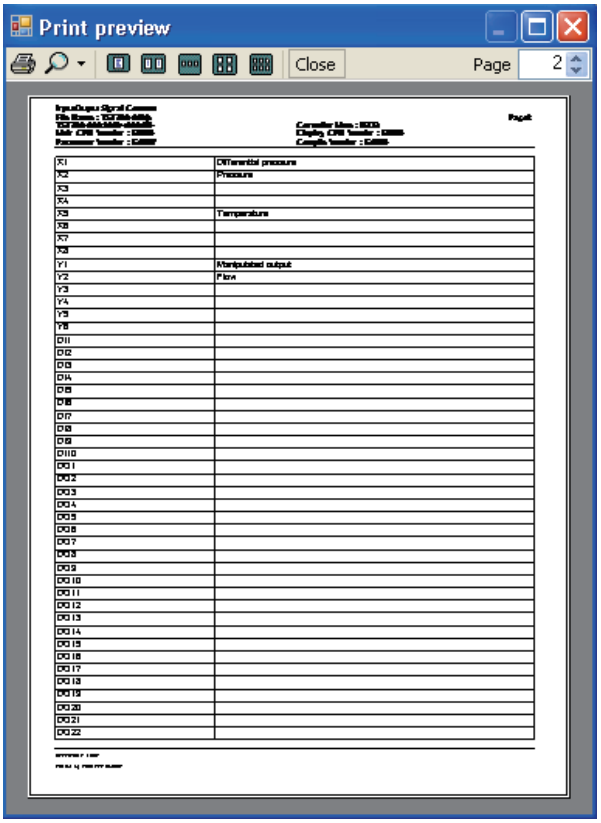
Description

The following are samples of how data is printed out.



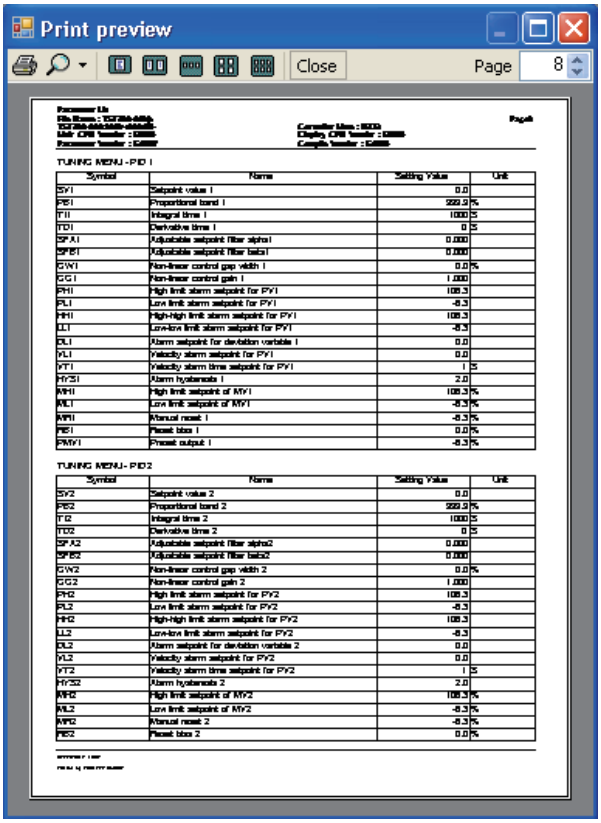
0270E.ai

File Information



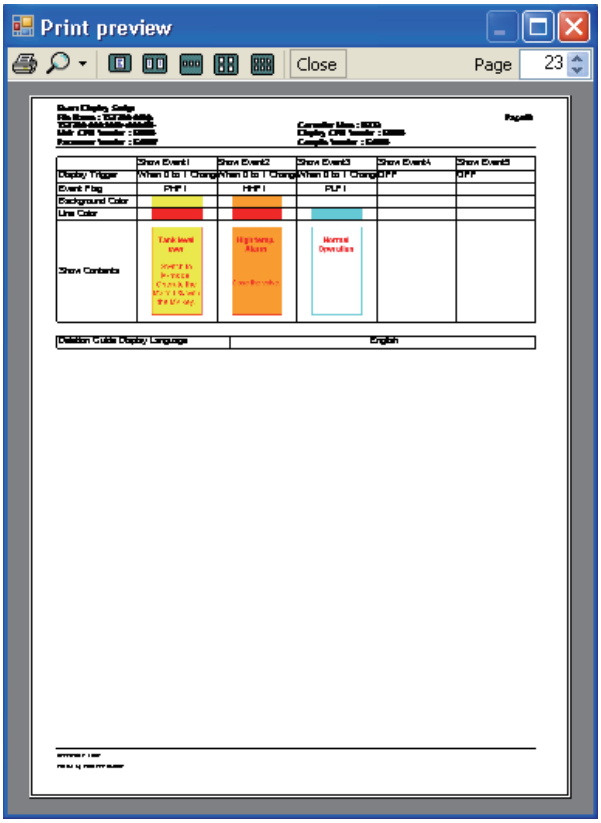
0271E.ai

Input/Output Signal Comment



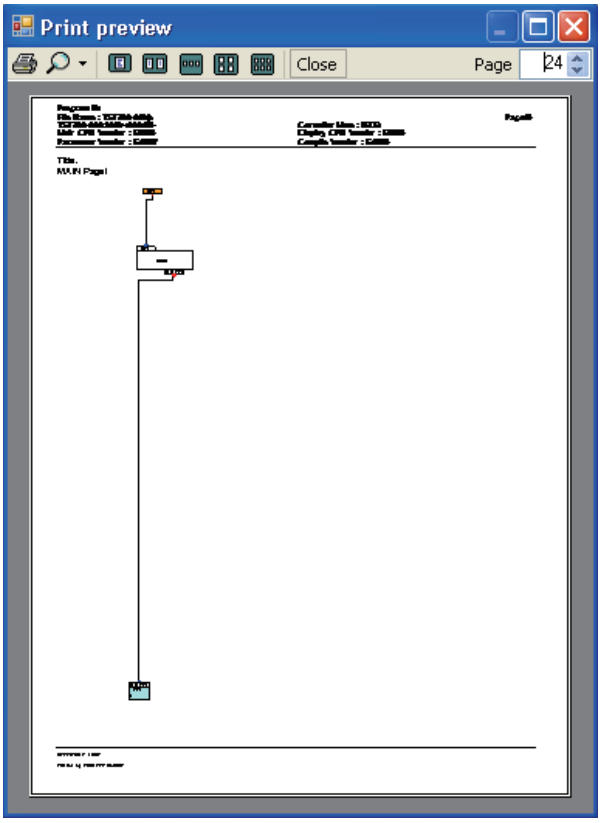
0272E.ai

Parameter List



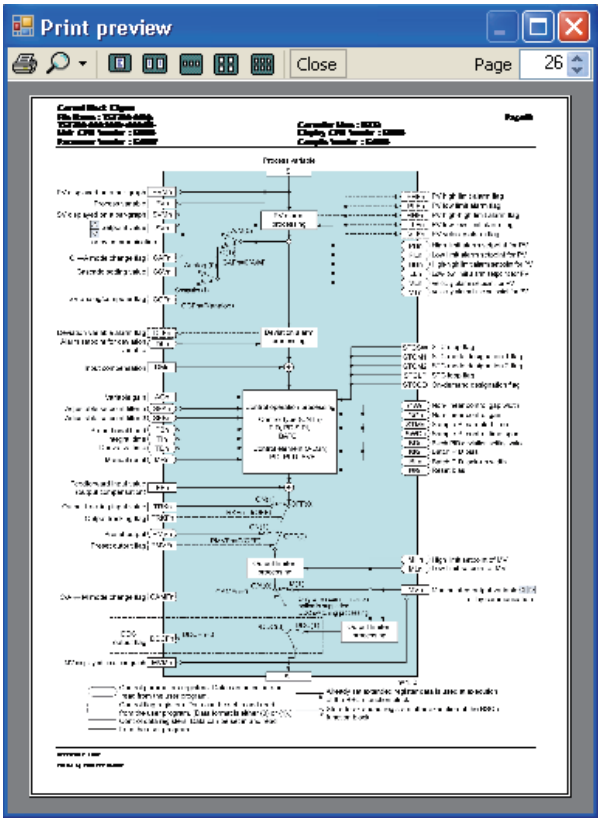
0273E.ai

Event Display Settings



0274E.ai

Program



0275E.ai

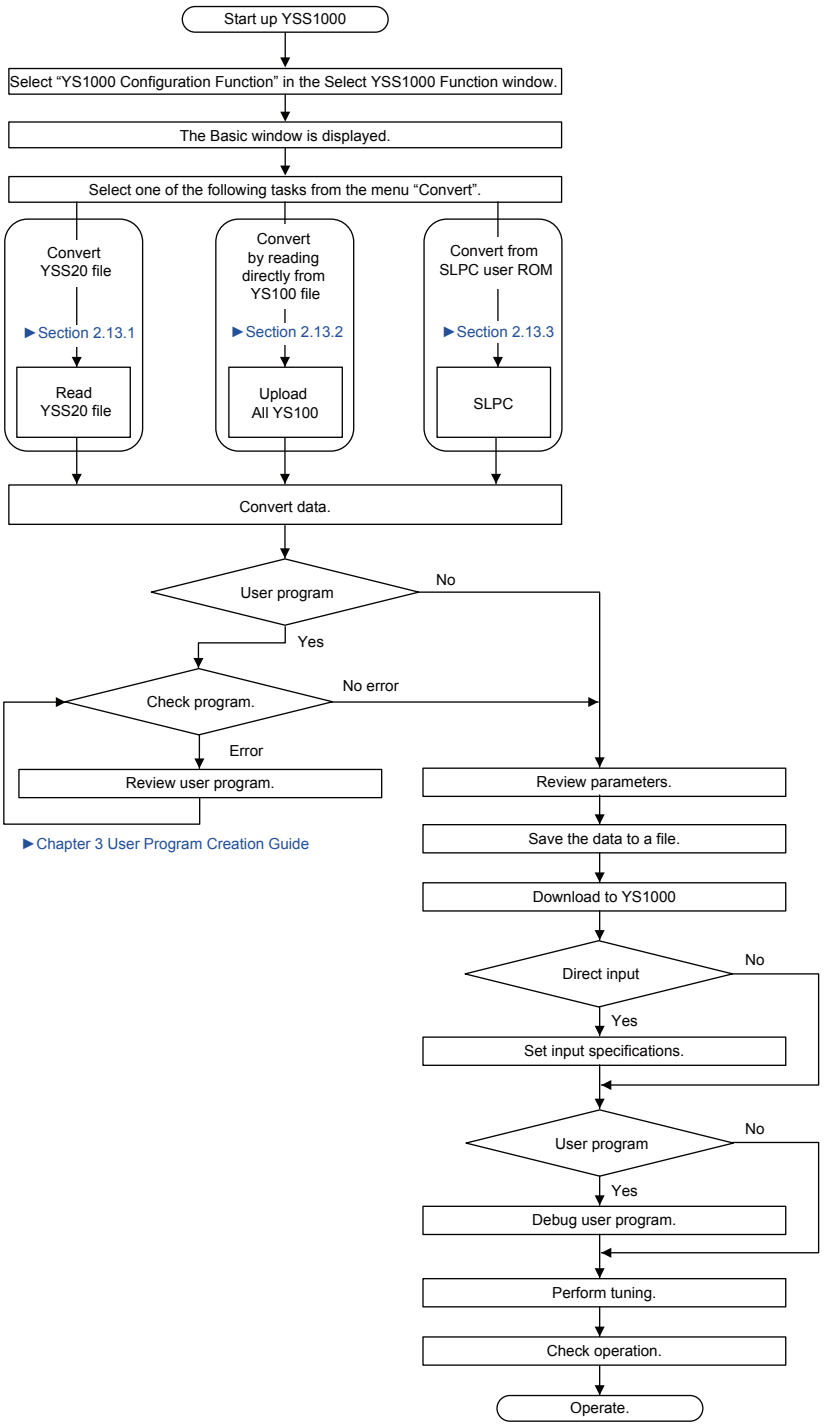
Control Block Diagram

2.13 Converting the Data of Old Models

The names of models applicable to data conversion becomes as shown in the table below.

Before Conversion	After Conversion
YS170, SLPC	YS1700
YS150	YS1500
YS131	YS1310
YS135	YS1350
YS136	YS1360

Conversion Workflow



2.13.1 Converting User Programs and Parameter Data of YSS20 R1.04 Format

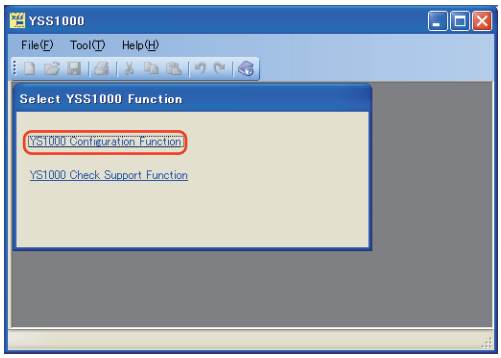
CAUTION

Be sure to review the converted data and check the operation after converting from YSS20 to YSS1000.

Read the R1.04 format files (extension: .hd) of YSS20 programming package and convert the data to YSS1000 format. The data is converted to system data, data sheets, parameter data, and user program data (when using a user program) of YSS1000. In the case of the user program, the text program and program comments are converted. Conversion to the function block method is not possible.
The YSS20 files have the following four extensions: ".hd," ".obj," ".pmt," and ".prg." Conversion is not possible if all the files are not in the same folder.

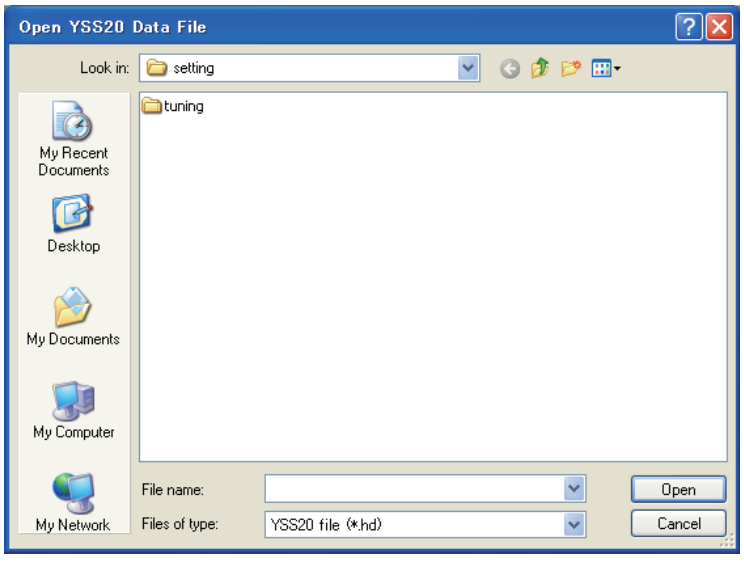
Operation

1. Start up YSS1000, and click "**YS1000 Configuration Function**" in the Select YSS1000 Function window.



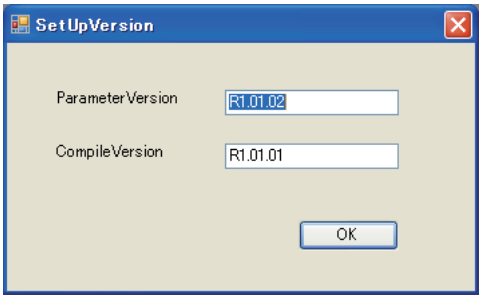
0275-01E.ai

2. Click **Convert>Read YSS20 File** in the Basic window to display the Open YSS20 Data File window. Specify the directory containing the four YSS20 files.



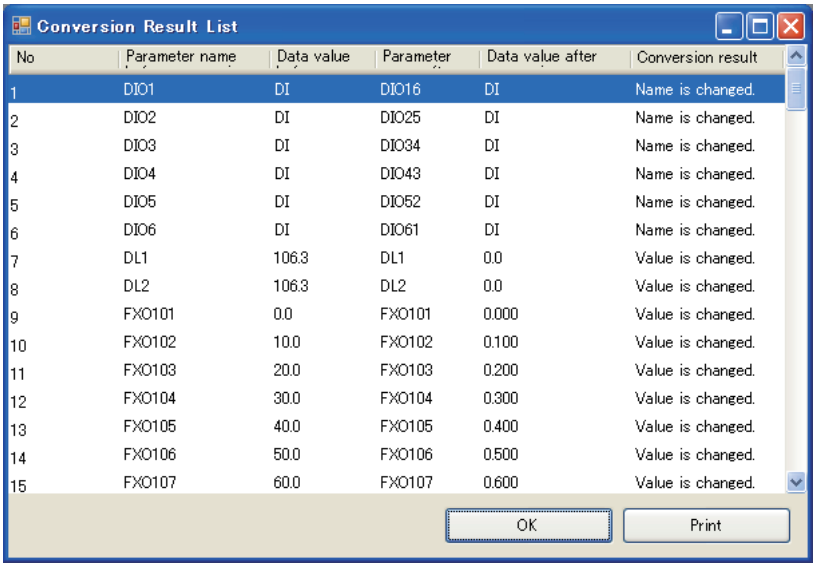
0275-02E.ai

3. Select the user file (extension: .hd) to convert, and click **OK** to display the Version Setting window. Set the same parameter version and compile version as those of YS1000 for downloading data.



0275-03E.ai

4. Set the parameter version and compile version of YS1000 for downloading, and click **OK** to display the Conversion Result List window. (The Conversion Result List window appears when a value is changed, name is changed, or conversion error is generated.) The conversion results are saved in the directory :\Data\Convert\ (See "2.11.10 Setting Path for Saving Files") under the file name Convert Month Day Hour Minute.csv. (Example: When the conversion is performed at 19:30 on September 30, the file name is Convert09301930.csv.)



0275-03E.ai

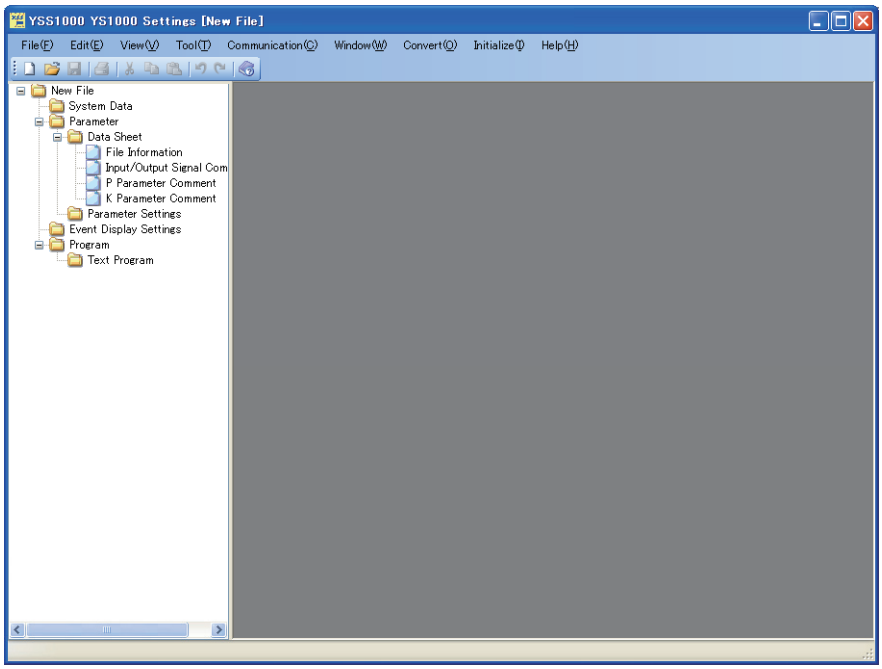
Confirm that there is no conversion error in the conversion results.

Conversion Errors

When data is converted from YSS20 to YSS1000, an error is displayed in the following cases. If an error is displayed, the default values of YS1000 are set in YSS1000.

- CMP is set for CMOD2 with the multi-function type of YS150/YS170.
- Part of the YS100 data has for some reason become damaged.

5. Click **OK** to display the Basic window.



0275-05E.ai

6. When you want convert data that contains a user program, perform a program check by selecting **Tool>Program Check** after the Text Programming window opens. If a compile error is generated, a review of the user programs is required.

7. After the data is saved to file, download it to YS1000 and then perform an operation check.

Description

Review the data after the conversion and save it as a file in YSS1000 format.

► Saving files: "2.11.7 Saving Files under a Different Name"

In the case of the system data, the model name, controller mode, programming method, and user program name are converted. Review other system data after conversion.

Parameter data is converted as shown in the table below. For parameters that are not converted or not applicable, the default parameter values of YSS1000 are set.

Display Group	Parameter Category	YS100 Parameter Symbol	YS1000 Parameter Symbol	YS1700 PROG	YS1700/YS1500			YS1310	YS1350	YS1360	Remarks
					SINGLE	CAS	SELECT				
Tuning Parameter	PID Setting 1	PID1	PID1	✓	✓	✓	✓				
	PID Setting 2	PID2	PID2	✓		✓	✓				
	STC Setting 1	STC1	STC1	✓	✓	✓	✓				
	STC Setting 2	STC2	STC2	✓		✓	✓				
	Setting 1	SETTING1	SETTING1					✓	✓	✓	
	Setting 2	SETTING2	SETTING2					✓			
	P&T Register Setting	P&T REG	P&T REG	✓							
	Parameter Setting	PARAMETER	PARAMETER		✓	✓	✓				
Engineering Parameter	Configuration 1	CONFIG1	CONFIG1	✓	✓	✓	✓	✓	✓	✓	
	Configuration 2	CONFIG2	CONFIG2	✓	✓	✓	✓	✓	✓	✓	
	Configuration 3	CONFIG3	CONFIG3		✓	✓	✓	✓	✓	✓	
	Input Specification Setting	SC MAINT	SC MAINT	–	–	–	–	–	–	–	Only settable with YS1000.
	Password Setting	PASSWORD	PASSWORD	–	–	–	–	–	–	–	
	Sample & Batch Setting	SMPL&BATCH	SMPL&BATCH	✓							
	Sample Setting	None	SMPL								
	Setting for Operation Display	CONFIG2	DISPLAY	✓	✓	✓	✓	✓	✓	✓	
	LCD Setting	LCD	LCD	–	–	–	–	–	–	–	
	Communication Setting	CONFIG1	COMM	–	–	–	–	–	–	–	
	DI/DO Setting	CONFIG3	DI/DO	✓	✓	✓	✓				
	FX Table Setting	FX TABLE	FX TABLE	✓	✓	✓	✓				
	GX1 Table Setting	GX1 TABLE	GX1 TABLE	✓							
	GX2 Table Setting	GX2 TABLE	GX2 TABLE	✓							
	Program-setting-unit 1 Setting	PGM1 SET	PGM1 SET	✓							
	Program-setting-unit 2 Setting	None	PGM2 SET								
	Preset PID Setting	PID TABLE	PID TABLE	✓							
	K Constant Setting	K CONST	K CONST	✓							
	P Scale Setting	None	None	✓							Only settable with YSS1000.
	X Scale Setting	None	None	✓							
	Y Scale Settting	None	None								

✓: Converted, –: Not converted, /: Not applicable

The “LD RAM” program instruction of YS170 does not exist in YS1700. Review the program after data conversion.

Note

When multiple different control instructions (BSC1, BSC2, CSC, SSC) exist in the program before conversion, the control function near the last program step is selected as control function.

For files of the YSS10 programming package and YSS20 programming package R1.03 or earlier, first convert the data to YSS20 R1.04 format, and then to YSS1000 format. The setup files of YSS20 R1.04 are included on YSS1000 Setting Software for YS1000 Series (CD-ROM). Storage folder: CD-ROM drive:\YSS20\english\

Convert the data to YSS20 R1.04 format in the following order. For details, refer to the user's manual for YSS20. The operation below is when converting in Windows 2000/XP, and Windows Vista.

(1) Install YSS20.
(2) Start up YSS20.
(3) Open the files of YSS10 or YSS20 R1.03 or earlier.
(4) Save the file under a different name.
(5) Exit YSS20.

The user's manual for YSS20 is included as an electronic manual on YS1000 Series Manuals (CD-ROM). Storage folder: CD-ROM drive:\YSS20\english\

YSS20 will not operate if files with the read-only attribute set are read in YSS20 R1.04. It is not possible to read data files directly from the CD-ROM. Furthermore, when the four files with the ".hd," ".obj," ".pmt," and ".prg." extensions are copied from the CD-ROM, it is necessary to remove the read-only attribute.

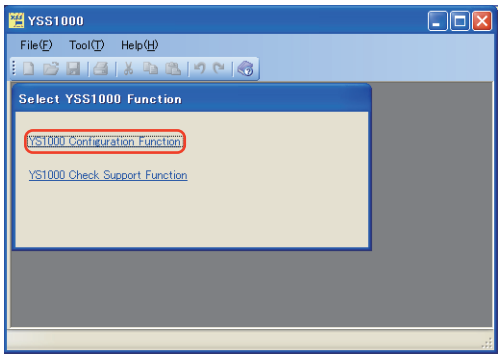
2.13.2 Reading and Converting User Programs and Parameter Data from YS100

Note

Be sure to review the converted data and check the operation after converting from YS100 to YSS1000.

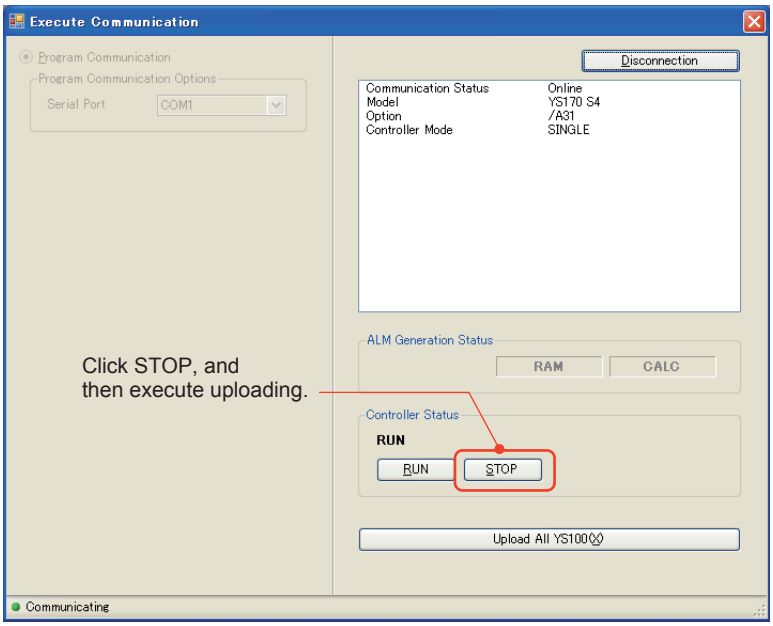
Operation

1. Start up YSS1000 while YS100 is connected, and click **"YS1000 Configuration Function"** in the Select YSS1000 Function window. For details on connecting, see "1.3 Connecting the YS100 to a PC."



0275-06E.ai

2. Click **Convert**>**Upload All YS100** in the Basic window to display the Execute Communication window.

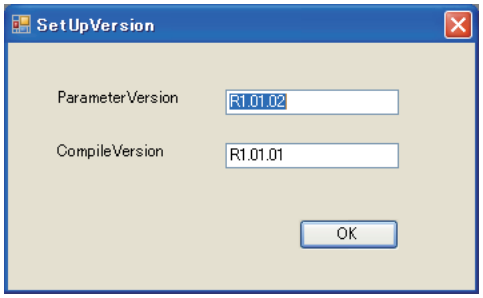


0275-07E.ai

3. Set the communication conditions, set the controller status to **STOP**, and click **Upload All YS100**. Message is displayed when the upload is completed.

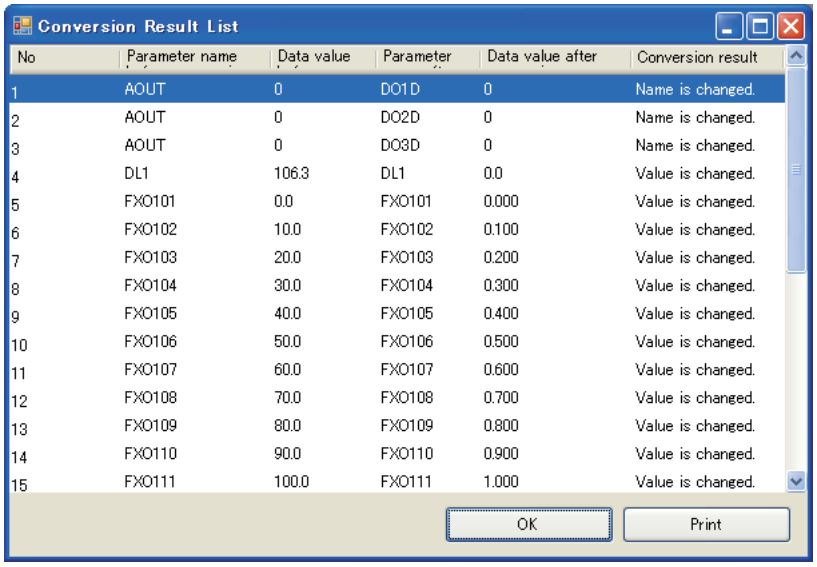
4. Click **OK** to display the Excute Communication window. Re-set the controller status to **RUN**.

5. Close the Excute Communication window to display the SetUp Version window. Set the parameter version and compile version of YS1000 for downloading. Click **OK** in SetUp Version window.



0275-03E.ai

6. Click **OK** to display the Conversion Result List window. (The Conversion Result List window appears when a value is changed, name is changed, or conversion error is generated.) The conversion results are saved in the directory :\Data\Convert\ (see "2.11.10 Setting Path for Saving Files") under the file name Convert Month Day Hour Minute.csv. (Example: When the conversion is performed at 19:30 on September 30, the file name is Convert09301930.csv.)



0275-09E.ai

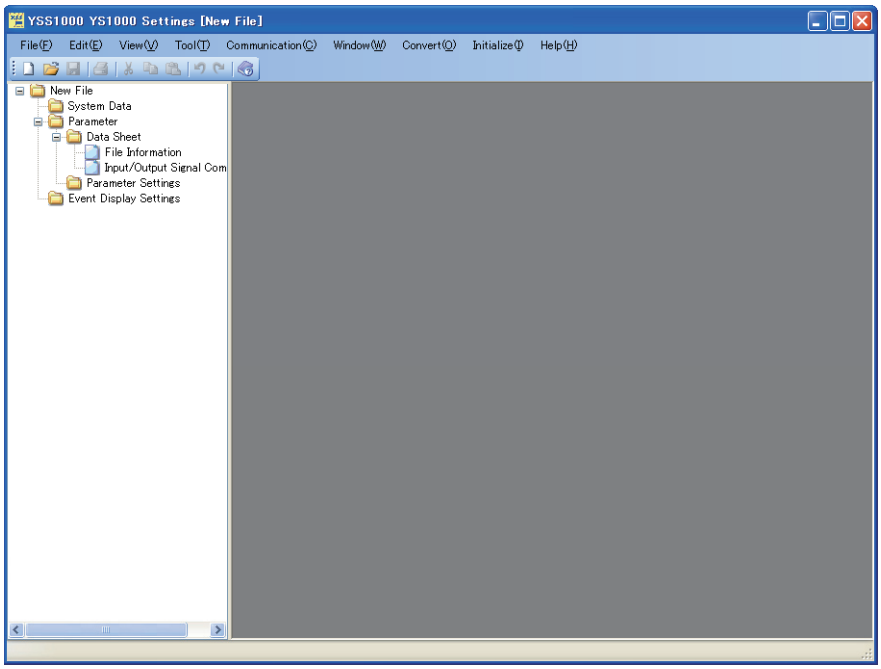
Confirm that there is no conversion error in the conversion results.

Conversion Errors

When data is converted from YS100 to YSS1000, an error is displayed in the following cases. If an error is displayed, the default values of YS1000 are set in YSS1000.

- CMP is set for CMOD2 with the multi-function type of YS150/YS170.
- Part of the YS100 data has for some reason become damaged.

7. Click **OK** to display the Basic window.



0275-10E.ai

8. When you want convert data that contains a user program, perform a program check by selecting **Tool**>**Program Check** after the Text Programming window opens. If a compile error is generated, a review of the user programs is required.

9. After the data is saved to file, download it to YS1000 and then perform an operation check.

Description

The data is read from the YS100 unit, and then converted to YSS1000 format. The data is converted to system data, parameter data, and user program data (when using a user program) of YSS1000.
The version of YS170 and YS150 needs to be REV1.6 or later in order to convert data. There are no restrictions on the version of YS131, YS135, and YS136.
Confirm the version of each instrument in Configuration Display 1 (CONFIG 1).

► Saving files: "2.11.7 Saving Files under a Different Name"

In the case of the system data, the model name, controller mode, programming method, and user program name are converted. Review other system data after conversion.
YS net of YS100 is not converted because it does not exist in YS1000.

When the COM port for the internal modem of laptop PC etc. is specified for the Serial Port of the Program Communication Options in the Excute Communication window, and Upload All YS100 is excuted, the message "This is not a YS100. Connect a YS100." may be displayed. In this case, specify the COM port connected to YS100.

2.13 Converting the Data of Old Models

Parameter data is converted as shown in the table below. For parameters that are not converted or not applicable, the default parameter values of YSS1000 are set.

Display Group	Parameter Category	YS100 Parameter Symbol	YS1000 Parameter Symbol	YS1700 PROG	YS1700/YS1500			YS1310	YS1350	YS1360	Remarks
					SINGLE	CAS	SELECT				
Tuning Parameter	PID Setting 1	PID1	PID1	✓	✓	✓	✓				
	PID Setting 2	PID2	PID2	✓		✓	✓				
	STC Setting 1	STC1	STC1	✓	✓	✓	✓				
	STC Setting 2	STC2	STC2	✓		✓	✓				
	Setting 1	SETTING1	SETTING1					✓	✓	✓	
	Setting 2	SETTING2	SETTING2					✓			
	P&T Register Setting	P&T REG	P&T REG	✓							
	Parameter Setting	PARAMETER	PARAMETER		✓	✓	✓				
Engineering Parameter	Configuration 1	CONFIG1	CONFIG1	✓	✓	✓	✓	✓	✓	✓	
	Configuration 2	CONFIG2	CONFIG2	✓	✓	✓	✓	✓	✓	✓	
	Configuration 3	CONFIG3	CONFIG3		✓	✓	✓	✓	✓	✓	
	Input Specification Setting	SC MAINT	SC MAINT	–	–	–	–	–	–	–	Only settable with YS1000.
	Password Setting	PASSWORD	PASSWORD	–	–	–	–	–	–	–	
	Sample & Batch Setting	SMPL&BATCH	SMPL&BATCH	✓							
	Sample Setting	None	SMPL								
	Setting for Operation Display	CONFIG2	DISPLAY	✓	✓	✓	✓	✓	✓	✓	
	LCD Setting	LCD	LCD	–	–	–	–	–	–	–	
	Communication Setting	CONFIG1	COMM	✓	✓	✓	✓	✓	✓	✓	
	DI/DO Setting	CONFIG3	DI/DO	✓	✓	✓	✓				
	FX Table Setting	FX TABLE	FX TABLE	✓	✓	✓	✓				
	GX1 Table Setting	GX1 TABLE	GX1 TABLE	✓							
	GX2 Table Setting	GX2 TABLE	GX2 TABLE	✓							
	Program-setting-unit 1 Setting	PGM1 SET	PGM1 SET	✓							
	Program-setting-unit 2 Setting	None	PGM2 SET								
	Preset PID Setting	PID TABLE	PID TABLE	✓							
	K Constant Setting	K CONST	K CONST	✓							
	P Scale Setting	None	None	✓							Only settable with YSS1000.
	X Scale Setting	None	None	✓							
	Y Scale Settting	None	None								

✓: Converted, –: Not converted, /: Not applicable

The “LD RAM” program instruction of YS170 does not exist in YS1700. Review the program after data conversion.
Program comments are not converted because they are not downloaded to YS170.

Difference between Specifications of YS100 and YS1000 (1)

The following parameters and setting values are changed when the data is converted.

YS100 Parameter	YS1000 Parameter	Before Conversion	After Conversion	Remarks
K01 to K16	K01 to K16	0.0 to 100.0	0.000 to 1.000	
FX101 to FX111	FX101 to FX111	0.0 to 100.0	0.000 to 1.000	
FX201 to FX211	FX201 to FX211	0.0 to 100.0	0.000 to 1.000	
GXI101 to GXI111	GXI101 to GXI111	0.0 to 100.0	0.000 to 1.000	
GXO101 to GXO111	GXO101 to GXO111	0.0 to 100.0	0.000 to 1.000	
GXI201 to GXI211	GXI201 to GXI211	0.0 to 100.0	0.000 to 1.000	
GXO201 to GXO211	GXO201 to GXO211	0.0 to 100.0	0.000 to 1.000	
PGO01 to PGO10	PGO101 to PGO110	0.0 to 100.0	0.000 to 1.000	
START	START	TIM1	M-COLD	
		AUT	AUT	
		TIM2	M-COLD	
DL1	DL1	106.3	0.0	For YS1700, the alarm does not work with a setting equivalent to 0.0%.
VL1	VL1	106.3	0.0	
TD1	TD1	0	0	The derivative time for YS100 is off with both 0 and 1.
		1	0	
DL2	DL2	106.3	0.0	For YS1700, the alarm does not work with a setting equivalent to 0.0%.
VL2	VL2	106.3	0.0	
TD2	TD2	0	0	The derivative time for YS100 is off with both 0 and 1.
		1	0	

Difference between Specifications of YS100 and YS1000 (2)

The following parameters are changed when the data is converted.

Name of Parameter	YS100 Parameter	YS1000 Parameter	Remarks
Alarm Contact Type	AOUT1 to AOUT6	DO1D to DO6D	
Time Setting for PGM1	PGT01 to PGT10	PGT101 to PGT110	
Output Setting for PGM1	PGO01 to PGO10	PGO101 to PGO110	
DI1/DO6 Specification	DIO1	DIO16	
DI2/DO5 Specification	DIO2	DIO25	
DI3/DO4 Specification	DIO3	DIO34	
DI4/DO3 Specification	DIO4	DIO43	
DI5/DO2 Specification	DIO5	DIO52	
DI6/DO1 Specification	DIO6	DIO61	

Note

When multiple different control instructions (BSC1, BSC2, CSC, SSC) exist in the program before conversion, the control function near the last program step is selected as control function.

2.13.3 Converting the User ROM Data of YS80 (SLPC)

Note

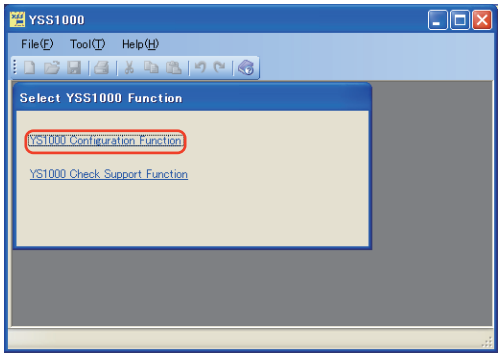
Be sure to review the converted data and check the operation after converting from SLPC to YSS1000.

For details on conversion, see the **Description** below.

Operation

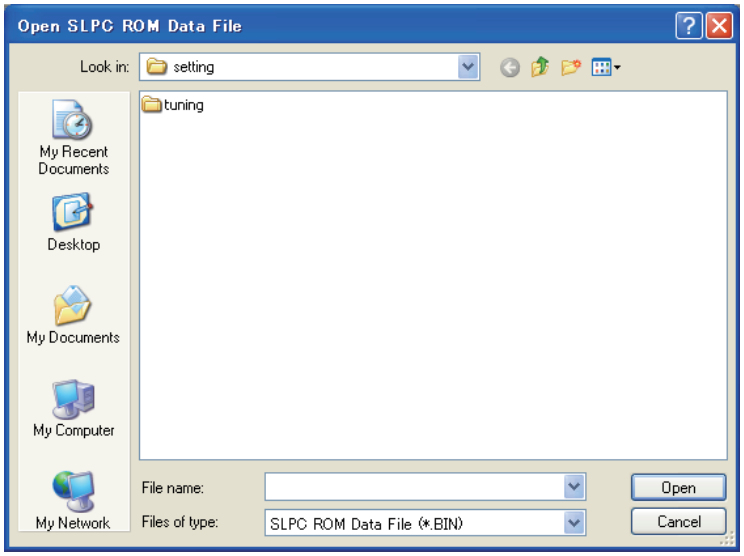
1. Use the software supplied with a commercially available ROM writer to read the user ROM data. The data to read is a file in binary format. Make the extension ".bin." When reading the data, read the image of the whole ROM. For details on the operation, refer the user's manual for the ROM writer.

2. Start up YSS1000, and click "**YS1000 Configuration Function**" in the Select YSS1000 Function window.



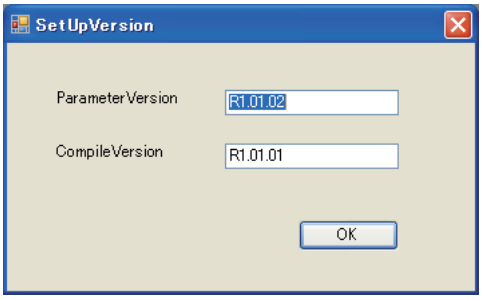
0275-12E.ai

3. Click **Convert>SLPC** in the Basic window to display the Open SLPC ROM Data File window.



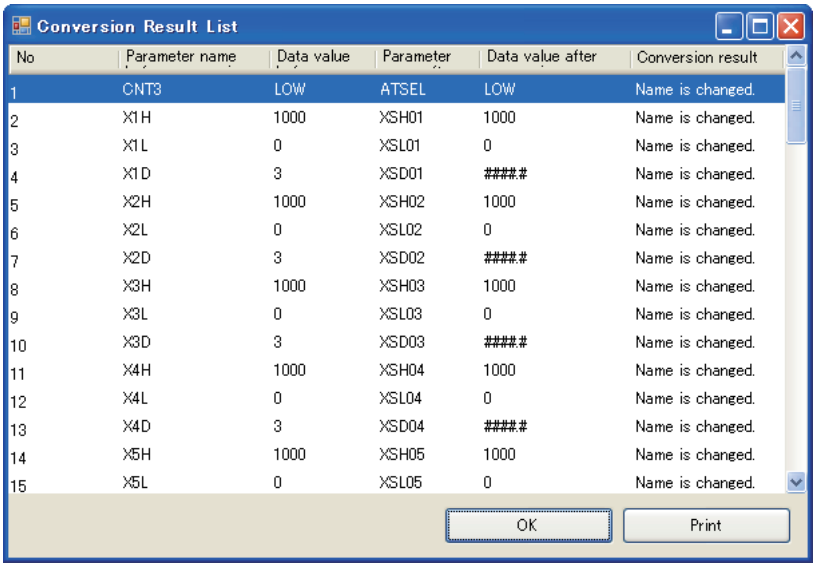
0275-13E.ai

4. Select the user file (extension: .bin) to convert, and click **OK** to display the SetUp Version window. Set the same parameter version and compile version as those of YS1000 for downloading data.



0275-14E.ai

5. Set the parameter version and compile version of YS1000 for downloading, and click **OK** to display the Conversion Result List window. (The Conversion Result List window appears when a value is changed, name is changed, or conversion error is generated.) The conversion results are saved in the directory :\Data\Convert\ (see "2.11.10 Setting Path for Saving Files") under the file name Convert Month Day Hour Minute.csv. (Example: When the conversion is performed at 19:30 on September 30, the file name is Convert09301930.csv.)



0275-15E.ai

Confirm that there is no conversion error in the conversion results.

Conversion Errors

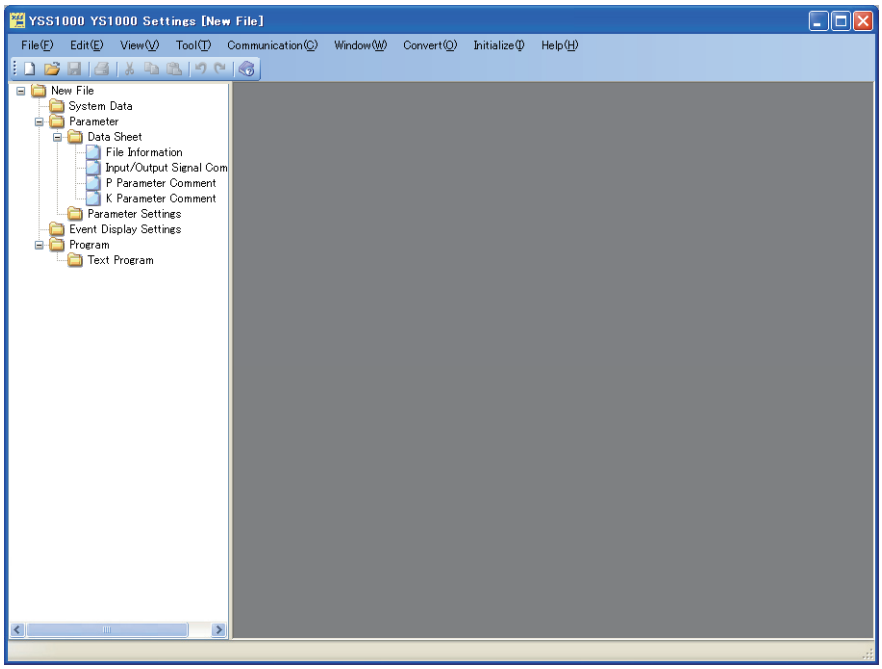
Data conversion stops if any of the following files is selected.

- A file other than an SLPC ROM data file
- A file of a size other than ROM 28-pin (part number: A1123LQ) (8 kb) and ROM 24-pin (part number: G9003LT) (2 KB)

When data is converted from SLPC to YSS1000, an error is displayed in the following cases. If an error is displayed, the default values of YS1000 are set in YSS1000.

- Part of the SLPC ROM data has for some reason become damaged.

6. Click **OK** to display the Basic window.



0275-16E.ai

7. When you want convert data that contains a user program, perform a program check by selecting **Tool>Program Check** after the Text Programming window opens. If a compile error is generated, a review of the user programs is required.

8. After the data is saved to file, download it to YS1000 and then perform an operation check.

Description

The data (binary file) is read from YS80 (SLPC) user ROM, and then converted to YSS1000 format. In order to read the data from the user ROM, a commercially available ROM writer is required to create a binary file.
The data is converted to system data, parameter data, and user program data of YSS1000. You can convert the data of the user ROM of “SLPC*A” and “SLPC*E.” Review the data after conversion and save it as a file in YSS1000 format.

► Saving files: "2.11.7 Saving Files under a Different Name"

Model Names and Suffix Codes of SLPCs Applicable for Conversion

SLPC-100*A, SLPC-110*A, SLPC-200*A, SLPC-210*A
SLPC-140*E, SLPC-150*E, SLPC-170*E, SLPC-240*E, SLPC-250*E, SLPC-270*E
SLPC-151*E, SLPC-181*E, SLPC-251*E, SLPC-281*E

The user ROM of SLPCs applicable for conversion is shown in the table below.

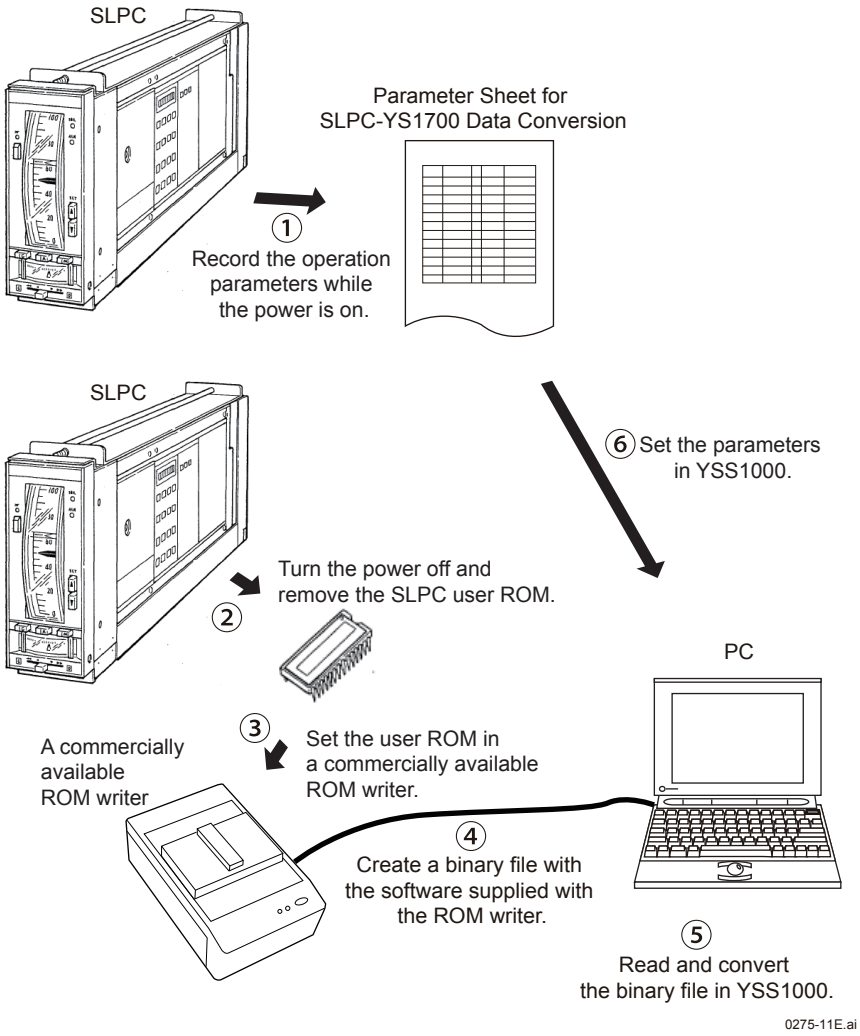
Yokogawa Electric’s Part No.	Manufacturer	Manufacturer’s Part No.	No. of Pins
A1123LQ	Fujitsu	MBM27C64	28
G9003LT	Fujitsu	MB8516	24

Note

The user ROM mounted to an SLPC contains the user program and default parameters. Data such as tuning data that is actually operated on SLPC is not included in user ROM. If you want to convert data, read the operation parameters and ACTION switch settings from the side panel of SLPC, and record them on a parameter sheet. Set the recorded operation parameters in YSS1000. When setting the parameters in YSS1000, set the scale parameters first. Use the parameter sheet in "11.4 Parameter Sheet for SLPC-YS1700 Data Conversion."

In the case of the system data, the model name, controller mode, programming method, and user program name are converted. Review other system data after conversion.

Overview of Data Conversion



Parameter data is converted as shown in the table below. For parameters that are not converted or not applicable, the default parameter values of YSS1000 are set.

Display Group	Parameter Category	YS1000 Display Symbol	YS1700 PROG	Remarks
Tuning Parameter	PID Setting 1	PID1	✓	When PH1 and PH2 are 106.3 (default value) and PL1 and PL2 are -6.3 (default value), the alarm in YS1000 is off. When DL1, VL1, DL2, and VL2 are 0, the alarm in YS1000 is off.
	PID Setting 2	PID2	✓	
	STC Setting 1	STC1	–	
	STC Setting 2	STC2	–	
	P&T Register Setting	P&T REG	✓	
Engineering Parameter	Configuration 1	CONFIG1	✓	
	Configuration 2	CONFIG2	✓	
	Input Specification Setting	SC MAINT	/	No parameters exist in SLPC.
	Password Setting	PASSWORD	/	
	Sample & Batch Setting	SMPL& BATCH	✓	
	Setting for Operation Display	DISPLAY	/	No parameters exist in SLPC.
	LCD Setting	LCD	/	
	Communication Setting	COMM	–	
	DI/DO Setting	DI/DO	✓	
	FX Table Setting	FX TABLE	✓	
	GX1 Table Setting	GX1 TABLE	✓	
	GX2 Table Setting	GX2 TABLE	✓	
	Program-setting-unit 1 Setting	PGM1 SET	✓	
	Program-setting-unit 2 Setting	PGM2 SET	/	No parameters exist in SLPC.
	Preset PID Setting	PID TABLE	/	
	K Constant Setting	K CONST	✓	
	P Scale Setting	None	✓	Only settable with YSS1000.
	X Scale Setting	None	✓	
	Y Scale Settting	None	✓	

✓: Converted, –: Not converted, /: Not applicable

Difference between Specifications of SLPC and YS1000 (1)

The following parameters and setting values are changed when the data is converted.

SLPC Parameter	YS1000 Parameter	Before Conversion	Aftger Conversion	Remarks
F01 to F11	FX101 to FX111	0.0 to 100.0	0.000 to 1.000	
G01 to G11	FX201 to FX211	0.0 to 100.0	0.000 to 1.000	
H01 to H11	GXI101 to GXI111	0.0 to 100.0	0.000 to 1.000	
I01 to I11	GXO101 to GXO111	0.0 to 100.0	0.000 to 1.000	
L01 to L11	GXI201 to GXI211	0.0 to 100.0	0.000 to 1.000	
M01 to M11	GXO201 to GXO211	0.0 to 100.0	0.000 to 1.000	
P30 to P39	PGO101 to PGO110	0.0 to 100.0	0.000 to 1.000	
MODE1	START	COLD	M-COLD	
		HOT	AUT	
TD1	TD1	0	0	The derivative time for SLPC is off with both 0 and 1.
		1	0	
TD2	TD2	0	0	
		1	0	

Difference between Specifications of SLPC and YS1000 (2)

The following parameters are changed when the data is converted.

Name of Parameter	SLPC Parameter	YS1000 Parameter	Remarks
Adjustable setpoint filter α	PX1 to PX2	SFA1 to SFA2	
Adjustable setpoint filter β	PY1 to PY2	SFB1 to SFB2	
Start mode	MODE1	START	
C-mode 1	MODE2	CMOD1	
Backup mode 1	MODE4	BMOD1	
Autoselector selection	CNT3	ATSEL	
Control cycle	CNT4	CYCL	Only settable with YSS1000.
Control operation formula	CNT5	ALG1 to ALG2	
100% value of scale	HI1 to HI2	SCH1 to SCH2	
0% value of scale	LO1 to LO2	SCL1 to SCL2	
Decimal point position	DP1 to DP2	SCDP1 to SCDP2	
Sample PI sampled time 1	ST1	STM1	
Sample-and-hold PI control time span 1	SW1	SWD1	
Sample PI sampled time 2	ST2	STM2	
Sample-and-hold PI control time span 2	SW2	SWD2	
DI1/DO6 specification	DI01	DIO16	
DI2/DO5 specification	DI02	DIO25	
DI3/DO4 specification	DI03	DIO34	
DI4/DO3 specification	DI04	DIO43	
DI5/DO2 specification	DI05	DIO52	
DI6/DO1 specification	DI06	DIO61	
Setting of FX1	F01 to F11	FX101 to FX111	
Setting of FX2	G01 to G11	FX201 to FX211	
Input setting of GX1	H01 to H11	GXI101 to GXI111	
Output setting of GX1	I01 to I11	GXO101 to GXO111	
Input setting of GX2	L01 to L11	GXI201 to GXI211	
Output setting of GX2	M01 to M11	GXO201 to GXO211	
Time setting for PGM1	P20 to P29	PGT101 to PGT110	
Output setting for PGM1	P30 to P39	PGO101 to PGO110	
100% value of P scale	P1H to P8H	PSH1 to PSH8	Only settable with YSS1000.
0% value of P scale	P1L to P8L	PSL1 to PSL8	
Decimal point position of P scale	P1D to P8D	PSD1 to PSD8	
100% value of X scale	X1H to X5H	XSH1 to XSH5	
0% value of X scale	X1L to X5L	XSL1 to XSL5	
Decimal point position of X scale	X1D to X5D	XSD1 to XSD5	
100% value of Y scale	Y1H to Y6H	YSH1 to YSH6	
0% value of Y scale	Y1L to Y6L	YSL1 to YSL6	
Decimal point position of Y scale	Y1D to Y6D	YSD1 to YSD6	

The following table shows instructions of SLPC and YS1000 with the same specifications but different instructions. SLPC instructions are converted to YS1000 instructions.

Load Instruction				Store Instruction			
SLPC	YS1000	SLPC	YS1000	SLPC	YS1000	SLPC	YS1000
LD A1	LD CSV1	LD B25	LD GG2	ST A1	ST CSV1	ST B25	ST GG2
LD A2	LD DM1	LD B26	LD PH2	ST A2	ST DM1	ST B26	ST PH2
LD A3	LD AG1	LD B27	LD PL2	ST A3	ST AG1	ST B27	ST PL2
LD A4	LD FF1	LD B28	LD DL2	ST A4	ST FF1	ST B28	ST DL2
LD A5	LD CSV2	LD B29	LD VL2	ST A5	ST CSV2	ST B29	ST VL2
LD A6	LD DM2	LD B30	LD VT2	ST A6	ST DM2	ST B30	ST VT2
LD A7	LD AG2	LD B31	LD MH2	ST A7	ST AG2	ST B31	ST MH2
LD A8	LD FF2	LD B32	LD ML2	ST A8	ST FF2	ST B32	ST ML2
LD A9	LD TRK1	LD B33	LD STM2	ST A9	ST TRK1	ST B33	ST STM2
LD A10	LD EXT	LD B34	LD SWD2	ST A10	ST EXT	ST B34	ST SWD2
LD A11	LD SEL	LD B38	LD SFA2	ST A11	ST SSW	ST B38	ST SFA2
LD A12	LD SV1	LD B39	LD SFB2	ST A12	ST SV1	ST B39	ST SFB2
LD A13	LD SV2	LD FL1	LD PHF1	ST A13	ST SV2	ST FL1	ST PHF1
LD A14	LD MV1	LD FL2	LD PLF1	ST A14	ST MV1	ST FL2	ST PLF1
LD A15	LD PVM1	LD FL3	LD DLF1	ST A15	ST PVM1	ST FL3	ST DLF1
LD A16	LD SVM1	LD FL4	LD VLF1	ST A16	ST SVM1	ST FL4	ST VLF1
LD B1	LD PB1	LD FL5	LD PHF2	ST B1	ST PB1	ST FL5	ST PHF2
LD B2	LD TI1	LD FL6	LD PLF2	ST B2	ST TI1	ST FL6	ST PLF2
LD B3	LD TD1	LD FL7	LD DLF2	ST B3	ST TD1	ST FL7	ST DLF2
LD B4	LD GW1	LD FL8	LD VLF2	ST B4	ST GW1	ST FL8	ST VLF2
LD B5	LD GG1	LD FL9	LD TRKF1	ST B5	ST GG1	ST FL9	ST TRKF1
LD B6	LD PH1	LD FL10	LD CAF1	ST B6	ST PH1	ST FL10	ST CAF1
LD B7	LD PL1	LD FL11	LD CAMF1	ST B7	ST PL1	ST FL11	ST CAMF1
LD B8	LD DL1	LD FL12	LD FL12	ST B8	ST DL1	ST FL12	ST FL12
LD B9	LD VL1	LD FL13	LD CCF1	ST B9	ST VL1	ST FL13	ST CCF1
LD B10	LD VT1	LD FL14	LD DDCF1	ST B10	ST VT1	ST FL14	ST DDCF1
LD B11	LD MH1	LD FL15	LD COMS02	ST B11	ST MH1	ST FL15	ST COMS02
LD B12	LD ML1	LD FL18	LD STCSW	ST B12	ST ML1	ST FL18	ST STCSW
LD B13	LD STM1	LD FL19	LD CAL	ST B13	ST STM1	ST FL30	ST STCM1
LD B14	LD SWD1	LD FL20	LD IOP	ST B14	ST SWD1	ST FL31	ST STCM2
LD B15	LD BD1	LD FL23	LD OOP	ST B15	ST BD1		
LD B16	LD BB1	LD FL25	LD IOPX01	ST B16	ST BB1		
LD B17	LD BL1	LD FL26	LD IOPX02	ST B17	ST BL1		
LD B18	LD SFA1	LD FL27	LD IOPX03	ST B18	ST SFA1		
LD B19	LD SFB1	LD FL28	LD IOPX04	ST B19	ST SFB1		
LD B21	LD PB2	LD FL29	LD IOPX05	ST B21	ST PB2		
LD B22	LD TI2	LD FL30	LD STCM1	ST B22	ST TI2		
LD B23	LD TD2	LD FL31	LD STCM2	ST B23	ST TD2		
LD B24	LD GW2			ST B24	ST GW2		

The SLPC load instruction is converted to the YS1000 instructions as follows according to the type or the presence of the control instruction.

	SLPC	YS1000
When BSC instruction is used or control instruciton is not used	LD FL12	LD FL12 (Note)
When CSC instruction is used	LD FL12	LD OCF
When SSC instruction is used	LD FL12	LD LRF

Note: LD FL12 instruction do not exist in YS1000. They generate an error. Review them after conversion.

The SLPC store instruction is converted to the YS1000 instructions as follows according to the type or the presence of the control instruction.

	SLPC	YS1000
When BSC instruction is used or control instruciton is not used	ST FL12	ST FL12 (Note)
When CSC instruction is used	ST FL12	ST OCF
When SSC instruction is used	ST FL12	ST LRF

Note: ST FL12 instruction do not exist in YS1000. They generate an error. Review them after conversion.

The following table shows instructions of SLPC that do not exist in YS1700. They generate an error. Review them after conversion.

Load Instruction		
LD B20	LD E6	LD D4
LD FL12	LD E7	LD D5
LD FL16	LD E8	LD D6
LD FL17	LD E9	LD D7
LD FL21	LD E10	LD D8
LD FL22	LD E11	LD D9
LD FL24	LD E12	LD D10
LD FL32	LD E13	LD D11
LD E1	LD E14	LD D12
LD E2	LD E15	LD D13
LD E3	LD D1	LD D14
LD E4	LD D2	LD D15
LD E5	LD D3	

Store Instruction		
ST B20	ST FL27	ST D10
ST FL12	ST FL28	ST D11
ST FL15	ST FL29	ST D12
ST FL16	ST FL32	ST D13
ST FL17	ST D1	ST D14
ST FL19	ST D2	ST D15
ST FL20	ST D3	
ST FL21	ST D4	
ST FL22	ST D5	
ST FL23	ST D6	
ST FL24	ST D7	
ST FL25	ST D8	
ST FL26	ST D9	

The following table shows instructions of SLPC (for peer-to-peer communication) that generate an error because the specifications differ from those of the instructions of YS1700. After conversion, “#” is added to the beginning of the instruction. Review them after conversion.

Load Instruction		Store Instruction
LD CI1	LD CO1	ST CO1
LD CI2	LD CO2	ST CO2
LD CI3	LD CO3	ST CO3
LD CI4	LD CO4	ST CO4
LD CI5	LD CO5	ST CO5
LD CI6	LD CO6	ST CO6
LD CI7	LD CO7	ST CO7
LD CI8	LD CO8	ST CO8
LD CI9	LD CO9	ST CO9
LD CI10	LD CO10	ST CO10
LD CI11	LD CO11	ST CO11
LD CI12	LD CO12	ST CO12
LD CI13	LD CO13	ST CO13
LD CI14	LD CO14	ST CO14
LD CI15	LD CO15	ST CO15

Note

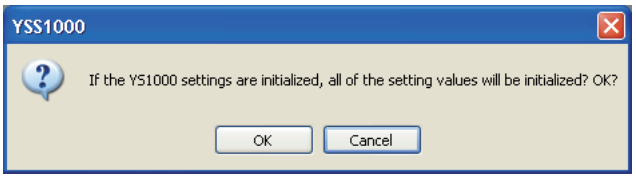
When multiple different control instructions (BSC1, BSC2, CSC, SSC) exist in the program before conversion, "NONE (No control function)" is selected as control function.

2.14 Initializing the YS1000

When the YS1000 is initialized, all settings return to their factory defaults.

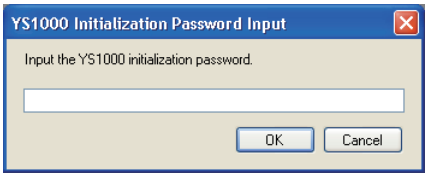
Operation

1. Click **Initialize>Initialize YS1000** from the menu to display the following window, and click **OK**.



0277E.ai

2. Enter the YS1000 initialization password "**YS1000-INIT**", and click **OK**. (Use uppercase characters only.)



0276E.ai

3. With the Excute Communication window, Click **OK** to initialize. Click  at the top right of the window to cancel the initialization.

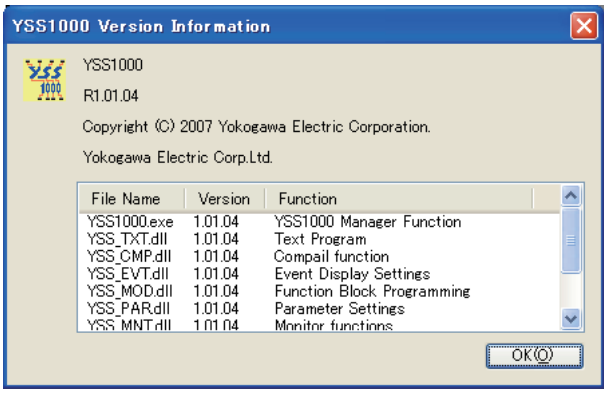
Description

Use this feature if you have forgotten the password to the user program that is downloaded to the YS1700 and you want to download a new program.
Prevent the password described in this section from being used inadvertently.

2.15 Checking the Software Version

Operation

1. Click **Help**>**About Software** from the menu to display the YSS1000 Version Information window.



0278E.ai

2. To close this window, click either **OK** or  at the top right of the window.

2.16 Checking Release Information

The release information is a record of version upgrades made to the YS1000 controller.

Operation

1. Click **Help**>**Release Information** from the menu to display the Release Information window.

0279E.ai

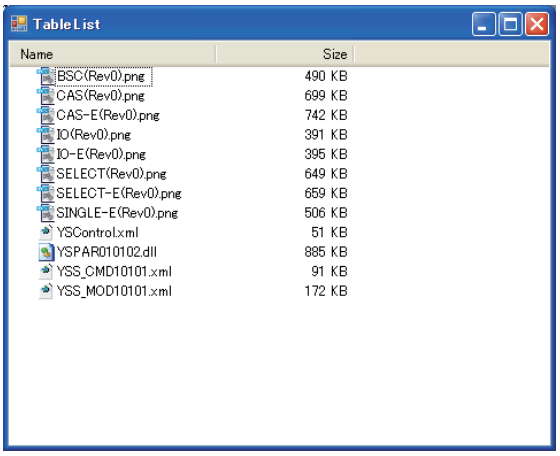
2. To close this window, click  at the top right of the window.

2.17 Viewing Table Lists

A "table list" is a list of DLL and xml files that are contained in the Table folder of the YSS1000 Setting Software.

Operation

1. Click **Help**>**Table List** from the menu to display the Table List window.



0280E.ai

2. To close this window, click  at the top right of the window.

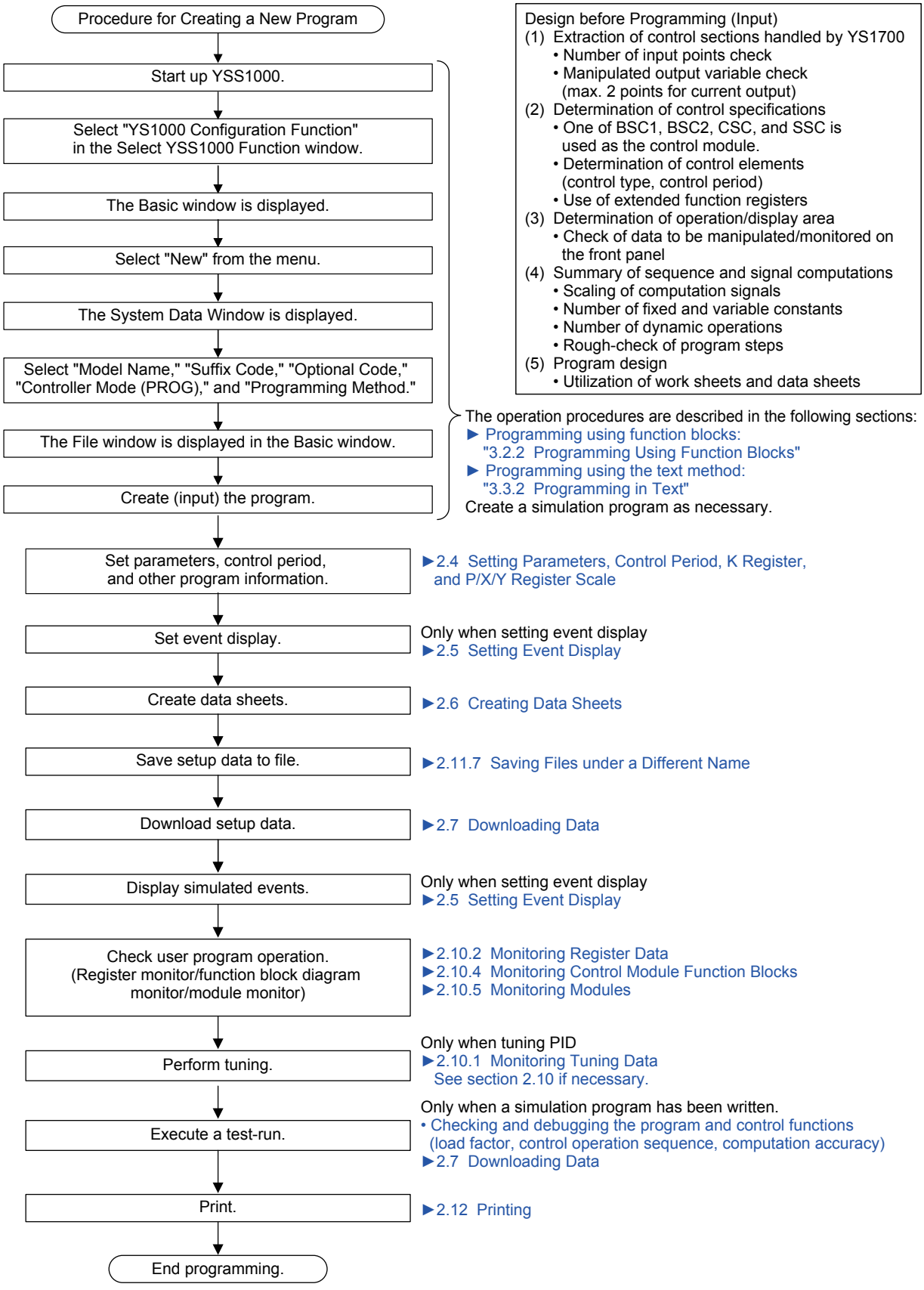
Blank Page

3.1 User Program Programming Flow

This chapter describes how to create user programs for the YS1700. For details on starting up and exiting, downloading, monitoring, and test-running of software, file management operations, and printing, see the "Chapter 2 YSS1000 Operation Guide."

Note

Choose either of function block programming or text programming as the programming method for writing the user program. Once you have selected a programming method, you cannot change it midway.

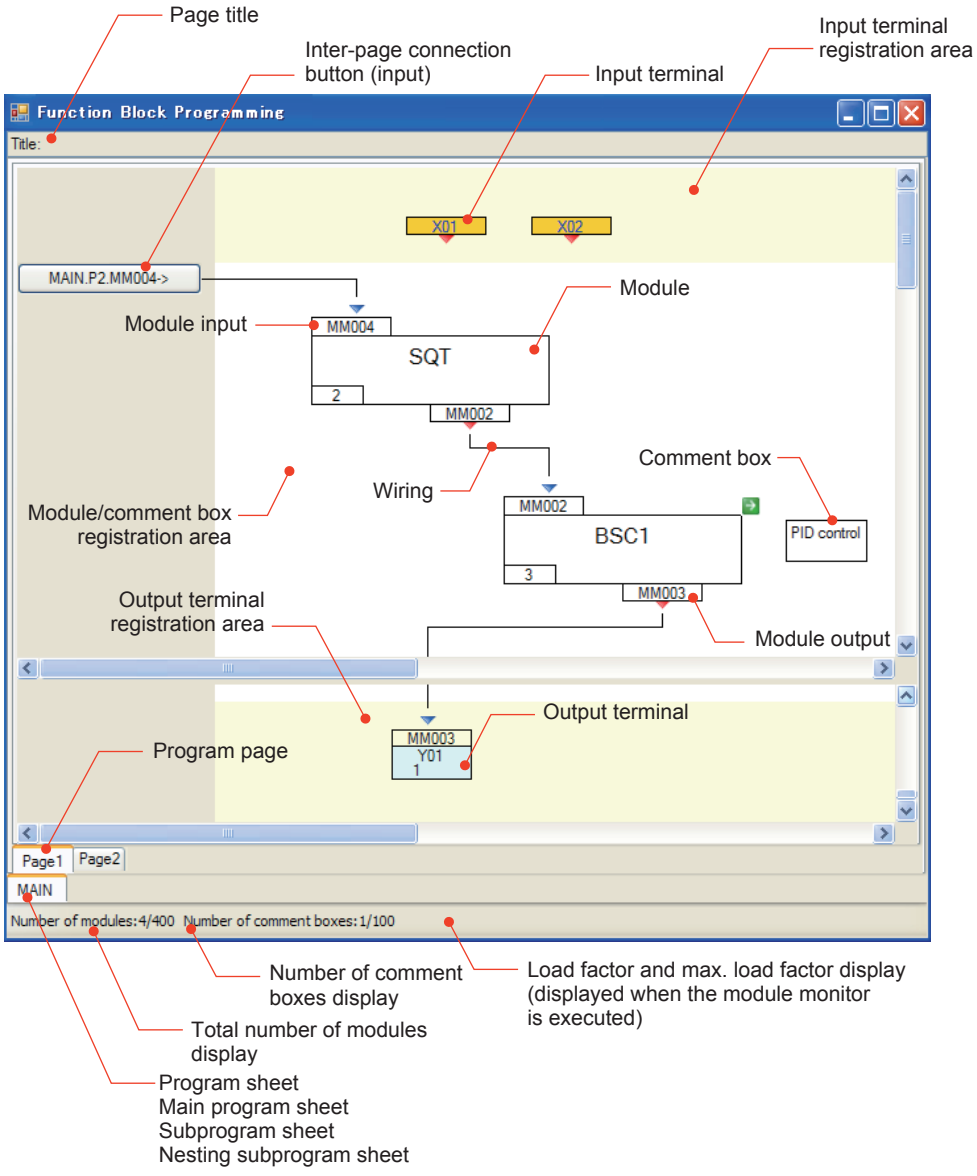

Procedure for Creating a New Program
Start up YSS1000.
Select "YS1000 Configuration Function" in the Select YSS1000 Function window.
The Basic window is displayed.
Select "New" from the menu.
The System Data Window is displayed.
Select "Model Name," "Suffix Code," "Optional Code," "Controller Mode (PROG)," and "Programming Method."
The File window is displayed in the Basic window.
Create (input) the program.
Set parameters, control period, and other program information.
Set event display.
Create data sheets.
Save setup data to file.
Download setup data.
Display simulated events.
Check user program operation. (Register monitor/function block diagram monitor/module monitor)
Perform tuning.
Execute a test-run.
Print.
End programming.
Design before Programming (Input)
(1) Extraction of control sections handled by YS1700
• Number of input points check
• Manipulated output variable check (max. 2 points for current output)
(2) Determination of control specifications
• One of BSC1, BSC2, CSC, and SSC is used as the control module.
• Determination of control elements (control type, control period)
• Use of extended function registers
(3) Determination of operation/display area
• Check of data to be manipulated/monitored on the front panel
(4) Summary of sequence and signal computations
• Scaling of computation signals
• Number of fixed and variable constants
• Number of dynamic operations
• Rough-check of program steps
(5) Program design
• Utilization of work sheets and data sheets
The operation procedures are described in the following sections:
► Programming using function blocks: "3.2.2 Programming Using Function Blocks"
► Programming using the text method: "3.3.2 Programming in Text"
Create a simulation program as necessary.
►2.4 Setting Parameters, Control Period, K Register, and P/X/Y Register Scale
Only when setting event display
►2.5 Setting Event Display
►2.6 Creating Data Sheets
►2.11.7 Saving Files under a Different Name
►2.7 Downloading Data
Only when setting event display
►2.5 Setting Event Display
►2.10.2 Monitoring Register Data
►2.10.4 Monitoring Control Module Function Blocks
►2.10.5 Monitoring Modules
Only when tuning PID
►2.10.1 Monitoring Tuning Data
See section 2.10 if necessary.
Only when a simulation program has been written.
• Checking and debugging the program and control functions (load factor, control operation sequence, computation accuracy)
►2.7 Downloading Data
►2.12 Printing

0301E.ai

3.2 Function Block Programming

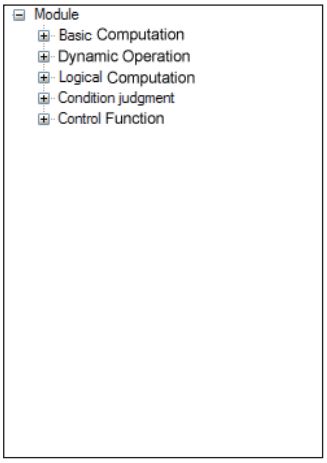
3.2.1 Names of Parts in the Function Block Programming Window

For details on each function, see "3.2.3 Explanation of Operations in the Function Block Programming Window."



0302E.ai

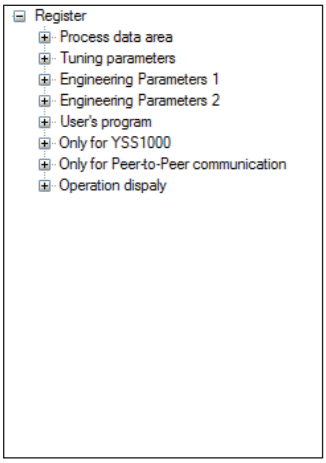
Module window



0303E.ai

Item	Operation
Window display/hide method	Click **View**>**Module window** from the menu, and check the check mark to display the window. Cancel the check mark to hide a window.
Registration of modules	Drag-and-drop the computing module to be used from the Module window and place it in the module/ comment box registration area.
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder. Right-click selects [Expand] or [Collapse] in the shortcut menu. When the tree is expanded, the module search function can be used by input from the keyboard.

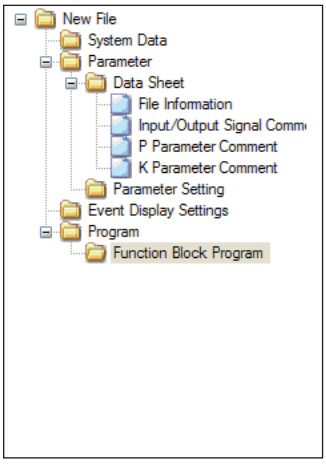
Register window



0304E.ai

Item	Operation
Window display/hide method	Click **View**>**Register window** from the menu, and check the check mark to display the window. Cancel the check mark to hide a window.
Set registers	Drag-and-drop registers to areas where they can be registered. The registers that cannot be registered cannot be registered to the registration area. • Input terminal registration area • Module input • Module parameters • Output terminal registration area • Input of storage register terminals • Input of storage register terminals with enable switch (EN) • Enable switch of storage register terminals with enable switch (EN) • Input of SUB OUT
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder. Right-click selects [Expand] or [Collapse] in the shortcut menu. When the tree is expanded, the register seach function can be used by input from the keyboard.

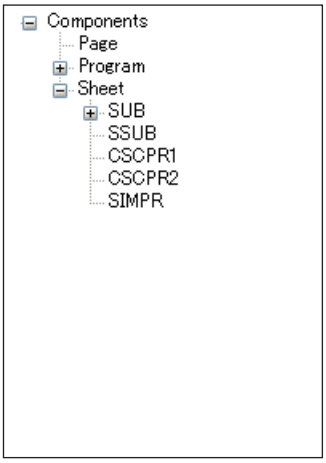
File window



0305E.ai

Item	Operation
Window display/hide method	Click **View**>**File window** from the menu, and check the check mark to display the window. Cancel the check mark to hide a window.
Open page	Click the folder of the window you want to display. The corresponding window is displayed.
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder.

Component Program window

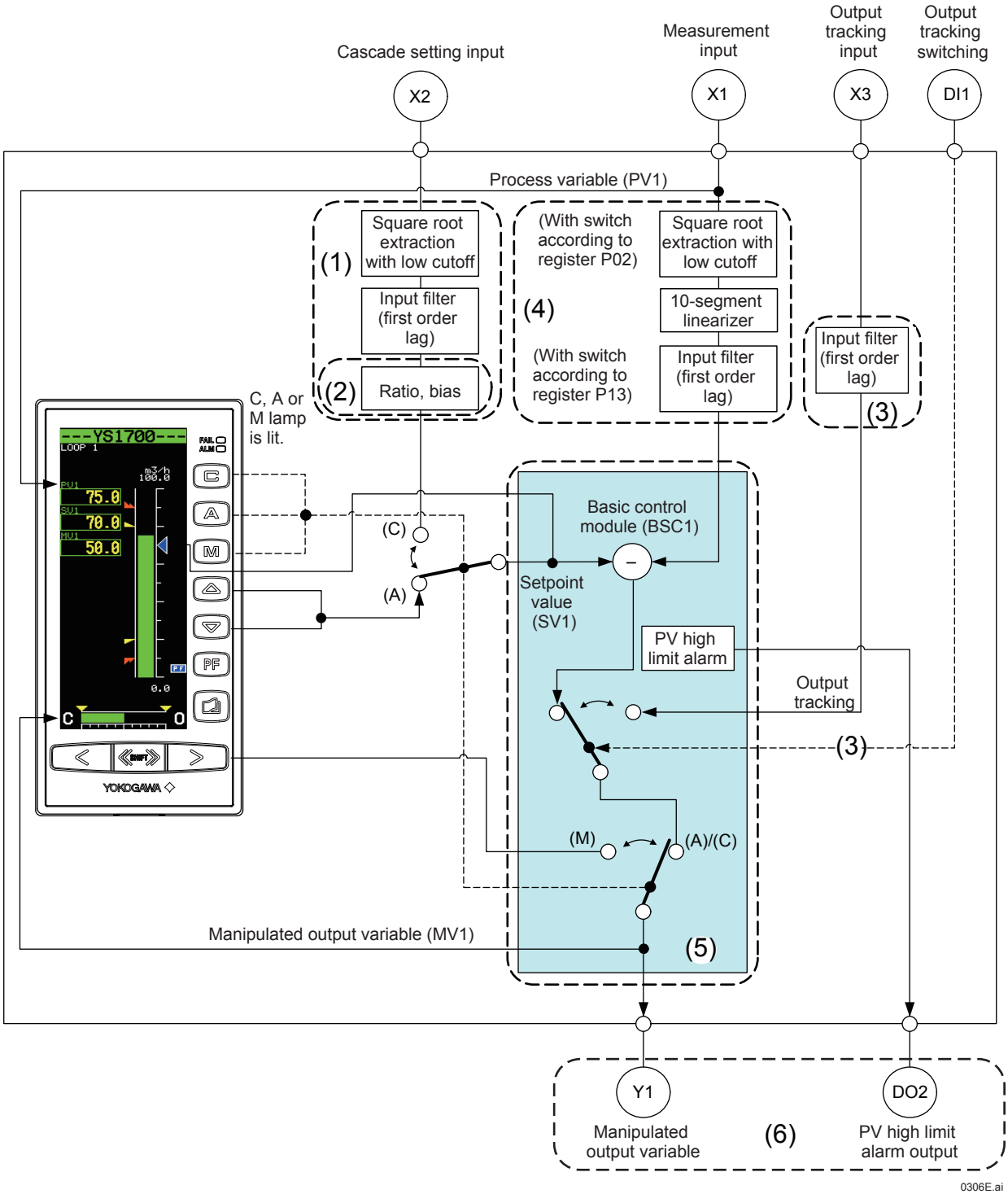


0305-01E.ai

Item	Operation
Window display/hide method	Click **View>Component Program Window** from the menu, and check the check mark to display the window. Cancel the check mark to hide a window.
Use component	Click the component file you want to use to display the Confirm Component Program window and Confirm Component Program Information window.
Expand/collapse tree	Click the + or - buttons to the left of the folder display. +: Displays the content under the current folder. -: Hides the content under the current folder. Right-click selects [Expand] or [Collapse] in the shortcut menu.

3.2.2 Programming Using Function Blocks

This section describes a simple programming procedure.
As an example of the procedure, create the function block diagram as follows.
Sample programs (single-loop control, cascade control, and selector control) are stored in the YSS1000 Setting Software. You can create a program by changing the sample program.



Create a program of the above function block diagram as described in the table below.

	Function	Sheet for Placing Functions	Execution Order (No. in figure)
Measurement input	Square root extraction with low cutoff 10-segment linearizer Input filter (first order lag)	Main program sheet (MAIN)	(4)
Cascade setting input	Square root extraction with low cutoff Input filter (first order lag)	Subprogram sheet (SUB@1)	(1)
	Ratio, bias	Nesting subprogram sheet (SSUB@201)	(2)
Output tracking input	Input filter (first order lag)	Subprogram sheet (SUB@2)	(3)
Basic control	Execution of basic control module	Main program sheet (MAIN)	(5)
Output	Manipulated output variable PV high limit alarm output		(6)

Note

Programs are executed in the order in which modules are registered. When registering modules, take the execution order into consideration.
The basic execution order is as follows: 1. input processing, 2. control computation, and 3. output processing.

Figure A: Explanation of Main Program (MAIN)

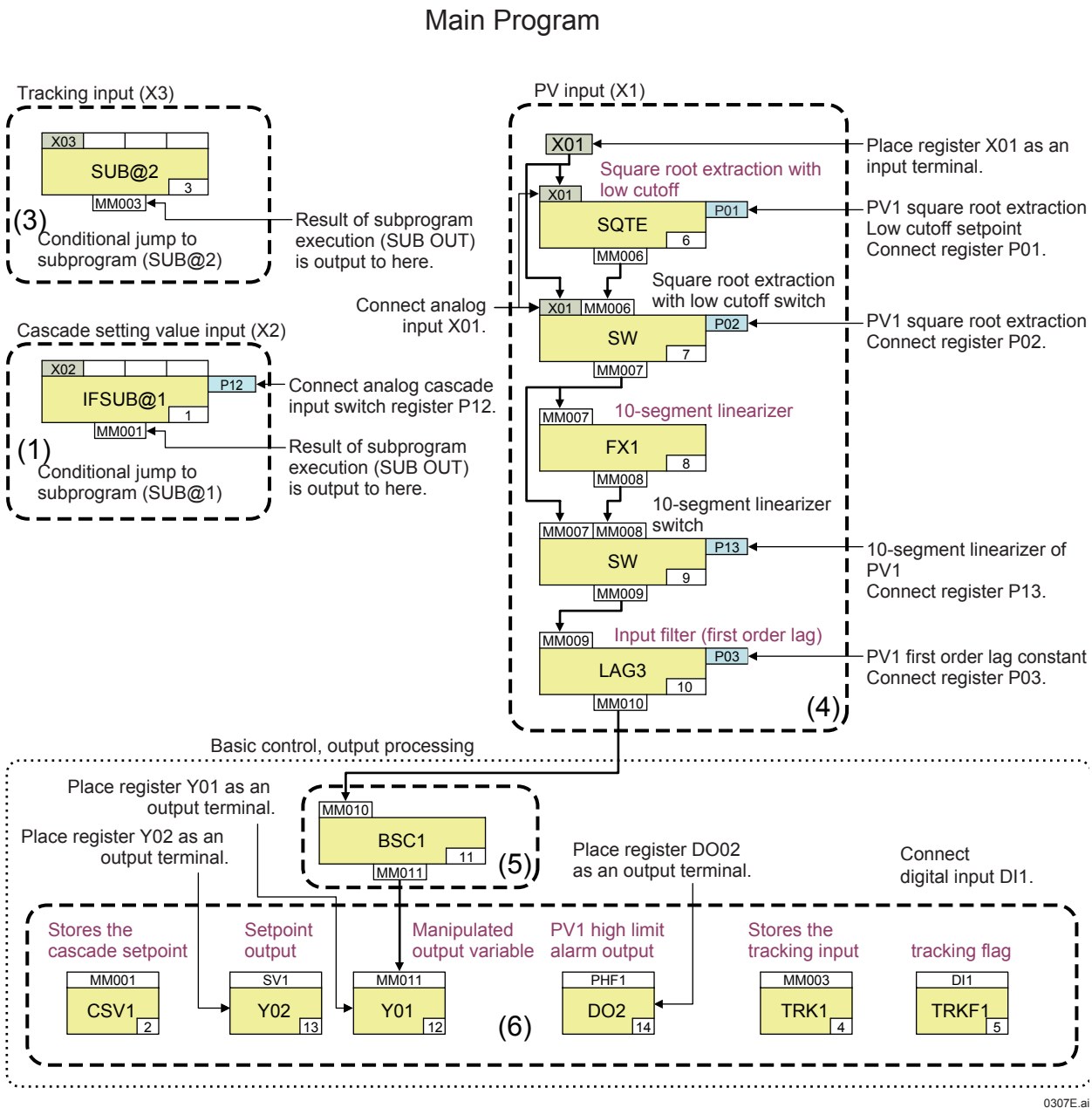


Figure B: Explanation of Subprogram (SUB@1)

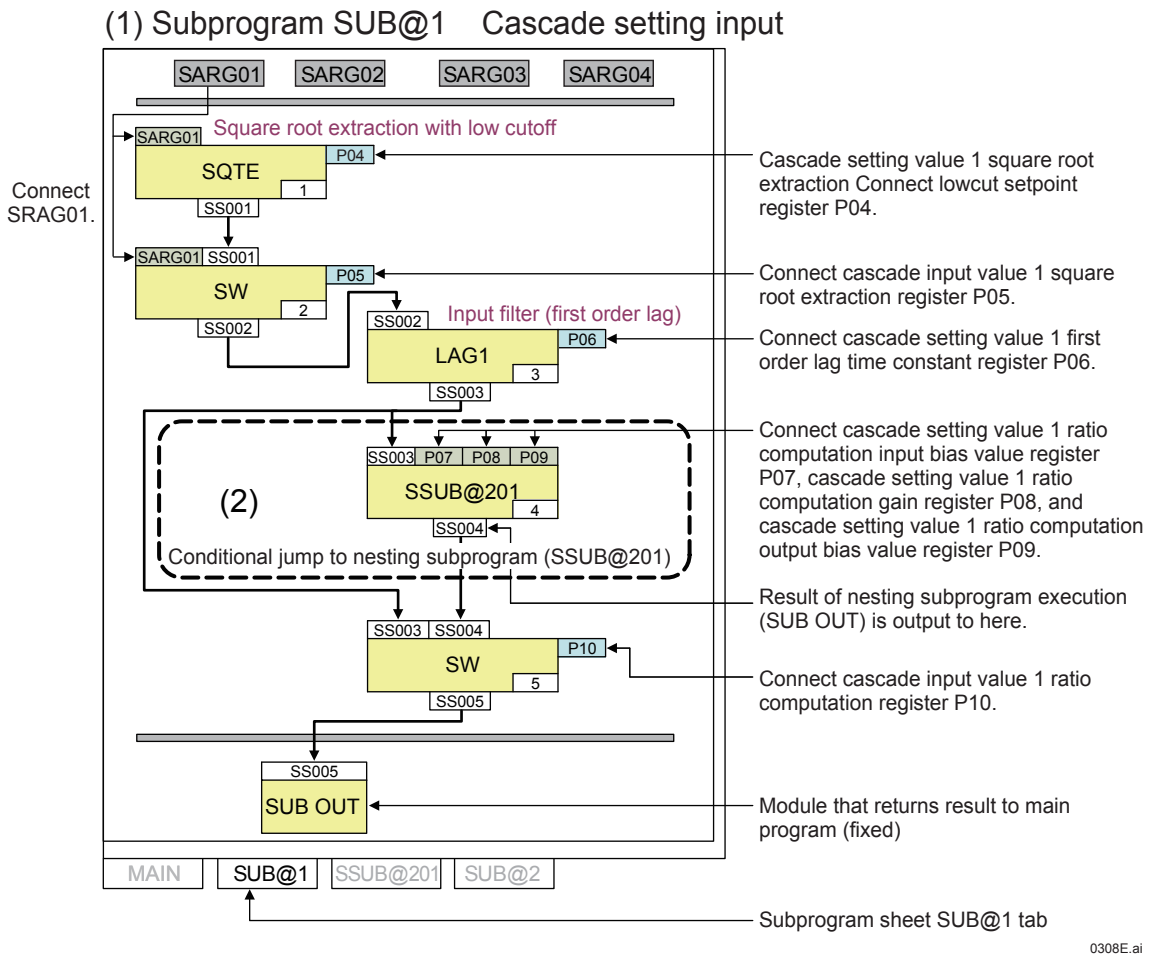


Figure C: Explanation of Nesting Subprogram (SSUB@201)

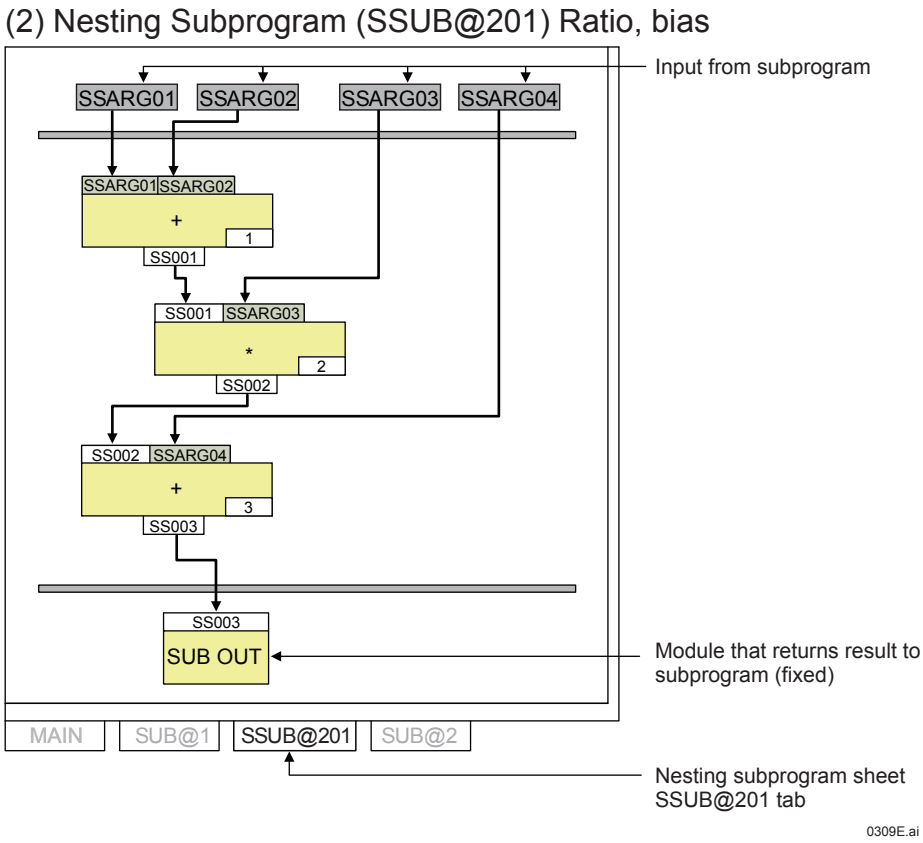
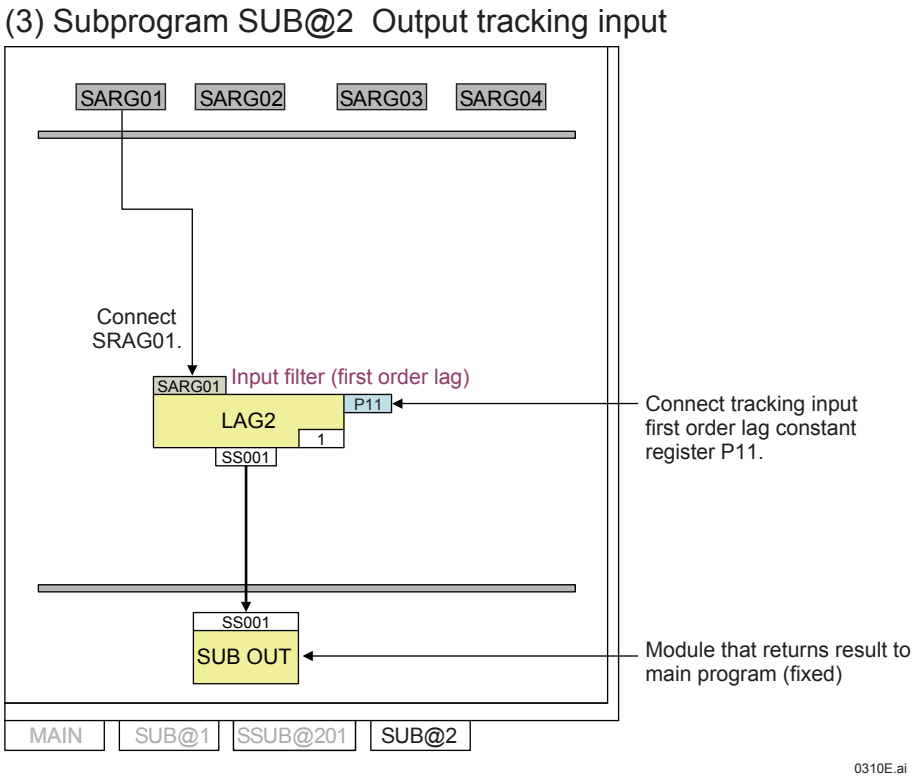
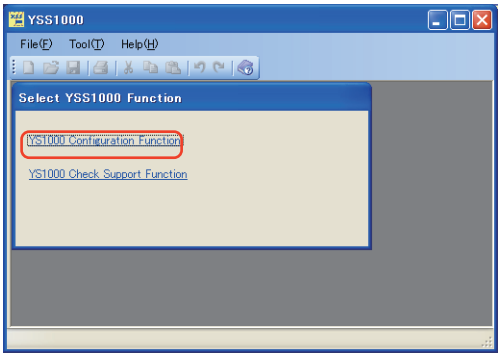


Figure D: Explanation of Subprogram (SUB@2)



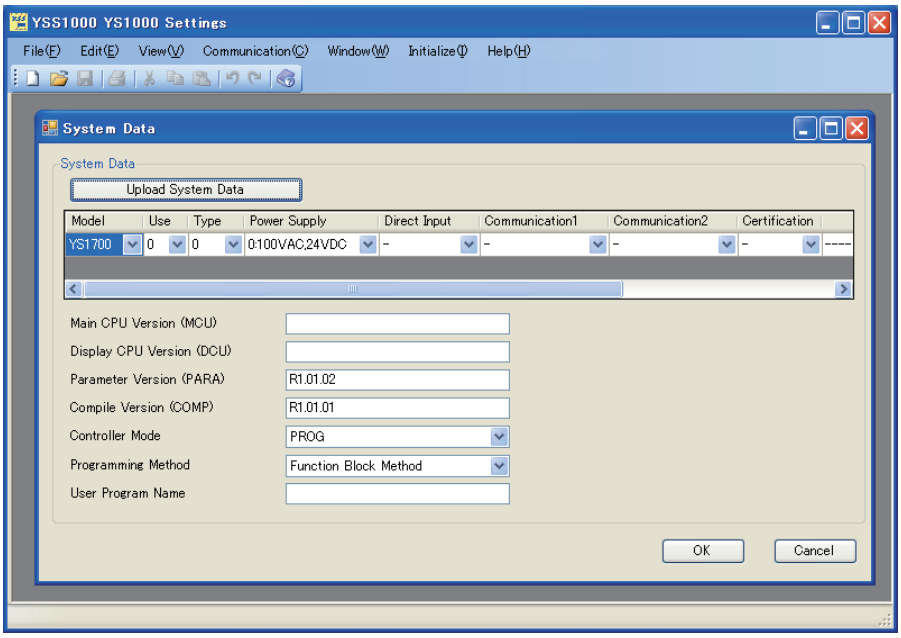
Operation

1. Start up YSS1000, and click "**YS1000 Configuration Function**" in the Select YSS1000 Function window.



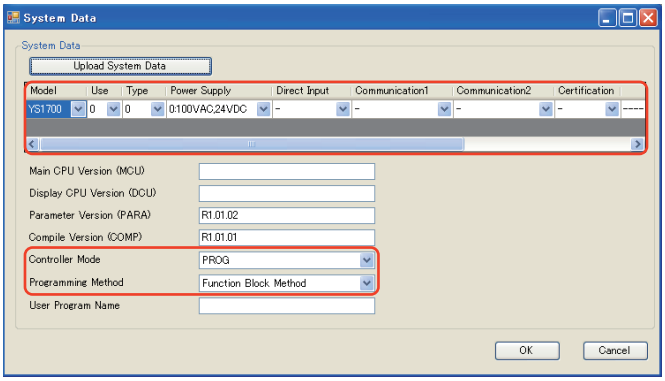
0311E.ai

2. Click **File>New** from the menu in the Basic window or on the toolbar to display the System Data window.



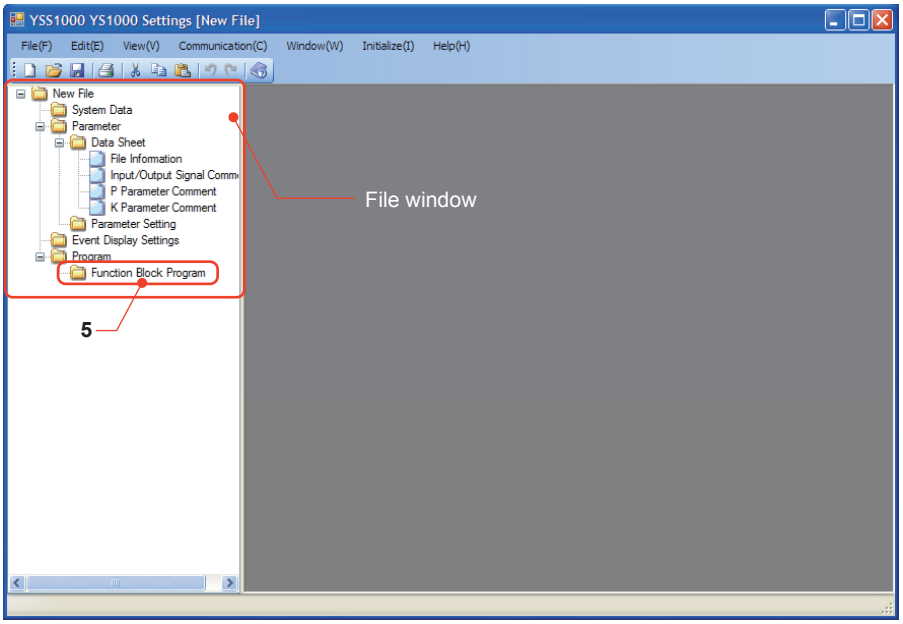
0312E.ai

3. Select model name "YS1700", suffix code and optional code of the controller type to be set, and set the controller mode and the programming method respectively to "PROG" (programmable mode) and "Function Block Method", and click **OK**.



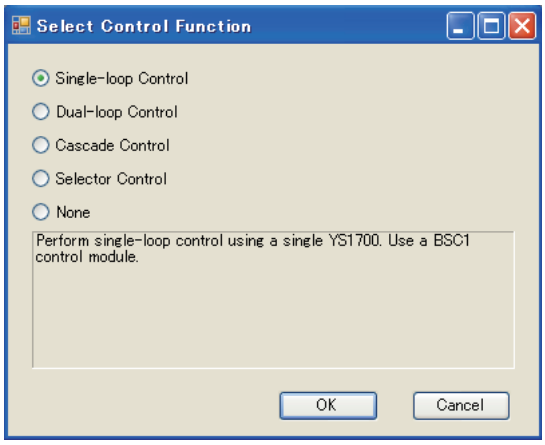
0313E.ai

4. The File window is displayed on the Basic window.



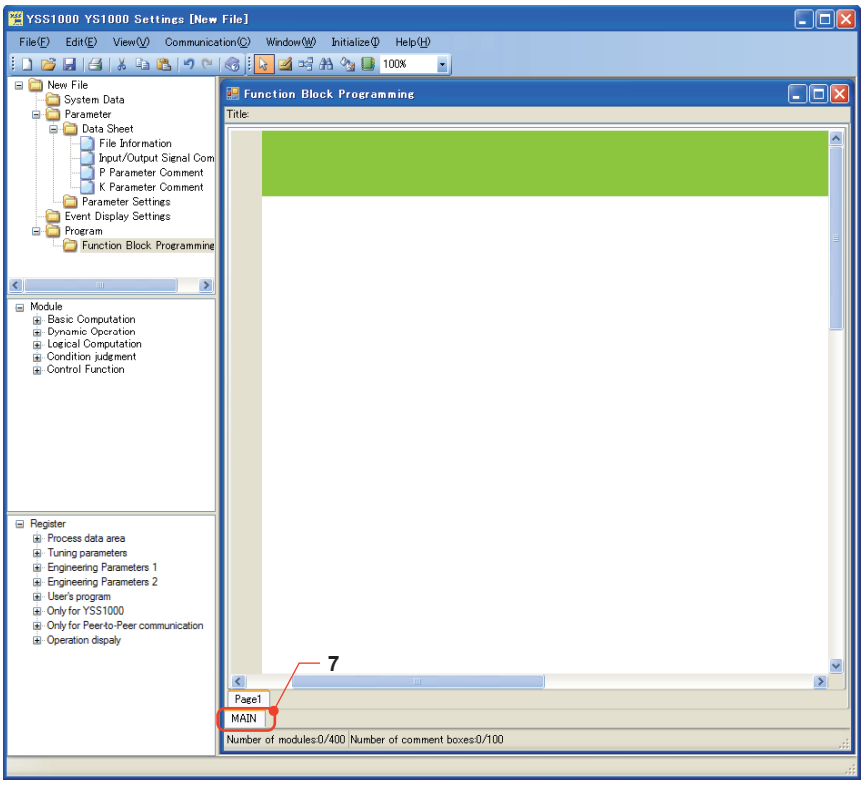
0314E.ai

5. Click **Function Block Programming** in the File window to display the Select Control Function window.



0315E.ai

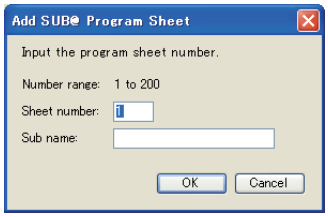
6. Select the control function, and click **OK** to display the Function Block Programming window.



0316E.ai

7. Before starting programming, create a sheet. Sheets and pages can be added also during programming.

The following procedure describes addition of a subprogram. The same procedure can be applied for a nesting subprogram.
With the programming window active, click **Edit**>**Add Program Sheet**>**Add SUB@n Program** from the menu, or right-click the **MAIN** tab on the programming window, and click **Add SUB@n Program**.



0316E-2.ai

8. Start programming.

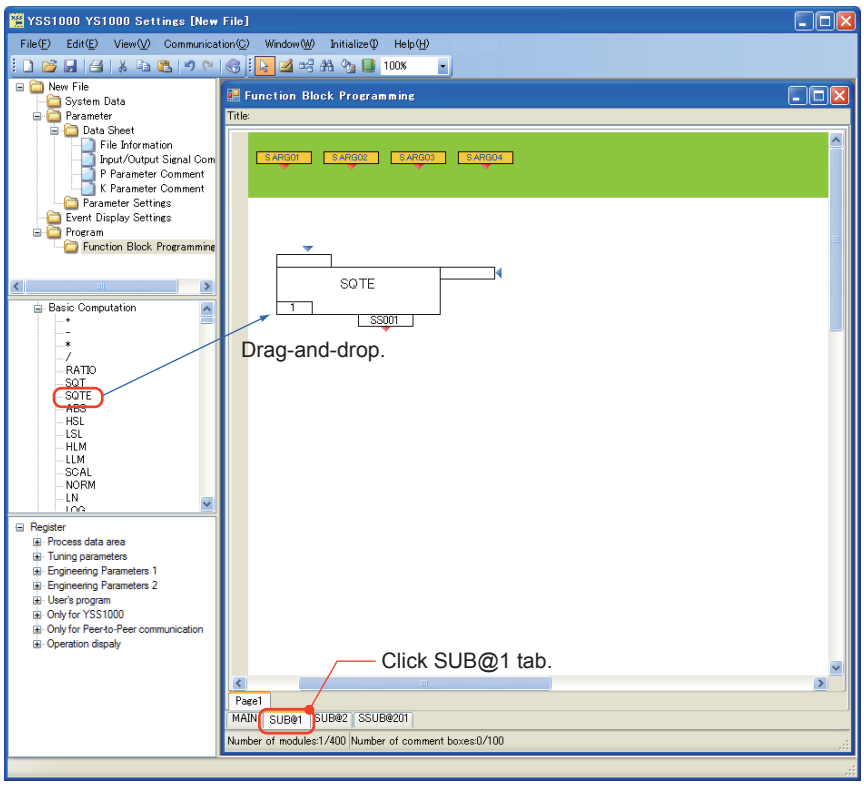
Note

The order in which computing modules are placed is important in programming. Computation processing is executed in the order in which computing modules are placed. Programming is also made easier by deciding on the positions for placing computing modules beforehand. The execution order and position of computing modules can also be changed midway.

Module search function

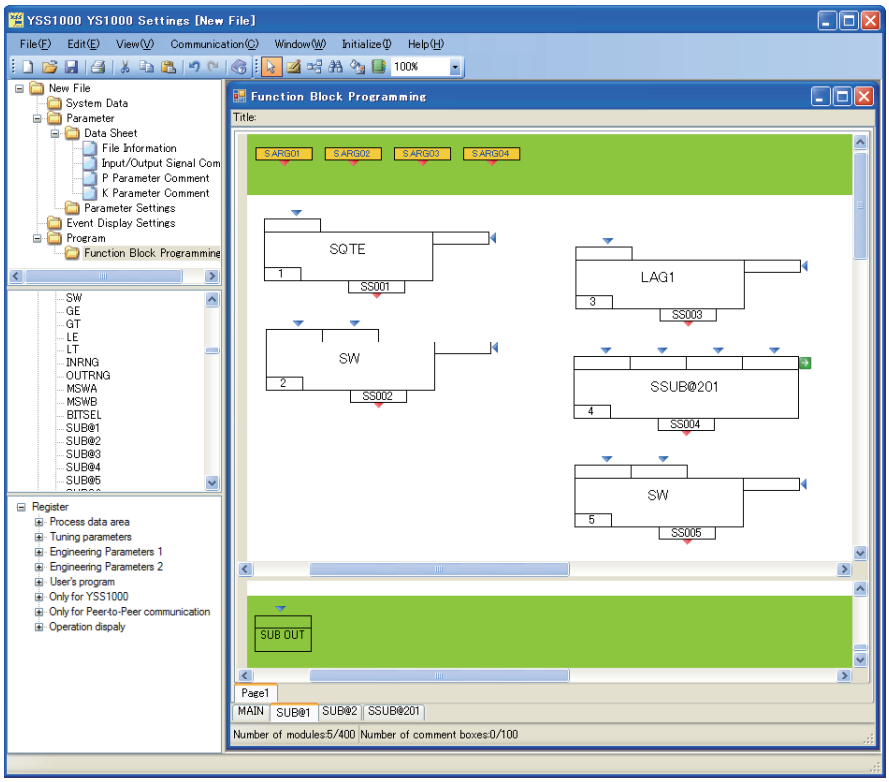
When the Module window is active, right-click the window, select [Expand], and enter the module symbol in alphanumerics on the keyboard.

9. First, place the square root extraction with low cutoff module on the subprogram sheet (SUB@1) while viewing Figure B: Subprogram (SUB@1).



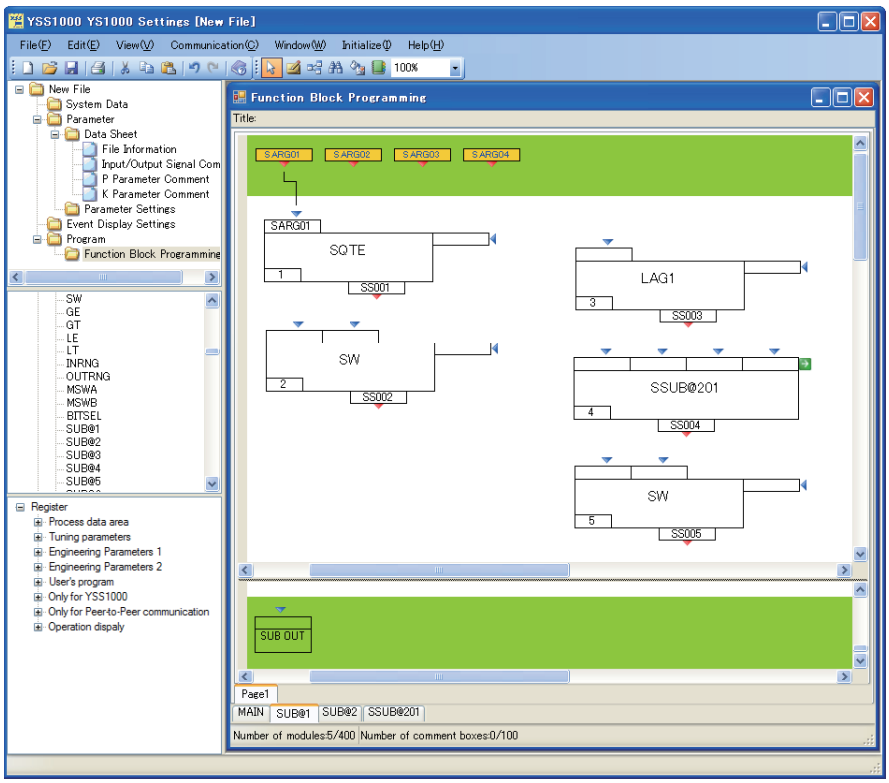
0317E.ai

10. Place other computing modules by the same method.



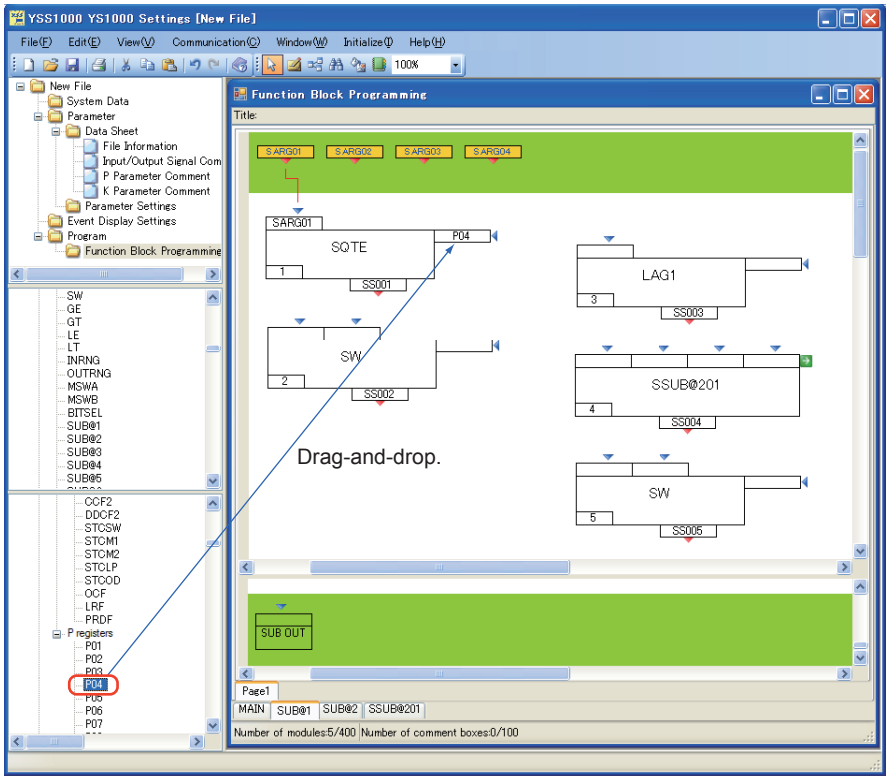
0318E.ai

11. Next, connect the inputs and outputs of each computing module. Connect analog input register X02 to the input of the SQTE module.



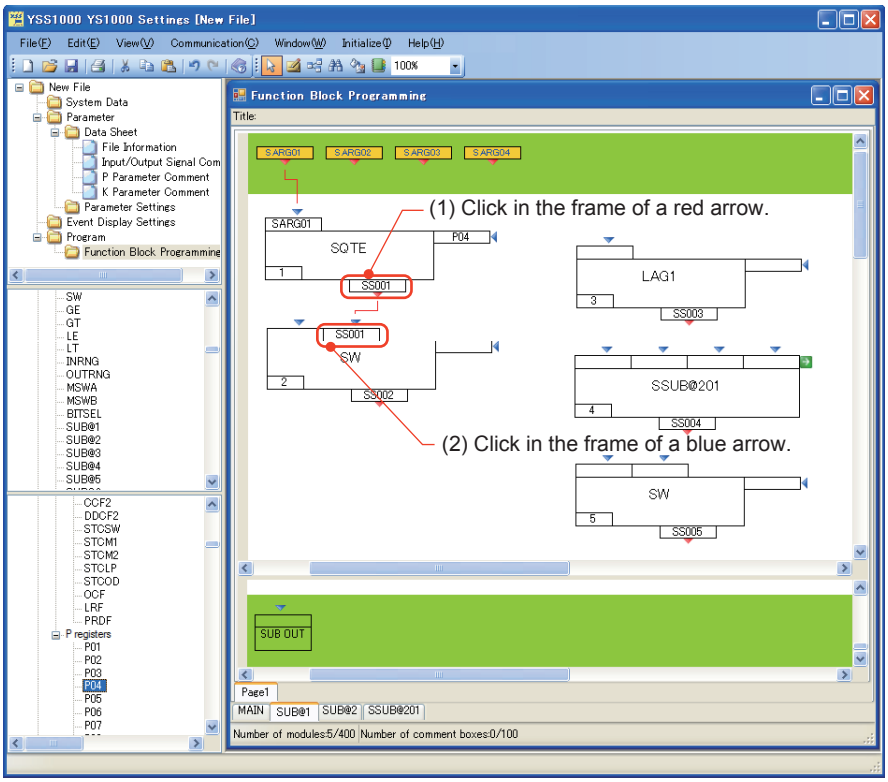
0319E.ai

12. Connect cascade setting value 1 square root extraction low cutoff setpoint register P04 to the parameters of the SQTE module.



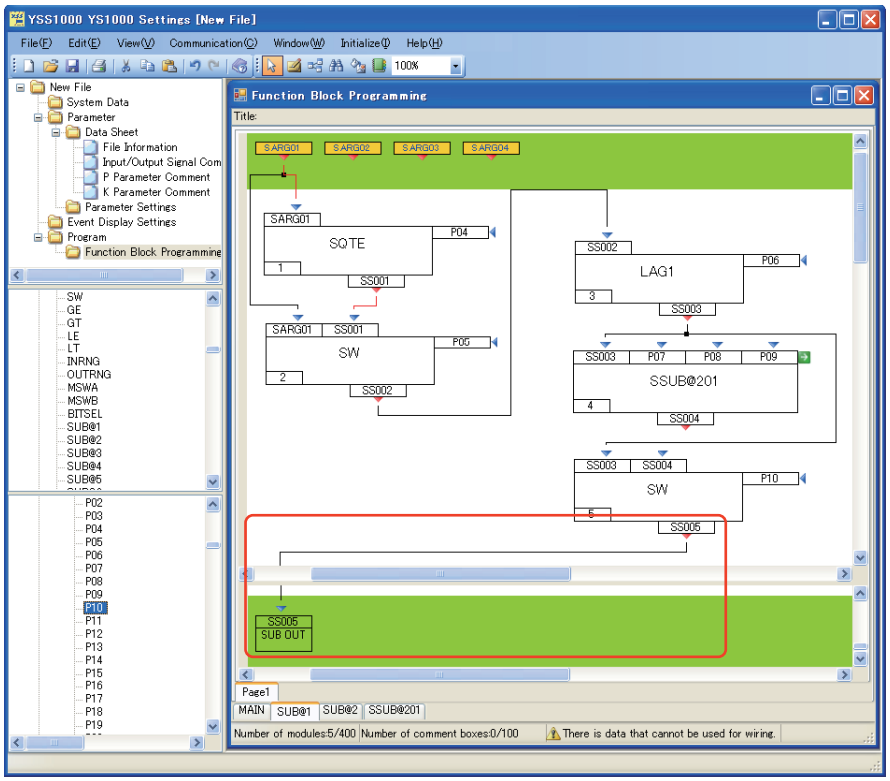
0321E.ai

13. Connect the output of the SQTE module to the input of the SW module. Click the red line of the connection source, and the blue line of the connection destination. Lines of different color can be connected together. Lines of the same color cannot be connected together.



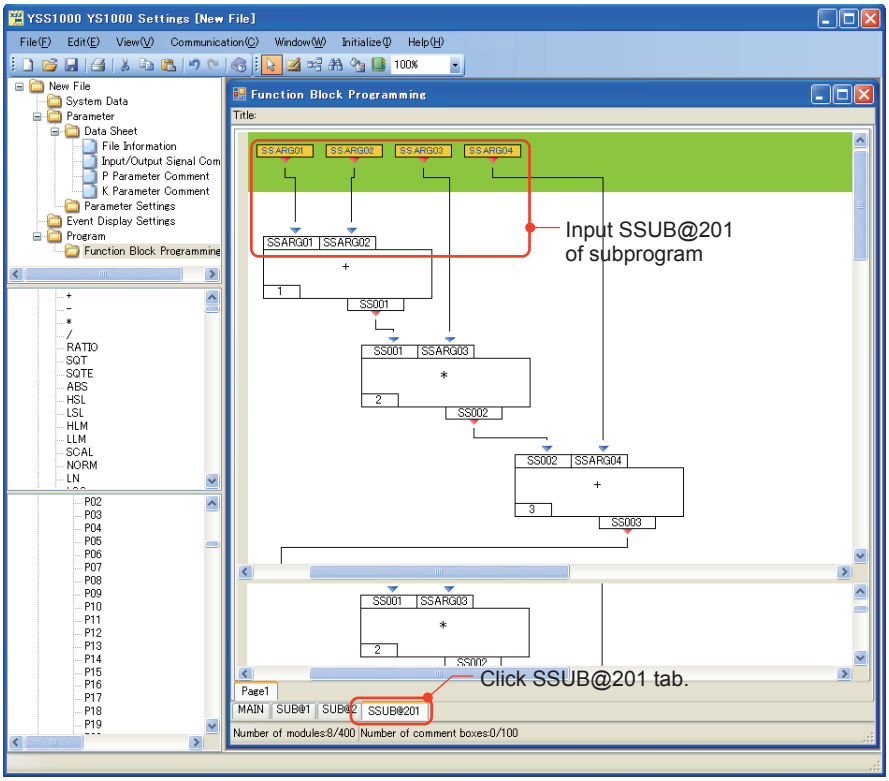
0322E.ai

14. Connect other modules. Connect to the SUB OUT module so that the result of subprogram execution is returned to the main program. This completes programming of subprogram SUB@1.



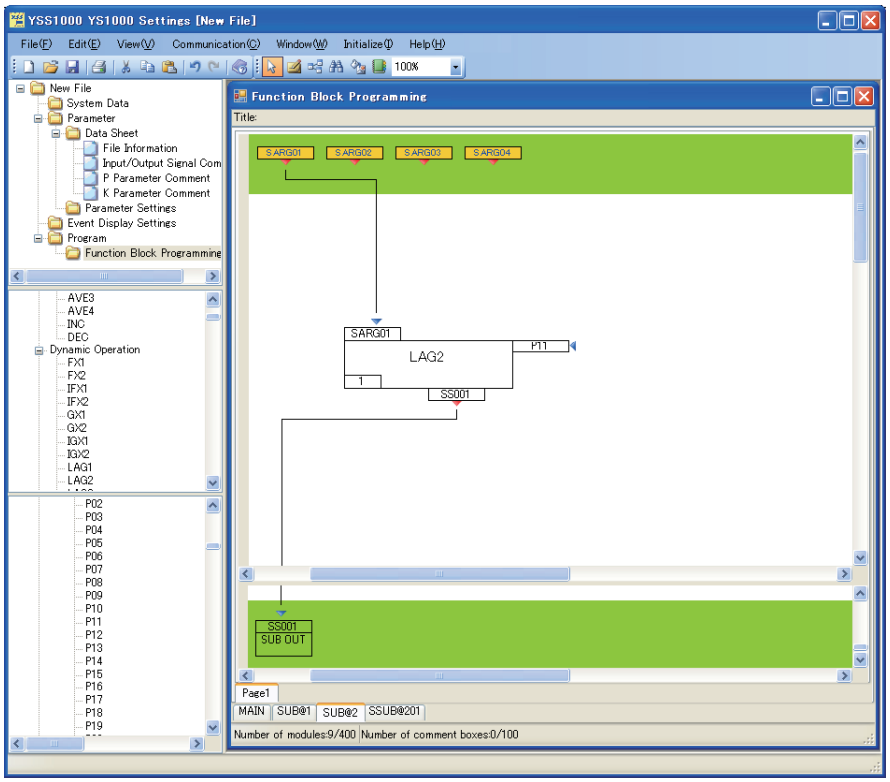
0323E.ai

15. The nesting subprogram SSUB@201 is located in Figure B: Subprogram (SUB@1). Create the nesting subprogram. Place and connect modules in the same way as for the subprogram while viewing Figure C: Nesting Subprogram (SSUB@201). Programming of the nesting subprogram involves input of arguments. Perform the connections in the same way.



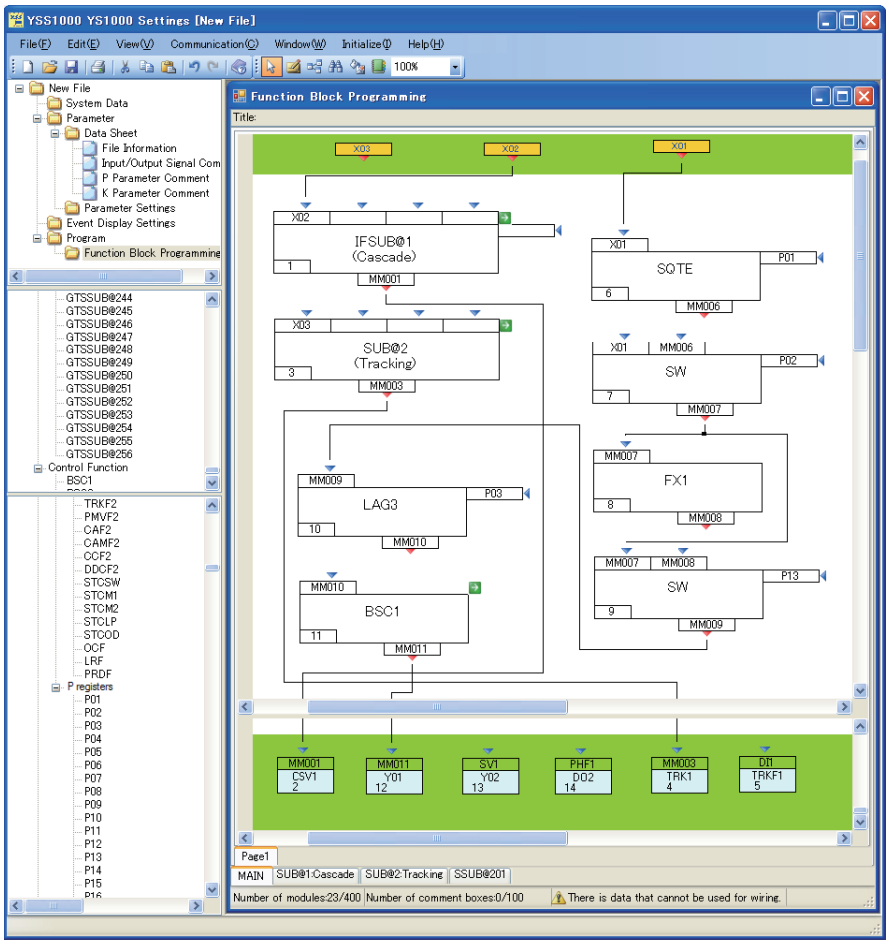
0324E.ai

16. Create the other subprogram while viewing Figure D: Subprogram (SUB@2).



0325E.ai

17. Create the main program. Place and connect modules in the same way as for the subprograms and nesting subprogram. This completes programming of all programs.



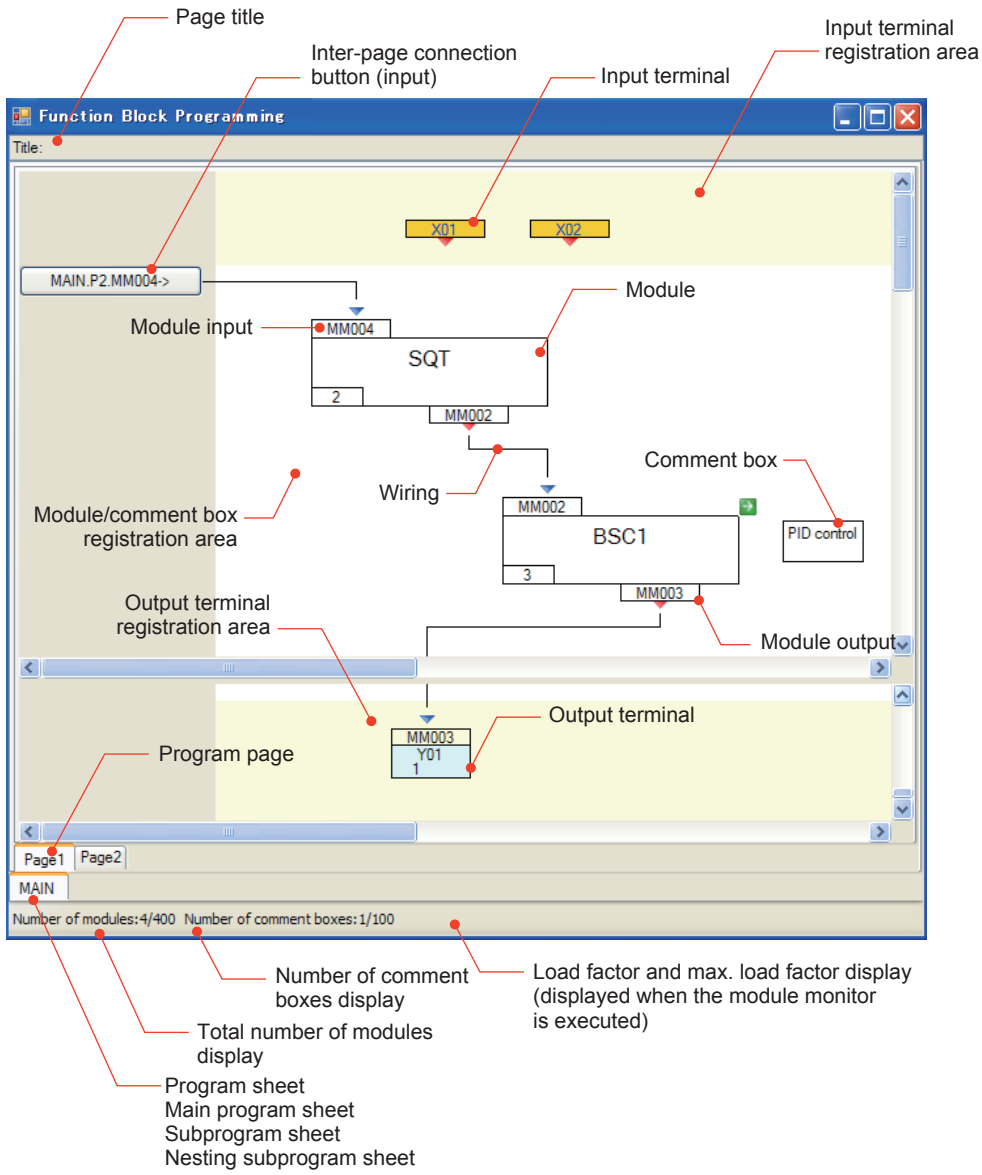
0326E.ai

18. Perform a program check by selecting **Tool>Program Check** when the Programming window is open. If a compile error is generated, a review of the user programs is required.

19. After the data is saved to file, download it to YS1000 and then perform an operation check.

20. Creating the user program is completed.

3.2.3 Explanation of Operations in the Function Block Programming Window



0327E.ai

(1) Page title

Page title	Operation
Switch to Input (Edit) mode	Switch by one of the following methods: (1) Double-click inside an area. (2) Click inside an area and press the F2 key. (3) Click and select a comment, and enter an accepted input character. This enables editing and the existing comment is overwritten. When the Edit mode is switched to, the cursor is displayed in the page title.
Accepted input character:	All alphanumerics and symbols that can be entered on the keyboard (However, tabs cannot be input.)
Input (Edit)	Input by one of the following methods: (1) Input and delete from the keyboard. (2) By the “Copy,” “Cut,” “Paste,” and “Delete” functions in the shortcut menu
Max. number of entered characters	160 characters

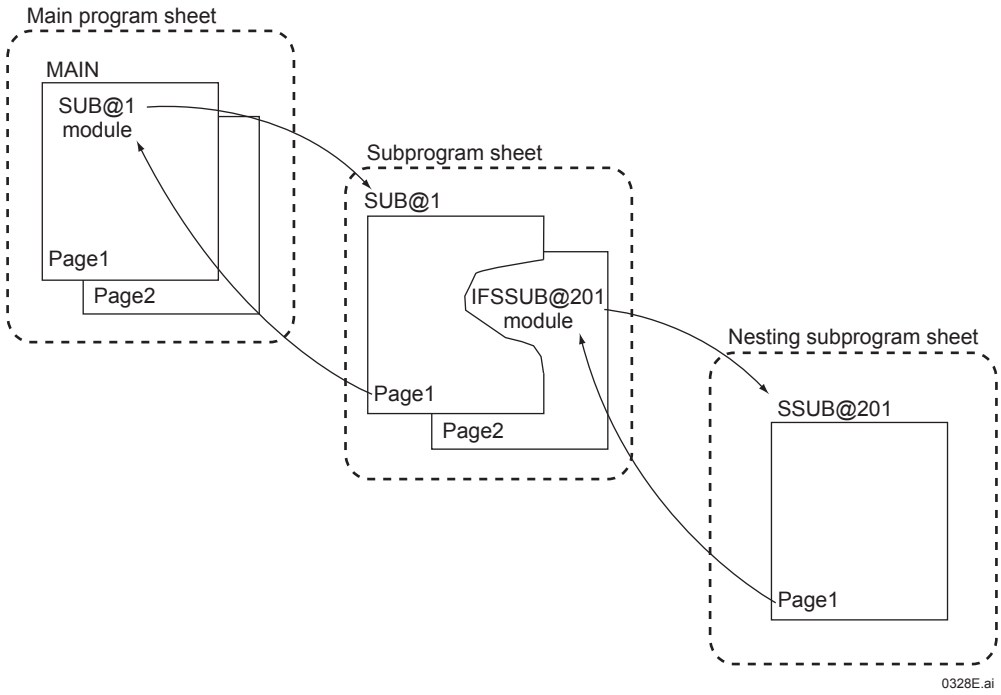
(2) Program sheets

Types of program sheets

Name	Description of Program	Number of Sheets
MAIN	Main program	1
SUB@n (*1)	Subprogram	0 (SUB@n unused) or n = 1 to 200
SSUB@n (*1)	Subprogram for nesting	0 (SSUB@n unused) or n = 201 to 256
SUB@CSCPRn	Computation subprogram between cascade loops	0 (CSCPRn unused) or n = 1 and 2
SUB@SIMPR	Simulation program	0 (SIMPR unused) or 1

*1: The sub name is displayed on the program sheet name when it is set.
Note: Program sheets cannot be moved.

The figure below shows the basic method of use of the main program sheet, subprogram sheet and nesting subprogram sheet.



Program sheet operations

Add and Delete		Operation
Adding program sheets	(1)	The corresponding subprogram sheet is automatically added when a subprogram module is added.
	(2) (*1)	Right-click an existing program sheet, and add from the shortcut menu that is displayed. (*3) “Add Program>Add SUB@n Program” “Add Program>Add SSUB@n Program” “Add Program>Add SUB@CSCPRn Program” “Add Program>Add SUB@SIMPR Program”
	(3) (*1)	**Edit**>**Add Program Sheet**>(select one of following)(*3) from the menu “Add SUB@n Program” “Add SSUB@n Program” “Add SUB@CSCPRn Program” “Add SUB@SIMPR Program”
Deleting program sheets	(1) (*2)	Right-click an existing program sheet, and execute “Delete Program Sheet” from the shortcut menu that is displayed. The main sheet cannot be deleted.
	(2) (*2)	Select **Edit**>**Delete Program Sheet** from the menu. The program sheet displayed now is deleted. The main sheet cannot be deleted.
Switching between program sheets		Switch between program sheets by one of the following methods: (1) Click the Program Sheet tab. (2) Other method (e.g. jump to the desired sheet from the search results or from Function Block Window of Control Module)

*1: Add by this method when adding a subprogram sheet after a subprogram sheet has been deleted, or when adding a SUB@CSCPRn or SUB@SIMPR program sheet.

*2: All pages in the program sheet are deleted when a program sheet is deleted. Deletions cannot be undone.

*3: In the case of “Add SUB@n Program” and “Add SSUB@n Program”, set the number of the sheet (@n). This number is the number of the subprogram module when the subprogram is called.

(3) Program page

With program pages, a running page number is added to each program sheet.
Each program sheet can hold up to 256 program pages. Pages cannot be moved.

Add, Delete, Display		Operation
Adding pages	(1)	Right-click an existing program page tab, and execute “Add Page” from the shortcut menu that is displayed. A page is added as the last page.
	(2)	Select **Edit**>**Add Page** from the menu. A page is added as the last page.
	(3) (*1)	Select **Edit**>**Insert page** from the menu. The page is inserted before the active page.
	(4) (*1)	Right-click an existing program page tab, and execute “Insert Page” from the shortcut menu that is displayed. The page is inserted before the active page.
Deleting pages	(1) (*2)(*3)	Right-click the page tab to be deleted, and execute “Delete Page” from the shortcut menu that is displayed.
	(2) (*2)(*3)	Set the page to be deleted to the active page, and click **Edit**>**Delete Page** from the menu.
Switching between program pages		Switch between program pages by one of the following methods: (1) Click the page tab. (2) Other method (e.g. jumping to the desired page from the search results or from Function Block window of Control Module)
Page title display		The page title is displayed on the tip of the tool when the mouse pointer is moved to the page tab.

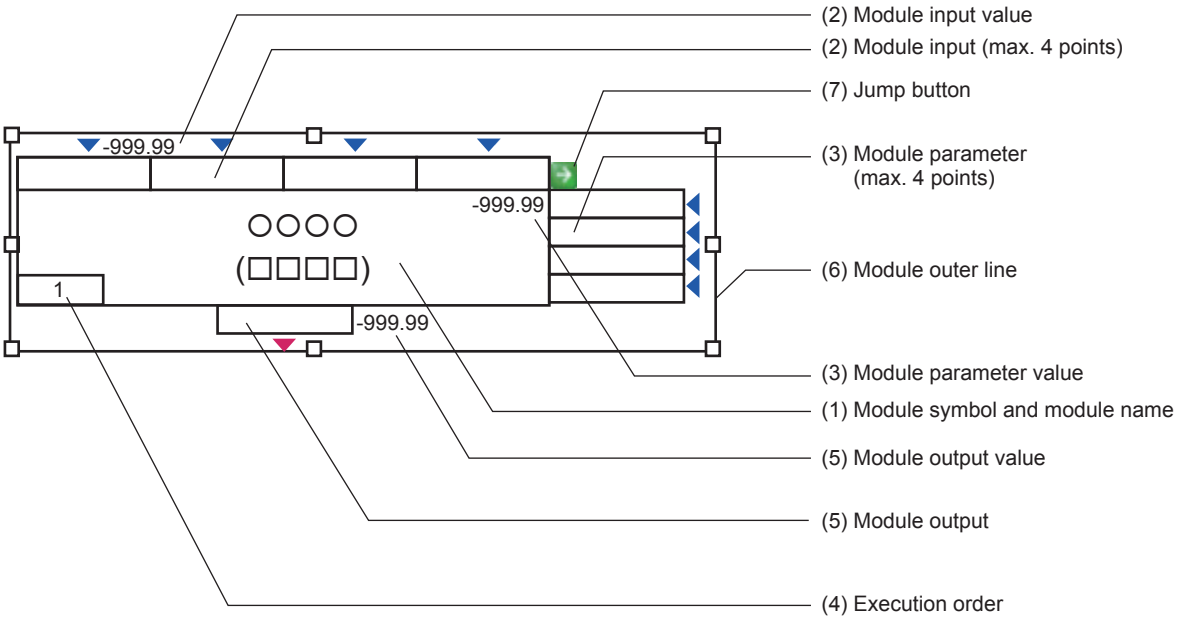
*1: When a page has been inserted, the page numbers from the newly inserted page onwards are automatically corrected.
Note: The page number displayed on the inter-page connection button also is corrected.
*2: This method is invalid when there is only page 1. Deletions cannot be undone. After a delete, the page number is automatically corrected.
Note: The page number displayed on the inter-page connection button also is corrected.
*3: Pages programmed with a SUB OUT output terminal cannot be deleted. To delete these pages, move the SUB OUT output terminal to another page by cutting-and-pasting, and delete the original page.

(4) Module/comment box registration area

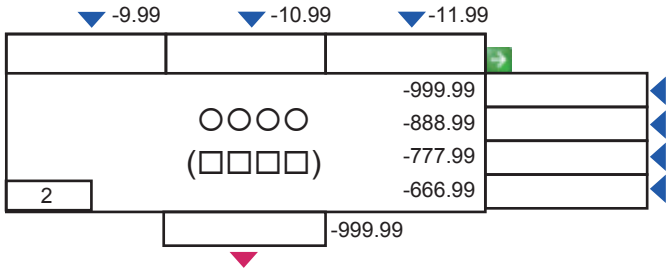
This area is for registering modules and comment boxes. As a guideline for the number of modules that can be registered, about 32 computing modules of two input points per page can be placed.
When two modules are set as a pair, the two modules are placed together. (The modules will be registered together if you drag-and-drop either of the modules.)
Module pairs: PGM1_A and PGM1_B, PGM2_A and PGM2_B, PGMM1_A and PGMM1_B, PGMM2_A and PGMM2_B

Registration	Operation
Module registration	Drag-and-drop the module symbol from the Module window to the module registration area and register it. Modules can be registered anywhere in the module registration area.
Comment box registration	Register by one of the following methods: (1) Right-click the registration area, and execute “Register Comment Box” from the shortcut menu that is displayed. (2) Click the Comment Box button on the Toolbar, and click the desired location to register the comment box. Comment boxes can be registered anywhere in the module registration area.

(5) Modules



0329E.ai



0330E.ai

(1) Module symbol and module name
The module symbol indicates the functions of the module. Register module names as necessary.
The module colors can be changed. To change a module color, see “Module operations” described later.

(2) Module input/module input value
The number of inputs differs according to the module. For details, see “Chapter 7 Operations and Application of Computing Module (Instructions).”
Module input values are displayed only when the module monitor is activated.

(3) Module parameter/module parameter value
The number of parameters differs according to the module. For details, see “Chapter 7 Operations and Application of Computing Module (Instructions).”
Module parameter values are displayed only when the module monitor is activated.

(4) Execution order
The execution order is displayed. The execution order is assigned to modules in the order that modules are registered. The execution order of modules can be changed midway. See “3.2.4.”

(5) Module output/module output value
The module output symbol is displayed.
Module output values are displayed only when the module monitor is activated.
The register name is automatically determined by the order in which modules are registered.

Module output registers for main program (MM001 to MM400)

Register Name	Description
MM001	Output data of execution order No.1 of main program
MM002	Output data of execution order No.2 of main program
MM003	Output data of execution order No.3 of main program
・ ・	
MM400	Output data of execution order No.400 of main program

Module output registers for subprogram (SS001 to SS400)
Example) @1 subprogram: 3 modules
@2 subprogram: 2 modules
@3CSCPR1 program: 4 modules

Register Name	Description
SS001	Output data of execution order No.1 of subprogram @1
SS002	Output data of execution order No.2 of subprogram @1
SS003	Output data of execution order No.3 of subprogram @1
SS001	Output data of execution order No.1 of subprogram @2
SS002	Output data of execution order No.2 of subprogram @2
SS001	Output data of execution order No.1 of @CSCPR1
SS002	Output data of execution order No.2 of @CSCPR1
SS003	Output data of execution order No.3 of @CSCPR1
SS004	Output data of execution order No.4 of @CSCPR1
・ ・	
SS400	

Module output registers for simulation program (SM001 to SM010)

Register Name	Description
SM001	Output data of execution order No.1 of simulation program
SM002	Output data of execution order No.2 of simulation program
SM003	Output data of execution order No.3 of simulation program
・ ・	
SM010	Output data of execution order No.10 of simulation program

(6) Module outer line (box specification)
The monitor value is displayed inside the line when the module monitor is activated.

(7) Jump button

The Jump button is available for only the following modules.

Clicking the Jump button switches the display to the jump destination.

Module		Jump Destination
Control module	BSC1	Function Block window of Control Module
	BSC2	
	CSC	
	SSC	
Subprogram module	SUB@n	1st page of corresponding n subprogram sheet
	IFSUB@n	
	GTSUB@n	
	SSUB@n	
	IFSSUB@n	
	GTSSUB@n	

Maximum number of registerable modules

Program Sheet	Maximum Number of Registerable Modules
MAIN SUB@1 to 200 SSUB@201 to 256 SUB@CSCPR1 SUB@CSCPR2	Total of program sheets on left 400
SUB@SIMPR	10

Note: Including storage register terminals and storage register terminals with enable switch (EN)

Registable sheets of subprogram modules

Subprogram module	Registable Sheets of Subprogram Modules
SUB@1 to 200 IFSUB@1 to 200 GTSUB@1 to 200	MAIN
SSUB@201 to 256 IFSSUB@201 to 256 GTSSUB@n201 to 256	MAIN SUB@n SUB@CSCPR1 SUB@CSCPR2

Module operations

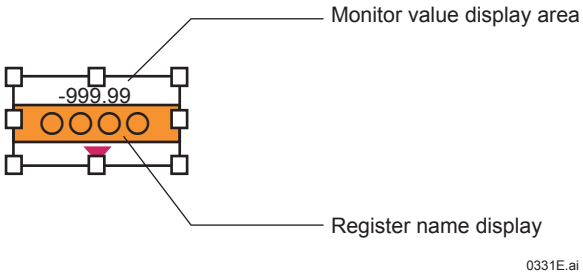
Item	Operation
Setting registers to module inputs	Set by one of the following methods: (1) Drag-and-drop registers from the Register window. (2) Click between the wiring start and end points. When setting of registers is completed, the register names are displayed at module inputs. For details, see "(11) Wiring" described later.
Setting registers to module parameters	Set by one of the following methods: (1) Drag-and-drop registers from the Register window. (2) Click between the wiring start and end points. When setting of registers is completed, the register names are displayed at module parameters. For details, see "(11) Wiring" described later.
Changing the execution order	Click **Edit**>**Change Execution Order** from the menu. See "3.2.4 Changing the Program Execution Sequence."
Switching the window display by a module's Jump button	When a module is provided with a Jump button, click this Jump button to jump to the corresponding page or window.
Change Subprogram Number	The subprogram number of subprogram module can be changed. See "3.2.7 Setting the Numbers of Subprogram Modules and Nesting Subprogram Modules."
Cutting modules	Select the module to be cut (*1), and perform one of the following: (1) Select **Edit**>**Cut** from the menu. (2) Click the Cut button on the toolbar. (3) Right-click, and select "Cut" in the shortcut menu that is displayed.
Copying modules	Select the module to be copied (*1), and perform one of the following: (1) Select **Edit**>**Copy** from the menu. (2) Click the Copy button on the toolbar. (3) Right-click, and select "Copy" in the shortcut menu that is displayed.
Pasting modules	Click the position to paste the module at, and perform one of the following: (1) Select **Edit**>**Paste** from the menu. (2) Click the Paste button on the toolbar. (3) Right-click, and select "Paste" in the shortcut menu that is displayed. If you paste a module after a cut in the same program sheet, the execution order of the modules stays the same. if you paste a module after a copy, a module is added to the program, so the pasted module is assigned following the final line of the execution order.
Moving modules	Modules can be moved only inside the module registration area in the same page. Move modules by one of the following methods: (1) To move only one module: Click the module to make it active, and drag-and-drop to move it. (2) To move multiple modules selected by box specification: Specify BOX for the desired modules and comment box to be moved, and drag-and-drop it to move.
Deleting modules	Select the module to be deleted (*1), and perform one of the following: (1) Select **Edit**>**Delete** from the menu. (2) Right-click, and select "Delete" in the shortcut menu that is displayed. (3) Press the Delete key on the keyboard.
Setting the module color	Select the module to be colored (*1), and perform one of the following: (1) Click the Color Fill button on the toolbar. (2) Right-click, and select "Color Fill " in the shortcut menu that is displayed.

*1: Selection Methods
- Click the desired module to make it active.
- Select multiple modules and comment boxes by box specification.
- Select menu command "Select All".
- Select multiple objects by clicking the mouse with the Ctrl key held down.

Objects selected by the above methods can be deselected by clicking elsewhere.

(6) Input terminals and registration area

This area is for registering registers to which values are read by the program.
Even if input terminals are not registered, it will not cause any problems in the program.
However, registering makes programs easier to follow.



Program Sheet	Default State
MAIN	Not registered
SUB@n	SARGn (n = 01 to 04) is displayed on 1st page.
SSUB@n	SSARGn (n = 01 to 04) is displayed on 1st page.
SUB@CSCPRn	SARG01 is displayed on 1st page.
SUB@SIMPR	Not registered

Item	Operation
Registering input terminals	Drag-and-drop register names from the Register window. Input terminals can be registered anywhere in the input terminal registration area.
Cutting input terminals	Select the input terminal to be cut (*1), and perform one of the following: (1) Select **Edit**>**Cut** from the menu. (2) Click the Cut button on the toolbar. (3) Right-click, and select “Cut” in the shortcut menu that is displayed.
Copying input terminals	Select the input terminal to be copied (*1), and perform one of the following: (1) Select **Edit**>**Copy** from the menu. (2) Click the Copy button on the toolbar. (3) Right-click, and select “Copy” in the shortcut menu that is displayed.
Pasting input terminals	Click the position to paste the input terminal at, and perform one of the following: (1) Select **Edit**>**Paste** from the menu. (2) Click the Paste button on the toolbar. (3) Right-click, and select “Paste” in the shortcut menu that is displayed.
Moving input terminals	Input terminals can be moved only inside the input terminal registration area in the same page. Move input terminals by one of the following methods: (1) To move only one input terminal: Click the input terminal to make it active, and drag-and-drop to move it. (2) To move multiple input terminals selected by box specification: Specify BOX for the desired modules and comment box to be moved, and drag-and-drop it to move. (3) Click the mouse with the Ctrl key held down to make multiple input terminals active. Drag-and-drop them.
Deleting input terminals	Select the input terminal to be deleted (*1), and perform one of the following: (1) Select **Edit**>**Delete** from the menu. (2) Right-click, and select “Delete” in the shortcut menu that is displayed. (3) Press the Delete key on the keyboard.

*1: Selection Methods
- Click the desired input terminal to make it active.
- Select multiple input terminals by box specification.
- Select menu command “Select All”.
- Click the mouse with the Ctrl key held down to select multiple input terminals.

Objects selected by the above methods can be deselected by clicking elsewhere.

(7) Output terminals and registration area

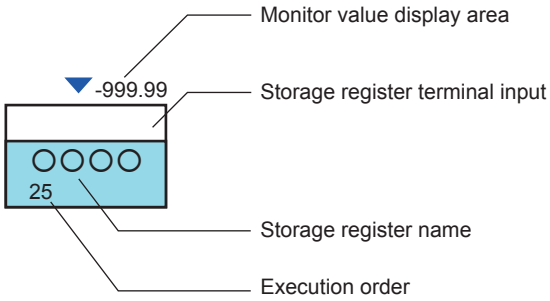
This area is for registering output terminals. There are three types of output terminals as follows:

(1)Storage register terminal
(2)Storage register terminal with enable switch (EN)
(3)SUB OUT terminal

Program Sheet	Default state
MAIN	Not registered
SUB@n	SUB OUT is displayed at 1st page.
SSUB@n	SUB OUT is displayed at 1st page.
SUB@CSCPRn	SUB OUT is displayed at 1st page.
SUB@SIMPR	Not registered

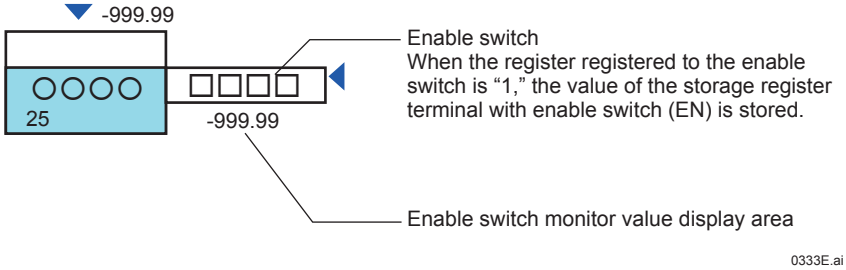
(1)Storage register terminal

This output terminal stores computation results and register values.



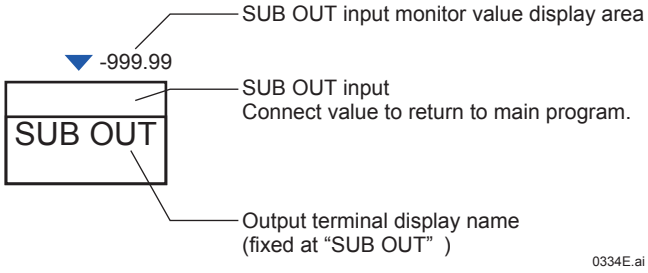
(2)Storage register terminal with enable switch (EN)

Storage register terminals can be changed to storage register terminal with enable switch (EN).



(3)SUB OUT terminal

The SUB OUT terminal is displayed at the 1st page of the SUB@n or SSUB@n program sheet. The SUB OUT terminal can be cut-and-pasted to other pages in the same program sheet.



Item	Operation
Registering storage register terminals	The display changes to “storage register terminal” by dragging-and-dropping a register name from the Register window. Registers can be registered anywhere in the output terminal registration area.
Registering storage register terminals with enable switch (EN)	After storing an output terminal as a storage register terminal, change it to a storage register terminal with enable switch (EN). To change this, right-click an already registered “storage register terminal,” and select “Change to Register Terminal with Enable Switch” in the shortcut menu that is displayed.
Changing to storage register terminals	Change a storage register terminal with enable switch (EN) to a storage register terminal. To change this, right-click an already registered “storage register terminal with enable switch (EN),” and select “Change to Storage Register Terminal” in the shortcut menu that is displayed.
Setting registers to output terminal inputs	Set by one of the following methods: (1) Drag-and-drop registers from the Register window. (2) Click between the wiring start and end points. When setting of registers is completed, the register names are displayed.
Setting an enable switch for EN storage register terminals	The procedure for changing is the same as setting registers to output terminal inputs (above).
Changing the execution order	Click **Edit**>**Change Execution Order** from the menu. See “3.1.4.”
Cutting output terminals	Select the output terminal to be cut (*1), and perform one of the following: (1) Select **Edit**>**Cut** from the menu. (2) Click the Cut button on the toolbar. (3) Right-click, and select “Cut” in the shortcut menu that is displayed.
Copying output terminals	Select the output terminal to be copied (*1), and perform one of the following: (1) Select **Edit**>**Copy** from the menu. (2) Click the Copy button on the toolbar. (3) Right-click, and select “Copy” in the shortcut menu that is displayed.
Pasting output terminals (*1)	Click the position to paste the output terminal at, and perform one of the following: (1) Select **Edit**>**Paste** from the menu. (2) Click the Paste button on the toolbar. (3) Right-click, and select “Paste” in the shortcut menu that is displayed. If you paste an output terminal after a cut, the execution order of the output terminals stays the same. if you paste an output terminal after a copy, the output terminal is added to the program, so the pasted output terminal is assigned following the final line of the execution order. The SUB OUT output terminal cannot be copied or pasted. The SUB OUT output terminal can be cut-and-pasted only to other pages in the same program sheet.
Moving output terminals	Output terminals can be moved only inside the output terminal registration area in the same page. Move output terminals by one of the following methods: (1) To move only one output terminal: Activate the output terminal, and drag-and-drop it to move. (2) To move multiple output terminals selected by box specification: Specify BOX for the desired modules and comment box to be moved, and drag-and-drop it to move.
Deleting output terminals	The SUB OUT terminal cannot be deleted. To delete other output terminals, select the output terminal to be deleted (*1), and perform one of the following: (1) Select **Edit**>**Delete** from the menu. (2) Right-click, and select “Delete” in the shortcut menu that is displayed. (3) Press the Delete key on the keyboard.

*1: Selection Methods
- Click the desired output terminal to make it active.
- Select multiple output terminals by box specification.
- Select menu command “Select All”.
- Click the mouse with the Ctrl key held down to select multiple output terminals.

Objects selected by the above methods can be deselected by clicking elsewhere.

Note: Storage register terminals and storage register terminals with enable switch (EN) cannot be registered to SUB@n and SSUB@n program sheets.

(8) Inter-page connection button (input)

Inter-page connection button (output) is automatically made when module output is placed to the module input or module parameter on another page.
Click this button to jump to the corresponding page, and activate the corresponding module.
During browsing the module output registers of other pages, the button is displayed on the left side of the page.
The program sheet name, number of pages, and module output register name are displayed on the button.

(9) Inter-page connection button (output)

Inter-page connection button (input) is automatically made when module output on another page is placed to the module input or module parameter.
Click this button to jump to the corresponding page, and activate the corresponding module.
If the module output register is being browsed on another page, the button is displayed on the right side of the page.
The program sheet name, number of pages, and module output register name are displayed on the button.
(In the case of storage register terminals and storage register terminals with enable switch (EN), the storage register name is displayed. In the case of SUB OUT input, SUB OUT is displayed.)

(10) Comment boxes

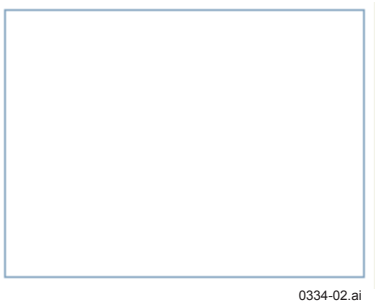
Comment boxes are for entering program comments.
Move the cursor to a comment box and click to enter comment text. Tabs cannot be entered in comment boxes.
The maximum number of characters that can be entered in comment boxes is 160 single-byte characters. The color of comment boxes can be changed, and up to 100 comment boxes can be registered.

Delete, Move, etc.	Operation
Switch to Input (Edit) mode	Switch by one of the following methods: (1) Double-click inside a comment box. (2) Click inside a comment box and press the F2 key. When the Edit mode is switched to, the cursor is displayed in the comment box.
Input (Edit)	Input by one of the following methods: (1) Input and delete from the keyboard. (2) Select the “Copy,” “Cut,” or “Paste” shortcut menu. (3) Click and select a comment, and enter an accepted character. This enables editing and the existing comment is overwritten. * To exit the Input (Edit) mode, select another object, or press the ESC key.
Cutting comment boxes	Select the comment box to be cut (*1), and perform one of the following: (1) Select **Edit**>**Cut** from the menu. (2) Click the Cut button on the toolbar. (3) Right-click, and select “Cut” in the shortcut menu that is displayed.
Copying comment boxes	Select the comment box to be copied (*1), and perform one of the following: (1) Select **Edit**>**Copy** from the menu. (2) Click the Copy button on the toolbar. (3) Right-click, and select “Copy” in the shortcut menu that is displayed.
Pasting comment boxes	Click the position to paste the comment box at, and perform one of the following: (1) Select **Edit**>**Paste** from the menu. (2) Click the Paste button on the toolbar. (3) Right-click, and select “Paste” in the shortcut menu that is displayed.
Move	Comment boxes can be moved inside the comment box registration area.
Deleting comment boxes	Select the comment box to be deleted (*1), and perform one of the following: (1) Select **Edit**>**Delete** from the menu. (2) Right-click, and select “Delete” in the shortcut menu that is displayed. (3) Press the Delete key on the keyboard.
Specifying the color of comment boxes	Select the comment box to be colored (*1), and perform one of the following: (1) Click the Color Fill button on the toolbar. (2) Right-click, and select “Color Fill “ in the shortcut menu that is displayed.

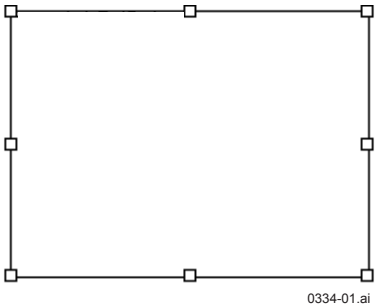
*1: Selection Methods
- Click the desired comment box to make it active.
- Select multiple modules and comment boxes by box specification.
- Select menu command “Select All”.
- Click the mouse with the Ctrl key held down to select multiple comment boxes.

Objects selected by the above methods can be deselected by clicking elsewhere.

Comment box



In the Edit mode



When comment box is selected

(11) Wiring

Wire between modules and between inputs and outputs. Objects can be wired only inside the same page.
The inter-page connection button is displayed for connecting between pages. Wiring cannot be moved by dragging-and-dropping. Registers can be re-wired by setting the registers again. When a jump is programmed in wiring, "■" is indicated. Wiring cannot be performed if there is no area for drawing the wiring.

Wiring Method	Explanation
Click wiring	Wiring can be performed by clicking between the start and end points. (Wiring is possible whichever of the start and end points you click first.) To cancel a start point (before clicking the end point), click the module registration area. Start points can also be wired with module outputs between different pages by clicking. However, module outputs of subprograms cannot be registered to other program sheets.
Wiring when dragging-and-dropping registers	**Register Drop Point** / **Wiring Destination** Input terminal: Wired automatically when the same register is located at the start point of the same page. Start point (*1): Wired automatically when the same register is located at the end point of the same page. *1: Registers cannot be dragged-and-dropped onto the inter-page connection button (output).

Note: Wiring with inter-page connection buttons is automatic.

Wiring Start and End Points

Wiring is performed between the start and end points in the following table. Wiring between two start points and two end points is not possible.

✓: Wiring possible −: Wiring not possible		End point (▼ mark: red color)		
		Input terminal	Module output	Inter-page connection button (input) (Note)
Start point (▼ mark: blue color)	Module input	✓	✓	✓
	Module parameter	✓	✓	✓
	Input of storage register terminal	✓	✓	✓
	Input of EN storage register terminal	✓	✓	✓
	Enable switch of EN storage register terminal	✓	✓	✓
	Input of SUB OUT	✓	✓	✓
	Inter-page connection button (output)	−	✓ (Note)	−

Note: Cannot be wired.

Click Wiring Enhanced Display

Right-click the following objects, and select “Display Wire Enhancement” on the shortcut menu with a check mark to enhanced-display the wiring currently connected to the corresponding object (enhanced in red):

- Module
- Input terminal
- Storage register terminal
- Storage register terminal with enable switch (EN)
- SUB OUT terminal

To cancel the enhanced display, click “Cancel Display Wire Enhancement” on the shortcut menu.

(12) Total number of modules display

Active Program Sheet	Display Total Number
Other than SUB@SIMPR	Limit values and total number of modules and output terminals currently registered to all pages other than SUB@SIMPR are displayed. (except the SUBOUT terminal) Example) When 50 modules are registered, this is displayed as “50/400”.
SUB@SIMPR	Limit values and total number of modules and output terminals currently registered to all pages in SUB@SIMPR are displayed. (except the SUBOUT terminal) Example) When 5 modules are registered, this is displayed as “5/10”.

(13) Number of comment boxes display

Limit values and total number of comment boxes currently registered to all pages are displayed.
Example) When 5 comment boxes are registered, this is displayed as “5/100”.

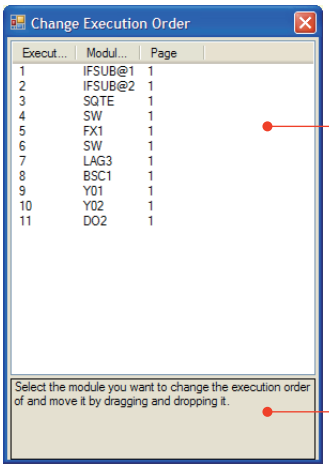
(14) Load factor and max. load factor value

This is displayed only when the module monitor is activated. See “2.10.5.”

3.2.4 Changing the Program Execution Sequence

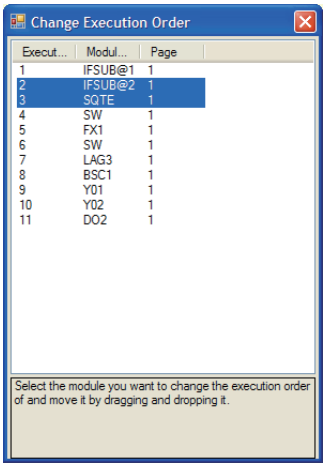
Operation

1. With the Programming window active, click **Edit>Change Execution Order** from the menu. The execution order of the module storage register terminals and the storage register terminals with enable switch (EN) of the active program sheet is displayed in the Change Execution Order window.



0335E.ai

2. Click the line of the program whose execution order is to be changed. To change the execution order of multiple lines, click the desired lines with the SHIFT key held down. For example, to move the 2nd and 3rd lines to before the 1st line, click the 2nd line and the 3rd line with the SHIFT key held down.
To cancel a selected line(s), click any other line.



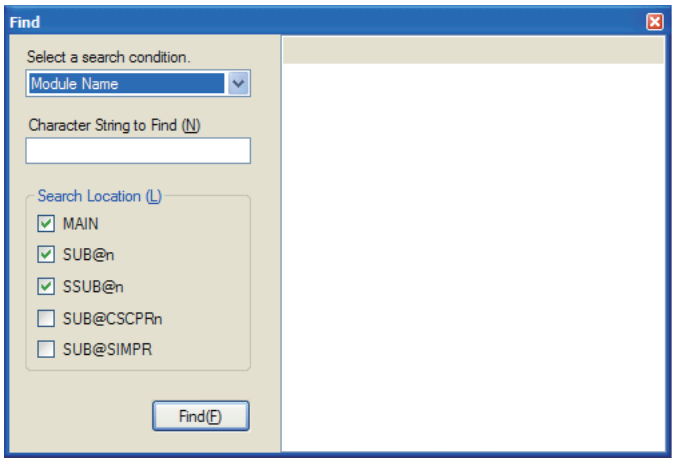
0336E.ai

3. Drag the selected line and drop it at the move destination. The selected line will be inserted before the line that you dropped it at.

3.2.5 Searching in Programs

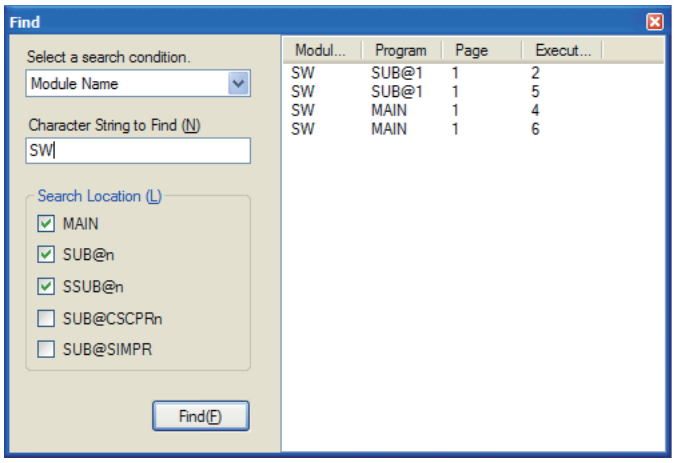
Operation

1. With the Programming window active, click **Edit**>**Find** from the menu.



0337E.ai

2. Select the search conditions, enter the text string to be **searched**, select the search range, and click **Find**. Entry of the text string to be searched is not case-sensitive.



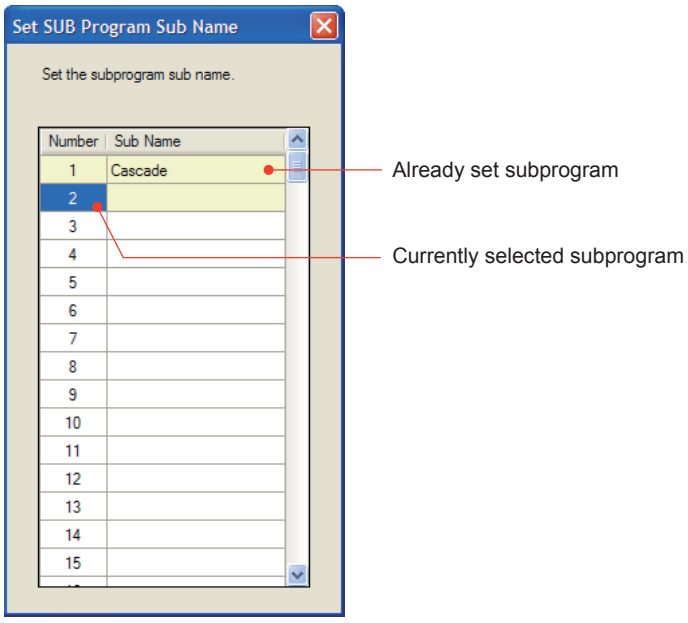
0338E.ai

3. Click a line of the search result to jump to the corresponding page, and activate the corresponding object.

3.2.6 Setting the Sub-names of Subprogram Modules and Nesting Subprogram Modules

Operation

1. Select the subprogram module to be set with a sub-name, and click either **Edit>Set SUBProgram Sub Name** or **Edit>Set SSUB Program Sub Name** from the menu. The figure below shows an example of setting the SUB program sub-name.



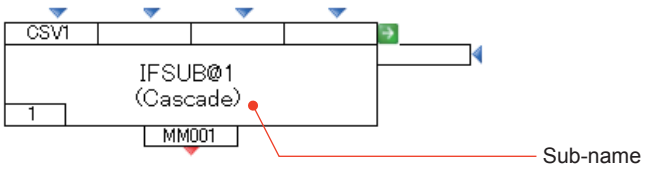
0339E.ai

2. Enter the sub-name.

Description

Subprogram modules can be given a name that summarizes their function as their sub-name.
Accepted input characters: alphanumerics and underbar (max. 10 characters)
The sub-name is displayed at the lower half of (i.e. under) the module symbol of the subprogram module, on the program sheet tab, and at the subprogram folder of the File window.

Example: Enter “Cascade” as the sub-name.

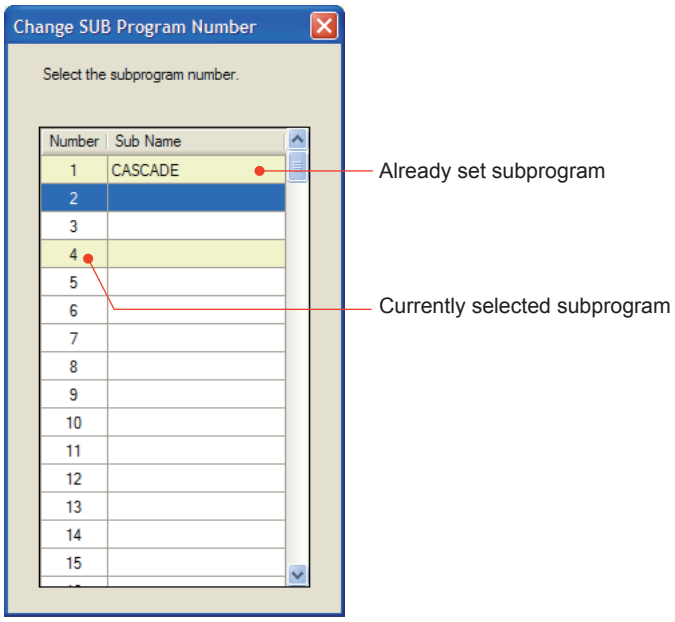


0340E.ai

3.2.7 Changing the Numbers of Subprogram Modules and Nesting Subprogram Modules

Operation

1. Select the subprogram module whose subprogram number is to be changed, and click either **Edit>Change SUB Program Number** or **Edit>Change SSUB Program Number** from the menu. The figure below shows an example of changing the SUB program number.

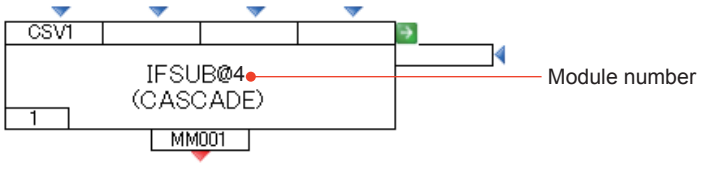


2. Click the cell of the number to be changed, and click at the top right of the window.

Description

To display the window, right-click an already registered module (SUB@n, IFSUB@n, GTSUB@n, SSUB@n, IFSUB@n, or GTSSUB@n), and click “Change Subprogram Number” from the shortcut menu.

Example: Change the subprogram number to “4.”



3.2.8 Splitting Windows

The Create Program can be split horizontally or vertically. The following menu commands are available for splitting windows:

- **Window>Remove Split**
- **Window>Split Horizontal**
- **Window>Split Vertical**

3.2.9 Enlarging/reducing Windows

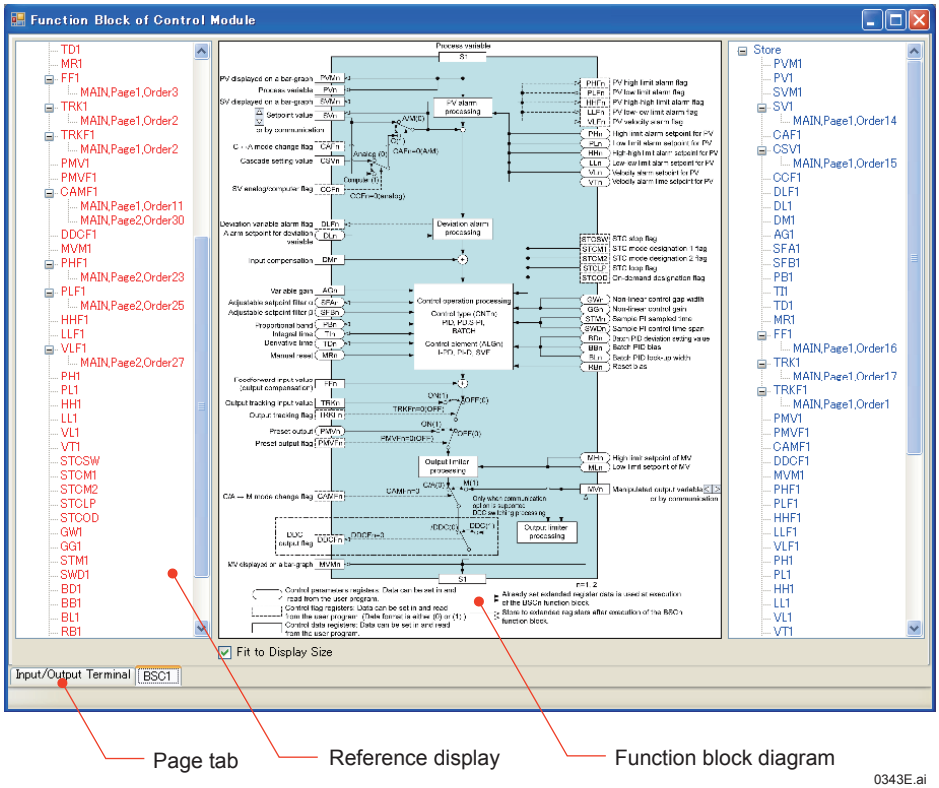
You can zoom in on or zoom out (changing the display enlargement ratio) of the Programming window. Zoom ratio: 50 to 100% in 10% increments to 50%, 60%, 70%, 80%, 90% and 100%.

Perform one of the following, to change zoom ratio:

- Select **View>ZOOM** from the menu to specify the zoom ratio.
- Click ZOOM on the toolbar to specify the zoom ratio.

3.2.10 Displaying the Links of Used Registers

The function block diagram of the control function selected in the Select Control Function window is displayed. Links are displayed in this window when the registers of the function block are used by the program.



Item	Operation
Window display method	(1) When the Function Block Programming window is open, select the check mark at **View>Function Block Window of Control Module** from the menu. (2) Click the icon on the toolbar. (3) Click the control module's jump button.
Closing the window	(1) Cancel the check mark at **View>Function Block Window of Control Module** from the menu. (2) Click the Close button on the title bar.
Page tab	Click the Page tab to switch between windows.
Function block diagram, input/output terminal diagram	These items are only displayed.
Browse view Storage view	Double-click these items to jump to the corresponding page of the corresponding program, and display the corresponding object as active. The register information is displayed when you bring the mouse pointer close to these items.

Item	Specification
Browse view	When the registers of the function block are registered to the following objects, the registers and the browse view content are displayed on the left side of the figure: • Module input • Module parameters • Storage register terminal inputs • Storage register terminal with enable switch (EN) input • Enable switch of storage register terminal with enable switch (EN) • Input of SUB OUT
Browse view content	Program sheet name, page number, module execution order Example: MAIN, Page1, Order1 SUB OUT is displayed as SUB OUT in the module execution order as it has no execution order.
Storage view	When the registers of the function block are registered to the following objects, the registers and the browse view content are displayed on the right side of the figure: • Storage register terminal input • Storage register terminal with enable switch (EN) input
Storage view content	Program sheet name, page number, module execution order Example: MAIN, Page1, Order1

3.2.11 Re-creating Programs

Operation

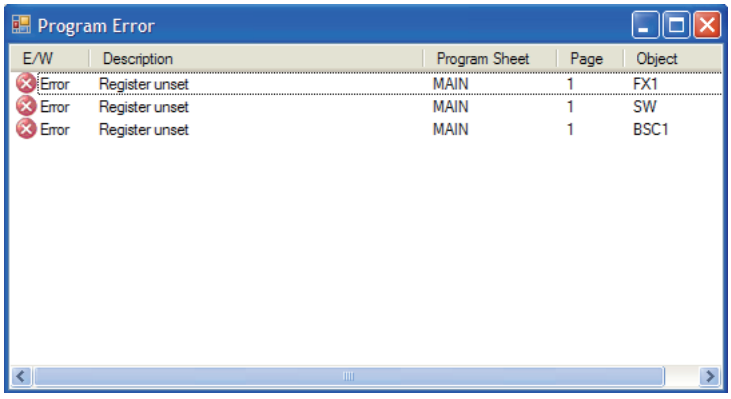
1. Click **File**>**New Program** from the menu.

Description

Perform this operation to re-create only a program. Previous parameters and other data remain.

3.2.12 Program Errors

The Program Error window as shown below is displayed if an error occurs in the program when creating a program and closing the Programming window.



0344E.ai

Errors and warnings are displayed in the Program Error window.
Errors must be corrected before programs can be downloaded to the YS1700.
When an error occurs, double click the line of the error in the Program Error window to jump to the location of the program where the error occurred, and the display is activated.

To remove the status warnings from the program check, select **Tool**>**Option**>**Warning Exception** and add a check. When a check is added to this item, warnings are also excluded from the program checks performed automatically during saving and downloading.

Status	Error Message	Cause
Error	The control function selection and control module are incompatible.	The relationship between the control function selected in the Select Control Function window and the control module are not feasible.
	Register unset	Registers are not set to the following registered objects: • Module inputs (Note 1) • Module parameters • Storage register terminal inputs • EN storage register terminal inputs • EN storage register terminal enable switches
	Subprogram undefined	The subprogram sheet called by the subprogram module does not exist.
	PGM module registration error	Only one of the PGM modules used as a pair exists.
	PGM module execution order error	The execution order of the PGM modules is incorrect.
	SUB OUT error	Two or more SUB OUT output terminals exist on the same program sheet.

(Note 1): An error will not occur even if input of the subprogram module is not set.

Status	Error Message	Cause
Warning	Duplicate storage register terminals	Storage register terminals or EN storage register terminals for the same register exist in a single program.
	The maximum number of times of use is exceeded.	Two or more of same dynamic operation modules are registered to the same program. However, the following dynamic operation modules can be registered multiple times: 10-segment linearizer function (FXm), inverse conversion of 10-segment linearizer function (IFXm), arbitrary segment linearizer function (GXm), inverse conversion of arbitrary segment linearizer function (IGXm)
	The maximum number of times of use is exceeded.	The same control modules are registered twice or more.
	Output terminal registration to SUB and SSUB program sheets	Terminals other than SUB OUT are registered to the output terminal registration area of the SUB and SSUB program sheets.
	The maximum number of errors or alarms is exceeded.	Error display appears in priority to the program error window when this error occurs with 5 or more of errors or alarms.
	Register unset	Register is not set to the SUB OUT input.

3.2.13 Saving Programs as Components

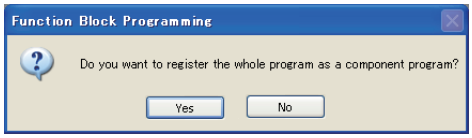
A created program can be reused in another user program.

(1) Saving as a Component

(1-1) Saving a whole program as a component

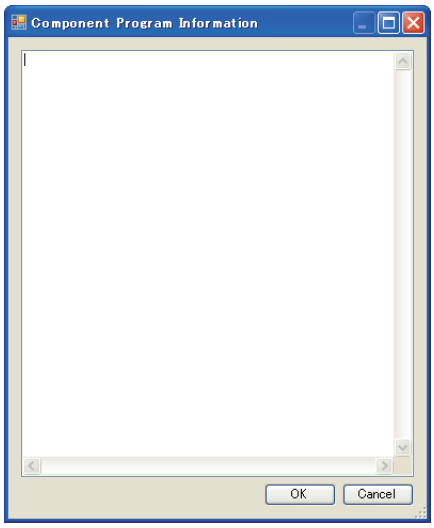
Operation

1. Click **File**>**Save Program as Component**>**Whole program** when the programming window is open to display the save as component confirmation message.



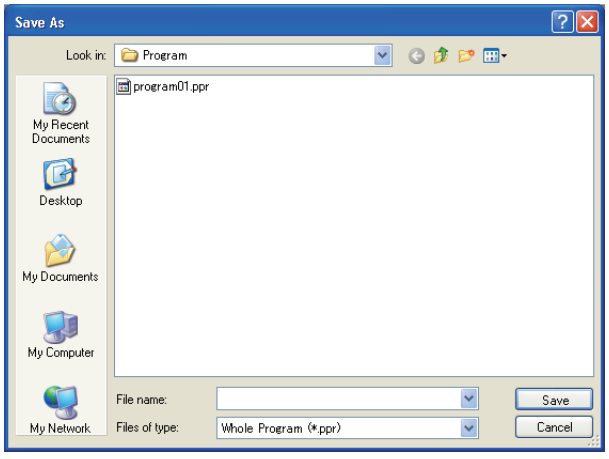
0344-01E.ai

2. Click **Yes** to display the Component Program Information window. Clicking **No** ends the process.
If you include an overview of the component program, it will be helpful when you want to reuse the component program in another user program.



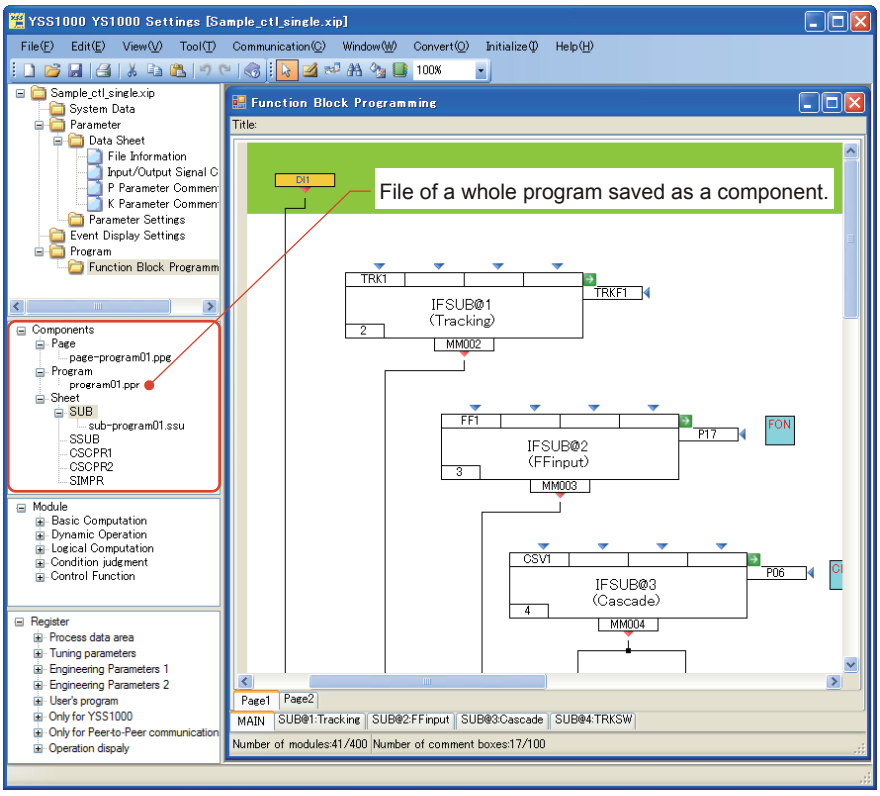
0344-02E.ai

3. Enter a comment for the component program in the Component Program Information window, and click **OK** to display the Save As window.



0344-03E.ai

4. Enter a file name, and click **Save**.



0344-04E.ai

Description

When you save a whole program as a component, only the program data can be saved as a file and reused. Data such as parameter data and event data is not included.

The saved component file is stored in the folder shown in the table below. (Factory default)

Program	Extension	Storage Folder	
		Windows 2000/XP	Windows Vista
Whole program (main program)	.ppr	C:\Program Files\YSS1000\Components\Program\	C:\Users\<UserName>\Documents\YSS1000\Components\Program\

► Changing the path: "2.11.10 Setting Path for Saving Files"

You can enter up to 5,000 single-byte characters for the component program information.

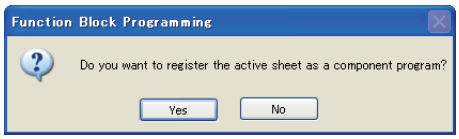
Editing and deleting of Component Files

Edit the component files in the Programming window after using the component files. Delete the component files by file manipulation of PC (Explorer).

(1-2) Saving a sub-program sheet as a component

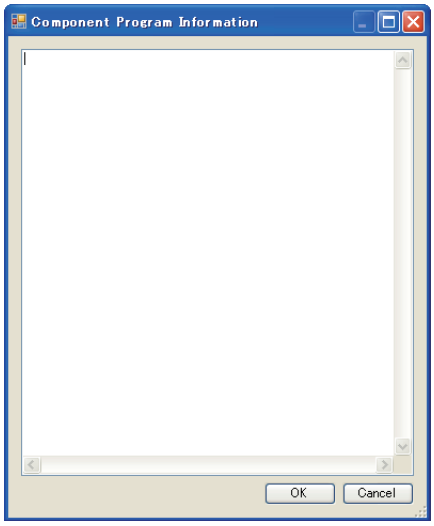
Operation

1. Make the sub-program sheet you want to save as a component active and click **File>Save Program as Component>Program Sheet** to display the save as component confirmation message.



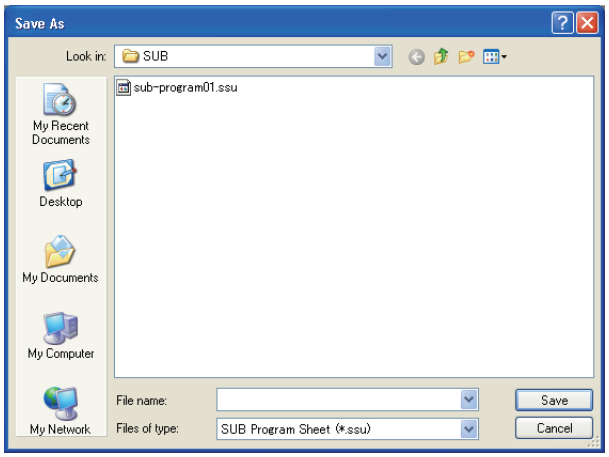
0344-05E.ai

2. Click **Yes** to display the Component Program Information window. Clicking **No** ends the process.
If you include an overview of the component program, it will be helpful when you want to reuse the component program in another user program.



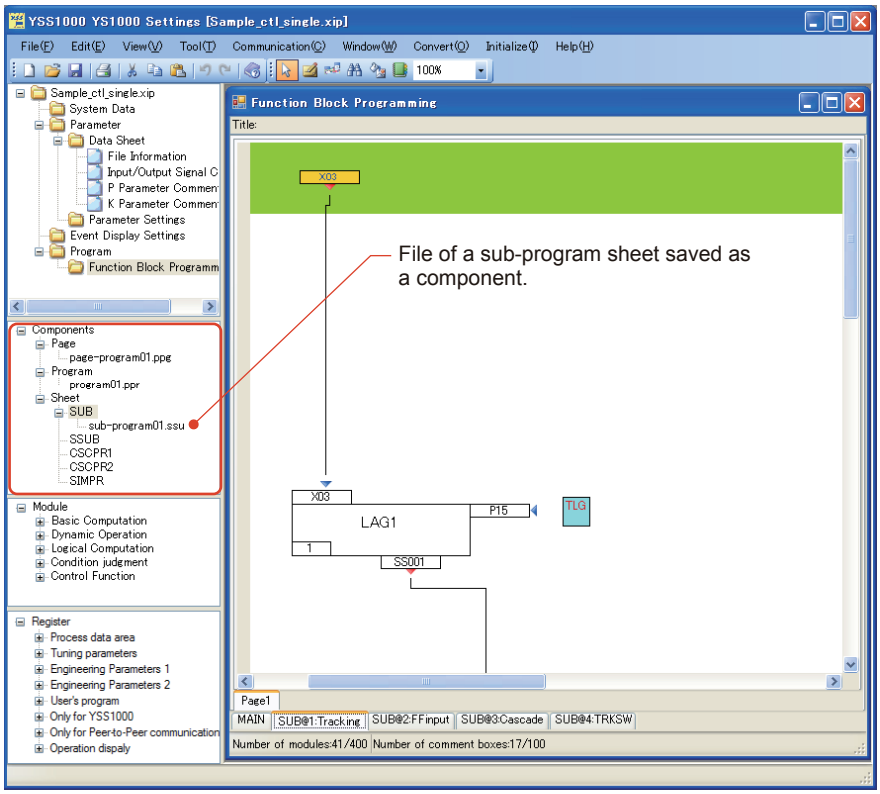
0344-06E.ai

3. Enter a comment for the component program in the Component Program Information window, and click **OK** to display the Save As window.



0344-07E.ai

4. Enter a file name, and click **Save**.



0344-08E.ai

Description

The active sub-program sheet can be saved as a component. To save the main program sheet as a component, save the whole program as a component. See “(1) Saving a whole program as a component.”
If an SSUB module, IFSSUB module, and GTSSUB@n module are registered to the program sheet, the calling sub-program sheets are also saved together as a component.

Note

If module output of the main program sheet is registered to the register registration section (example: module input, etc.) of the sub-program sheet to be saved as a component, there will no longer be a connection.

The saved component file is stored in the folder shown in the table below. (Factory default)

Sheet	Extension	Storage Folder	
		Windows 2000/XP	Windows Vista
SUB program	.ssu	C:\Program Files\YSS1000\Components\Sheet\SUB\	C:\Users\<UserName>\Documents\YSS1000\Components\Sheet\SUB\
SSUB program	.sss	C:\Program Files\YSS1000\Components\Sheet\SSUB\	C:\Users\<UserName>\Documents\YSS1000\Components\Sheet\SSUB\
CSCPR1 program	.1sc	C:\Program Files\YSS1000\Components\Sheet\CSCPR1\	C:\Users\<UserName>\Documents\YSS1000\Components\Sheet\CSCPR1\
CSCPR2 program	.2sc	C:\Program Files\YSS1000\Components\Sheet\CSCPR2\	C:\Users\<UserName>\Documents\YSS1000\Components\Sheet\CSCPR2\
SIMPR program	.ssi	C:\Program Files\YSS1000\Components\Sheet\SIMPR\	C:\Users\<UserName>\Documents\YSS1000\Components\Sheet\SIMPR\

► Changing the path: "2.11.10 Setting Path for Saving Files"

You can enter up to 5,000 single-byte characters for the component program information.

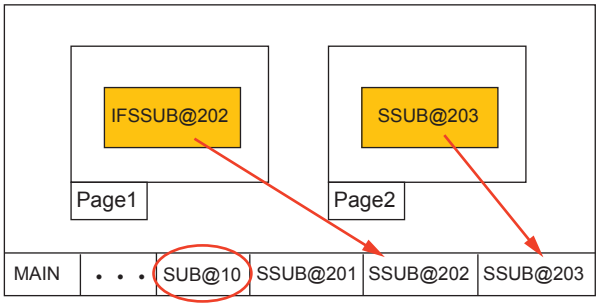
Editing and deleting of Component Files

Edit the component files in the Programming window after using the component files.
Delete the component files by file manipulation of PC (Explorer).

Updating Programs Automatically

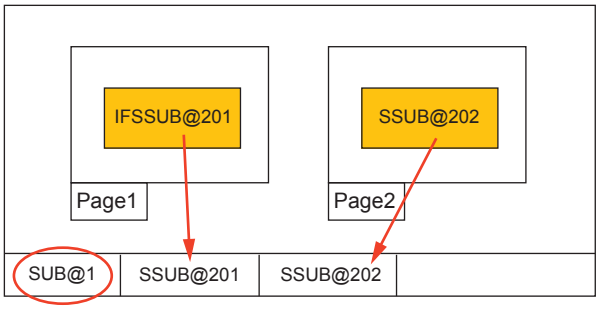
Example: Saving a sub-program sheet (SUB@10) including nesting sub-program sheets (SSUB@202 and SSUB@203) as a component.

Sub-program sheets to save as a component



0344-09E.ai

Sub-program sheets saved as a component



0344-10E.ai

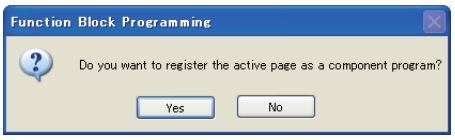
Nesting sub-program sheets called in the SUB@10 sheet are saved together as a component.
The SUB@10 sheet to be saved as a component is automatically revised to SUB@1.
The nesting sub-program sheets SSUB@202 and SSUB@203 are automatically revised to SSUB@201 and SSUB@202, respectively. (The sub-program sheet and nesting sub-program sheets are automatically revised in order starting from the first program number.)

The SUB program sub-name and SSUB program sub-name are saved under the same name as a component.

(1-3) Saving a main program page as a component

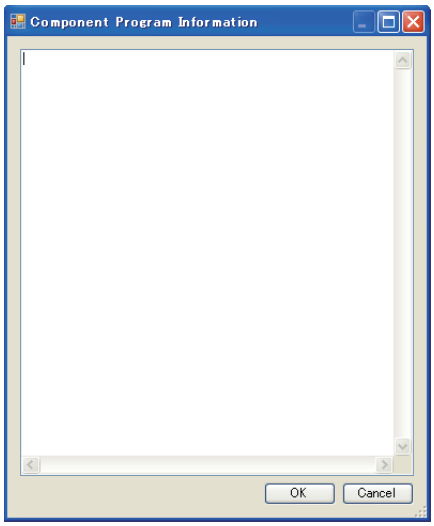
Operation

1. Make the main program page you want to save as a component active and click **File**>**Save Program as Component**>**Page** to display the save as component confirmation message.



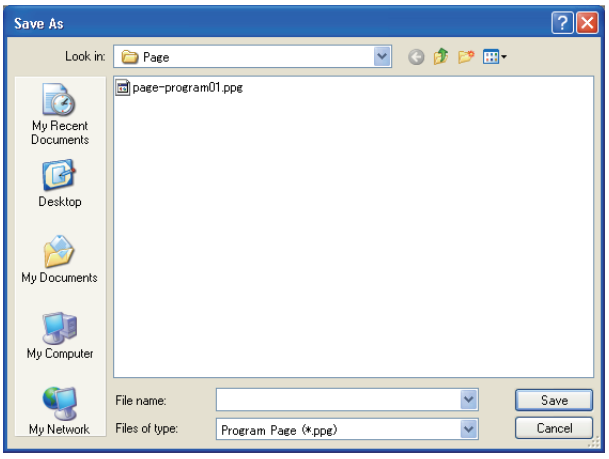
0344-11E.ai

2. Click **Yes** to display the Component Program Information window. Clicking **No** ends the process.
If you include an overview of the component program, it will be helpful when you want to reuse the component program in another user program.



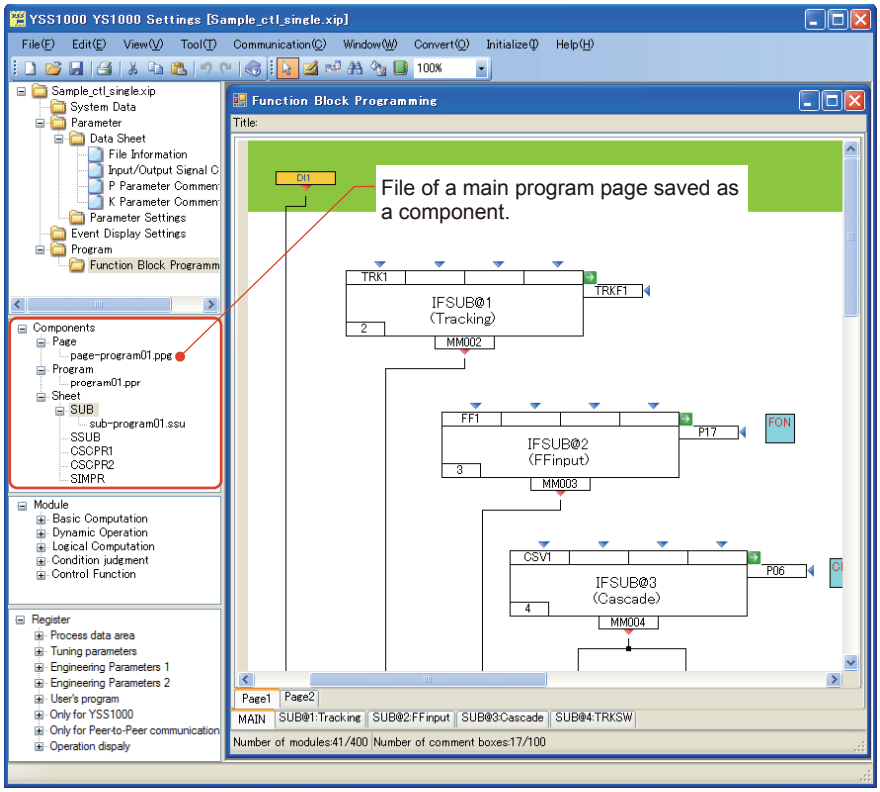
0344-12E.ai

3. Enter a comment for the component program in the Component Program Information window, and click **OK** to display the Save As window.



0344-13E.ai

4. Enter a file name, and click **Save**.



0344-14E.ai

Description

It is only possible to save main program pages individually as components. The active main program page can be saved as a component.
If an SUB module, IFSUB module, GTSUB@n module, SSUB module, IFSSUB module, and GTSSUB@n module are registered to the program page, the calling sub-program sheets are also saved together as a component.

Note

If module output of the main program sheet is registered to the register registration section (example: module input, etc.) of the sub-program sheet saved together as a component, there will no longer be a connection.

Note

If you want to use module output of the main program page to be saved as a component with the sub-program sheets saved together, register the module output of the main program page to be saved to the input of the sub-module, and enable it to be passed to the sub-program as an argument.

The saved component file is stored in the folder shown in the table below. (Factory default)

Page	Extension	Storage Folder	
		Windows 2000/XP	Windows Vista
Main program page	.ppg	C:\Program Files\YSS1000\Components\Page\	C:\Users\<UserName>\Documents\YSS1000\Components\Program\

► Changing the path: "2.11.10 Setting Path for Saving Files"

You can enter up to 5,000 single-byte characters for the component program information.

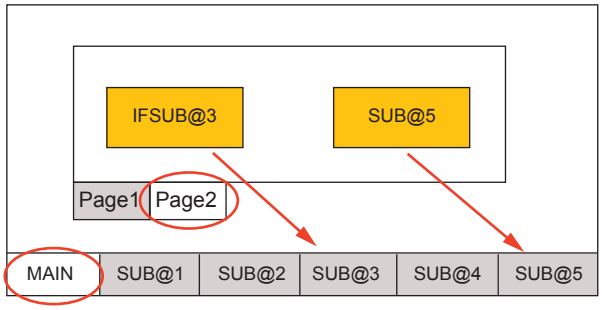
Editing and deleting of Component Files

Edit the component files in the Programming window after using the component files. Delete the component files by file manipulation of PC (Explorer).

Updating Programs Automatically

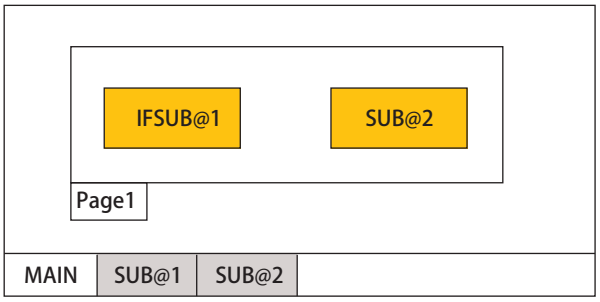
Example: Saving a page (Page 2) of a main program sheet including sub-program sheets (IFSUB@3 and SUB@5) as a component.

Program page to save as a component



0344-15E.ai

Program page saved as a component



0344-16.ai

Sub-program sheets called in the main program page are saved together as a component. The sub-program sheets IFSUB@3 and SUB@5 to be saved as a component are automatically revised to IFSUB@1 and SUB@2, respectively. The nesting sub-program sheets are also automatically revised in the same way. (The sub-program sheets and nesting sub-program sheets are automatically revised in order starting from the first number.)

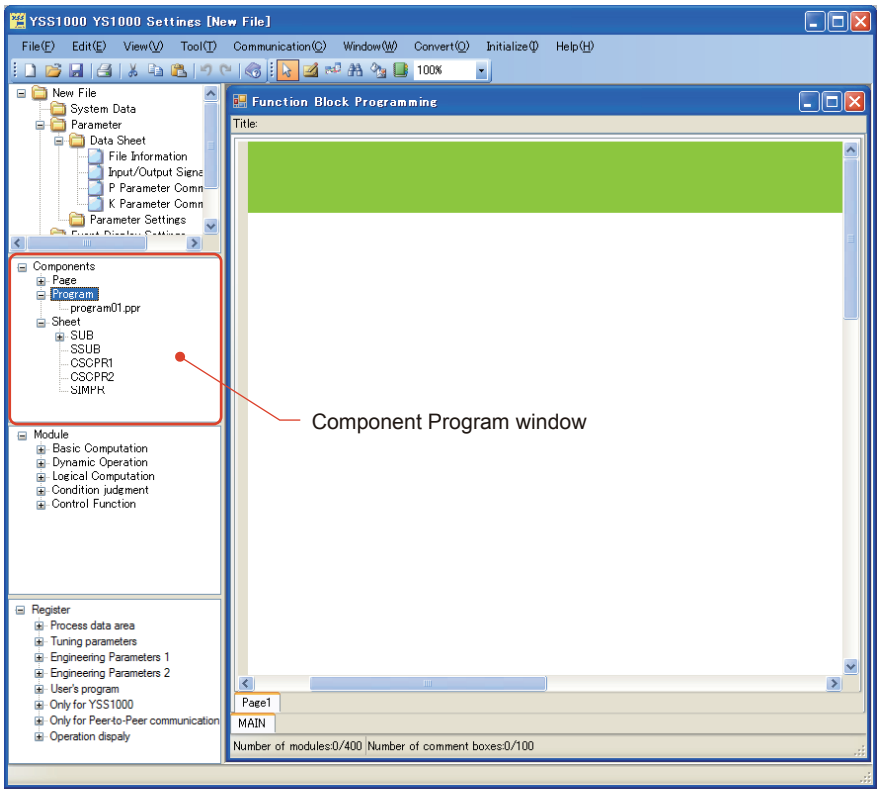
The SUB program sub-name and SSUB program sub-name are left as is. The module output of the program page saved as a component is automatically revised from MM001. If the module output register of another page is registered, it is revised so that it is no longer registered.

(2) Using a Program Saved as a Component

The following operation example describes the procedure for using a program saved as a component in another user file. A sub-program sheet and a main program page can be used in the same way.

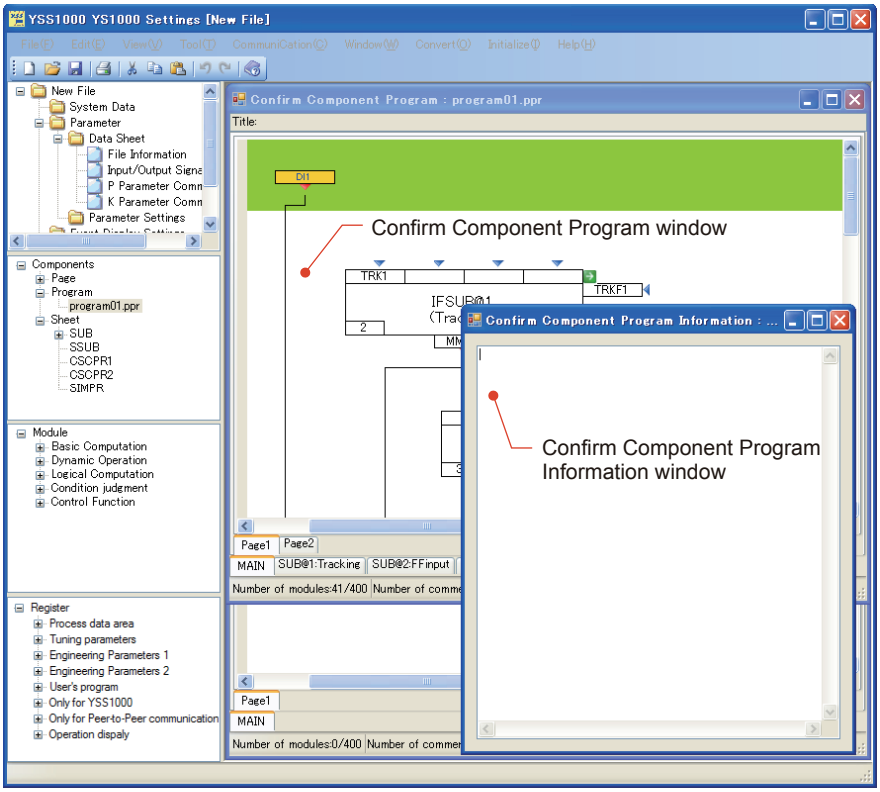
Operation

1. Click **View**>**Component Program Window** and add a check when the programming window is open to display the Component Program window.



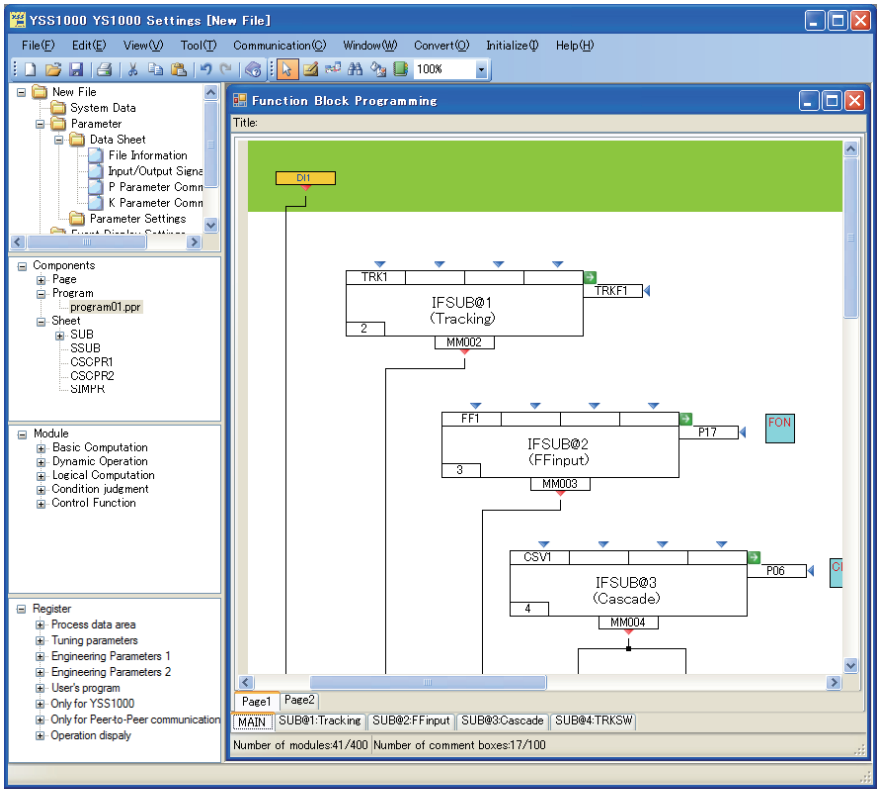
0344-30E.ai

2. Click the component file you want to use to display the Confirm Component Program window and Confirm Component Program Information window.



0344-31E.ai

3. Confirm the program to be used and click of the Confirm Component Program window to display the confirmation message, and then click **Yes** to use the component program in the programming window. Click **No** to not use the component program.



0344-32E.ai

Description

Main Program Sheet (Whole Program)

Save any user program you are working on in the programming window if needed, because it will be deleted.

Sub-program Sheets

If you use component programs, they are registered in order starting from the largest existing sub-program number plus 1. However, if SUB@200 or SSUB@256 exists and the sub-program numbers to be added exceed SUB@200 or SSUB@256, you cannot use the component programs. In such a case, change the numbers of the sub-program numbers, and confirm the empty sheets up to SUB@200 and SSUB@256 before use. If there are programs in cascade (CSCPR1 and CSCPR2) or a simulation program (SIMPR), the process stops without them being overwritten.

Main Program Pages

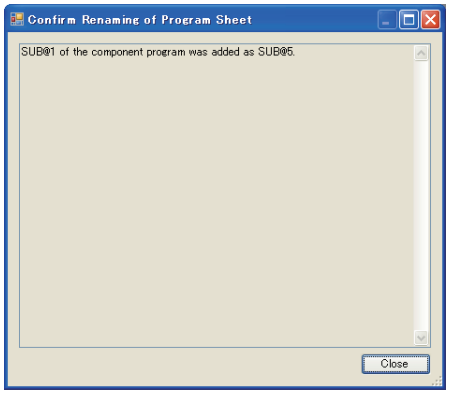
If you use component programs, they are registered in order starting from the largest existing program page number plus 1. However, if there are 256 pages in the main program and the main program pages to be added exceed the 256th page, the component programs cannot be used. Confirm the empty pages of the main program before use.

Editing Component Program Information

The content displayed in the Confirm Component Program Information window cannot be edited. If you want to edit the component program information, right-click the component file in the Component Program window and select Edit Component Program Information from the shortcut menu to display the Component Program Information window so that you can edit the content.

Confirm Renaming of Program Sheet Window

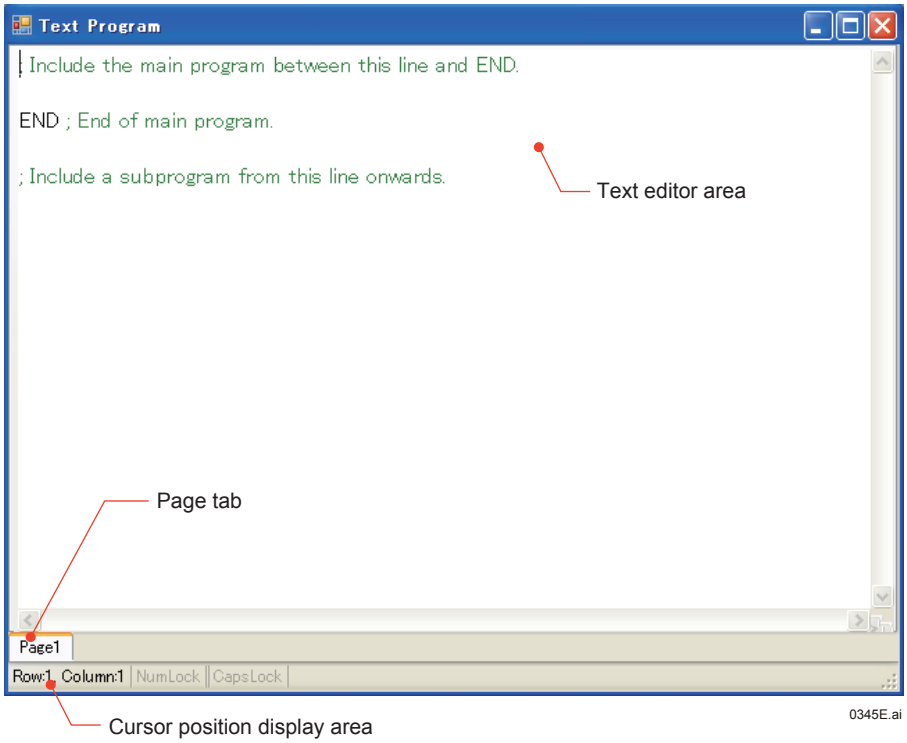
The Confirm Renaming of Program Sheet window below is displayed when a sub-program sheet or program page (when a sub-program is included) is used.
The content of change is described in PartsUselog.txt of the folder in the directory: \Temp\ (see "2.11.10 Setting Path for Saving Files").



0344-31-1E.ai

3.3 Text Programming

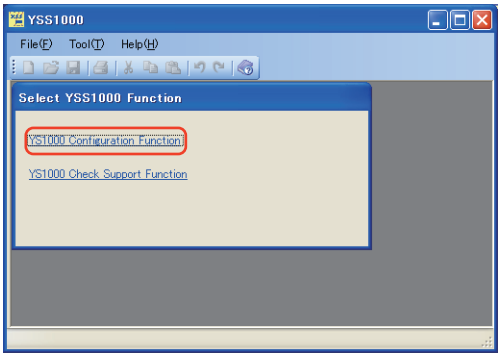
3.3.1 Names of Parts in the Text Programming Window



3.3.2 Programming in Text

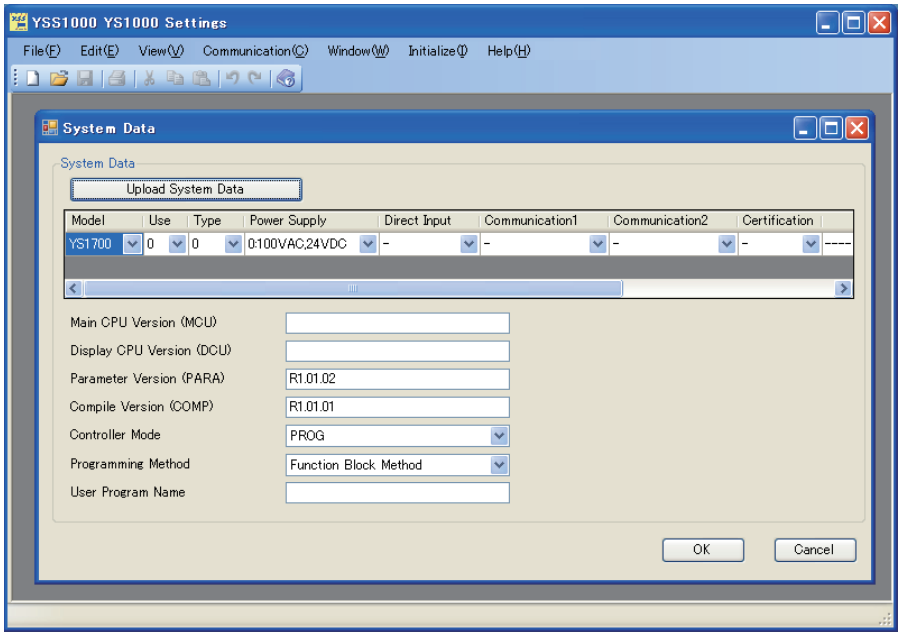
Operation

1. Start up YSS1000, and click "**YS1000 Configuration Function**" in the Select YSS1000 Function window.



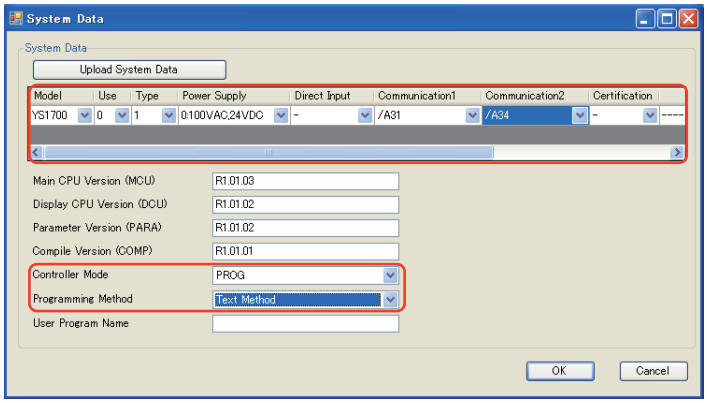
0346E.ai

2. Click **File**>**New** from the menu in the Basic window or on the toolbar to display the System Data window.



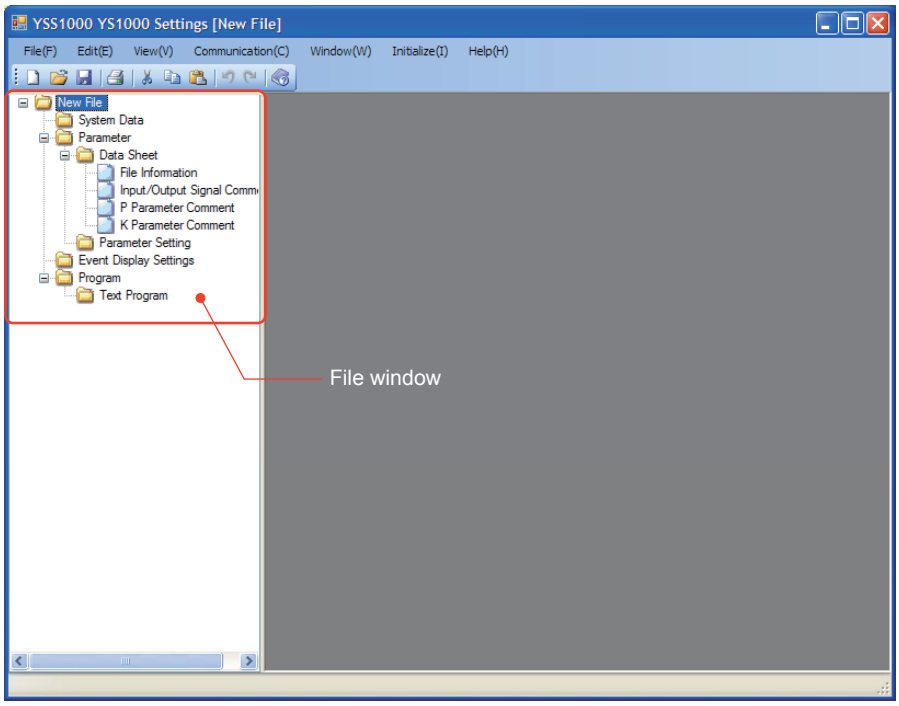
0347E.ai

3. Select model name "YS1700," suffix code, and optional code to be set, and set the controller mode and the programming method respectively to "PROG" (programmable mode) and "Text Method," and click **OK**.



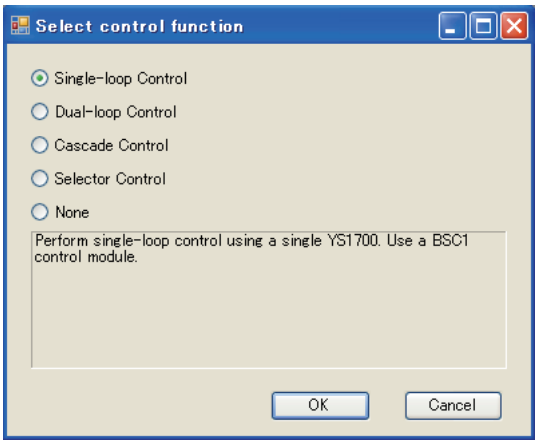
0348E.ai

4. The File window is displayed on the Basic window.



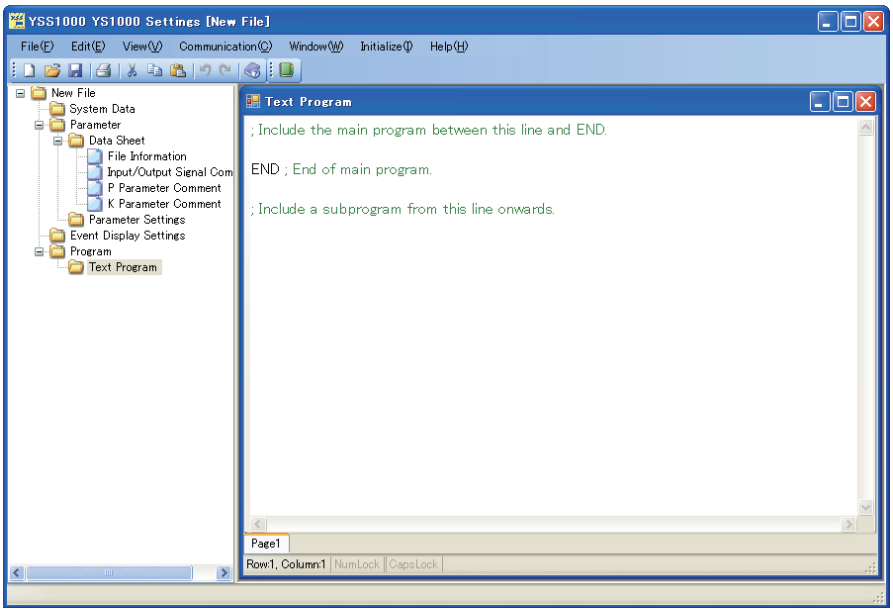
0349E.ai

5. Click **Text Program** in the File window to display the Select Control Function window.



0350E.ai

6. Select the control function, and click **OK** to display the Text Programming window.



0351E.ai

7. Start programming.

Before starting programming, make sure that you have thoroughly read and understood the operations described in Chapters 4 to 12.

8. After completing programming, perform a program check by selecting **Tool>Program Check** when the Programming window is open. If a compile error is generated, a review of the user programs is required.

9. After the data is saved to file, download it to YS1000 and then perform an operation check.

3.3.3 Explanation of Operations in the Text Programming Window

(1) Page tab

<Inserting new page breaks>

(1)Place the cursor at the position to insert the page break, right-click, and execute "Page Break" from the shortcut menu.

(2)Place the cursor at the position to insert the page break, and click **Edit**>**Page Break** from the menu.

Recommended maximum number of pages: 50 pages

<Canceling inserted page breaks>

Press the Delete key at the last line of the page where the page break is inserted.

<Switching pages>

(1)Click the Page tab.

(2)Keyboard operation

Ctrl+PageUP: Displays the previous page. The last page is displayed when the current page is the first page of the program.

Ctrl+PageDOWN: Displays the next page. The first page is displayed when the current page is the last page of the program.

(2) Cursor position display area

The line and row of the cursor position are displayed.

(3) Text editor area

Accepted input characters:

- All alphanumerics and symbols that can be entered on the keyboard
- Tabs
- Spaces

Max. Number of Lines and Rows	Maximum Value
Max. number of lines	2,000
Max. number of rows (1 character per row)	80

3.3.4 Auto-conversion of Entered Characters

The color of entered characters is automatically changed as follows to distinguish program text and comments, and prevent input errors.

Character	Color
Valid computation instructions	Black
Label name, subprogram name (From @ up to spaces, from @ up to page breaks, from @ up to ";" (semi-colons))	Blue
Comments From ";" (semi-colon) up to an entered page break	Green
Other than above Invalid computation instructions, etc.	Orange

Supplementary Explanation: Instructions that contain numbers are valid up to the 3rd digit regardless of the number of digits of the number range.
Example: LD X1, LD X01, LD X001

Auto-conversion of Lowercase Characters

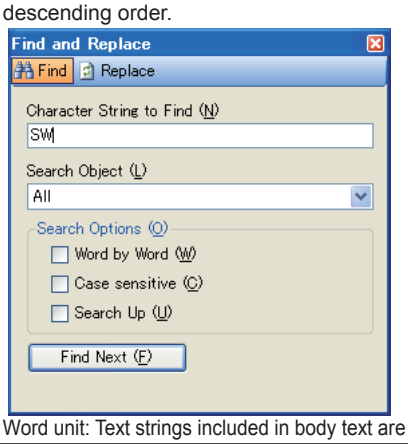
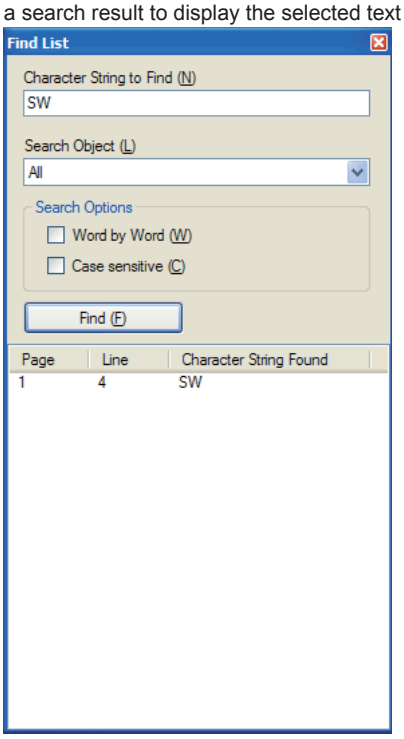
Lowercase characters are converted to uppercase characters if they are entered for valid computation instructions.

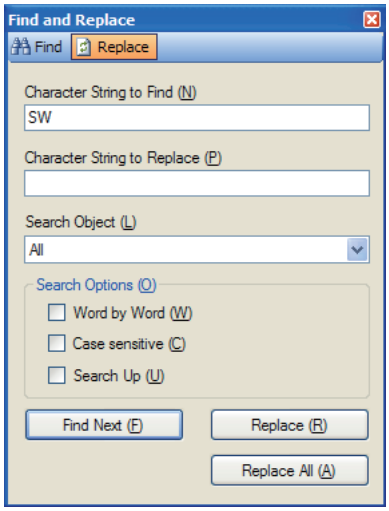
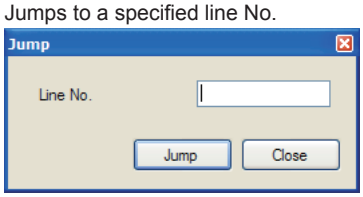
Example: When "s" is entered followed by "t" (both lowercase characters), they are automatically converted to "ST" (uppercase).
However, this function is not applicable to characters in comments, label names and subprogram names.

3.3.5 List of Text Program Editing Functions

Text program editing functions can be executed from the menu in the Text Programming window.

✓: Available, –: Not available

Editing Function	Menu Command	Shortcut Menu (right-click to display)	Remarks
Undo	✓	–	Max. 10 times
Redo	✓	–	Max. 10 times
Cut	✓	✓	–
Copy	✓	✓	–
Paste	✓	✓	Pastes the content of the Clipboard copied or cut by other Text Editors.
Delete	✓	✓	Deletes are possible by the Delete key on the keyboard.
Select All	✓	–	
Find	✓	–	Searches are executed in two directions, ascending order and descending order.  Word unit: Text strings included in body text are not searched for.
Find List	✓	–	Displays a list of text strings found in the search. Double-click a search result to display the selected text string as active.  0353E.ai Word unit: Text strings included in body text are not searched for.

Editing Function	Menu Command	Shortcut Menu (right-click to display)	Remarks
Replace	✓	–	There are two types of replace, Find and Replace and Replace All.  0354E.ai
Jump	✓	–	Jumps to a specified line No.  0355E.ai
Page Break	✓	✓	See "3.3.3."

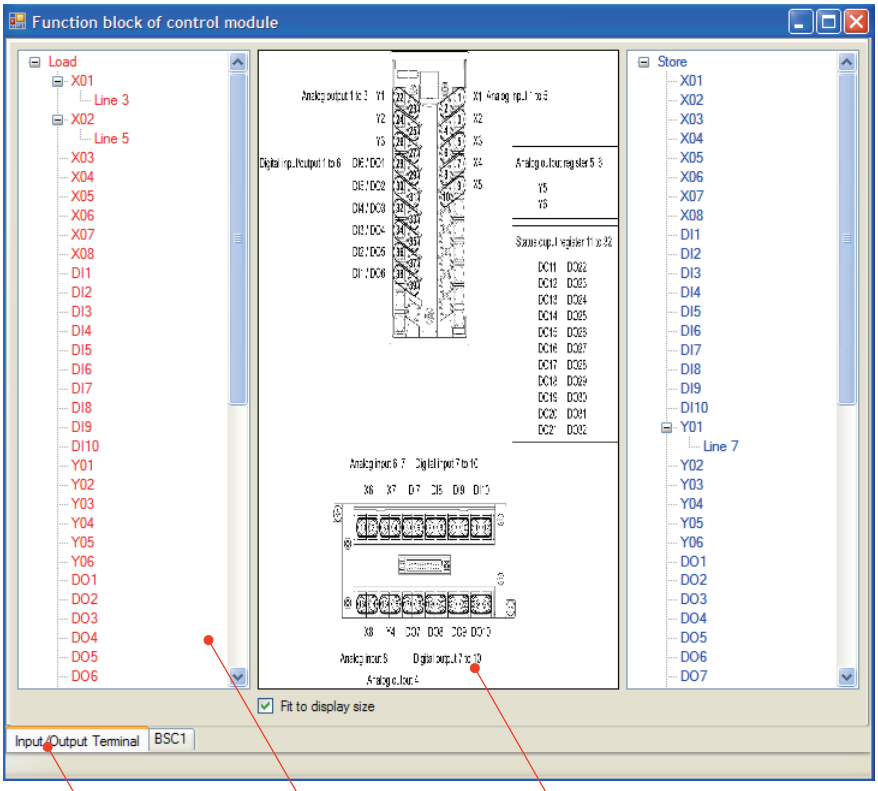
3.3.6 Link Display of Registers in Use

"Link display" displays which of the registers in the function block diagram have been programmed with the LD (Load), ST (Store) or STE (Store with enable switch) instruction.

In the Text Programming window, when LD xxx, ST yyy, or STE yyy is input, and xxx or yyy are registers displayed in the function block diagram, "Link Display" is displayed to the side of registers in the function block diagram.

Click "Link display" to jump to the corresponding line in the Text Programming window.
Note: The LD and ST commands in this case function as follows:

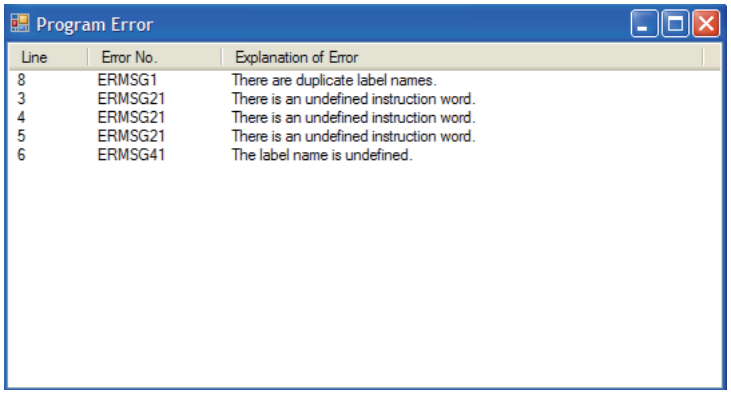
LD xxx = Reads the value from the register xxx.
ST yyy = Stores the value to the register yyy.
STE yyy = Stores the value of register S2 to register yyy when the register S1 ≥ 0.5.



	Operation
Window display method	(1) Check the check mark at **View>Function Block Window of Control Module** from menu. (2) The window can be displayed from the icon on the toolbar.
Closing windows	(1) Cancel the check mark at **View>Function Block Window of Control Module** from menu. (2) Click the Close button on the title bar to close the window.
Page tab	Click the Page tab to switch between pages.
Function block diagram, input/output terminal diagram	These diagrams cannot be accessed (clicked, etc.)
Browse view Storage view	Double-click these items to jump to the corresponding page of the corresponding program, and display the corresponding line as active. Move the mouse pointer close to these items to display the register name.

3.3.7 Error Messages

Double-click a line that is indicating an error to jump to the program page and display the corresponding line as active.



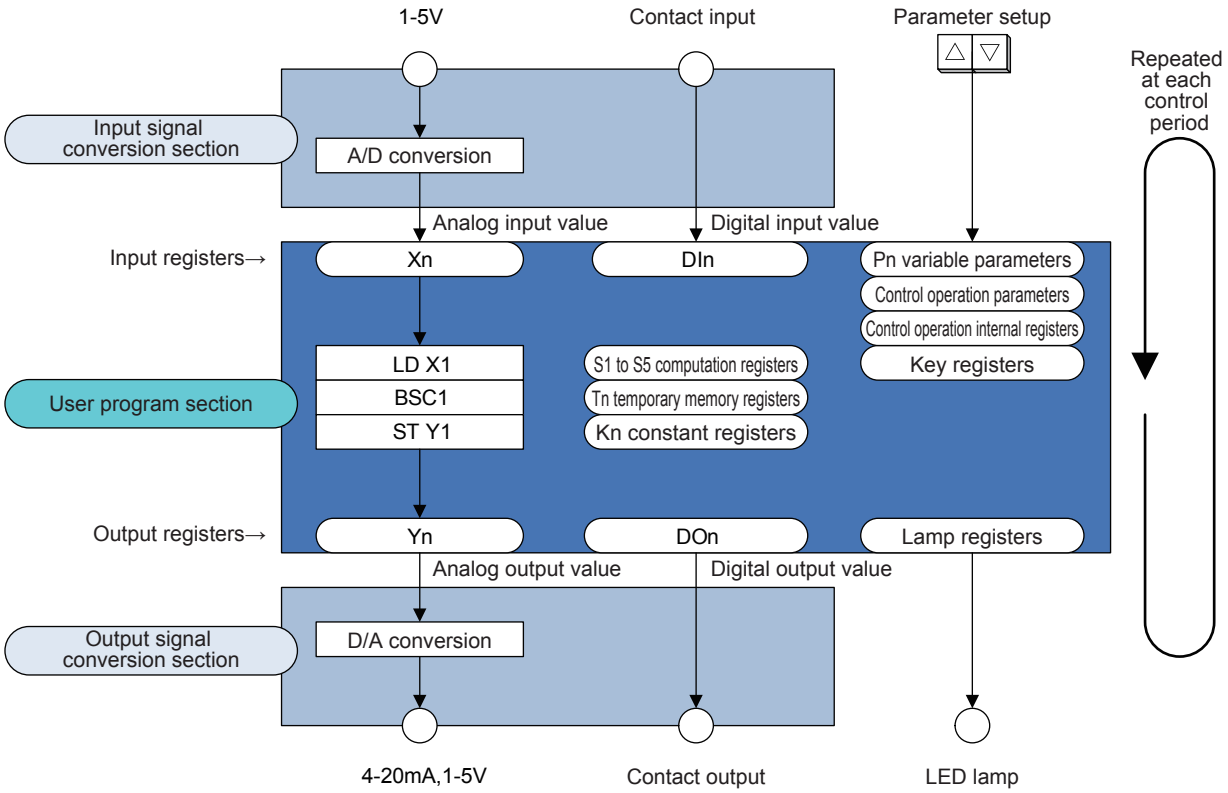
0357E.ai

Error No.	Description
ERMSG1	There are duplicate label names.
ERMSG2	There are duplicate subprogram names.
ERMSG11	The maximum number of steps has been exceeded.
ERMSG12	The maximum number of steps of the simulation program has been exceeded.
ERMSG21	There is an undefined instruction word.
ERMSG22	Unaccepted characters have been used for the label.
ERMSG23	There is a wrong instruction word after the instructions words.
ERMSG24	The label is being used incorrectly.
ERMSG25	There is no label.
ERMSG31	There is no END.
ERMSG32	There is a subprogram without END.
ERMSG33	There is END for a subprogram.
ERMSG41	The label name is undefined.
ERMSG51	The subprogram is undefined.
ERMSG52	There is RTN within the main program.
ERMSG61	There is no RTN for the subprogram.
ERMSG62	There is no RTN for SUB@SIMPR.
ERMSG63	GOSUB, GIFSUB and GTSUB statements are being used within the subprogram.
ERMSG71	GO and GIF statements are being used between the main program and subprogram.
ERMSG72	GO and GIF statements are being used between SUB@n or SUB@ CSCPRn and SUB@SIMPR.
ERMSG81	The subprogram is before the main program.
ERMSG82	Another subprogram is after the simulation program.
ERMSG92	There are too many characters for the label.
ERMSG112	A different control operation is being used at the same time.
ERMSG901	There is no program.
ERMSG999	The number of errors has been exceeded. Compiling has been stopped.

Blank Page

4.1 Basic Operation

The figure below shows an overview of the flow of signals and computations on the YS1700. The user program is positioned between the input signal conversion section and output signal conversion section. The I/O registers function as the interface between the input signal conversion section and the output signal conversion section.



Basic Operation of YS1700 (example of text programming)

Input conversion section

Analog inputs, digital inputs and various setup data (simply called "parameters" from here on) are automatically converted to internal data at the input signal conversion section, and stored in the input registers.

User program section

When input conversion ends, the user program is started up to read the various data and execute the target computation and control.
After computation, the results are stored to the output registers.

Output conversion section

The data stored in the output registers is automatically converted to analog signals and digital signals, which are output from the controller.

In this way, the series of operations input conversion → computation → output conversion is executed at each control period.

► I/O registers: "4.4.8 I/O Signals and I/O Signal Registers"

4.1.1 Control Modules

Control modules are the basic modules for building control functions. Control modules are a type of computing module.
By using these control modules, you can achieve the control functions listed in the table below.

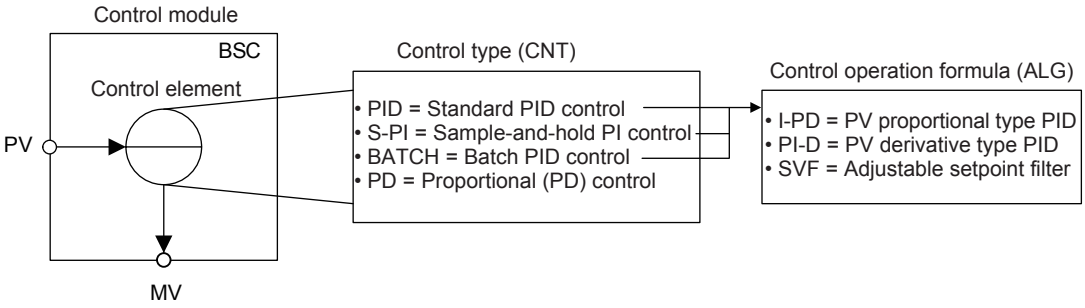
Control Types and Control Modules

Control Type	Applicable Control Module
Single-loop control	BSC1
Dual-loop control	BSC1 and BSC2
Cascade control	CSC
Selector control	SSC

4.1.2 Control Types (CNT) and Control Operation Formulas (ALG)

To determine the control operation formulas of control modules, set the control type (CNT) and control operation formulas (ALG) in the engineering parameters.

► Control type and control operation formula settings: "Operating the Engineering Displays" in YS1500 Indicating Controller/YS1700 Programmable Indicating Controller Operation Guide



Control Elements in Control Modules

The following table shows the combinations of control types (CNT) and control operation formulas (ALG) for each of the control modules.

Combinations of Control Types and Control Operation Formulas

Control Type (CNT)	Control Operation Formula (ALG)	BSC1	BSC2	CSC		SSC	
				Loop 1	Loop 2	Loop 1	Loop 2
Standard PID control	PV proportional type PID (I-PD)	✓	✓	✓	✓	✓	✓
	PV derivative type PID (PI-D)	✓	✓	✓	✓	✓	✓
	Adjustable setpoint filter (SVF)	✓	✓	✓	✓	✓	✓
Proportional (PD) control	PV proportional type PID (I-PD)	–	–	–	–	–	–
	PV derivative type PID (PI-D)	✓	✓	–	–	–	–
	Adjustable setpoint filter (SVF)	–	–	–	–	–	–
Sample-and-hold PI control	PV proportional type PID (I-PD)	✓	✓	✓	✓	✓	✓
	PV derivative type PID (PI-D)	✓	✓	✓	✓	✓	✓
	Adjustable setpoint filter (SVF)	✓	✓	✓	✓	✓	✓
Batch PID control	PV proportional type PID (I-PD)	✓	✓	–	–	–	–
	PV derivative type PID (PI-D)	✓	✓	–	–	–	–
	Adjustable setpoint filter (SVF)	✓	✓	–	–	–	–

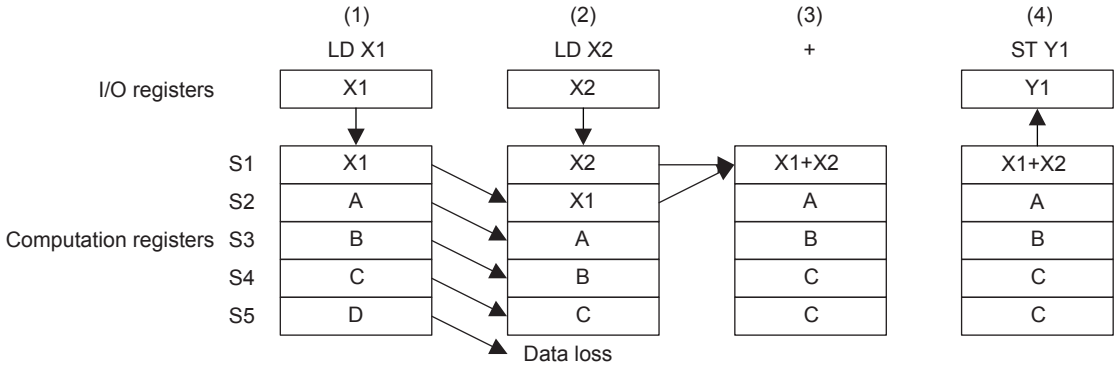
✓: Available, –: Not available (Do not set this combination.)

► Explanation of control types (CNT) and control operation formulas (ALG): “1.2 Selecting the Control Method (Selecting Control Type CNT and Control Operation Formula ALG)” in YS1500 Indicating Controller/YS1700 Programmable Indicating Controller User's Manual

4.2 Principles of Computation

Programming in text is generally known as the "reverse Polish notation."
Computation is executed in a five-stage computation register called the "S register." The Read instruction (annotated as "LD") is used for transferring data from the input registers to the S registers. Each of the computation and control instructions ("+," "HSL," "BSC1," etc.) perform computations on the content of the S registers, and store the result to the S registers. The Store instruction (annotated as "ST") is used to transfer the computation results to the output registers. The content of S registers is sometimes lost or shifted depending on the instructions that are executed.
Computation registers S1 to S5 are common to the main, sub- and simulation programs. The content of these registers is processed in accordance with the latest instruction regardless of jump and return in the program.

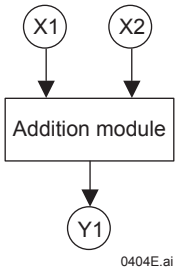
Example of 2-input addition text program



Operation of S Registers

	Instruction	S1	S2	S3	S4	S5	Explanation
–	(default state)	A	B	C	D	E	Contains no data.
(1)	LD X1	X1	A	B	C	D	Loads data X1 in the input registers to S1. The content of each S register is sequentially pushed down. The data in S5 is lost.
(2)	LD X2	X2	X1	A	B	C	Loads data X2 in the input registers. Operation is the same as (1).
(3)	+	X1+X2	A	B	C	C	Adds the values of S1 and S2, and stores the result to S1. The content of each S register is sequentially pushed up. The original value remains in S5.
(4)	ST Y1	X1+X2	A	B	C	C	Stores the values of register S1 to output register Y1. The content of each S register does not change.

When programming by function block programming, there is no need to be conscious of how S registers operate. This programming method has no Read and Store instructions, and is the equivalent of performing "wiring."
The following shows how the above text program is annotated in function block programming.



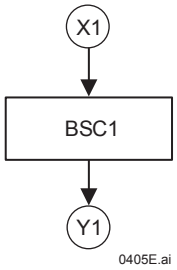
Example of Addition Program

4.3 Principles of Computation of PID Control

PID control computation also is a type of computation function, and programming methods are the same as those of other functions.

Instruction	S1	S2	S3	S4	S5	Explanation
LD X1	X1	A	B	C	D	Loads data X1 in the input registers to S1.
BSC1	MV1	A	B	C	D	Performs PID computation. The result (output value) is stored to S1.
ST Y1	MV1	A	B	C	D	Stores output values to Y1.
END						Ends the program.

The following shows the above annotated in function block programming.



The BSC instruction instructs the process variables, setpoint values and manipulated output variable, reads PID parameters, and includes various other operations.

Control operation can be programmed by a single program such as this.
The BSC instruction (BSC1 targets loop 1 and BSC2 targets loop 2) instructs the process variables, setpoint values and manipulated output variable, reads PID parameters, and includes various other operations. The following figure illustrates the functions of the BSC instruction.

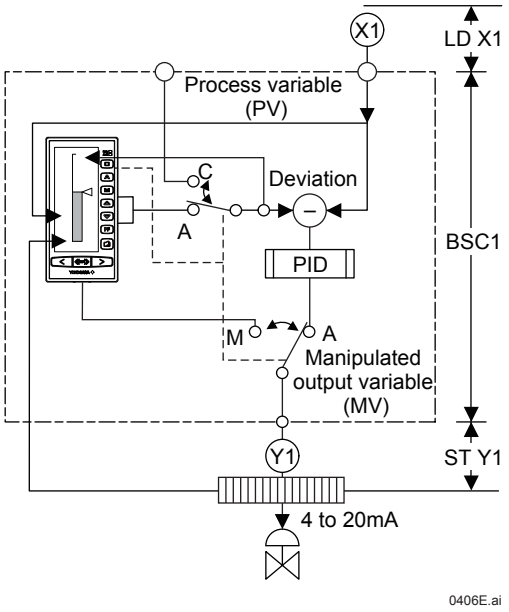
(1) Process variable (PV) display area
This area displays the content (PV value) in computation register (S1) just before the BSC instruction. It does not display the input signal as it is. It can also display PV signals that have undergone computation, such as square root extraction.

(2) Setpoint value (SV) display area
In the C mode, this area instructs external setpoint values, and in the A and M modes, instructs the internal setpoint values by the SV setting keys on the front panel.

(3) Control operation
Control such as standard PID control and proportional (PD) control is executed.

(4) Manipulated output variable (MV)
The MV is stored to register S1 after execution of the BSC instruction. In the M mode, the MV can be incremented and decremented by the MV operation keys.

(5) Current output (Y1) and output display area
Analog output 1 (Y1) is the current output terminal. In this example, the MV value is taken to be the manipulated output variable by the ST Y1 instruction. The MV display area at the bottom of the front panel displays analog output 1 (Y1).



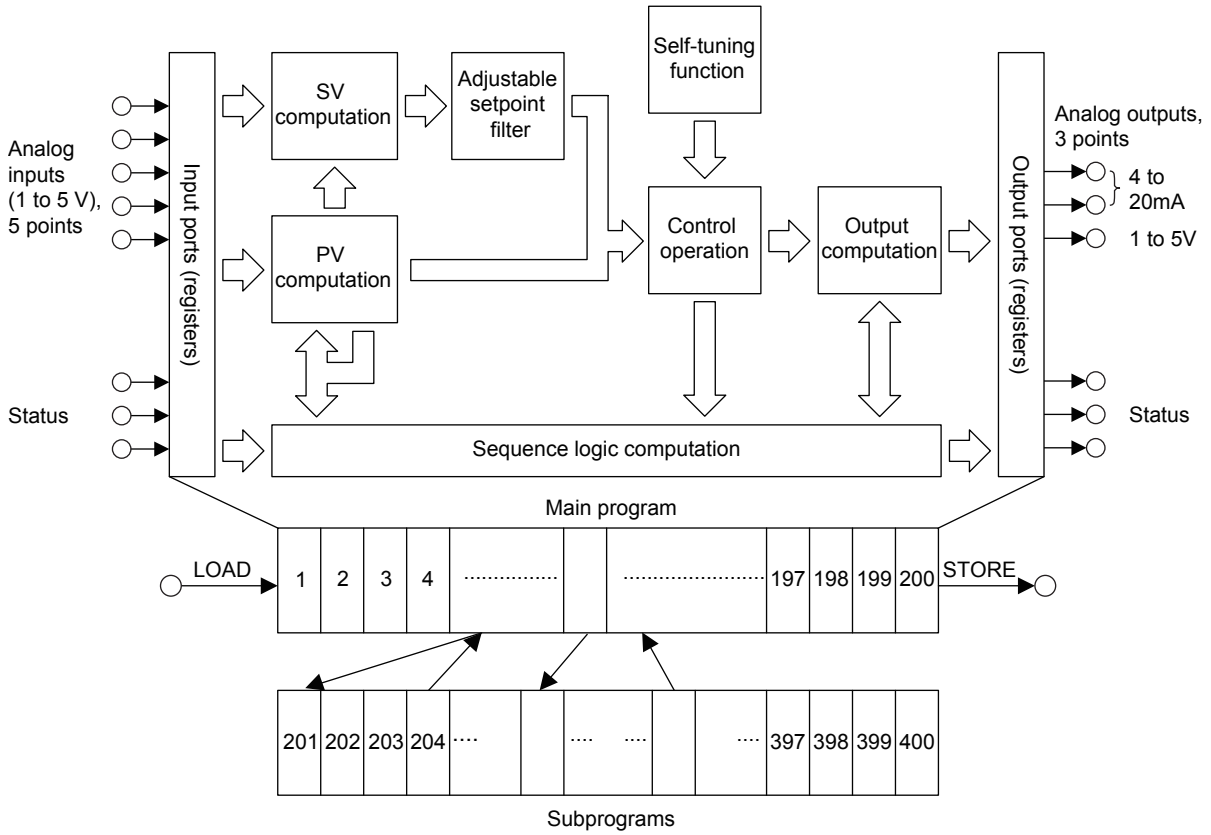
Correspondence between Control Functions (BSC) and Display/Operation Areas of the Controller

4.4 Structure, Size and Control Period of Programs

4.4.1 Program Structure and Size

User programs comprise a main program, a subprogram and a simulation program.

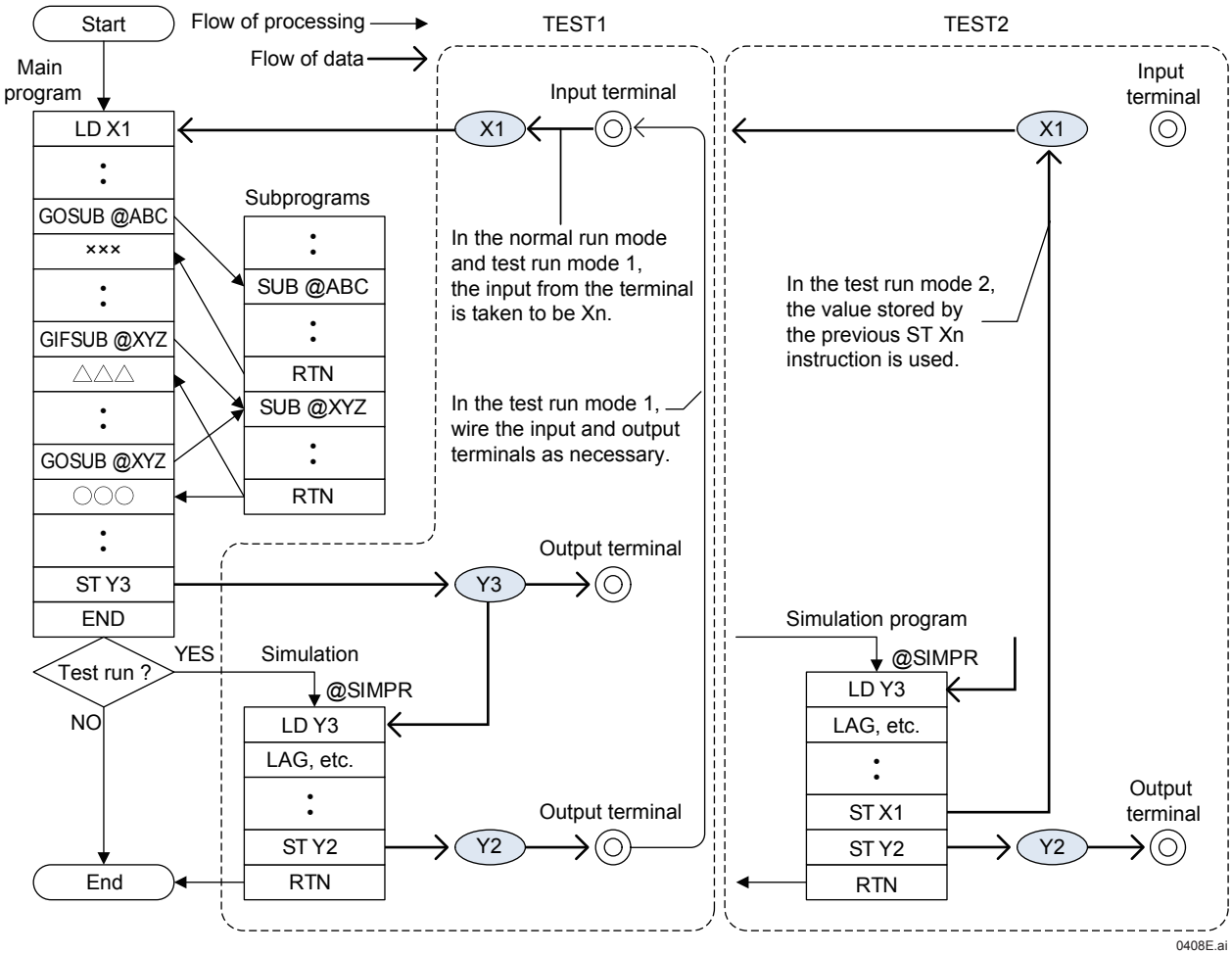
Programs start from Step 1 of the main program, subprograms are executed by branch instructions while sequentially executing steps in the main program, and finally the program ends by the END instruction written in the main program. The following figure is an example a main program and a subprogram comprising 200 steps each.



0407E.ai

During a test run, the simulation program is executed. A program for verifying operation of the main and subprograms is written to the simulation program.
Two modes, mode 1 (TEST1) and mode 2 (TEST2), are available for executing a test run.
In mode 1, the actual terminals are used for input and output. Assign the input and output of the simulation program to the terminals like TEST1 in the figure below and carry out external wiring before executing the simulation.
In mode 2, the Xn and DIn registers are isolated from the input terminals. Store instructions (ST Xn and ST DIn) can be written to the simulation program. Stored values are used as the input value of the next computation. Execute the simulation by storing the output of the simulation program to the input of the main program like TEST2 in the figure below.

	Main Program	Subprogram	Simulation Program
Description	From Step 1	SUB @<sub-program name>	SUB @SIMPR
Start of program	Step 1 at each control period	At GOSUB or GIFSUB instruction	The GOSUB or GIFSUB instruction is not required if the test run is being executed after the END instruction in the main program.
End of program	At END instruction	At RTN instruction	At RTN instruction
At a control period overflow	The computation overflow flag is raised to continue computation. The number of times that the computation overflow occurs is counted. If computation does not end after two control periods, program execution is forcibly ended. A system alarm occurs.		If computation does not end even after the control period, program execution is forcibly ended, and a system alarm occurs.



0408E.ai

Details of Text Program Errors

Error Item	Check Details
END instruction	An error results if there is no END instruction or there are multiple END instructions, or if an END instruction exists in a subprogram.
Subprogram	Jumping from one subprogram to another (i.e. nesting) is prohibited.
RTN instruction	Programming the RTN instruction in the main program results in an error.
Control operation/ dynamic operation	Multiple dynamic operations cannot be used in a single control period. However, multiple linearizer instructions can be used. An error occurs if the programming content is illegal.

Subprogram names "@CSCPR1" and "@CSCPR2" are exclusively for the cascade (CSC) control module.
Subprogram name "@SIMPR" is for the simulation program during a test run.
Do not program control modules (BSC1, BSC2, CSC, SSC) in subprograms.

Program Size

Programming Method	Program Size
Function block programming	A total of 400 modules can be programmed in the main and subprograms. Up to ten modules can be programmed in the simulation program.
Text programming	A total of 1000 steps can be programmed in the main and subprograms. Up to 50 steps can be programmed in the simulation program.

4.4.2 Program Execution Statuses

Run (normal run)

The following are performed at each control period:

(1)Updating of input terminal statuses to input registers
(2)Execution of instructions in the user program
(3)Updating of output register values to output terminals

Test run

Two modes, mode 1 and mode 2, are available for executing a test run. When the next control period arrives before computation has ended, computation is forcibly ended as soon as the currently executing step ends.

<Mode 1> (TEST1)

This mode performs an operation test by main program, simulation program and external wiring.

The following are performed at each control period:

(1)Updating of input terminal statuses to input registers
(2)Execution of instructions in the user program
(3)Updating of output register values to output terminals
(4)Execution of simulation program

<Mode 2> (TEST2)

This mode sets the simulated input to the input terminals by main program, simulation program, internal wiring of the program and various YSS1000 monitoring functions.

The following are performed at each control period:

(1)Execution of instructions in the user program
(2)Updating of output register values to output terminals
(3)Execution of simulation program

The values of the previous control period are carried over as the values of the input registers.

STOP

In this state, updating of I/O and execution of the user program are stopped.

(1)During engineering parameter setup
(2)During downloading of the user program

Set the YS1000 to STOP using the YSS1000 before downloading parameters or user program to the YS1000. After downloading is completed, set the YS1000 to RUN.

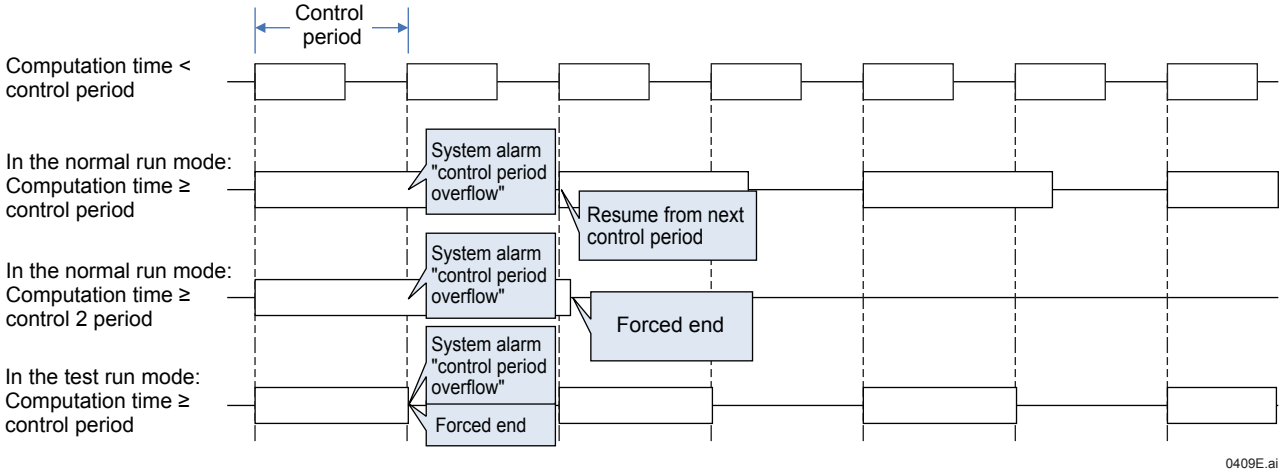
For how to check, the program execution status and how to set simulated input value in TEST2, see "2.10."

4.4.3 Control Period

The control period can be selected from 50 ms, 100 ms and 200 ms. When making a user program, set the control period in the YSS1000 Setting Software for YS1000 Series. The validity of the preset control period can be verified by the load factor during a normal run or a test run.

When computation does not end within the control period (i.e. load factor ≥ 100%)

- During a normal run, the system alarm "control period overflow" error occurs, and computation continues as it is. When computation does not end even after two control periods, the program is exited when the currently executing step ends. If computation ends before then, computation is resumed. However, control period- and time-related instructions no longer are normal at this time.
- During a test run, the program is exited when the currently executing step ends, and the system alarm "control period overflow" error occurs.
- The number of times that the computation overflow occurs is counted. When computation is forcibly ended, output will be updated only if the ST Yn instruction is executed.



4.4.4 Program Load Factor

The load factor is expressed by the following formula:

$$\text{Load factor (\%)} = \frac{\text{Computation execution time}}{\text{Possible computation time}} \times 100$$

0410E.ai

The "computation execution time" is a value on the entire system that combines the following:

(1) Execution time of system computations (e.g. display processing, communications processing)
(2) Execution time of user program computation

Possible Computation Time

Control Period	Possible Computation Time
50 ms	16.5 ms
100 ms	66.5 ms
200 ms	166.5 ms

When the STC function is not used, use a load factor of less than 100%. The load factor increases temporarily when the STC function is used. Use the reference values in the following table as the maximum load factor.

Control Period	Max. Load Factor (reference values)
50 ms	48%
100 ms	64%
200 ms	72%

4.4.5 Computing Data Format

Computing data is a 4-byte floating point number.

4.4.6 Computation Range and Computation Accuracy (Computation Resolution)

Computation range: $-3.402823 * 10^{38}$ to $3.402823 * 10^{38}$

Computation accuracy: $1.175494 * 10^{(-38)}$
However, A/D conversion error is not included.

4.4.7 Computation Overflow

The system alarm "computation overflow" error occurs when the computation process or result is a non-numeric or infinity.
When infinity occurs, computation is limited to the maximum value of that sign. For non-numerics, the result is stipulated for each instruction, and computation is continued.

4.4.8 I/O Signals and I/O Signal Registers

Types of I/O registers

I/O signals are stored to a data storage location called an "I/O register."
Create the user program so that it reads data to computation registers from these I/O registers, and stores the computation result to the I/O registers.

Input Register	Signal	Value
X1 to X5	Analog inputs 1 to 5 (voltage) Basic type: Unused -0.25 is read. Basic type (with expandable I/O): Analog inputs 6 to 8 (voltage)	The input value is used if a signal is being input to the terminals. (0.0 to 1.0 for 1 to 5 V) -0.25 if no signal is being input
X6 to X8		
DI1 to DI6	Digital inputs 1 to 6 (depending on assignment)	"0" is read for unassigned DI. The value is "0" for an open terminal, and "1" for a closed terminal.
DI7 to DI10	Digital inputs 7 to 10 basic type (with expandable I/O only)	
CX1 to CX16	Peer-to-peer communication analog inputs 1 to 16	"0" when peer-to-peer communication is not used. "0" even when the power supply is turned on.
CI1 to CI64	Peer-to-peer communication status inputs 1 to 64	

Output Register	Signal	Default
Y1	Analog output 1 (current)	<At a hot start, status change from STOP to run> Previous value <At a cold start, initial start> Current output: -0.2 Voltage output: -0.063 Communication output: 0
Y2	Analog output 2 (voltage)	
Y3	Analog output 3 (current/voltage)	
Y4	Basic type: Analog output register 4 Basic type (with expandable I/O): Analog output 4 (voltage)	
Y5, Y6	Analog output registers 5 and 6	
DO1 to DO6	Digital outputs 1 to 6 (depending on assignment) Unassigned DO are status output registers.	<At a hot start, status change from STOP to run> Previous value <At a cold start, initial start> 0
DO7 to DO10	Basic type: Status output registers 7 to 10 Basic type (with expandable I/O): Digital outputs 7 to 10	
DO11 to DO32	Status output registers 11 to 32	
CY1 to CY4	Peer-to-peer communication analog outputs 1 to 4	<At a hot start, status change from STOP to run> Previous value <At a cold start, initial start> 0
CO1 to CO16	Peer-to-peer communication status outputs 1 to 16	

X1 to X5 and Y1 to Y3 are directly linked to the I/O terminals. On the basic type (with expandable I/O), X6 to X8 and Y4 are also linked to the I/O terminals.
Y5 and Y6 are registers for saving data, and can be used as communication output registers.
Of the 12 digital I/O points (DI1 to DI6, DO1 to DO6), six points can be connected to digital I/O terminals. Select and set these in the YSS1000 Setting Software for YS1000 Series.
Data is stored only to DI1 to DI10 registers that are actually linked to terminals. The value is "0" when these registers are not linked to terminals.
Of DO1 to DO10, only the values of DO1 to DO10 registers that are linked to registers are output. When these registers are not linked to registers, they can be used as registers for saving status data.
DO11 to DO32 and basic type DO7 to DO10 are registers for saving status data.
The CX1 to CX16, CI1 to CI64, CY1 to CY4, and CO1 to CO16 I/O registers are exclusively for Peer-to-peer communication.

► Peer-to-peer communication: "Chapter 8 Using Peer-to-peer Communication"

In the test run mode 2, X1 to X6 and DI1 to DI10 are isolated from the terminals. These can be used as simulated inputs by using the ST Xn and ST DIn instructions in the simulation program.

■ Analog I/O signals

Relationship between signals and internal data

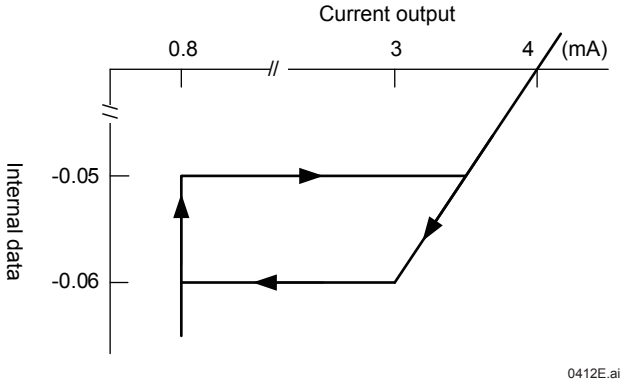
The analog input signals (1 to 5 V) are converted to internal data of 0.0 to 1.0 before program execution, and stored in the input registers.
After computation, data is stored to the output registers, and 0.0 to 1.0 are output as analog outputs of 1 to 5 V or 4 to 20 mA.

As the current output, in order to provide the YS1700 with a forced fully close function for a valve (final control element), when the signal decreases to -6% (3.04 mA), the current decreases rapidly to about at -20% (0.8 mA). When the signal increases to internal data -5% or higher, the current signal is released from the -20% state to return to about -5% (3.2 mA). Output 100% or more increases linearly, and is limited at about 106.25% (21 mA).

Analog I/O Signals

	Signal (internal data)	Accuracy %/Span	Signal Limit Value (corresponding internal data)
Input	1 to 5 V(0.0 to 1.0)	±0.1	0 to 6.35 V (-0.250 to 1.337)
Output	1 to 5 V (0.0 to 1.0)	±0.1	0.75 to 5.25 V (-0.0625 to 1.0625)
	4 to 20 mA (0.0 to 1.0)	±0.2	0.8 to 21.0 mA (-0.0625 to 1.0625)

Note 1: Accuracy is valid within internal data range 0.0 to 1.0. At sections where internal data range 0.0 to 1.0 is exceeded, accuracy becomes out-of-specification even though a proportional relationship is established between the signal and internal data.
Note 2: The input signal overrange alarm functions on the range -0.0625 to 1.0625.



YS1700 Current Output Characteristics (tight shut state)

Structure of internal data

Internal data is 4-byte floating point number (IEEE 754 single-accuracy floating point number type).

■ Digital I/O signals

There are six digital signal terminals shared for I/O, and these can be specified as desired for input or output. With the basic type (with expandable I/O), four points each are added exclusively for input and output, respectively. Select and set the programmable mode in the YSS1000 Setting Software for YS1000 Series.

Relationship between signals and internal data

Digital input signals are converted to internal data of "0" (contact open) or "1" (contact closed) before program execution, and stored in the input registers.
After computation, data is stored to the output registers, and the value is output as "contact open" if less than 0.5, and as "contact closed" if 0.5 or more.

Digital I/O Signal

	Signal (internal data)	
Input	Contact open (0.000)	Contact closed (1.000)
Output	Contact open (less than 0.500)	Contact closed (0.500 or more)

Structure of internal data

I/O register values are 4-byte floating point numbers (IEEE 754 single-accuracy floating point number type). In internal processing, however, these values are handled as two values of less than 0.5 or 0.5 or more.

4.4.9 Internal Registers

Structure of internal data

The values of temporary memory registers (Tn), variable registers (Pn), constant registers (Kn), key registers (KYn), and LED lamp registers (LPn) are 4-byte floating point numbers (IEEE 754 single-accuracy floating point number type).

Temporary memory registers (Tn) n=1 to 60

These registers are for temporarily saving intermediate values during computation, and can be read and written from and to the program. These registers are not reset at each control period. In the P&T Register Display, values are only displayed. There is no unit for these values. On the YS1700, these registers are displayed as temporary memory registers (T01 to T60).

Variable registers (Pn) n=1 to 30

These registers are used as constants and coefficients of computation, and can be read and written from and to the program. These registers are not reset at each control period. In the P&T Register Display, scaled values can be displayed and set. There is no unit for these values. Scaling can be set to each individual register in the YSS1000 Setting Software for YS1000 Series. On the YS1700, these registers are displayed as variable parameters (P01 to P30).
Scaling range: -99999 to 99999, default: 0 to 100, first decimal place
When read in the user program, the scale high limit (PSH) is set to 1 and scale low limit (PSL) is set to 0.

Constant registers (Kn) n=1 to 100

These registers are used as constants and coefficients of computation, and are set in the YSS1000 Setting Software for YS1000 Series during programming. They can be read but not changed by the program, and are not reset at each control period. In the K Constant Display, the values of n=1 to 60 can be verified. There is no unit for these values. On the YS1700, these registers are displayed as constant registers (K01 to K60).

Key register (KY1)

n=1: PF key
The "key pressed" status is read to this register.

LED lamp registers (LPn) n=1 to 7

n=1: PF key, n=2: Loop 1 C mode key, n=3: Loop 1 A mode key, n=4: Loop 1 M mode key, n=5: Loop 2 C mode key, n=6: Loop 2 A mode key, n=7: Loop 2 M mode key
The LED lamp status is read to these registers. Lit/out states can be written only by the PF key. A lit state is indicated by "1."

Backlight ON/OFF flag (CFL)

The backlight status (lit/out) can be read from and written to this flag.
Lit: 0, out: 1

4.4.10 Special Registers

Register	Meaning	Explanation
PON	At power ON	1.0 only during the initial control period at a power ON regardless of start type (hot or cold). 0.0 from then on
COLD	COLD	1.0 only during the initial control period at a cold start. 0.0 from then on
MNS1	Always -1	-1.0 can always be read.
ZERO	Always 0	0.0 can always be read.
PLS1	Always 1	1.0 can always be read.
PLS2	Always 2	2.0 can always be read.
PLS3	Always 3	3.0 can always be read.
PLS4	Always 4	4.0 can always be read.
PLS5	Always 5	5.0 can always be read.
PLS6	Always 6	6.0 can always be read.
PLS7	Always 7	7.0 can always be read.
PLS8	Always 8	8.0 can always be read.
PLS9	Always 9	9.0 can always be read.
PLS10	Always 10	10.0 can always be read.
ALT	Inversion	The initial control period at a cold start is 0.0. From the next control period, 1.0/0.0 can be read at each control period. Reading is delayed at a control period overflow error.
CLK1	1-second clock	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0/0.0 can be read at 0.5 second intervals. Reading is delayed at a control period overflow error. When the control period is 200 ms, from 0.0→1.0 to 1.0→0.0 is 0.4 second, and from 1.0→0.0 to 0.0→1.0 is 0.6 second.
CLK2	2-second clock	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0/0.0 can be read at 1 second intervals. Reading is delayed at a control period overflow error.
CLK10	10-second clock	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0/0.0 can be read at 5 second intervals. Reading is delayed at a control period overflow error.
CLK60	60-second clock	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0/0.0 can be read at 30 second intervals. Reading is delayed at a control period overflow error.
CLK1P	1-second clock pulse	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0 can be read at 1 second intervals for one control period only. Reading is delayed at a control period overflow error.
CLK2P	2-second clock pulse	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0 can be read at 2 second intervals for one control period only. Reading is delayed at a control period overflow error.
CLK10P	10-second clock pulse	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0 can be read at 10 second intervals for one control period only. Reading is delayed at a control period overflow error.
CLK60P	60-second clock pulse	A reset is applied (0.0 can be read) at the initial control period at a cold start. From the next control period, the internal count is started, and 1.0 can be read at 60 second intervals for one control period only. Reading is delayed at a control period overflow error.

5.1 Three Types of Control Modules

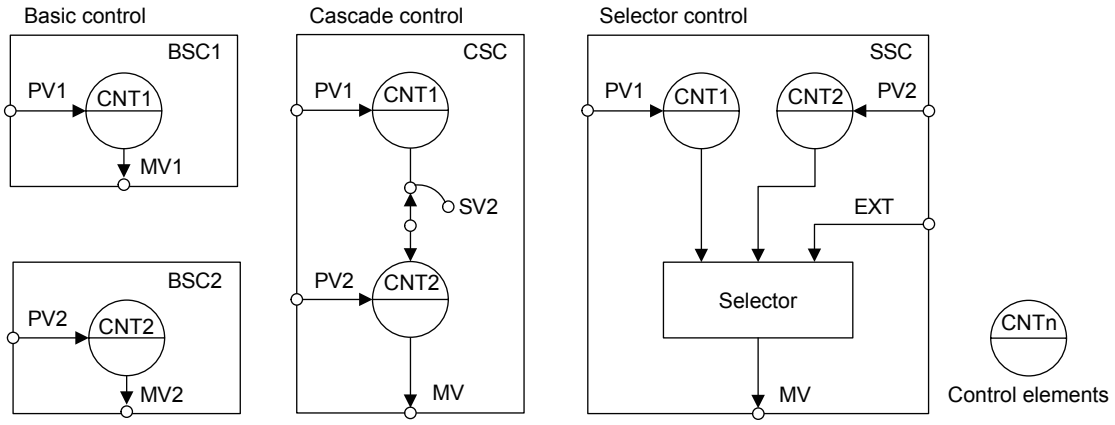
The YS1700 is provided with the following:
(1) Control modules for determining the structure of control operations
(2) Extended function registers for enabling various applied control operations
These modules and registers can be combined in programming to achieve the desired control.

► Extended function registers: "Chapter 6 Applied Usage of Control Modules"

There are three types of control modules as follows:
(1) Basic control module (computation instruction symbols BSC1, BSC2)
To execute 1-loop control functions, use BSC1, and to execute 2-loop control functions, use BSC1 and BSC2.
(2) Cascade control module (computation instruction symbol CSC)
This module is used when two loops are connected in series (in a cascade configuration), and cascade control is executed on one YS1700.
(3) Selector control module (computation instruction symbol SSC)
This module is used when two loops are connected in parallel, and autoselector control is executed on one YS1700.

Note

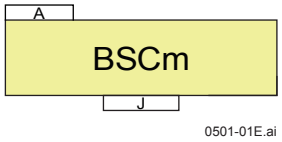
One type of control module can be executed only once on a single program. Note, however, that BSC1 and BSC2 can be executed once each.



Concept of Control Modules

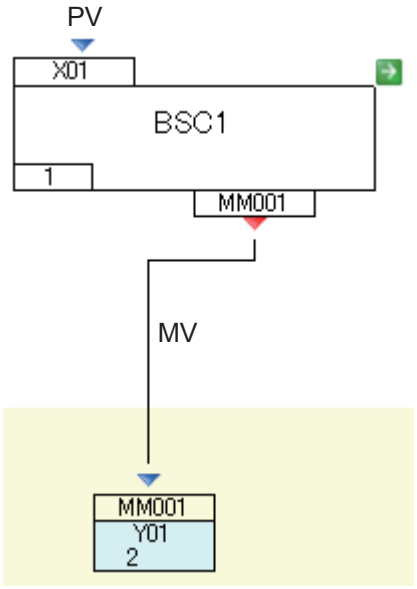
5.2 How to Use the Basic (BSC) Control Module

The BSC is provided with enough control functions for one loop. The BSC control module comes with two computing modules, BSC1 and BSC2, and computation instructions. When both computing modules are executed on a program, dual-loop control is performed on a single YS1700.
BSC1 is used as loop 1 in dual-loop control or single-loop control. BSC2 is used as loop 2 in dual-loop control.

Name of Computation	Computing Module	Computation Instruction Symbol
Basic control module	 0501-01E.ai	BSCm (m=1,2)

Operation

<Example of function block programming>
Connect the input and an output to the basic control module. When computation is executed, the computation is executed on the input value (process variable), and the result (manipulated output variable) is output. The keys and indication on the controller front are automatically updated.



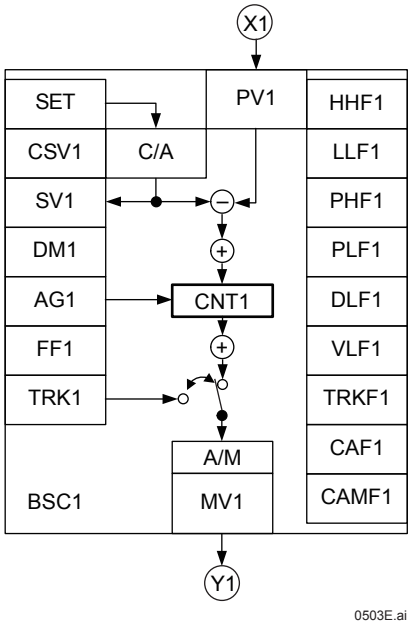
0502E.ai

<Example of text programming>
The input value (process variable) is stored to register S1 before execution of BSC computation. The BSC1 module targets loop 1, and the BSC2 module targets loop 2. After computation is executed, the computation result (manipulated output variable) is stored to register S1. The keys and indication on the controller front are automatically updated.

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads input X1 (process variable).
BSC1	MV1					Executes basic control.
ST Y1	MV1					Outputs result (manipulated output variable) to Y1.

Function Block Diagram

The following shows an overview of the basic control module (BSC).



BSC Function Block Overview

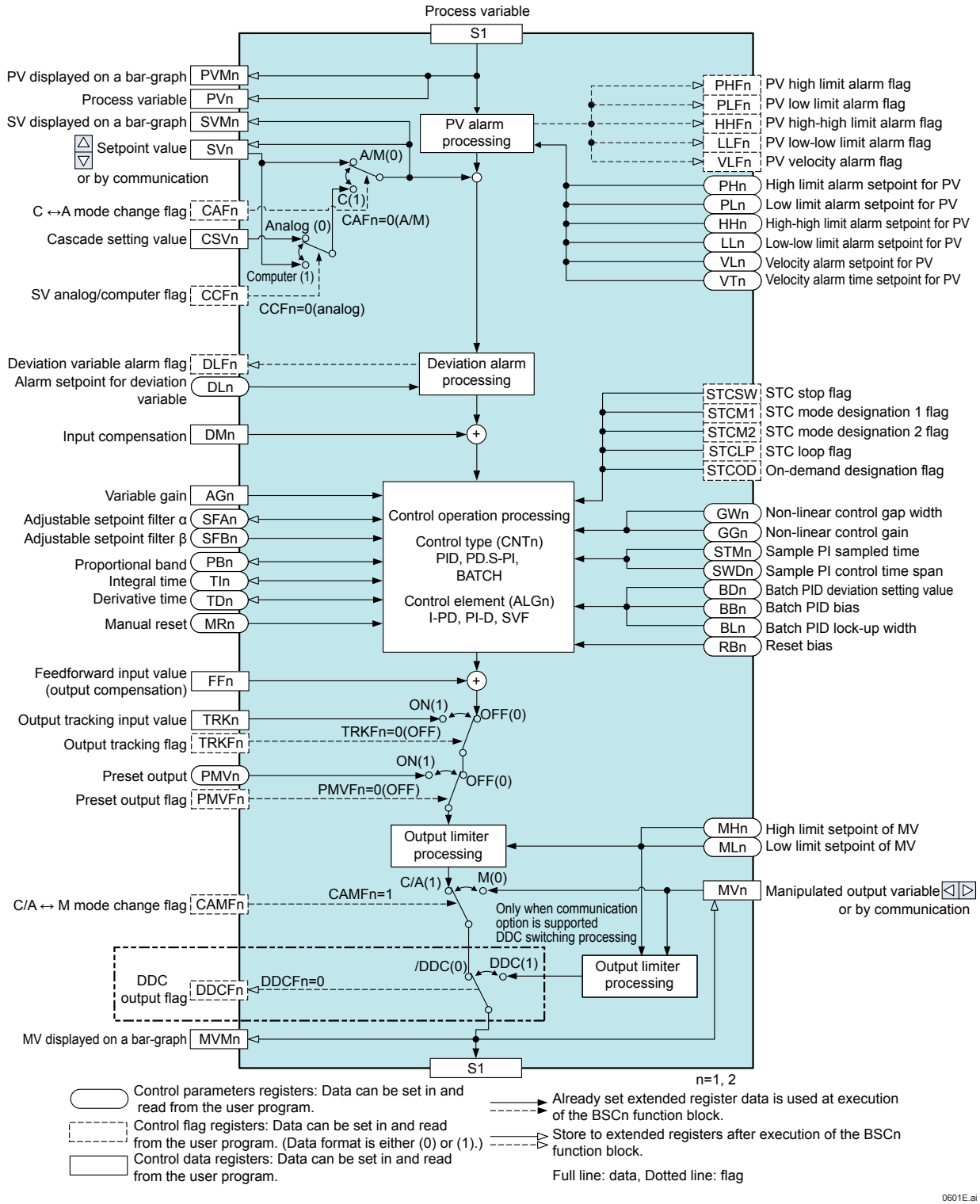
Note

Use Y1 as the output of BSC1, and Y3 as the output of BSC2.

5.2.1 Basic (BSC) Control Module

The figure below is the BSC function block including control elements (CNT1) and the extended function registers. S1 registers and extended function registers can be regarded as the signal terminals of the controller "BSC," and the specified functions are demonstrated by "wiring" data to these terminals.

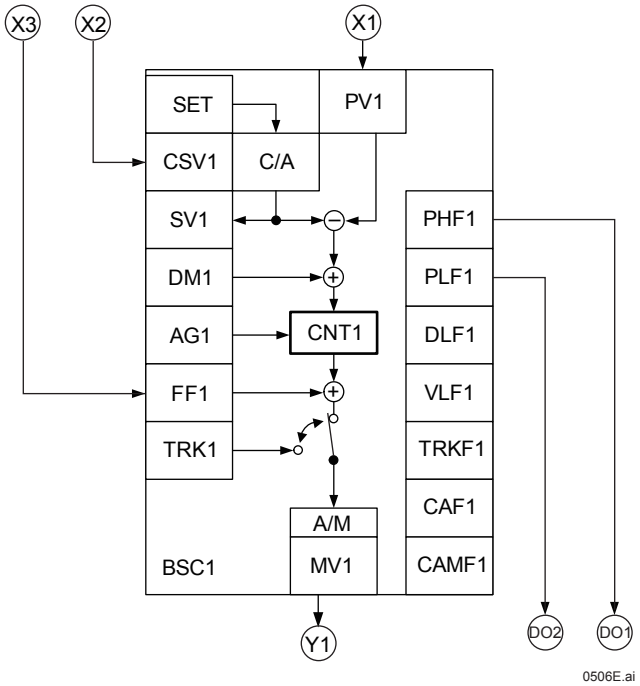
► S1 registers: "4.2 Principles of Computation"



Details of BSC Function Block

How to Use Extended Functions

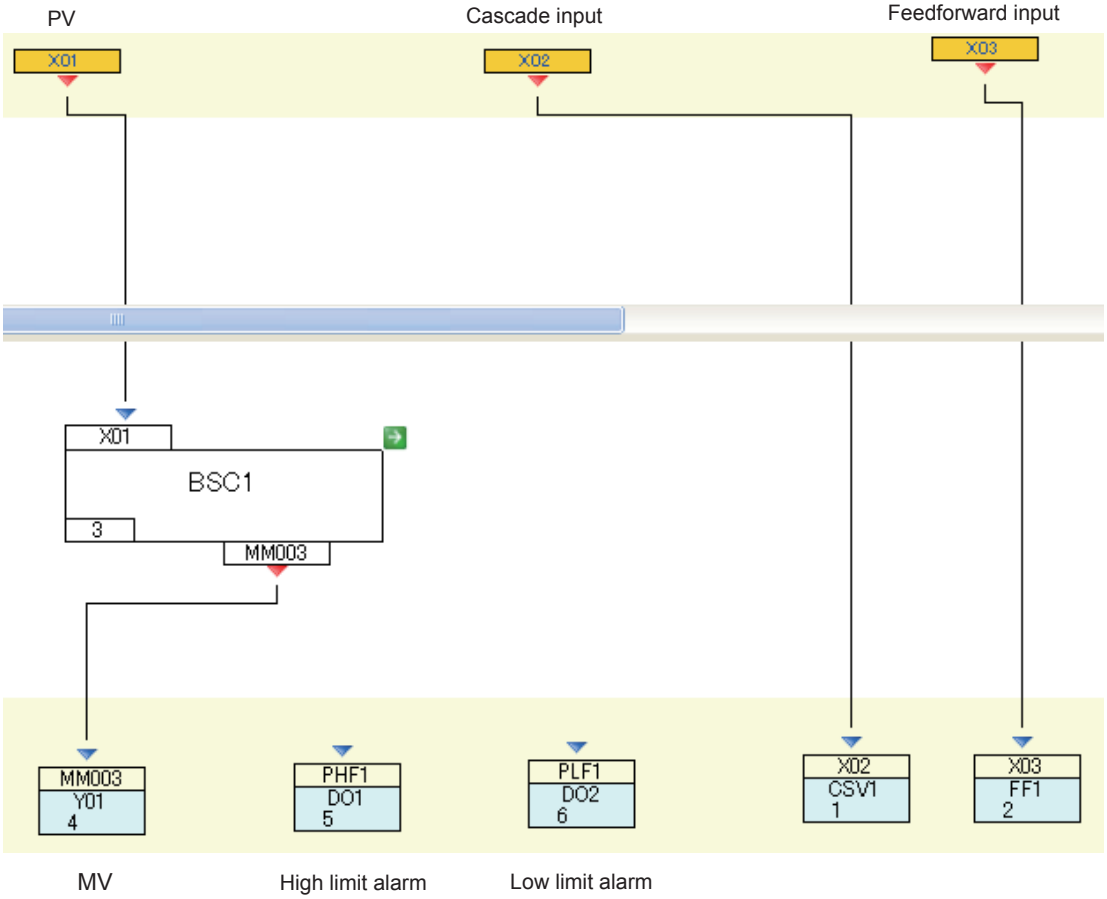
For example, when cascade is set, connect cascade setting input to CSV1 by the ST instruction.
When feedforward compensation is required, connect the signal to FF1.
Also, when outputting an input alarm, connect the content of PHF1 and PLF1 to register DOn.
Defaults are assigned to these control data registers and control flag registers so that these functions are disabled from the beginning. So, when these functions are not to be used, there is no need to be aware of these data and flags.
Data that is set on the YS1700 is stored to the control parameter registers. For details, see Chapter 6.



Using Extended Functions (loop 1 control)

The following shows the program for the control example in the figure above.

● **Example of function block programming**

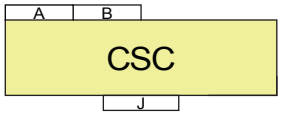


● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X2	X2					Cascade setting input (input 2)
ST CSV1	X2					Stores to the cascade input terminal.
LD X3	X3	X2				Feedforward input (input 3)
ST FF1	X3	X2				Stores to the FF1 input terminal.
LD X1	X1	X3	X2			PV (input 1)
BSC1	MV1	X3	X2			Basic control computation
ST Y1	MV1	X3	X2			Outputs the manipulated output variable.
LD PHF1	0/1	MV1	X3			Reads the high limit alarm (alarm ON: 1).
ST DO1	0/1	MV1	X3			Alarm output (digital output 1)
LD PLF1	0/1	0/1	MV1			Reads the low limit alarm (alarm ON: 1).
ST DO2	0/1	0/1	MV1			Alarm output (digital output 2)
END						Ends program execution.

5.3 How to Use the Cascade (CSC) Control Module

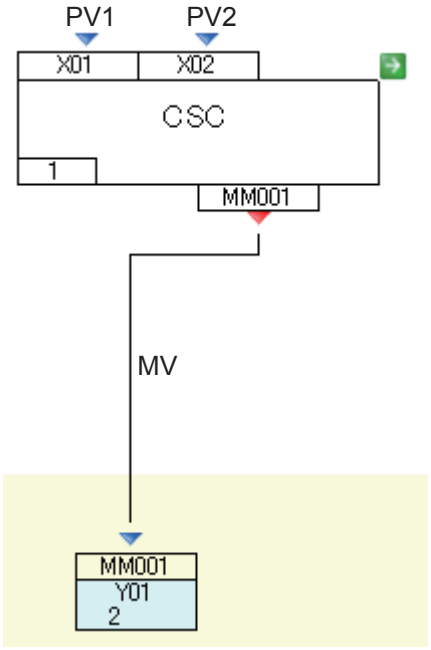
This module has two built-in PID control elements, and executes cascade control on a single YS1700.

Name of Computation	Computing Module	Computation Instruction Symbol
Cascade control module	 0503-01E.ai	CSC

Operation

<Example of function block programming>

Connect two inputs and an output to the cascade control module. When computation is executed, the computation is executed on the input value (process variable), and the result (manipulated output variable) is output. The keys and indication on the controller front are automatically updated.



0504E.ai

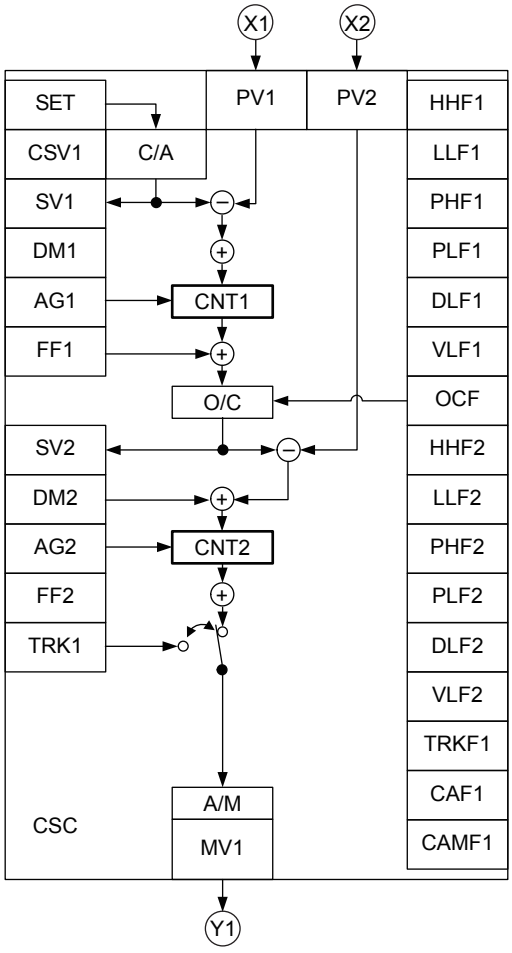
<Example of text programming>

The input value (process variable) of loop 1 is stored to register S2 and the input value (process variable) of loop 2 is stored to register S1 before execution of CSC computation. After computation is executed, the computation result (manipulated output variable) is stored to register S1. The keys and indication on the controller front are automatically updated.

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads input X1 (process variable).
LD X2	X2	X1				Reads input X2 (process variable).
CSC	MV1					Executes cascade control. CNT1, CNT2
ST Y1	MV1					Outputs result (manipulated output variable) to Y1.

Function Block Diagram

The following shows an overview of the cascade control module (CSC).

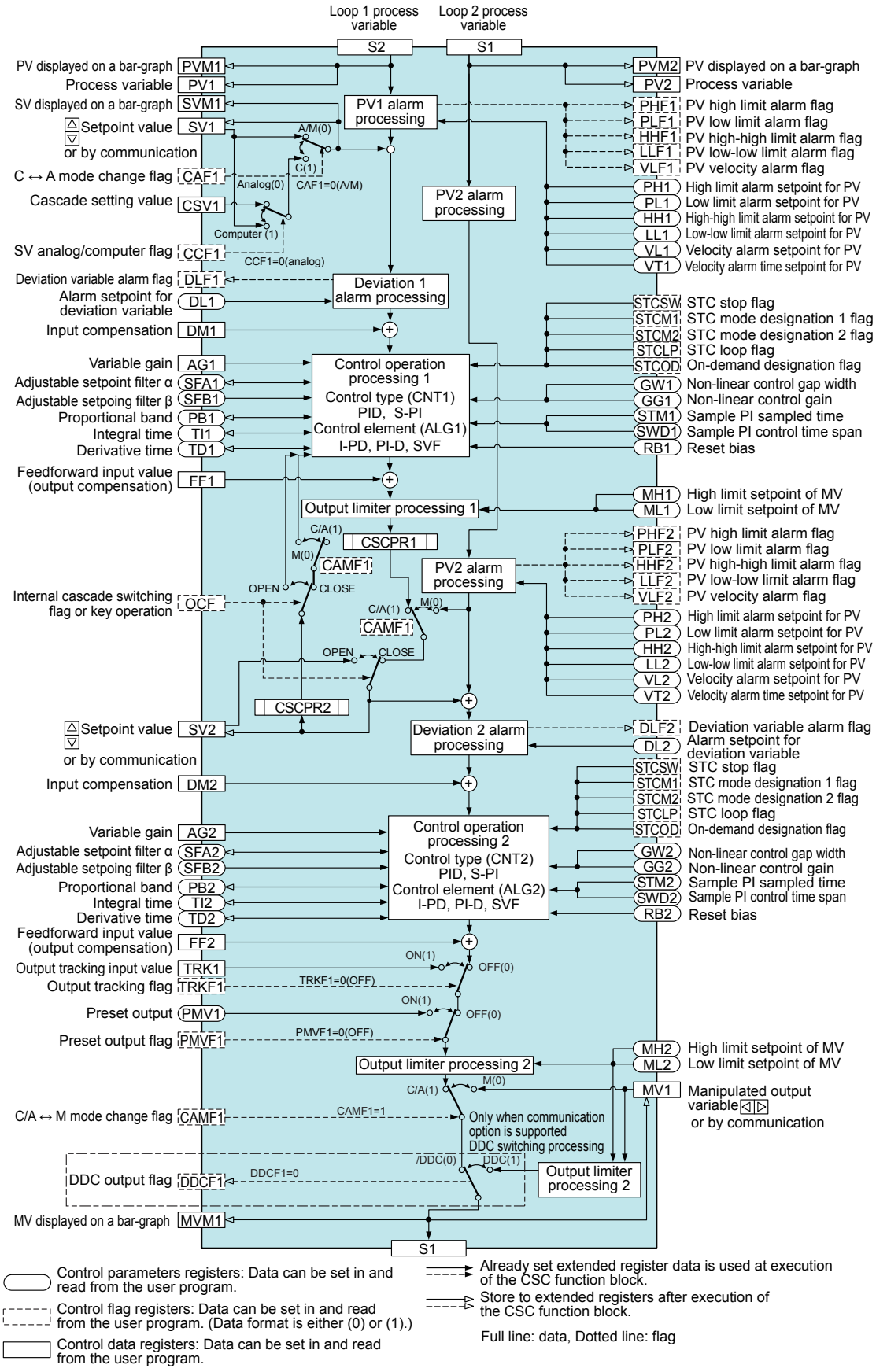


CSC Function Block Overview

5.3.1 Cascade (CSC) Control Module

The figure below is the CSC function block including control elements (CNT1) and the extended function registers. S1 registers and extended function registers can be regarded as the signal terminals of the controller "CSC," and the specified functions are demonstrated by "wiring" data to these terminals.

► S1 registers: "4.2 Principles of Computation"



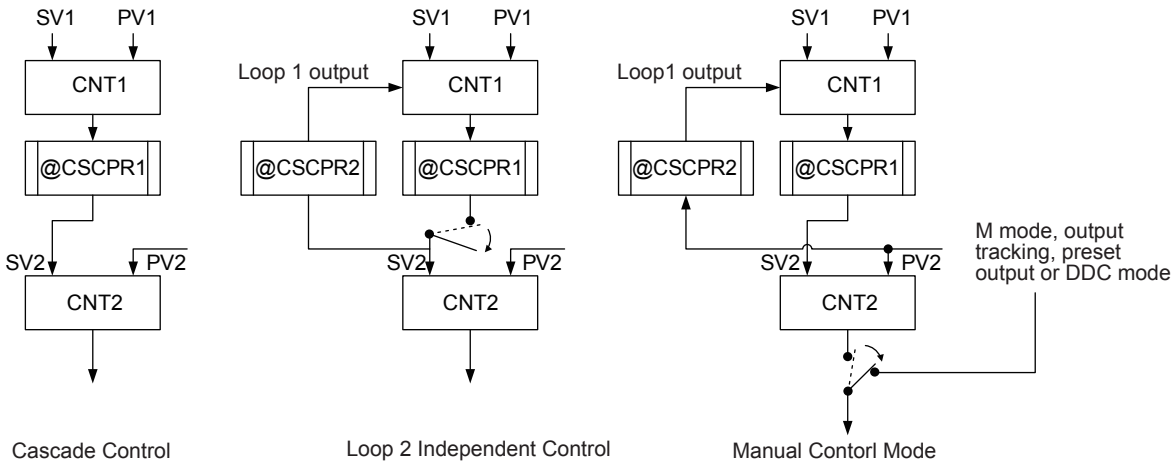
Details of CSC Function Block

Computation between cascade loops

With the CSC control modules, operation formulas are inserted between loop 1 (CNT1) and loop 2 (CNT2). Inter-loop operations comprise two subprograms, @CSCPR1 and @CSCPR2. The table below shows the roles of each subprogram.
Subprograms need to be programmed only when inter-loop operations are required. Otherwise, they are not required.

Both subprograms of @CSCPR1 and @CSCPR2 used or unused, @CSCPR2 only used:

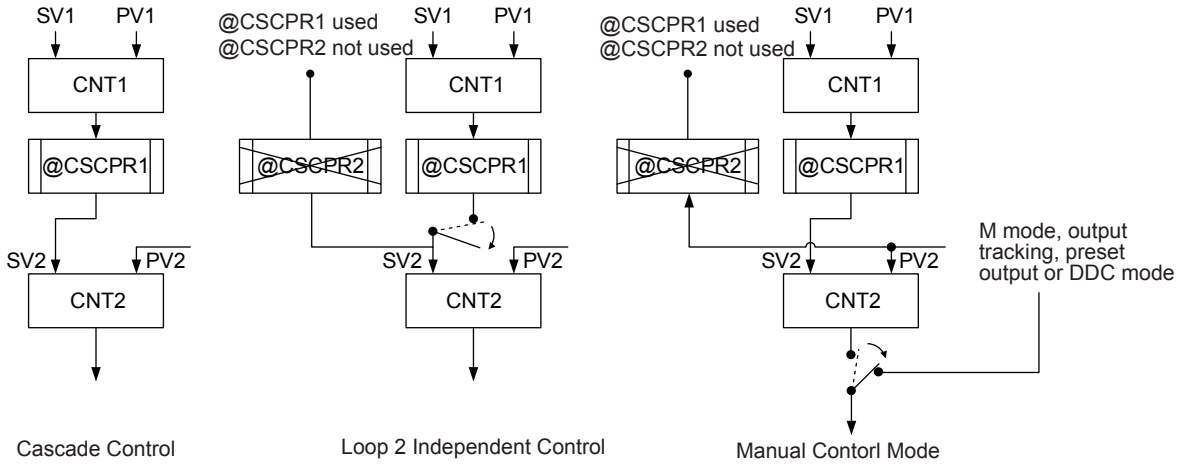
In cascade control, the output of CNT1 is used for calculation of @CSCPR1, and is input as the SV value (SV2) of CNT2.
In loop 2 independent control, the SV2 is used for calculation of @CSCPR2, and is input as the output value of CNT1.
In the manual control (M mode) in a cascade closed status, the PV2 is used for calculation of @CSCPR2, and is input as the output value of CNT1. In the manual control (M mode) in a cascade open status, the same operation as in loop 2 independent control is performed for calculation of @CSCPR2.
If @CSCPR1 includes a non-linear computation (e.g. limiter, selector), tracking sometimes is not possible in computation of @CSCPR2. In this case, too, balancing is required for switching from an open to a closed status. One balancing method is to switch to the manual control (M mode) once to set to a cascade closed status and switch back to the automatic control (A mode).



Computation between Cascade Loops in Each Mode

Only @CSCPR1 used, @CSCPR2 not used:

In this case, output for tracking SV2 is not input to CNT1 when the cascade is open. So, balancing is required when the cascade open status is switched to a closed status. One balancing method is to switch to the manual control (M mode) once to set to a cascade closed status and switch back to the automatic control (A mode).



Computation between Cascade Loops in Each Mode When @CSCPR2 is Unused

Table: Inter-loop Computation Subprogram

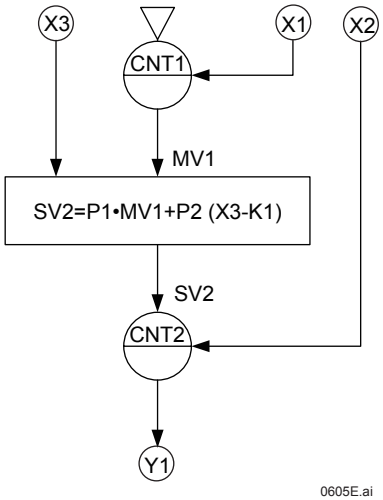
Subprogram	Application	S1 Internal Data at a Jump to the Subprogram	S1 Internal Data at a Return from the Subprogram
@CSCPR1	Computation between cascade loops when cascade is closed	Loop1 output	SV2
@CSCPR2 (Note 1)	(1) Computation for tracking loop 1 output to setpoint value of loop 2 when cascade is open	SV2	Loop 1 output
	(2) Computation of loop 1 output in M mode when cascade is closed (Note 2)	PV2	Loop 1 output

Note 1: The two applications of @CSCPR2 are switched automatically according to the operation mode.
Note 2: Equivalent to operations during output tracking, preset MV output operations and direct operation of the manipulated output variable from the host system (DDC mode)

Application program

The following explains an example program using the CSC control module.

Inserting a computation (operation formula) between loop 1 and loop 2:
The following explains an example of executing cascade control on a YS1700 and inserting the operation formula shown in the figure between CNT1 and CNT2.

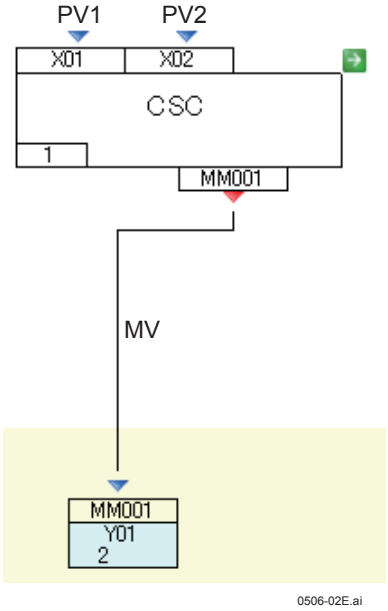


Example of Cascade Control

[Main Program]

Jump instructions for @CSCPR1 and @CSCPR2 need not be specified on the main program.

- Programming by function block programming



0506-02E.ai

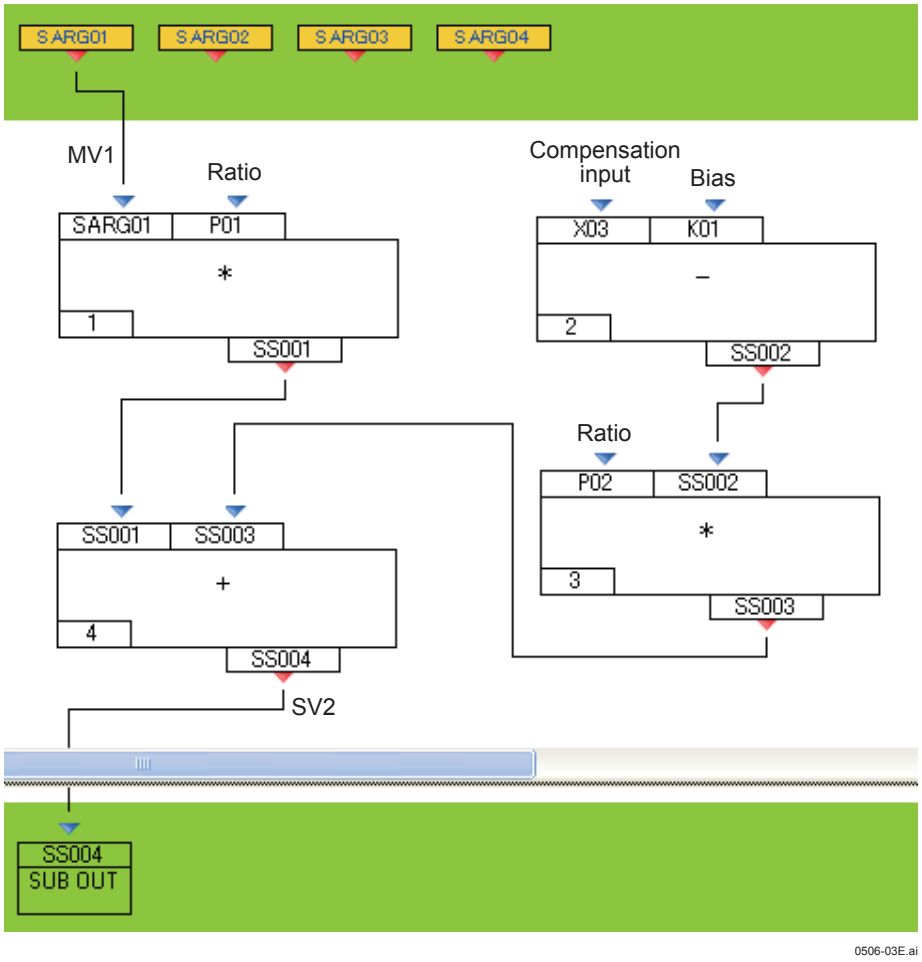
- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads process variable 1 (PV1).
LD X2	X2	X1				Reads process variable 2 (PV2).
CSC	MV1					Executes cascade control.
ST Y1	MV1					Outputs the manipulated output variable.
END						Ends program execution.

[@CSCPR1 sub-program]

Insertion computation SV2=P1•MV1+P2(X3-K1)

- Programming by function block programming



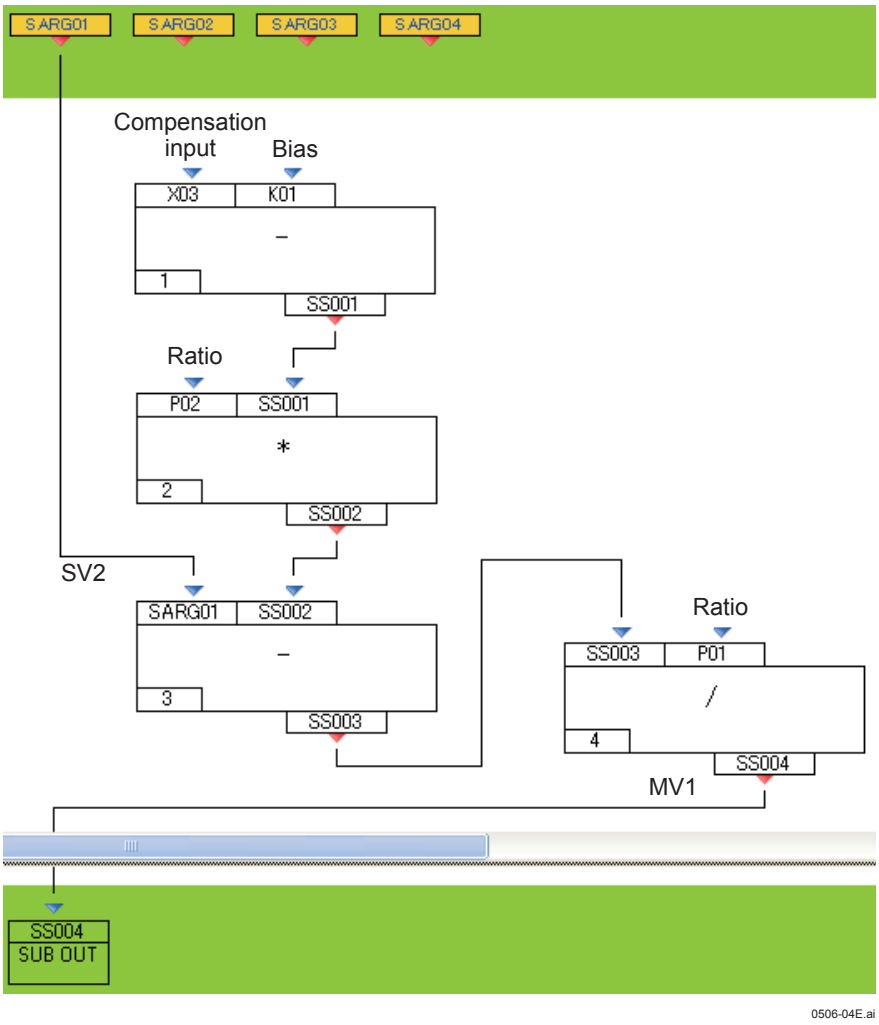
- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
SUB @CSCPR1	MV1					MV1 is currently stored to S1 at a jump.
LD P1	P1	MV1				Reads the ratio.
*	MV1*P1					Computes the ratio.
LD X3	X3	MV1*P1				Reads the compensation input.
LD K1	K1	X3	MV1*P1			Reads the bias.
-	X3-K1	MV1*P1				Computes the bias.
LD P2	P2	X3-K1	MV1*P1			Reads the ratio.
*	P2(X3-K1)	MV1*P1				Computes the ratio.
+	MV1*P1+P2(X3-K1)					Computes SV2.
RTN	Same as above					SV2=MV1•P1+P2(X3-K1)

[[@CSCPR2 sub-program]

Program the inverse computation of @CSCPR1 to @CSCPR2. Deform the operation formula of @CSCPR1 to calculate MV1.
MV1=(SV2-P2(X3-K1))/P1

- Programming by function block programming

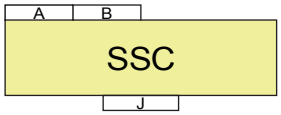


- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
SUB @CSCPR2	SV2					SV2 is currently stored to S1 at a jump.
LD X3	X3	SV2				Reads the compensation input.
LD K1	K1	X3	SV2			Reads the bias.
-	X3-K1	SV2				Computes the bias.
LD P2	P2	X3-K1	SV2			Reads the ratio.
*	P2(X3-K1)	SV2				Computes the ratio.
-	SV2-P2(X3-K1)					
LD P1	P1	SV2-P2(X3-K1)				Reads the ratio (P1).
/	(SV2-P2(X3-K1))/P1					Computes MV1.
RTN	Same as above					MV1=(SV2-P2(X3-K1))/P1

5.4 How to Use the Selector (SSC) Control Module

This control module executes autoselector control on a single YS1000. Autoselector control selects and outputs the smallest or largest value from three manipulated signals, the signal output from loop1, the signal output from loop2, and the selector external signal. The output selection signal can also be used to execute selection control for outputting any one signal of the three manipulated signals.

Name of Computation	Computing Module	Computation Instruction Symbol
Selector control module		SSC

Operation

Selector control has two operation modes, autoselector control and manual selection control, which are switched by the selector control switch (SSW).

► Detailed operation according to SSW setting: "Chapter 6 Applied Usage of Control Modules"

(SSW=0: autoselector)

To set the selector control switch (SSW) in the extended function registers of the user program, set SSW to "0."
Specify the high and low selectors of selector operation by autoselector specification (ATSEL) in Configuration Display 1 (CONFIG1).
By autoselector control, the control element on the non-selected side stands by in proportional operation (gain * deviation), and switches to bumpless operation according to changes in the processing circumstances.

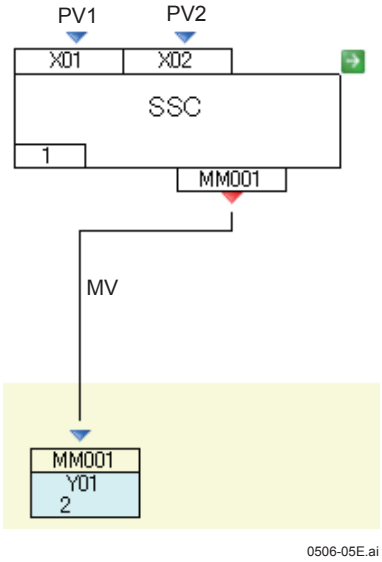
(SSW = 1, 2, 3: manual selection control)

Set the number of the manipulated variable to output to the SSW to select the output.
1=Output of CNT1
2=Output of CNT2
3=Output of external signal
Also, in manual selection control, the control element on the non-selection side tracks the manipulated output variable, and switches to bumpless control when the switching signal is received.

(SSW=4: autoselector (slave))

This setting is used to perform autoselector control on two YS1700s.
On the master, set SSW to "0," and on the slave, set SSW to "4."

<Example of function block programming>
Connect two inputs and an output to the selector control module. When computation is executed, the computation is executed on the input value (process variable), and the result (manipulated output variable) is output. The keys and indication on the controller front are automatically updated.



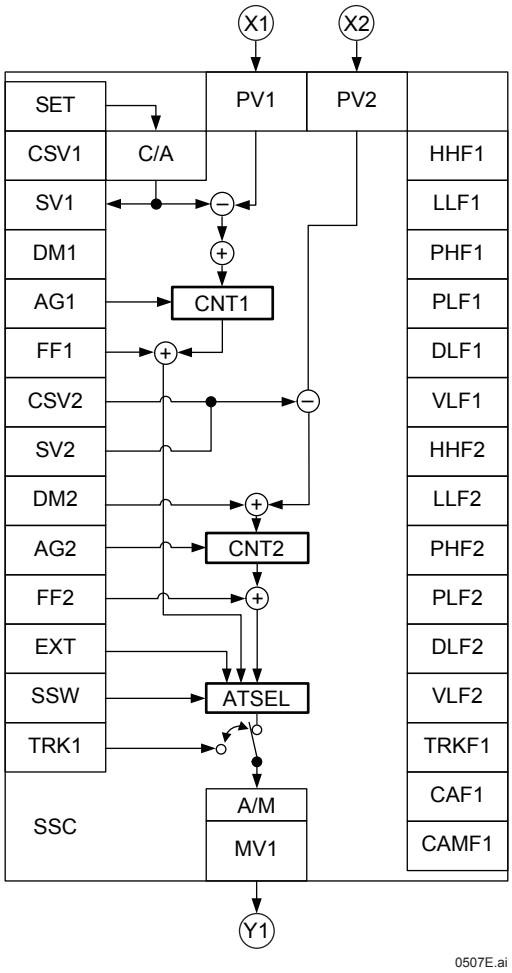
0506-05E.ai

<Example of text programming>
The input value (process variable) of loop 1 is stored to register S2 and the input value (process variable) of loop 2 is stored to register S1 before execution of SSC computation. After computation is executed, the computation result (manipulated output variable) is stored to register S1. The keys and indication on the controller front are automatically updated.

Program	S1	S2	S3	S4	S5	Explanation
LD K1	0.0					Sets the autoselector.
ST SSW	0.0					SSW=0.0
LD X1	X1					Reads input X1 (process variable).
LD X2	X2	X1				Reads input X2 (process variable).
SSC	MV1					Executes selector control. CNT1, CNT2
ST Y1	MV1					Outputs result (manipulated output variable) to Y1.

Function Block Diagram

The following shows an overview of the selector control module (SSC).

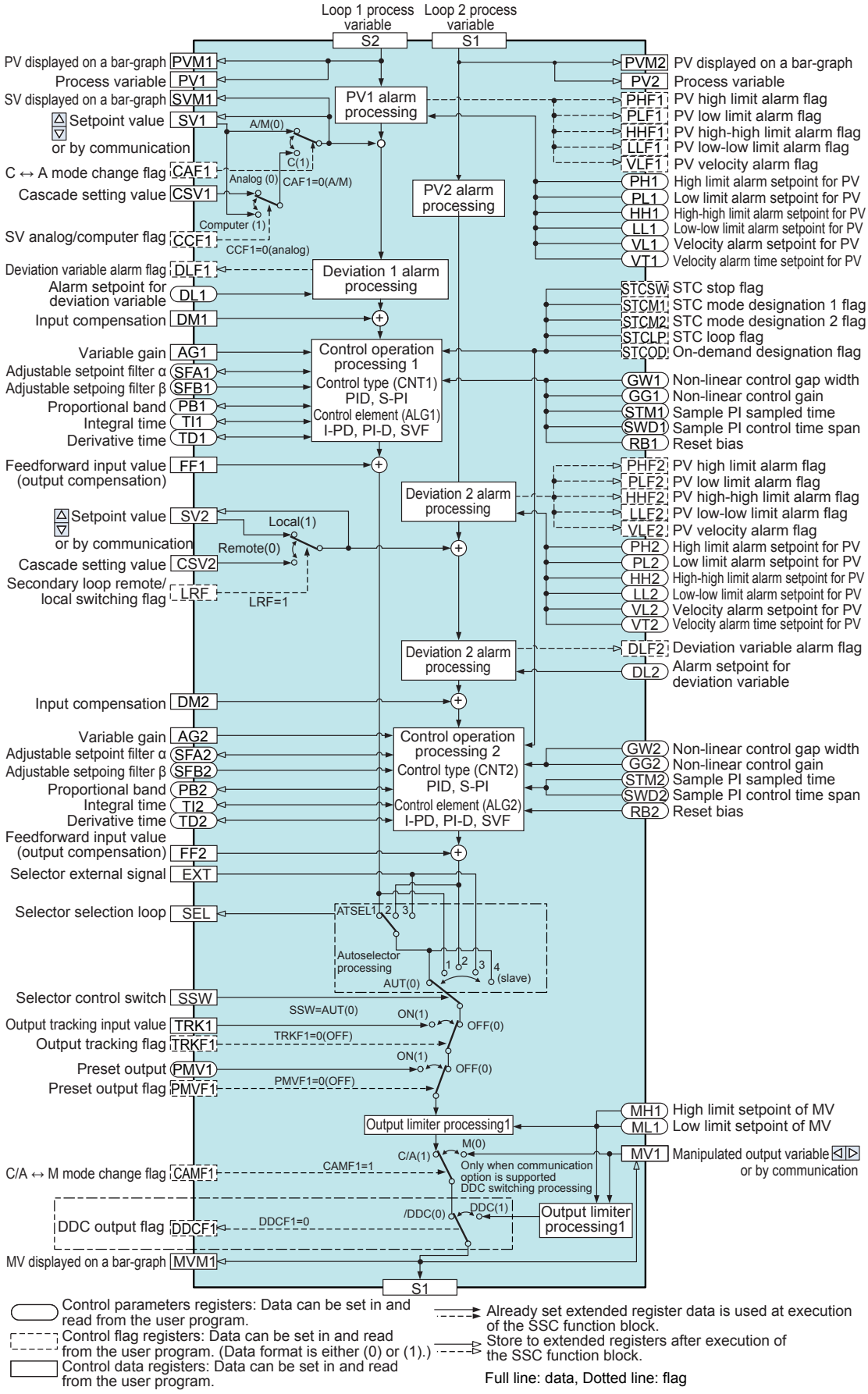


SSC Function Block Overview

5.4.1 Selector (SSC) Control Module

The figure below is the SSC function block including control elements (CNT1) and the extended function registers. S1 registers and extended function registers can be regarded as the signal terminals of the controller "SSC," and the specified functions are demonstrated by "wiring" data to these terminals.

► S1 registers: "4.2 Principles of Computation"

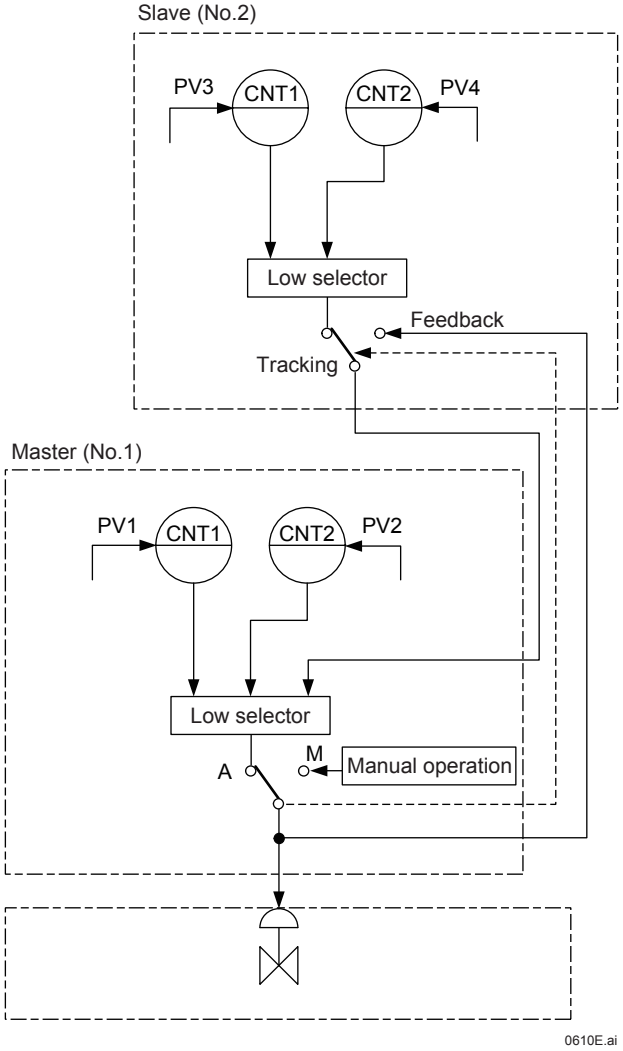


Details of SSC Function Block

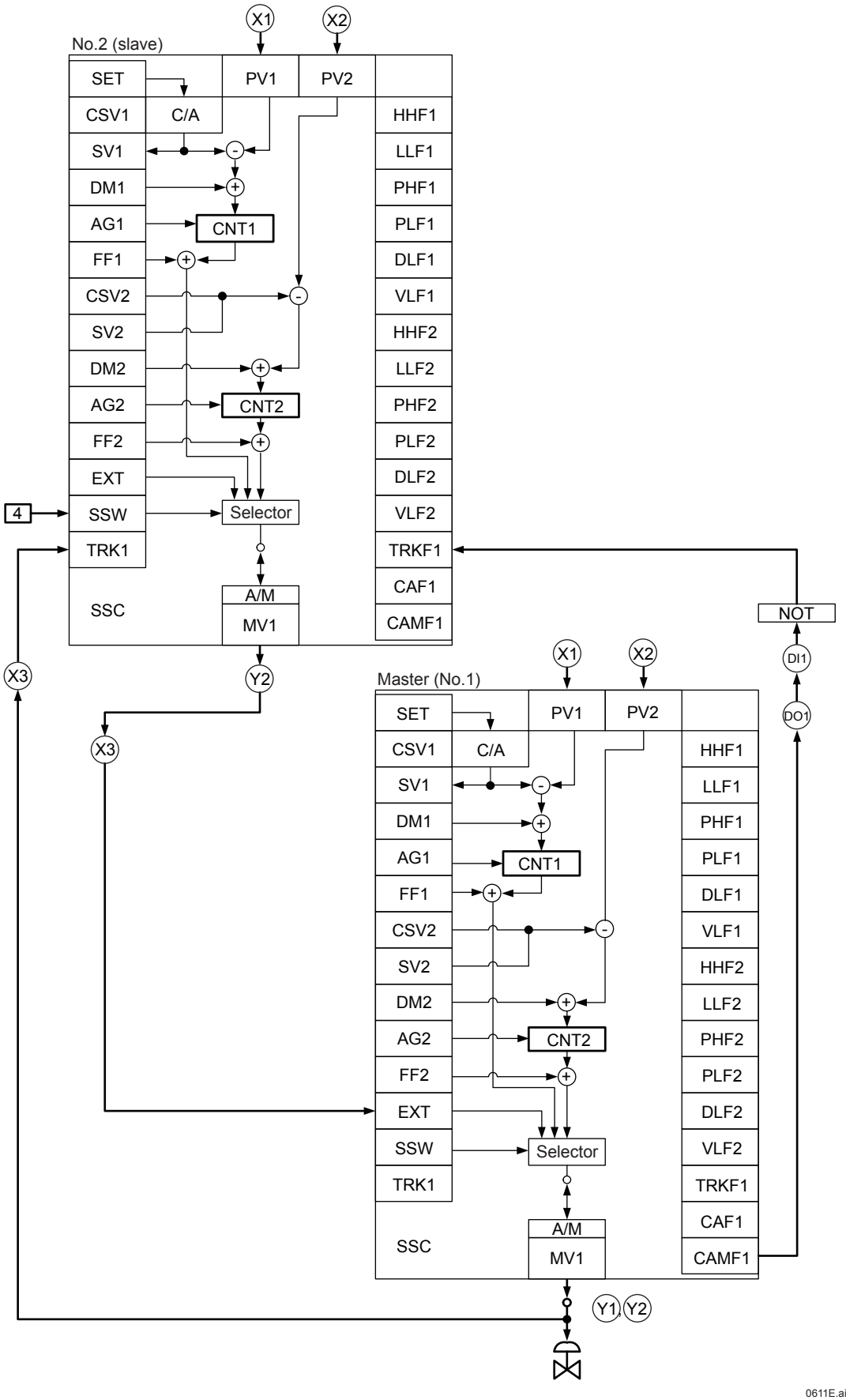
Autoselector control system of three or more loops

When two YS1700 units are combined, you can configure an Autoselector control system of 3 or 4 loops. At this time, set the SSW on the slave controller to "4," and set the SSW on the master controller to AUT (0).

The figure below shows the configuration of selector control on two controllers.



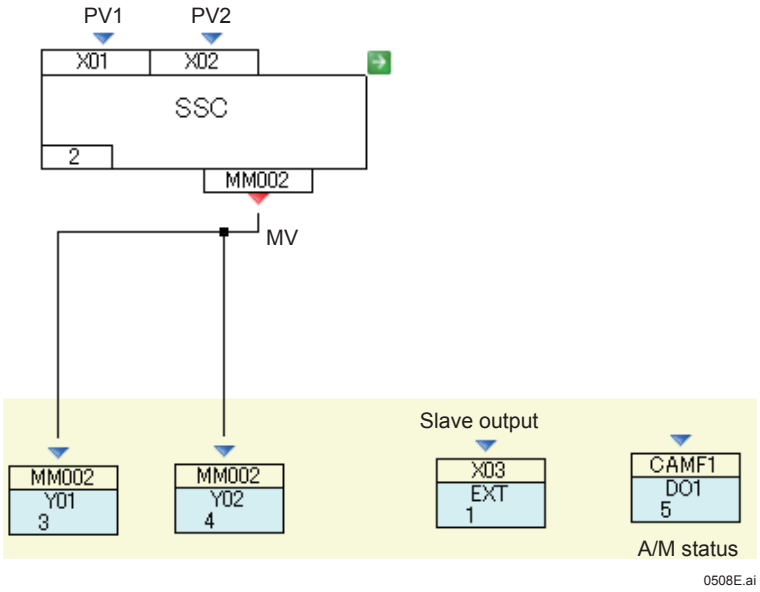
Configuration of Multi-loop Autoselector Control



Function Block Overview of Multi-loop Autoselector Control

[Master (No.1) program]

- Programming by function block programming

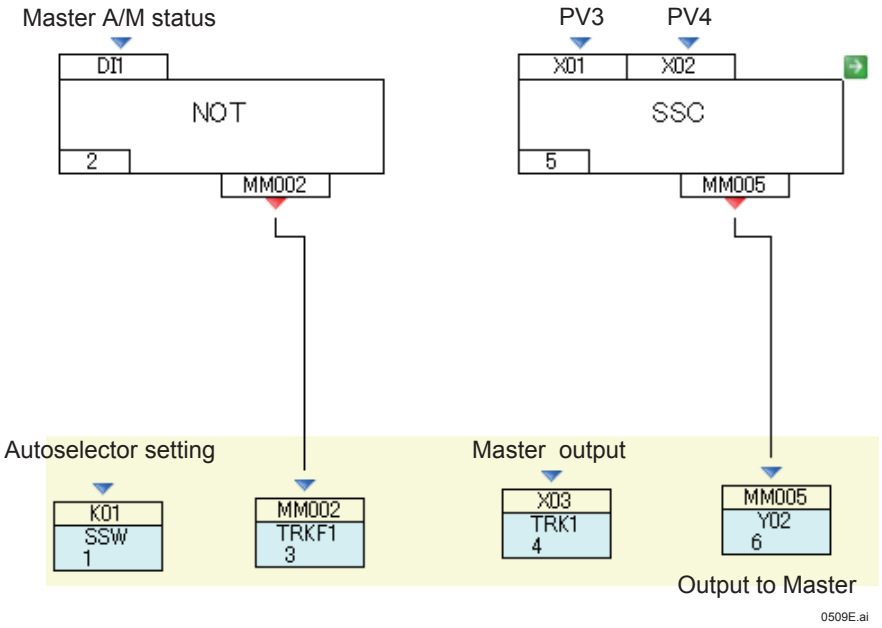


- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X3	X3					Reads slave output as an external signal.
ST EXT	X3					Stores to terminal (EXT).
LD X1	X1	X3				Reads PV1.
LD X2	X2	X1	X3			Reads PV2.
SSC	MV1	X3				Executes Autoselector control.
ST Y1	MV1	X3				Outputs to the final control element.
ST Y2	MV1	X3				Returns control to the slave.
LD CAMF1	0/1	MV1	X3			Reads the mode change flag.
ST DO1	0/1	MV1	X3			Outputs the A/M status to the slave.
END						Ends program execution.
CNT1, CNT2=PID (standard PID control) SSW=0 (Autoselector) Do not program this as the SSW register default is "0." ATSEL=LO (Low selector)						

[Slave (No.2) program]

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD K1	4.0					Sets the Autoselector on the slave.
ST SSW	4.0					SSW=4.0
LD DI1	0/1	4.0				Master A/M signal (This is tracked by the slave when the master is in manual control.)
NOT	1/0	4.0				
ST TRKF1	1/0	4.0				
LD X3	X3	1/0	4.0			Master manipulated output signal
ST TRK1	X3	1/0	4.0			
LD X1	X1	X3	1/0	4.0		Reads PV3.
LD X2	X2	X1	X3	1/0		Reads PV4.
SSC	MV1	X3	1/0	4.0	4.0	Executes Autoselector control.
ST Y2	MV1	X3	1/0	4.0		Outputs to the master.
CNT1, CNT2=PID (standard PID control) ATSEL=LO (Low selector)						

The No.2 controller (slave) sends the manipulated output variable to the No.1 controller (host). SSW is set to 4.0 (K1=400.0%) to set to Autoselector on the slave (in this case, the slave is a type that accepts feedback signals from an external device). The slave receives the A/M signal from the No.1 controller (host) to turn tracking ON if it is in the manual control. The slave also detects whether or not it is selected by this feedback signal. If the No.2 controller is not selected in the automatic control, it stands by in proportional operation.

Blank Page

6.1 Overview

Data in extended function registers (control data registers, control parameter registers, and control flag registers) is read and used for computation and alarm output, and appropriate signals are input to these registers so that their extended functions are used. Appropriate defaults that will not affect other controllers are stored to unused registers.

6.2 Extended Function Registers

6.2.1 Control Data Registers

Register (data)	Control Module				Name	Computation Effective Range	Default before Execution of Program at Control Period	Data Update after Execution of Control Module	Default
	BSC1	BSC2	CSC	SSC					
CSV1	●	–	●	●	Cascade setting value 1	0.000 to 1.000 (0.0 to 100.0%)	ON	Not updated	-8.000
DM1	●	–	●	●	Input compensation 1	-1.000 to 1.000 (-100.0 to 100.0%)			0.000
AG1	●	–	●	●	Variable gain 1	-8.000 to 8.000 (-800.0 to 800.0%)			1.000
FF1	●	–	●	●	Feedforward input value 1	-1.000 to 2.000 (-100.0 to 200.0%)			0.000
TRK1	●	–	●	●	Output tracking input value 1	0.000 to 1.000 (0.0 to 100.0%)			-8.000
CSV2	–	●	–	●	Cascade setting value 2	0.000 to 1.000 (0.0 to 100.0%)			-8.000
DM2	–	●	●	●	Input compensation 2	-1.000 to 1.000 (-100.0 to 100.0%)			0.000
AG2	–	●	●	●	Variable gain 2	-8.000 to 8.000 (-800.0 to 800.0%)			1.000
FF2	–	●	●	●	Feedforward input value 2	-1.000 to 2.000 (-100.0 to 200.0%)			0.000
TRK2	–	●	–	–	Output tracking input value 2	0.000 to 1.000 (0.0 to 100.0%)			-8.000
EXT	–	–	–	●	Selector external signal	0.000 to 1.000 (0.0 to 100.0%)			Low Selector: 8.000 High Selector: -8.000
SSW	–	–	–	●	Selector control switch	0 to 4			0
SEL	–	–	–	○	Selector selection loop	0,1,2,3		Updated	0

●: Available by read (LD) and store (ST) instructions, ○: Available only by the read (LD) instruction, –: Not available

When programming by function block programming, the read (LD) instruction is the equivalent of "wiring" to the module inputs and module parameters, and the store (ST) instruction is the equivalent of "wiring" from module outputs. "Default before execution of program at control period" indicates registers set with default data before the user program is executed at each control period. "Data update after execution of control module" indicates registers that are updated after a control module (BSCn, CSC or SSC) is executed.

Figures in parentheses "( )" for the setting range indicate data with unit appended.

Register (data)	Control Module				Name	Computation Effective Range	Default before Execution of Program at Control Period	Data Update after Execution of Control Module	Default
	BSC1	BSC2	CSC	SSC					
PV1	○	–	○	○	Process variable 1	-0.250 to 1.250 (-25.0 to 125.0%)	OFF	Updated	–
PV2	–	○	○	○	Process variable 2	-0.250 to 1.250 (-25.0 to 125.0%)			–
SV1	●	–	●	●	CNT1 setpoint value	0.000 to 1.000 (0.0 to 100.0%)			Normalization of parameter setpoint value
SV2	–	●	●	●	CNT2 setpoint value	0.000 to 1.000 (0.0 to 100.0%)			Normalization of parameter setpoint value
DV1	○	–	○	○	Deviation variable 1	–			PV1-SV1
DV2	–	○	○	○	Deviation variable 2	–			PV2-SV2
MV1	●	–	●	●	Manipulated output variable 1	-0.063 to 1.063 (-6.3 to 106.3%)			-0.063
MVM1	●	–	●	●	MV1 displayed on a bar-graph	-0.063 to 1.063 (-6.3 to 106.3%)			-0.063
PVM1	●	–	●	●	PV1 displayed on a bar-graph	-0.250 to 1.250 (-25.0 to 125.0%)			-0.200
SVM1	●	–	●	●	SV1 displayed on a bar-graph	-0.063 to 1.063 (-6.3 to 106.3%)			-0.063
MV2	–	●	–	–	Manipulated output variable 2	-0.063 to 1.063 (-6.3 to 106.3%)			-0.063
MVM2	–	●	–	–	MV2 displayed on a bar-graph	-0.063 to 1.063 (-6.3 to 106.3%)			-0.063
PVM2	–	●	●	●	PV2 displayed on a bar-graph	-0.250 to 1.250 (-25.0 to 125.0%)			-0.200
SVM2	–	●	●	●	SV2 displayed on a bar-graph	0.000 to 1.000 (0.0 to 100.0%)			0.000

●: Available by read (LD) and store (ST) instructions, ○: Available only by the read (LD) instruction, –: Not available
When programming by function block programming, the read (LD) instruction is the equivalent of "wiring" to the module inputs and module parameters, and the store (ST) instruction is the equivalent of "wiring" from module outputs. "Default before execution of program at control period" indicates registers set with default data before the user program is executed at each control period. "Data update after execution of control module" indicates registers that are updated after a control module (BSCn, CSC or SSC) is executed.
Figures in parentheses "( )" for the setting range indicate data with unit appended.

(1) Cascade setting value 1 (CSV1)

[Register]

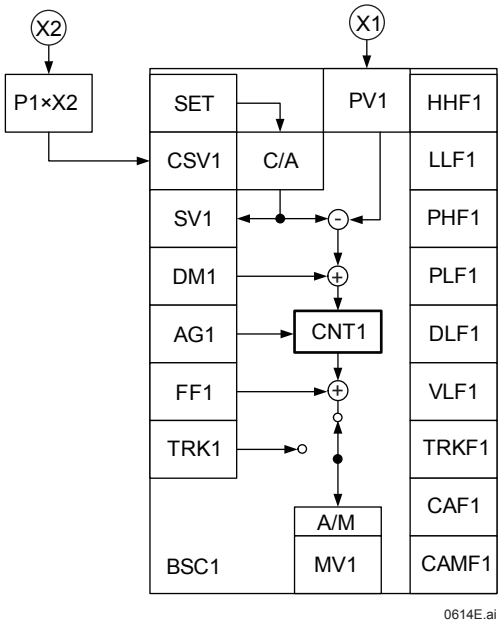
Register	Name	Function	Setting Range	Default
CSV1	Cascade setting value 1	Setpoint value of CNT1 in C mode	0.000 to 1.000 (0.0 to 100.0%)	-8.000

[Operation]

The input signal (X2) from the external device is stored to cascade setting value 1 (CSV1) before execution of the control module.
When the controller is set to the cascade setting automatic control (C) (CAS), the cascade setting value 1 (CSV1) becomes the setpoint value (SV1) of the control module. (The same applies to the cascade control module (CSC) and selector control module (SSC).) When the data of cascade setting value 1 (CSV1) is out of the setting range (-6.3 to 106.3%), the mode is not switched to the cascade setting automatic control (C) (CAS).

[Function block overview]

The figure below is an example of cascade setting value 1 (CSV1) with ratio computation. The product of the input signal (X2) from the external device and variable constant (P1) is taken as setpoint value (SV1).

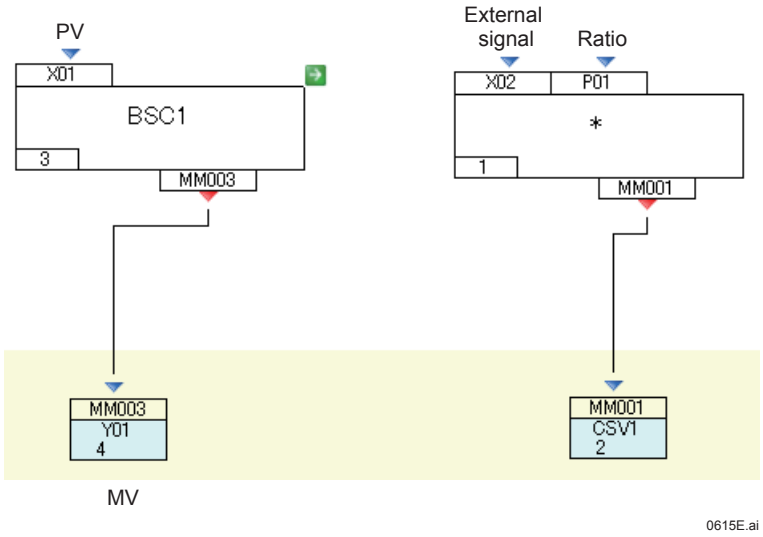


Function Block Overview of Cascade Setting Value 1 (CSV1)

[Program example]

The following shows the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X2	X2					Reads the input signal from the external device.
LD P1	P1	X2				Reads the ratio.
*	P1*X2					Computes cascade setting value 1 (CSV1).
ST CSV1	P1*X2					CSV1=P1*X2
LD X1	X1	P1*X2				Reads the input.
BSC1	MV1	P1*X2				Executes the control module.
ST Y1	MV1	P1*X2				Manipulated output variable
Next computation						

(2) Cascade setting value 2 (CSV2)/Secondary loop remote/local switching (LRF)

[Register]

Register	Name	Function	Setting Range	Default
CSV2	Cascade setting value 2	Setpoint value of CNT2 in selector control	0.000 to 1.000 (0.0 to 100.0%)	-8.000

[Flag]

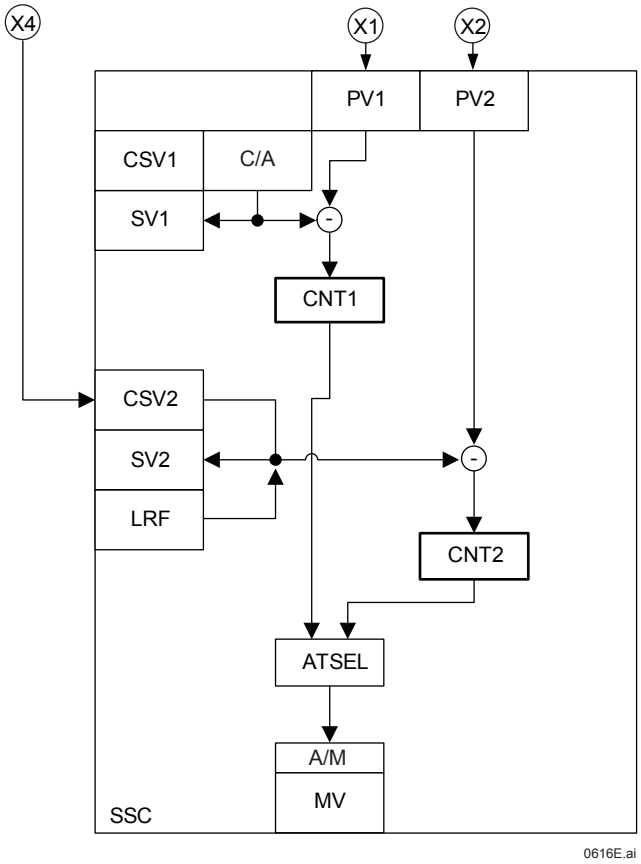
Flag	Name	Function	Setting Range	Default
LRF	Secondary loop remote/ local switching	Selection of loop 2 setpoint value of SSC ►"6.2.2 Control Flag Registers"	0: remote 1: local	Status of front panel keys

[Operation]

An external setting signal is stored to CSV2 by the selector control module (SSC), and the secondary loop remote/local switching flag (LRF) is set to "0" (remote) to enable setting of the loop 2 setpoint value from the external device. At this time, the cascade setting automatic control (C mode) is indicated on the loop 2 display on the YS1700. Also, the SV2 is displayed but cannot be changed. The setpoint value of SV2 can be changed by setting LRF to "1" (local) to set loop 2 to the automatic control (A mode).

[Function block overview]

The figure below is an example of the loop 2 setting value (CSV2) based on the secondary loop remote/local switching flag (LRF).



Function Block Overview of Cascade Setting Value 2 (CSV2)

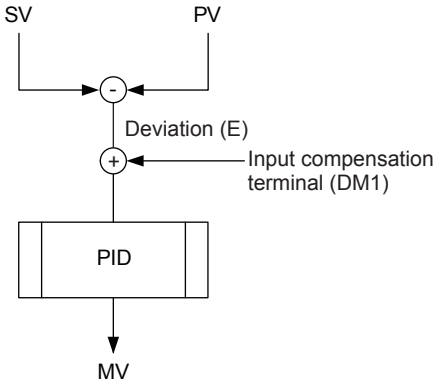
(3) Input compensation (DMn)

[Registers]

Register	Name	Function	Setting Range	Default
DM1	Input compensation 1	Addition of data to the deviation of CNT1	-1.000 to 1.000 (-100.0 to 100.0%)	0.000
DM2	Input compensation 2	Addition of data to the deviation of CNT2		

[Operation]

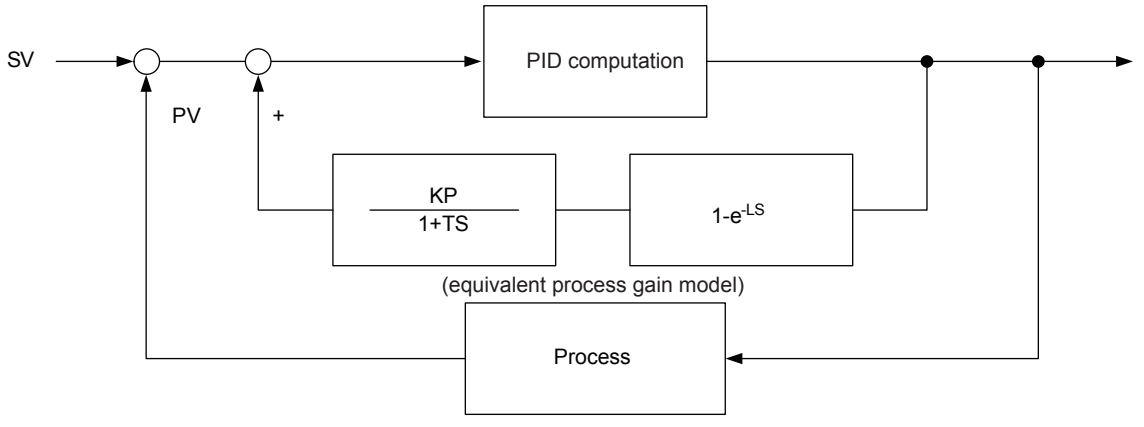
Data stored to input compensation (DMn) is added to the deviation (En=SVn-PVn). This is useful, for example, for Smith dead time compensation control. (n=1, 2)
When the basic control module (BSC2), cascade control module (CSC) and selector control module (SSC) are used, each of the two loops has respective input compensations (DM1 and DM2).



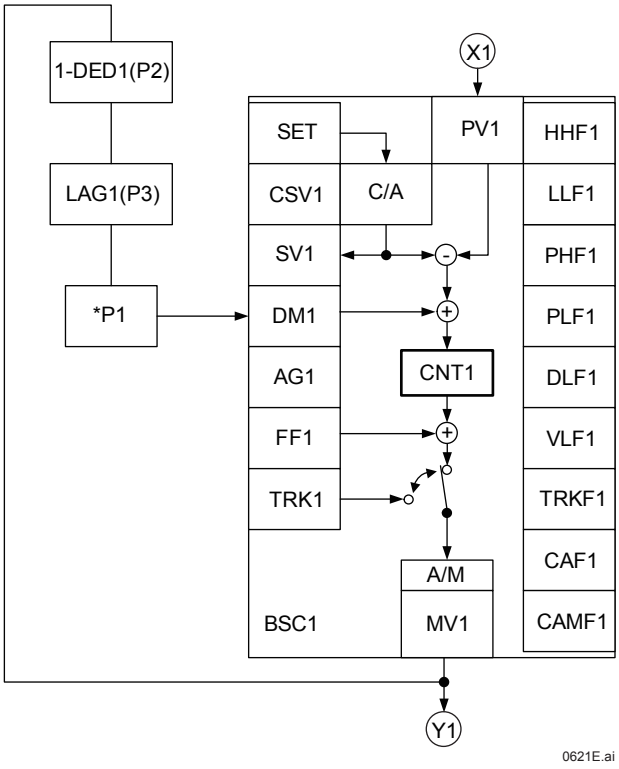
Concept of Input Compensation (DMn)

[Function block overview]

The function block overview shows a control example on a system having dead time using input compensation (DMn). In this example, input compensation (DM1) is used. Note, however, that the gain, dead time and first order lag time of the control target should be fully ascertained to configure an equivalent process gain model.



Example of Control on a System Having Dead Time

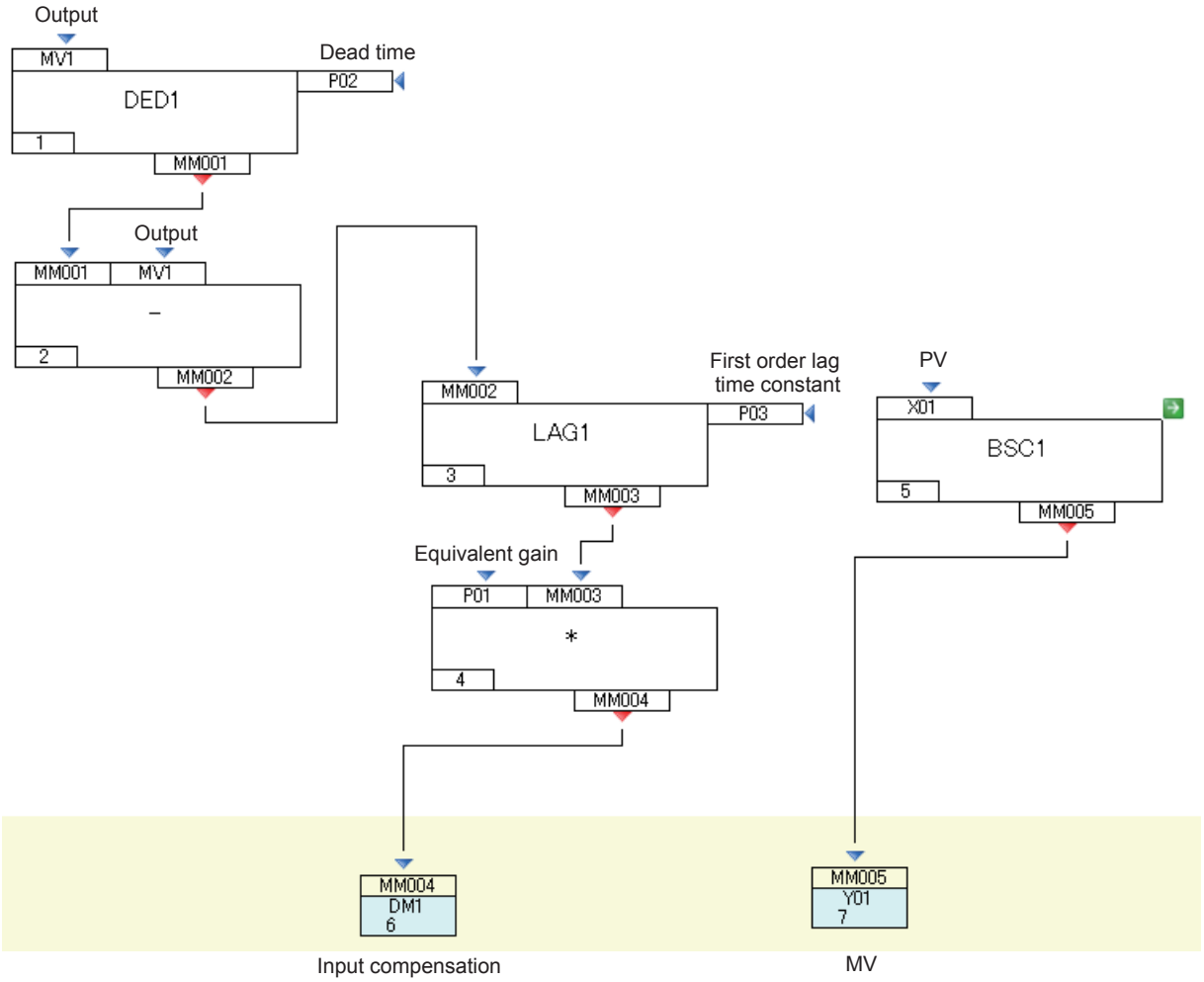


Function Block Overview of Dead Time Compensation

[Program example]

The following shows the program in the above figure.

- Programming by function block programming



● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD Y1	Y1					Reads the output.
LD Y1	Y1	Y1				Reads the output.
LD P2	P2	Y1	Y1			Sets the dead time (L=P2).
DED1	Y1(t-L)	Y1				Executes the dead time computation. Value of Y1 L seconds before
-	a					Compensates the dead time. $a=MV(1-e^{-LS})$
LD P3	P3	a				Sets the first order lag constant.
LAG1	b					Computes the first order lag. b=1/(1+TS)
LD P1	P1	b				Sets the KP of the equivalent gain to $KP(1-e^{-LS})MV1/(1+TS)$.
*	P1*b					
ST DM1	P1*b					Inputs to the input compensation (DM1) terminal.
LD X1	X1	P1*b				Reads the input.
BSC1	MV1	P1*b				Executes the control module.
ST Y1	MV1	P1*b				Manipulated output variable
Next computation						

(4) Variable gain (AGn)

[Registers]

Register	Name	Function	Setting Range	Default
AG1	Variable gain 1	Multiplication on the proportional gain of CNT1	-8.000 to 8.000 (-800.0 to 800.0%)	1.000
AG2	Variable gain 2	Multiplication on the proportional gain of CNT2		

[Operation]

Variable gain (AGn) is a coefficient for multiplying the proportional band (PB). So, proportional gain can be changed in the program. (n=1, 2)

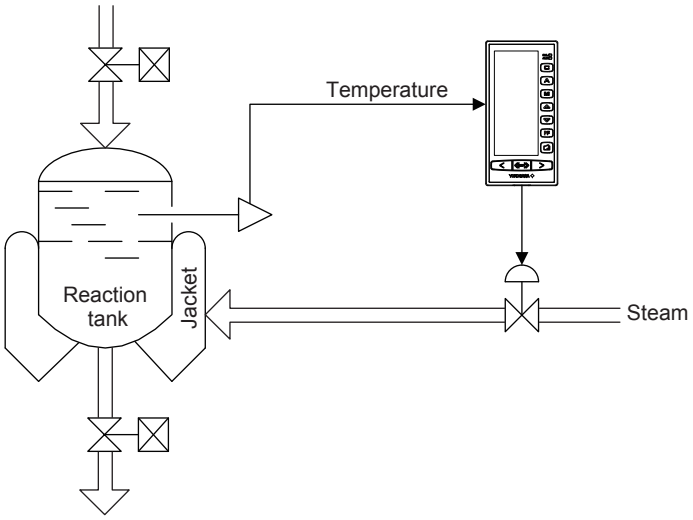
$$\text{Proportional gain} = \frac{100}{\text{Proportional band (PB)}} \times (\text{Variable gain (AGn)})$$

0623E.ai

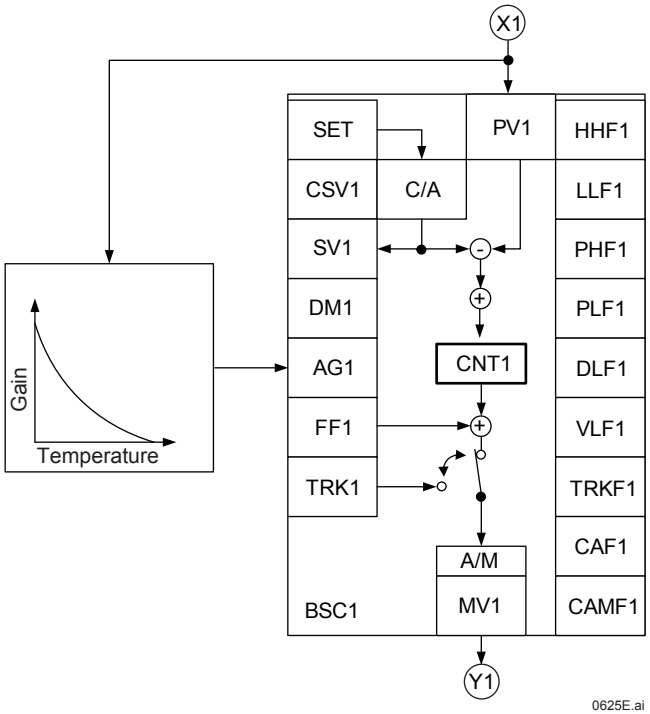
According to the above formula, if variable gain (AG1) is taken to be "2.0" when the proportional band (PB) is "50%," the proportional gain becomes "4" (proportional band (PB)=25%).

[Function block overview]

The function block overview shows a control example of a thermochemical reaction in which the reaction speed (process gain) changes according to the temperature.
As the reaction speed is slow in a low-temperature chemical reaction, increase the variable gain (AGn) to speed up the corrective action.
As the reaction speed is fast in a high-temperature chemical reaction, decrease the variable gain (AGn) to prevent hunting or overshooting.



Thermochemical Reaction Control

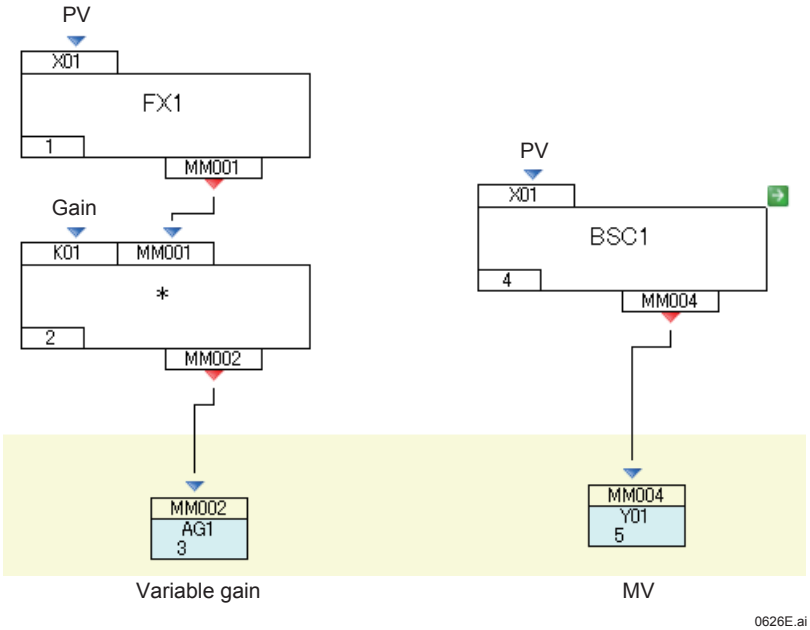


Function Block Overview of Thermochemical Reaction Control

[Program example]

The following shows the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads the input.
FX1	f(x)					K1=3.0 Converts the temperature to gain. Gain: 0 to 3
LD K1	K1					
*	K1*f(x)					
ST AG1	K1*f(x)					Inputs to the variable gain (AG1) terminal.
LD X1	X1	K1*f(x)				Reads the input.
BSC1	MV1	K1*f(x)				Executes the control module.
ST Y1	MV1	K1*f(x)				Manipulated output variable
Next computation						

(5) Feedforward input value (output compensation) (FFn)

[Registers]

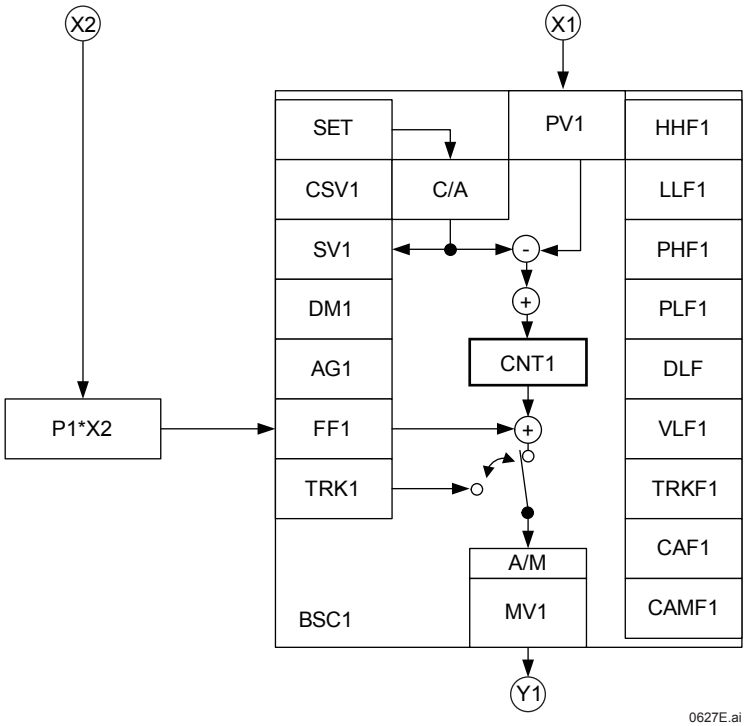
Register	Name	Function	Setting Range	Default
FF1	Feedforward input value 1 (Output compensation 1)	Addition of data to the manipulated output variable of CNT1	-1.000 to 2.000 (-100.0 to 200.0%)	0.000
FF2	Feedforward input value 2 (Output compensation 2)	Addition of data to the manipulated output variable of CNT2		

[Operation]

Output compensation (FFn) is used for feedforward control. (n=1, 2)
In the cascade setting automatic control (C mode) and automatic control (A mode), the signal stored to output compensation (FFn) is added to the manipulated output variable (MV).
In the manual control (M mode), manual operation has nothing to do with output compensation (FFn). Also, switching between the automatic control (A mode) and manual control (M mode) is performed bumplessly.

[Function block overview]

The feedforward signal generally undergoes compensation (e.g. gain multiplication and phase computation) before it is input to the output compensation (FF1) terminal. The following shows a function block overview of feedforward control.

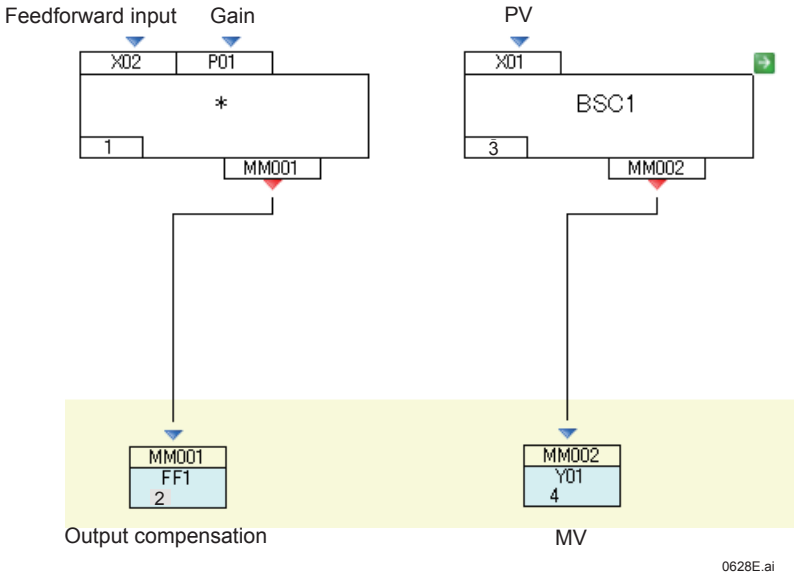


Function Block Overview of Feedforward Control

[Program example]

The following illustrates the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X2	X2					Reads the feedforward signal.
LD P1	P1	X2				Sets the gain.
*	FF1					Computes the gain. (FF1=P1*X2)
ST FF1	FF1					Stores to the output compensation (FF1) terminal.
LD X1	X1	FF1				Reads the input.
BSC1	MV1	FF1				Executes the control module.
ST Y1	MV1	FF1				Manipulated output variable
Next computation						

(6) Output tracking input value (TRKn, TRKFn)

[Registers]

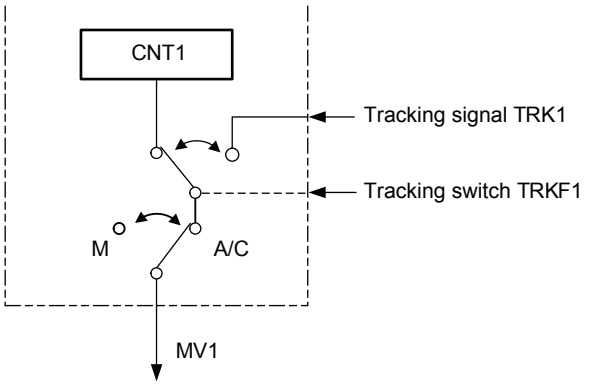
Register	Name	Function	Setting Range	Default
TRK1	Output tracking input value 1	Output of control module when TRKF1=1	0.000 to 1.000 (0.0 to 100.0%)	-8.000
TRK2	Output tracking input value 2	Output of control module when TRKF2=1		

[Flags]

Flags	Name	Function	Setting Range	Default
TRKF1	Output tracking 1 flag	Selection of control operation output or external signal output	0: Control operation output 1: External signal output	0
TRKF2	Output tracking 2 flag	Selection of control operation output or external signal output		

[Operation]

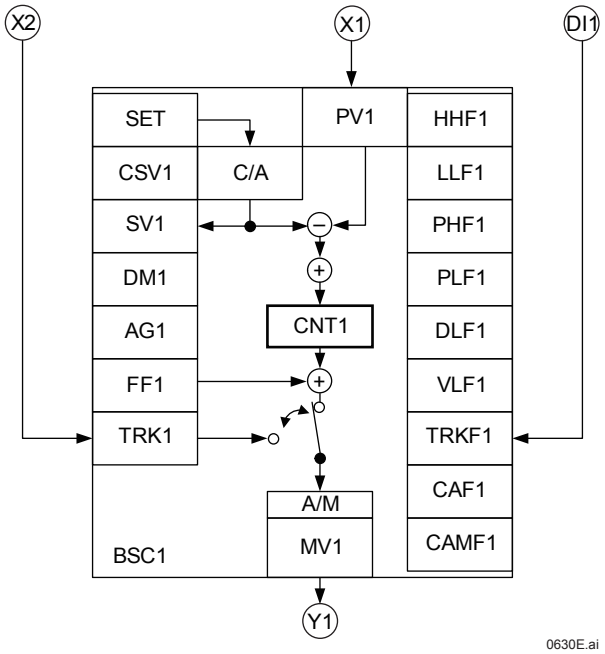
In the cascade setting automatic control (C mode) and automatic control (A mode), when output tracking flag (TRKFn) is "1," the tracking signal (output tracking input value (TRKn)) from the external device is output as the output of the control module.
In the manual control (M mode), manual operation has nothing to do with output tracking input value (TRKn). (n=1, 2)



Concept of Output Tracking (TRKn)

[Function block overview]

The following shows a function block overview of output tracking input value (TRKn). The input signal (X2) from the external device and digital input (DI1) are taken as the tracking signal (output tracking input value (TRKn)) and tracking switch (output tracking flag (TRKFn)), respectively.

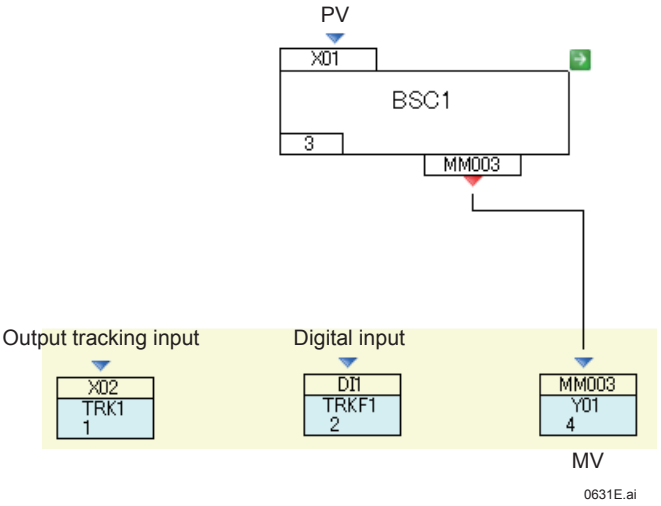


Function Block Overview of Output Tracking (TRKn)

[Program example]

The following illustrates the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X2	X2					Reads the tracking signal.
ST TRK1	X2					Stores to the output tracking (TRK1) terminal.
LD DI1	0/1	X2				Reads the digital input (DI1).
ST TRKF1	0/1	X2				Stores to the output tracking flag (TRKF1).
LD X1	X1	0/1	X2			Reads the input.
BSC1	MV1	0/1	X2			Executes the control module.
ST Y1	MV1	0/1	X2			Manipulated output variable
Next computation						

(7) Selector control (EXT, SSW, SEL)

[Registers]

Register	Name	Function	Setting Range	Default
EXT	Selector external signal	External manipulated signal in selector control	0.000 to 1.000 (0.0 to 100.0%)	Low selector: 8.000 High selector: -8.000
SSW	Selector control switch	Selection of Selector function	0: Automatic 1: Loop 1 selected 2: Loop 2 selected 3: External signal selected 4: Slave selected	0
SEL	Selector selection loop	In selector control, the currently selected loop.	0: Loop 1/2, Other than external (When manual mode) 1: Loop 1 selected 2: Loop 2 selected 3: External signal selected When SSW is 4 and output is from the master side, 1 or 2 can be read. If output is from the slave side, 3 can be read.	0

[Operation]

Autoselector control or selection control operates by setting data to the selector control switch (SSW). With Autoselector control, the smallest or largest value is selected from three manipulated signals, the signal output from loop 1, the signal output from loop 2, and the selector external signal (EXT), and is output. With selection control, the output selection signal is used to output any one of these three manipulated signals. The currently selected loop is output to the selector selection loop (SEL).

► Autoselector control system of three loops or more: "5.4.1 Selector (SSC) Control Module"

Parameter		Selector Function	Loop 1				Loop 2			
SSW	ATSEL		Operation Mode	SV	PV	MV	Operation Mode	SV	PV	MV
AUT (0)	LO	Selection of smallest value of Auto Low Selector CNT1, CNT1 or EXT	C	CSV1	PV1	When selected, AUTO output. When not selected, MV1=MV+Kp1 x e1	C	CSV2	PV2	When selected, AUTO output. When not selected, MV2=MV+Kp2 x e2
			A	Front panel keys			A	Front panel keys		
			M			MV tracked	M			MV tracked
	HI	Auto High Selector	The maximum value is selected. The status of each item is the same as above.							
1	–	Selection of loop 1 output at all times	C	CSV1	PV1	Automatic control	C	CSV2	PV2	MV1 tracked
			A	Front panel keys			A	Front panel keys		
			M			Manual control	M			MV tracked
2	–	Selection of loop 2 output at all times	In the loop 1 tracking mode, loop 2 becomes AUTO. The status of each item is the same as above.							
3	–	Selection of external signal EXT at all times	C	CSV1	PV1	EXT signal tracked	C	CSV2	PV2	EXT signal tracked
			A	Front panel keys			A	Front panel keys		
			M			MV tracked	M			MV tracked
4	LO	Auto Low Selector (slave)	C	CSV1	PV1	When selected, AUTO output. When not selected, MV1=master MV+Kp1 x e1	C	CSV2	PV2	When selected, AUTO output. When not selected, MV2=master MV+Kp2 x e2
			A	Front panel keys			A	Front panel keys		
			M			MV tracked	M			MV tracked
	HI	Auto High Selector (slave)	The maximum value is selected. The status of each item is the same as above.							

* Kp1=loop 1 proportional gain, Kp2=loop 2 proportional gain, e1=loop 1 deviation, e2=loop 2 deviation

* "Master MV" is when TRKF1 is set to "0" by tracking input on the slave. Action is regular tracking when TRKF1=1.

(8) Process variable (PVn)

[Registers]

Register	Name	Function	Setting Range	Default
PV1	Process variable 1	Process variable of loop 1	-0.250 to 1.250 (-25.0 to 125.0%)	–
PV2	Process variable 2	Process variable of loop 2		

[Operation]

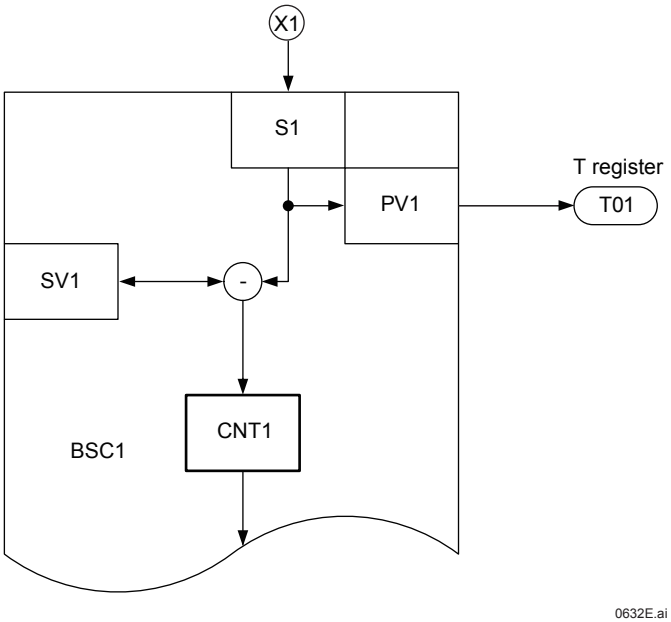
When the control module is excute, the value input to register S1 or S2 is output to PV1 and PV2.

Input Registers and Process Variable Registers

Control Module Name	Input Register	Process Variable Register
BSC1	S1	PV1
BSC2	S1	PV2
CSC	S1	PV2
	S2	PV1
SSC	S1	PV2
	S2	PV1

[Function block overview]

The following function block overview is an example of storing the control module input data into the T register.

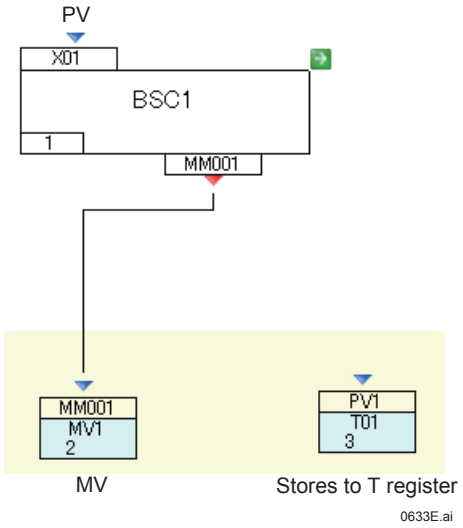


Function Block Overview of Process Variable

[Program example]

The following illustrates the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads the input.
BSC1	MV1					Executes the control module.
LD PV1	PV1	MV1				Reads the process variable.
ST T01	PV1	MV1				Stores to T01 register.

(9) Setpoint value 1 (SV1)

[Register]

Register	Name	Function	Setting Range	Default
SV1	Setpoint value 1	Setpoint value of CNT1	0.000 to 1.000 (0.0 to 100.0%)	Parameter setpoint value

[Operation]

Set the setpoint value to be used for control operation before execution of the control module. After execution of the control module, the setpoint value that was used for control operation of the control module is input to SV1.

(10) Setpoint value 2 (SV2)

[Register]

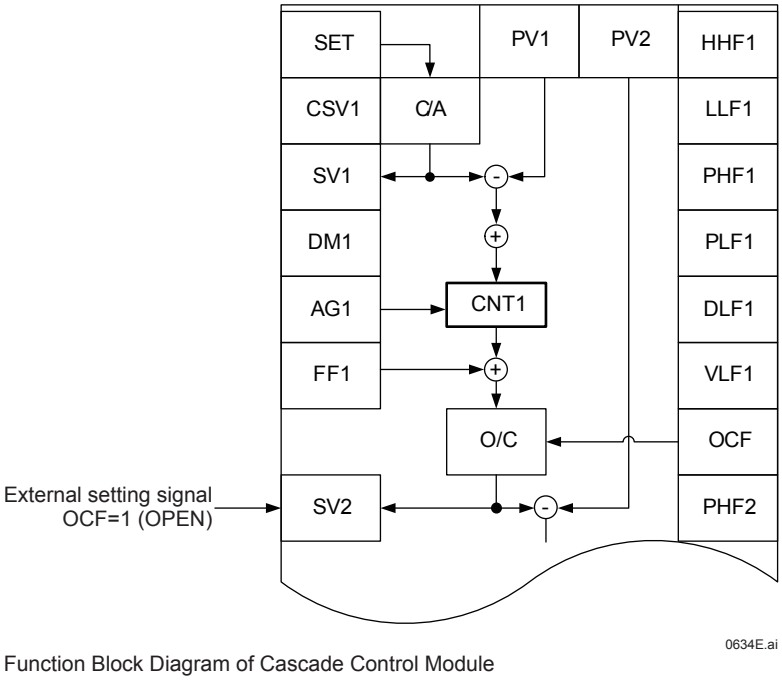
Register	Name	Function	Setting Range	Default
SV2	Setpoint value 2	Setpoint value of CNT2	0.000 to 1.000 (0.0 to 100.0%)	Parameter setpoint value

[Operation]

When the loop 1 and loop 2 of the cascade control module are isolated from each other (internal cascade switching flag OCF set to "1"), control becomes loop2 independent control, and the signal stored to SV2 becomes the setpoint value of the loop 2.
When the SV2 register is not used, SV2 can be set by the SV setting keys on YS1700.

[Function block overview]

The figure below is an example of the loop 2 setpoint value (SV2) based on the internal cascade switching flag (OCF).



Function Block Diagram of Cascade Control Module

(11) Deviation variable (DVn)

[Registers]

Register	Name	Function	Setting Range	Default
DV1	Deviation 1	PV1-SV1	–	PV1-SV1
DV2	Deviation 2	PV2-SV2	–	PV2-SV2

[Operation]

The following results are output from extended function registers PV1, SV1, PV2, and SV2:
DV1=PV1-SV1
DV2=PV2-SV2

(12) Manipulated output variable (MVn)

[Registers]

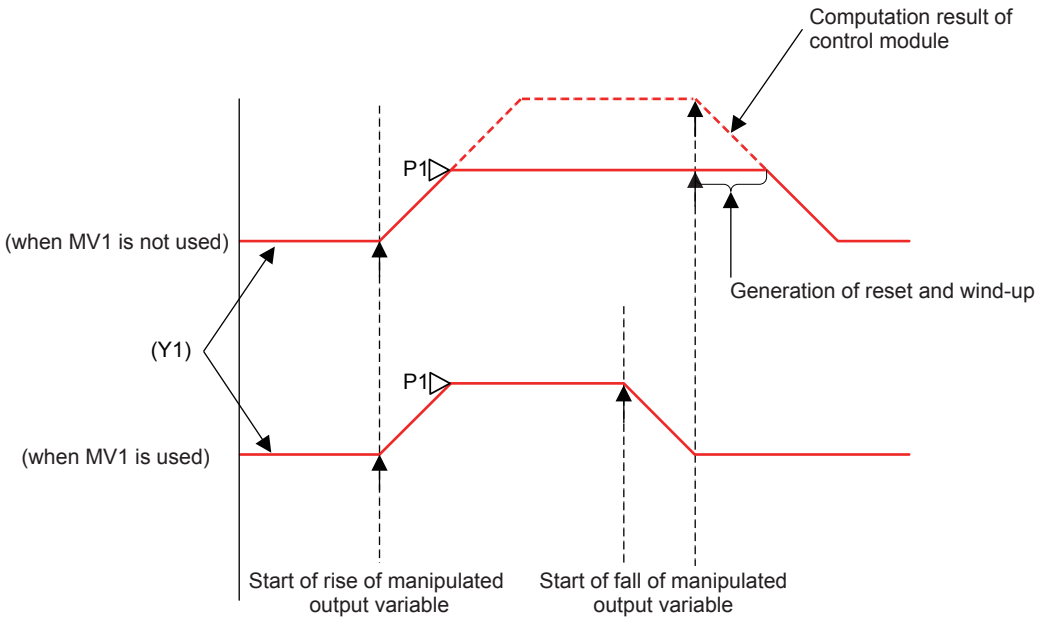
Register	Name	Function	Setting Range	Default
MV1	Manipulated output variable 1	Output of CNT1 control module	-0.063 to 1.063 (-6.3 to 106.3%)	-0.063
MV2	Manipulated output variable 2	Output of CNT2 control module		

[Operation]

The computation result of the control module is stored to manipulated output variable (MVn). (n=1, 2)
However, when data is stored to manipulated output variable n (MVn) after execution of the control module, the stored data becomes the manipulated output variable (MV).
The computation result of the control module is rewritten. (This applies to all operation modes.)
Note, however, that in the cascade setting automatic control (C mode) and automatic control (A mode), a reset and wind-up may occur if these operations are performed.

Reset and wind-up: As an example, the High Limiter (HLM) is used with the variable constant (P1) as the output limiter. High-limited (HLM) values are output to (Y1) and stored to manipulated output variable 1 (MV1) at the same time.

The following shows a concept diagram.



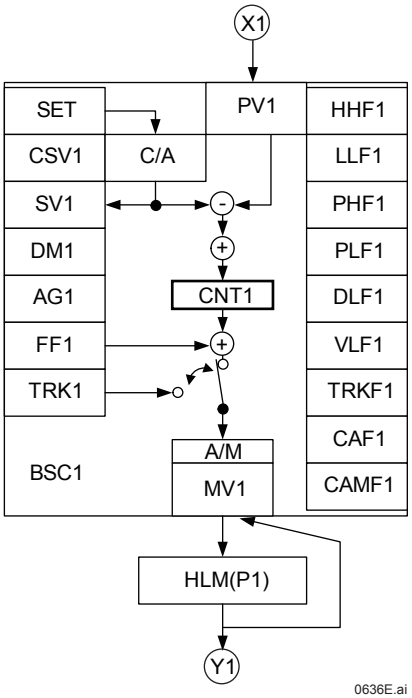
Concept Diagram of Reset and Wind-up

When decreasing output from a limit status, a reset and wind-up occurs as shown in the above figure as the manipulated output variable (MV) does not track the limit value when the limiter has been simply made to function on the manipulated output variable (MV) without the variable constant (P1) being returned to manipulated output variable 1 (MV).
In the cascade setting automatic control (C mode) and automatic control (A mode), the reset and wind-up occurs when the setting of the output limiter (high limit setpoint of MV1 (MH1)) of the control module is set to a value greater than P1.
Note, however, the reset and wind-up can be prevented by setting so that MH1 is equal to P1.

Output limiter: The output limiter operates in the cascade setting automatic control (C mode) and automatic control (A mode). It can also be made to operate in the manual control (M mode), too, by using MVn. (n=1, 2)

[Function block overview]

The figure below is a function block over of the output limiter.

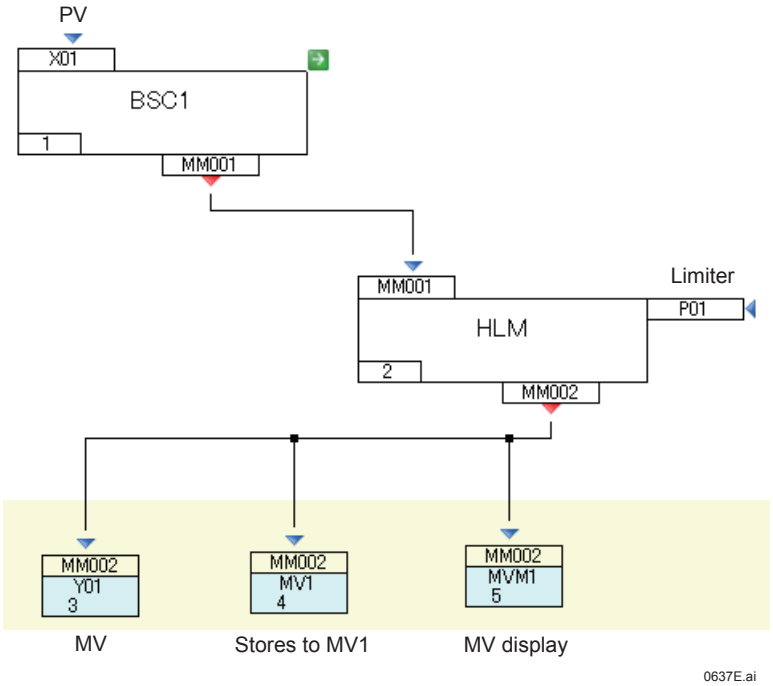


Function Block Overview of Output Limiter

[Program example]

The following illustrates the program in the above figure.

- Programming by function block programming



● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads the input.
BSC1	MV1					Executes the control module.
LD P1	P1	MV1				Reads the limiter value.
HLM	P1	MV1				High Limiter (MV1>P1)
ST MV1	P1					Stores to MV1.
ST MVM1	P1					Outputs to the display.
ST Y1	P1					Manipulated output variable
END						Ends program execution.

(13) MV displayed on a bar-graph (MVMn)

[Registers]

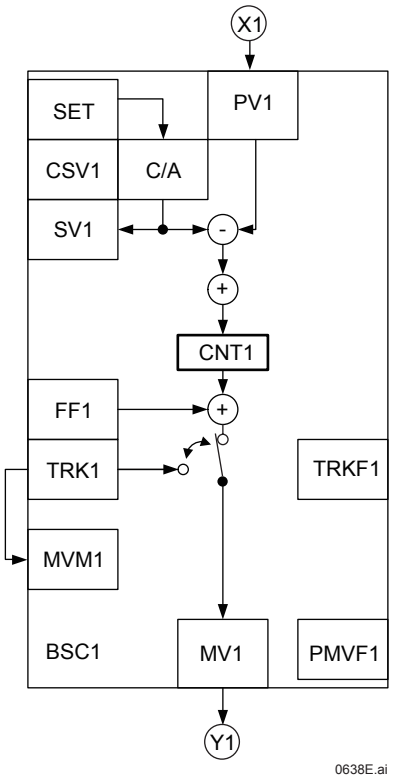
Register	Name	Function	Setting Range	Default
MVM1	MV1 displayed on a bar-graph	Display value	-0.063 to 1.063 (-6.3 to 106.3%)	-0.063
MVM2	MV2 displayed on a bar-graph	Display value		

[Operation]

With MV displayed on a bar-graph (MVM1, MVM2), any signal can be displayed (indicated) regardless of regular display items (result computed by the control module).
The MV displayed on a bar-graph is updated once by the control module. However, a desired data can be displayed by storing it in the MV displayed on a bar-graph. Be sure to store the desired display data in the MVM1 register or the MVM2 register after execution of the control module.

[Function block overview]

The currently set output tracking input value 1 (TRK1) is displayed as the MV displayed on a bar-graph (MVM1).

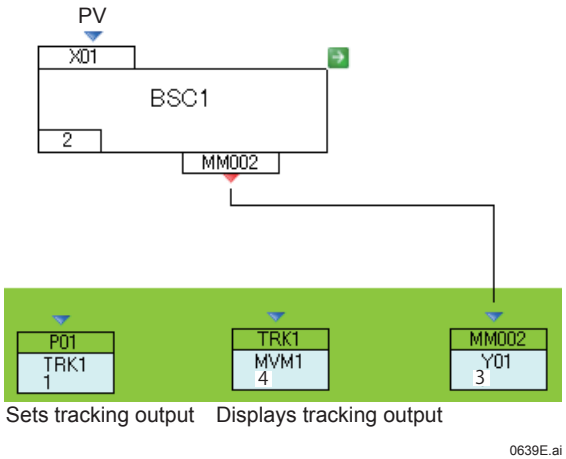


Function Block Overview of MV Displayed on a Bar-Graph

[Program example]

The following illustrates the program in the above figure.

- Programming by function block programming



● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD P01	P01					Reads reigster P01.
ST TRK1	P01					Sets tracking output.
LD X1	X1	P01				Reads the input.
BSC1	MV1	P01				Executes the control module.
ST Y1	MV1	P01				Outputs the manipulated output variable
LD TRK1	TRK1	MV1	P01			Reads the tracking output.
ST MVM1	TRK1	MV1	P01			Displays tracking output as the MV displayed on a bar-graph.
END						Ends program excution.

(14) PV displayed on a bar-graph (PVMn)

[Registers]

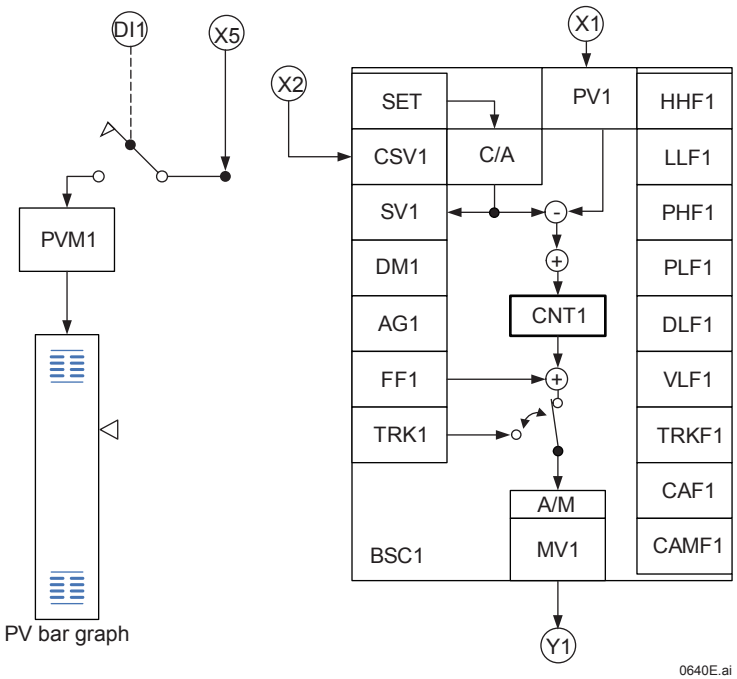
Register	Name	Function	Setting Range	Default
PVM1	PV1 displayed on a bar-graph	Display value	-0.063 to 1.063 (-6.3 to 106.3%)	-0.063
PVM2	PV2 displayed on a bar-graph	Display value		

[Operation]

With PV displayed on a bar-graph (PVM1, PVM2), any signal can be displayed (indicated) regardless of regular display items (result computed by the control module).
The PV displayed on a bar-graph is updated once by the control module. However, a desired data can be displayed by storing it in the PV displayed on a bar-graph. Be sure to store the desired display data in the PVM1 register or the PVM2 register after execution of the control module.

[Function block overview]

The following illustrates an example of displaying the input value (X5) as the PV displayed on a bar-graph when the digital input (DI1) turns OFF (0).
The following shows a function block overview of PV1 displayed on a bar-graph (PVM1).

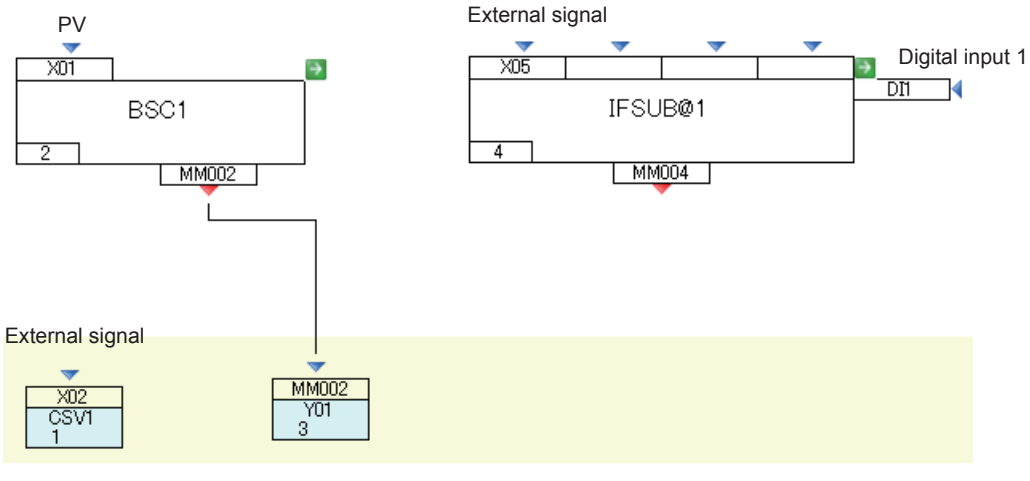


Function Block Overview of PV Displayed on a Bar-Graph

[Program example]

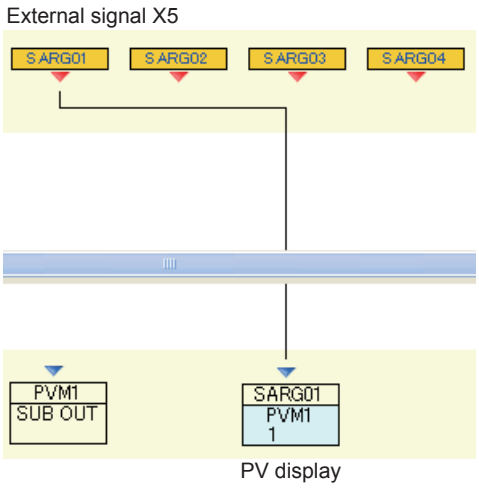
The following illustrates the program in the above figure.

- Programming by function block programming



0641E.ai

Main Program



PV display

0642E.ai

Subprogram (when DI1 ≥ 0.5)

● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X2	X2					Reads the input signal (X2) from the external device.
ST CSV1	X2					Writes to cascade setting value 1 (CSV1).
LD X1	X1	X2				Reads the input.
BSC1	MV1	X2				Executes the control module.
ST Y1	MV1	X2				Manipulated output variable.
LD DI1	0/1	MV1	X2			Reads the digital input (DI1).
GIF @JUMP	0/1	MV1	X2			
LD X5	X5	MV1	X2			Reads the input signal (X5) from the external device.
ST PVM1	X5	MV1	X2			Outputs to PV displayed on a bar-graph.
Computation after @ JUMP						

(15) SV displayed on a bar-graph (SVMn)

[Registers]

Register	Name	Function	Setting Range	Default
SVM1	SV1 displayed on a bar-graph	Display value	-0.063 to 1.063 (-6.3 to 106.3%)	-0.063
SVM2	SV2 displayed on a bar-graph	Display value		

[Operation]

With SV displayed on a bar-graph (SVM1, SVM2), any signal can be displayed (indicated) regardless of regular display items (result computed by the control module).
The SV displayed on a bar-graph is updated once by the control module. However, a desired data can be displayed by storing it in the SV displayed on a bar-graph. Be sure to store the desired display data in the SVM1 register or the SVM2 register after execution of the control module.

6.2.2 Control Flag Registers

Register (flag)	Control Module				Name	Signal		Default before Execution of Program at Control Period	Data Update after Execution of Control Module	Default
	BSC 1	BSC 2	CSC	SSC		0	1			
PHF1	○	–	○	○	PV1 high limit alarm flag	Normal	Alarm	OFF	Updated	0
PLF1	○	–	○	○	PV1 low limit alarm flag	Normal	Alarm			0
DLF1	○	–	○	○	Deviation variable 1 alarm flag	Normal	Alarm			0
VLF1	○	–	○	○	PV1 velocity alarm flag	Normal	Alarm			0
HHF1	○	–	○	○	PV1 high-high limit alarm flag	Normal	Alarm			0
LLF1	○	–	○	○	PV1 low-low limit alarm flag	Normal	Alarm			0
PHF2	–	○	○	○	PV2 high limit alarm flag	Normal	Alarm			0
PLF2	–	○	○	○	PV2 low limit alarm flag	Normal	Alarm			0
DLF2	–	○	○	○	Deviation variable 2 alarm flag	Normal	Alarm			0
VLF2	–	○	○	○	PV2 velocity alarm flag	Normal	Alarm			0
HHF2	–	○	○	○	PV2 high-high limit alarm flag	Normal	Alarm			0
LLF2	–	○	○	○	PV2 low-low limit alarm flag	Normal	Alarm			0
TRKF1	●	–	●	●	Output tracking 1 flag	Automatic	Tracking	ON (0)	Not updated	0
TRKF2	–	●	–	–	Output tracking 2 flag	Automatic	Tracking			0
PMVF1	●	–	●	●	Preset output 1 flag	OFF	ON			0
PMVF2	–	●	–	–	Preset output 2 flag	OFF	ON			0
CAF1	●	–	●	●	C ↔ A mode change 1 flag	A	C	OFF		Status of front panel keys
CAF2	–	●	–	–	C ↔ A mode change 2 flag	A	C			
CAMF1	●	–	●	●	C/A ↔ M mode change 1 flag	M	C, A			
CAMF2	–	●	–	–	C/A ↔ M mode change 2 flag	M	C, A			
OCF	–	–	●	–	Internal cascade switching flag	Cascade	Loop 2 independent			
LRF	–	–	–	●	Secondary loop remote/local switching flag	Remote	Local			
PRDF	–	–	●	–	Primary direct flag	OFF	ON	ON (0)		0
STCSW	●		●	●	STC stop flag	Not stopped	Stopped			0
STCM1	●		●	●	STC mode designation 1 flag	*1		OFF	*3	–
STCM2	●		●	●	STC mode designation 2 flag					–
STCLP	●		–	–	STC loop flag	Loop 1	Loop 2	*2	Not updated	0
STCOD	●		●	●	On-demand designation flag	OFF	ON	ON (0)	ON (0)	0
CCF1	●	–	●	●	SV1 analog/computer flag	Analog	Computer	ON (0)	Not updated	Parameter setpoint value
CCF2	–	●	–	–	SV2 analog/computer flag	Analog	Computer			Parameter setpoint value
DDCF1	○	–	○	○	DDC output 1 flag	Not DDC	DDC	OFF		0
DDCF2	–	○	–	–	DDC output 2 flag	Not DDC	DDC			0

●: Available by read (LD) and store (ST) instructions.
○: Available only by the read (LD) instruction.
–: Not available
When programming by function block programming, the read (LD) instruction is the equivalent of "wiring" to the module inputs and module parameters, and the store (ST) instruction is the equivalent of "wiring" from module outputs.

*1: These registers function on the loop selected at "STCLP".

STCM1	STCM2	STC Mode	Explanation
0	0	OFF	Function is OFF.
1	0	DISP	Only displays self-tuning.
0	1	ON	Executes self-tuning and updates PID.
1	1	ATSTUP	Executes self-tuning (startup).

*2: OFF in cascade or selector control. ON (0) in control other than cascade and selector.
*3: See 7.4.1 in YS1500 Indicating Controller/YS1700 Programmable Indicating Controller User's Manual

(1) PV high limit alarm (PHn, PHFn), PV low limit alarm (PLn, PLFn)

[Registers]

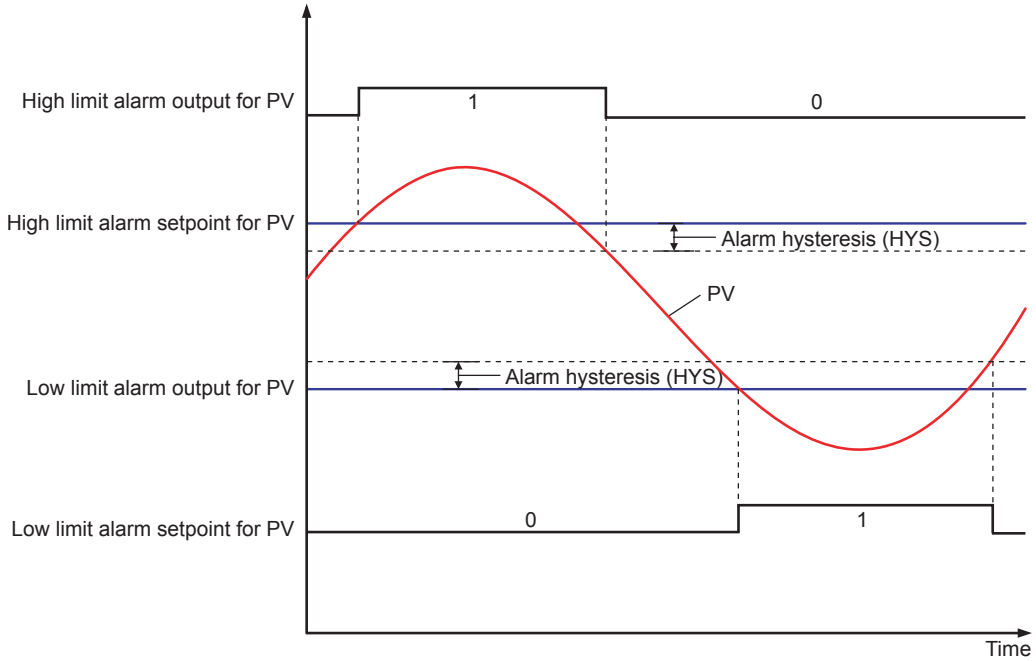
Register	Name	Setting Range	Default
PH1	High limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.3%)	Parameter setpoint value
PL1	Low limit alarm setpoint for PV1		
PH2	High limit alarm setpoint for PV2		
PL2	Low limit alarm setpoint for PV2		

[Flags]

Flag	Name	Setting Range	Default
PHF1	PV1 high limit alarm flag	0: Normal 1: Alarm	0
PLF1	PV1 low limit alarm flag		
PHF2	PV2 high limit alarm flag		
PLF2	PV2 low limit alarm flag		

[Operation]

The PV high limit and PV low limit alarms are built into the control modules. The figure below shows how these alarms operate.



For an example in the figure above, the contact type is such that the contact opens if an event occurs (factory default).

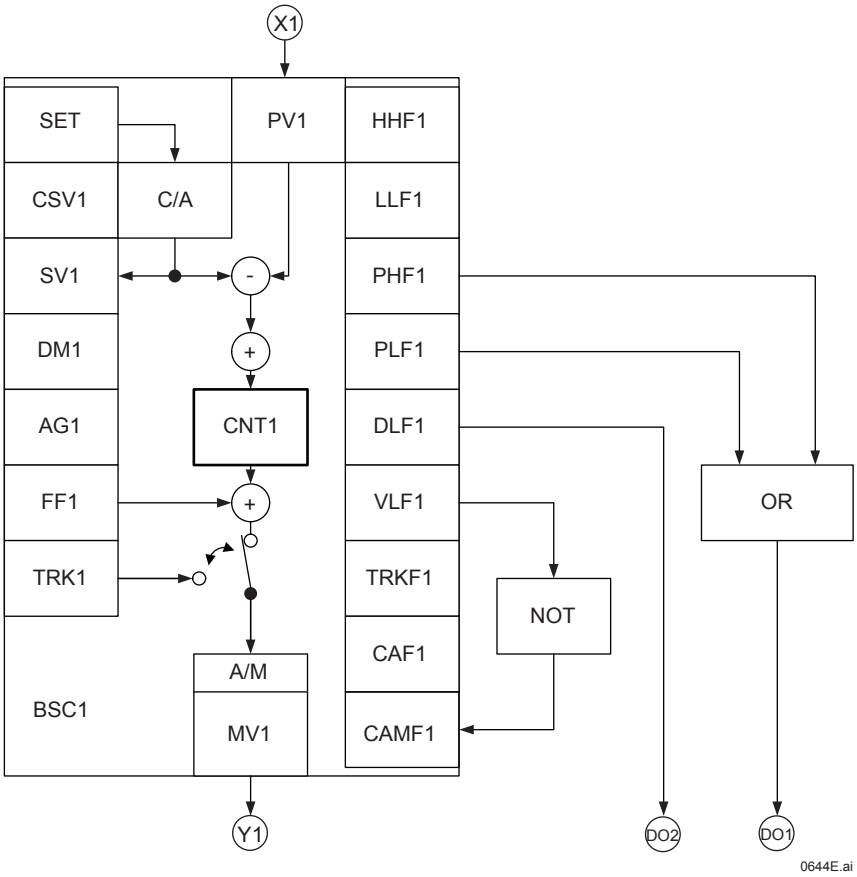
0643E.ai

Operation of PV High Limit and PV Low Limit Alarms

By the PV1 high limit alarm setting, an alarm is generated when process variable 1 exceeds the high limit alarm setpoint for PV1 (PH1). (PV1 high limit alarm flag (PHF1=1))
By the PV1 low limit alarm setting, an alarm is generated when process variable 1 exceeds the low limit alarm setpoint for PV1 (PL1). (PV1 low limit alarm flag (PLF1=1))

[Function block overview]

The following shows a function block overview of the PV high limit alarm.



Function Block Overview of PV High Limit Alarm

[Program example]

The following shows the program in the above figure.

● Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD VLF1	0/1					Reads the velocity alarm.
NOT	1/0					Inverts the signal.
GIF @JUMP	1/0					
LD K1	0	0				K1=0.0 (M mode is entered if in a velocity alarm status.)
ST CAMF1	0	0				
@JUMP LD X1	X1	1/0				Executes the control module.
BSC1	MV1	1/0				
ST Y1	MV1	1/0				
LD PHF1	0/1	MV1	1/0			Reads the high limit alarm.
LD PLF1	0/1	0/1	MV1	1/0		Reads the low limit alarm.
OR	0/1	MV1	1/0			Outputs the high/low limit non-recognition alarm.
ST DO1	0/1	MV1	1/0			Outputs to digital output 1.
LD DLF1	0/1	0/1	MV1	1/0		Reads the deviation alarm.
ST DO2	0/1	0/1	MV1	1/0		Outputs to digital output 2.
Next computation						

(2) Deviation alarm (DLn, DLFn)

[Registers]

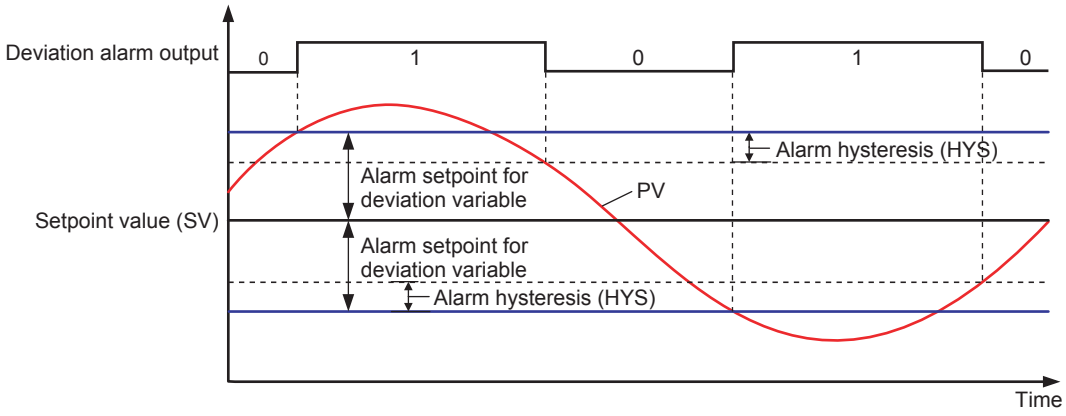
Register	Name	Setting Range	Default
DL1	Alarm setpoint for deviation variable 1	0.0 to 1.000 (0.0 to 100.0%)	Parameter setpoint value
DL2	Alarm setpoint for deviation variable 2		

[Flags]

Flag	Name	Setting Range	Default
DLF1	Deviation variable 1 alarm flag	0: Normal 1: Alarm	0
DLF2	Deviation variable 2 alarm flag		

[Operation]

The deviation alarm is built into the control modules. The figure below shows how this alarm operates.



For an example in the figure above, the contact type is such that the contact opens if an event occurs (factory default).

0646E.ai

Operation of the Deviation Alarm

Deviation is an absolute value.

(3) PV velocity alarm (VLn, VTn, VLFn)

[Registers]

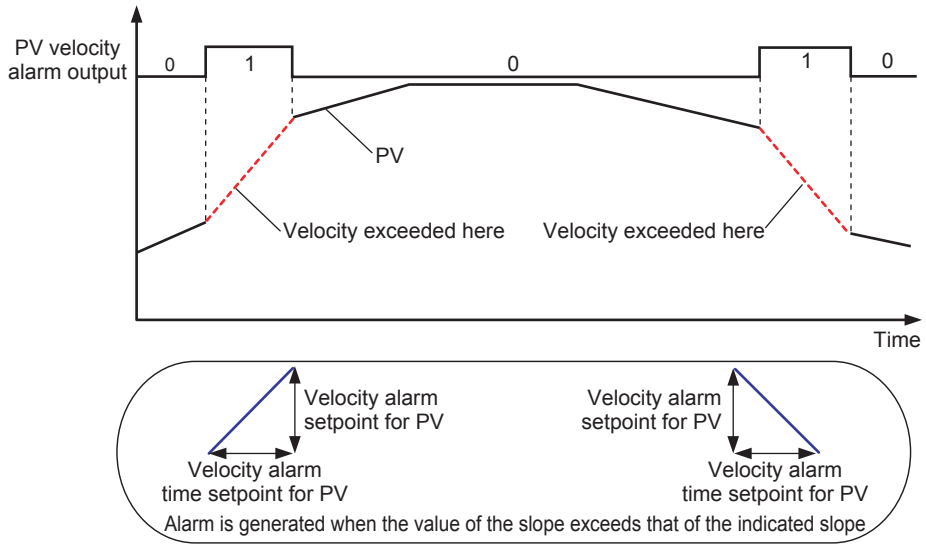
Register	Name	Setting Range	Default
VL1	Velocity alarm setpoint for PV1	0.0 to 1.000 (0.0 to 100.0%)	Parameter setpoint value
VT1	Velocity alarm time setpoint for PV1		
VL2	Velocity alarm setpoint for PV2	0.001 to 9.999 (1 to 9999 seconds)	
VT2	Velocity alarm time setpoint for PV2		

[Flags]

Flag	Name	Setting Range	Default
VLF1	PV1 velocity alarm flag	0: Normal 1: Alarm	0
VLF2	PV2 velocity alarm flag		

[Operation]

The PV velocity alarm is built into the control modules. The figure below shows how this alarm operates.



For an example in the figure above, the contact type is such that the contact opens if an event occurs (factory default).

0647E.ai

Operation of the PV Velocity Alarm

The PV velocity alarm can be expressed as the current process variable (PVn(t)) minus the past process variable (PVn(t-VTn)), which is obtained by subtracting the velocity alarm time setpoint for PV1 (VTn) from the current process variable. If the velocity exceeds the velocity alarm setpoint for PV, an alarm is generated. (PV velocity alarm flag (VLFn)=1) (n=1, 2)
This can be expressed as follows:
PV velocity alarm: $|PVn(t)-(PVn(t-VTn))|\geq VLn$
By setting a shorter velocity alarm time setpoint for PV, changes in the process variable can be monitored in more detail.

The alarm continues to be output for at least the duration of the velocity alarm time setpoint.

(4) PV high-high limit alarm (HHn, HHFn), PV low-low limit alarm (LLn, LLFn)

[Registers]

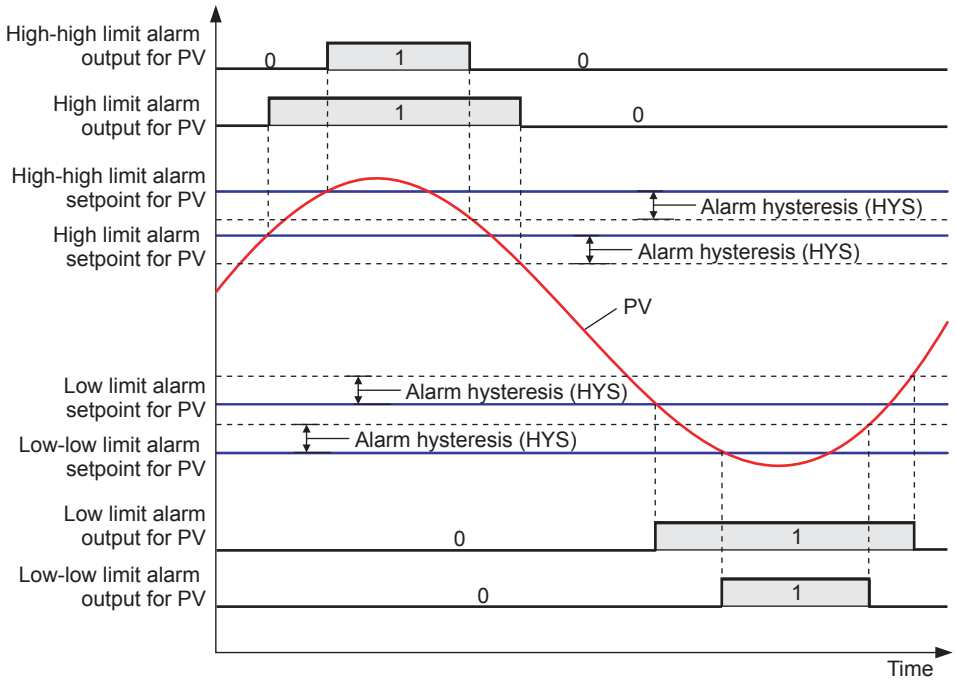
Register	Name	Setting Range	Default
HH1	High-high limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.3%)	Parameter setpoint value
LL1	Low-low limit alarm setpoint for PV1		
HH2	High-high limit alarm setpoint for PV2		
LL2	Low-low limit alarm setpoint for PV2		

[Flags]

Flag	Name	Setting Range	Default
HHF1	PV1 high-high limit alarm flag	0: Normal 1: Alarm	0
LLF1	PV1 low-low limit alarm flag		
HHF2	PV2 high-high limit alarm flag		
LLF2	PV2 low-low limit alarm flag		

[Operation]

The PV high-high limit alarm and PV low-low limit alarm are built into the control modules. The figure below shows how these alarms operate.



For an example in the figure above, the contact type is such that the contact opens if an event occurs (factory default).

0648E.ai

Operation of PV High-high Limit Alarm and PV Low-low Limit Alarm

(5) Alarm hysteresis (HYSn)

[Registers]

Register	Name	Setting Range	Default
HYS1	Alarm hysteresis 1	0.000 to 0.200 (0.0 to 20.0%)	Parameter setpoint value
HYS2	Alarm hysteresis 2		

[Operation]

See (1) to (4) above.

(6) Preset output (PMVn, PMVFn)

[Registers]

Register	Name	Setting Range	Default
PMV1	Preset output 1	-0.063 to 1.063 (-6.3 to 106.3%)	Parameter setpoint value
PMV2	Preset output 2		

[Flags]

Flag	Name	Setting Range	Default
PMVF1	Preset output 1 flag	0: OFF 1: ON	0
PMVF2	Preset output 2 flag		

[Operation]

In automatic control (A) or cascade setting automatic control (C), the preset output (PMVn) is output as manipulated output when a digital input signal is received. The value cannot be operated manually. When a digital signal is released, the result of control computation is output.

(7) Mode change flag (CAFn, CAMFn)

[Flags]

Flag	Name	Setting Range	Default
CAF1	C ↔ A mode change 1 flag	0: A mode 1: C mode	Front panel keys
CAF2	C ↔ A mode change 2 flag		
CAMF1	C/A ↔ M mode change 1 flag	0: M mode 1: C, A modes	
CAMF2	C/A ↔ M mode change 2 flag		

[Operation]

The C ↔ A mode change (CAFn) and C/A ↔ M mode change (CAMFn) flags are used to switch the operation mode.
The table below shows the operation mode that is selected according to the setting of these flags.

Operation Modes According to the Settings of the C ↔ A Mode Change (CAFn) and C/A ↔ M Mode Change (CAMFn) Flags

(CAFn) (*2) / **(CAMFn)** (*2)	**0**	**1**
1	Automatic control (A)	Cascade setting automatic control (C)
0	Manual control (M)	Manual control (M) (*1)

*1: When C/A ↔ M mode change flag (CAMFn) is set to "0", the mode changes to the manual control (M) mode regardless of the value of C ↔ A mode change flag (CAFn). Also, C ↔ A mode change flag (CAFn) is rewritten to "0" at execution of the control module.

*2: When mode change has been set by C ↔ A mode change flag (CAFn) and C/A ↔ M mode change flag (CAMFn), the operation mode cannot be switched by the C, A and M mode keys on controller's front panel.

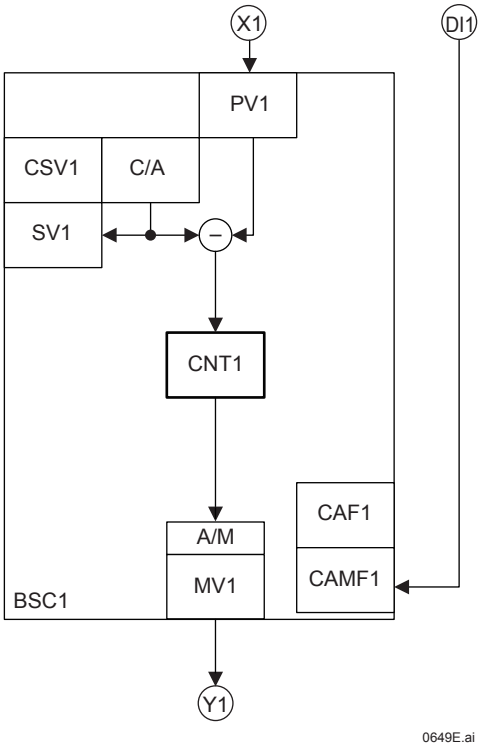
When the C mode (CMODn) is set to CMP, the controller shifts to the backup mode when an error occurs in communication with the host system. However, pay attention to the following points:

(1) When the backup mode is the automatic operation backup mode (BUA), C ↔ A mode change (CAFn) is set to "0" and C/A ↔ M mode change (CAMFn) is set to "1." The "C" lamp is lit and the "A" lamp blinks on the front panel. The operation changes to automatic control (A).
At this time, changing the C ↔ A mode change (CAFn) to "0" is invalid even if this is attempted in the user program, and the indicated lamp does not change to the "A" lamp.

(2) When the backup mode is the manual operation backup mode (BUM), C ↔ A mode change (CAFn) is set to "0" and C/A ↔ M mode change (CAMFn) is set to "0." The "C" lamp is lit and the "M" lamp blinks on the front panel. The operation changes to manual control (M).
At this time, changing the C ↔ A mode change (CAFn) to "0" or C/A ↔ M mode change (CAMFn) to "0" is invalid even if this is attempted in the user program, and the indicated lamp does not change to the "M" lamp.

[Function block overview (1)]

The figure below shows an example of switching between the automatic control (A) and manual control (M) by digital input (DI).



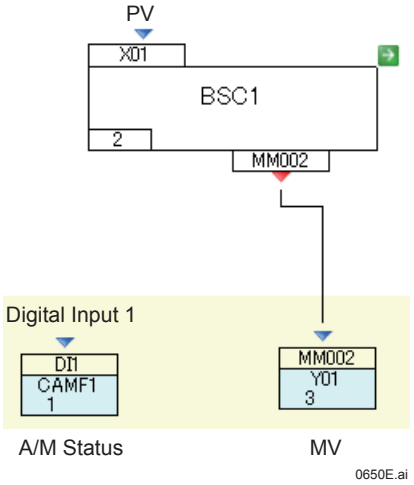
Function Block Overview of Operation Mode Switching by Digital Input (DI)

[Program example (1)]

The following shows the program in the above figure.

The status of the digital input (DI1) is read and stored to the C/A ↔ M mode change 1 flag (CAMF1).
When digital input (DI1) is "0" (OFF), the manual control (M) is entered.
When digital input (DI1) is "1" (ON), the automatic control (A) is entered.
At this time, as priority is given to digital input (DI1), the operation mode cannot be changed by the C, A and M mode keys on the controller's front panel.

- Programming by function block programming

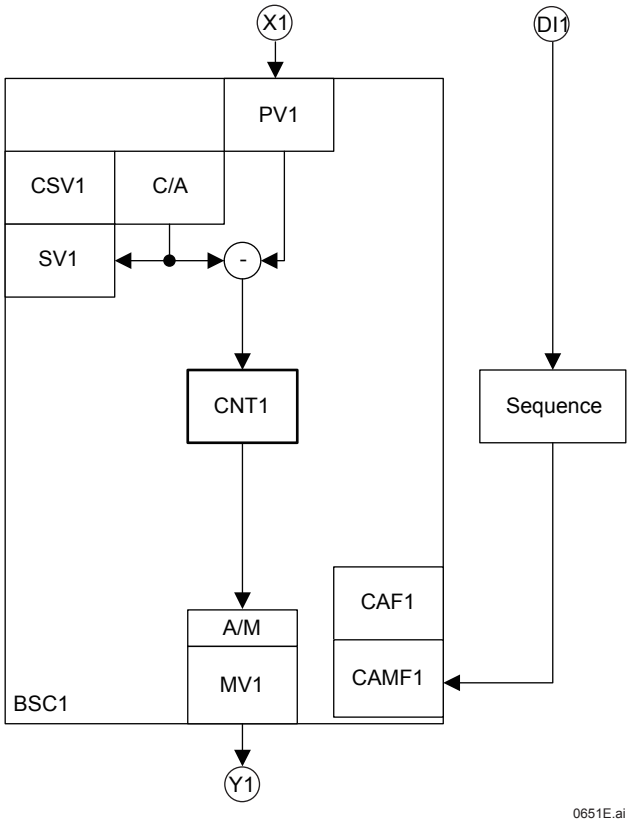


● Programming by function block programming

Program	S1	S2	S3	S4	S5	Explanation
LD DI1	0/1					Reads the digital input (DI1).
ST CAMF1	0/1					Changes the A/M modes. (0: M, 1: A)
LD X1	X1	0/1				Executes the control module.
BSC1	MV	0/1				
ST Y1	MV	0/1				
END						Ends program excution.

[Function block overview (2)]

The figure below shows an example of switching between the automatic control (A) and manual control (M) by non-lock type digital input (DI). At this time, the A and M mode keys on the controller's front panel are operable.

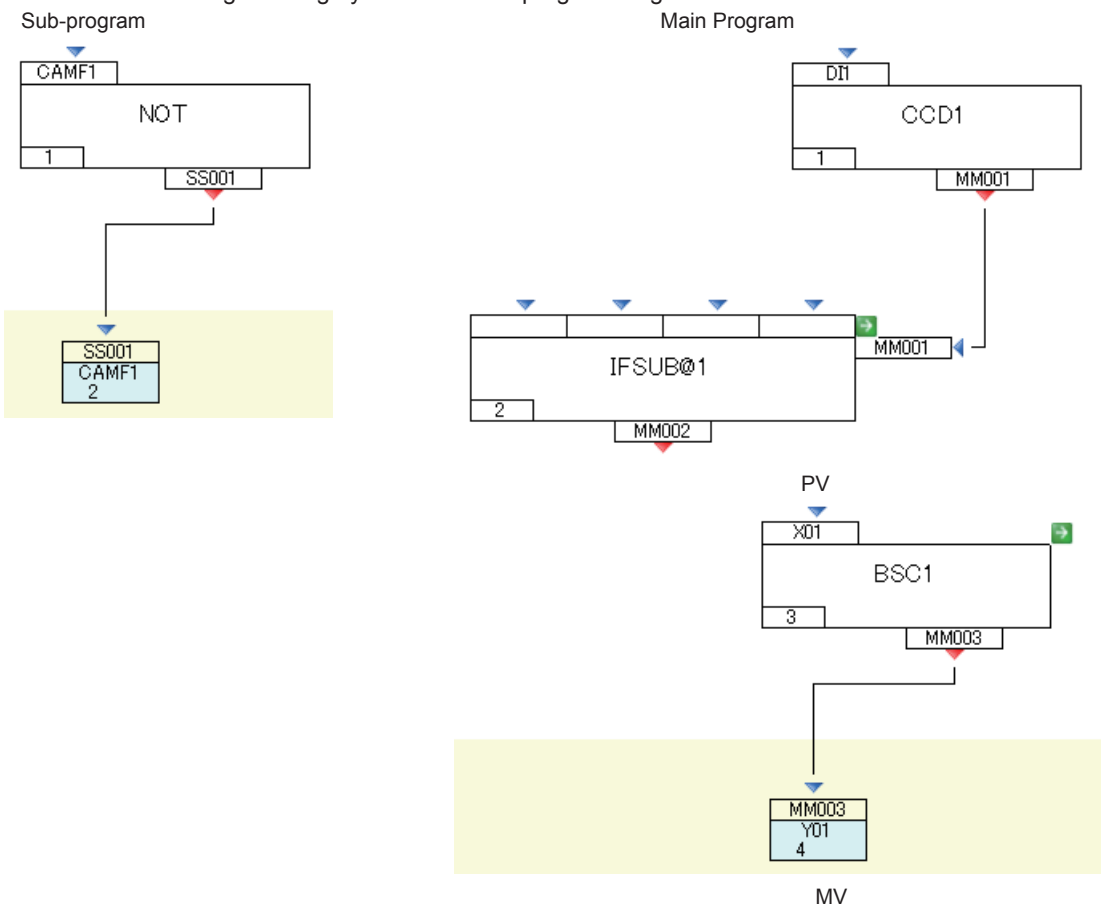


Function Block Overview of Operation Mode Switching by Non-lock Type Digital Input (DI)

[Program example (2)]

The following shows the program in the above figure.

- Programming by function block programming



- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD DI1	0/1					Reads the digital input (DI1).
CCD1	0/1					Detects change of value from 0 to 1
GIF @JUMP1						Goes to JUMP1 if there is a change
GO @JUMP2						Goes to JUMP2 if there is no change
@JUMP1 LD CAMF1	0/1					Switches between the automatic control (A) and manual control (M) ((A), (C)=1, (M)=0)
NOT	1/0					
ST CAMF1	1/0					
@JUMP2 LD X1	X1	1/0				Executes the control module.
BSC1	MV1	1/0				
ST Y1	MV1	1/0				
END						Ends program excution.

(8) Internal cascade switching flag (OCF)

[Flag]

Flag	Name	Function	Setting Range	Default
OCF	Internal cascade switching	Isolation of loops 1 and 2 of CSC	0: Cascade 1: Loop 2 independent	Status of front panel keys

[Operation]

Cascade open/close of loops 1 and 2 in cascade control.
By manipulating these flags, the lamp status of the C, A and M mode keys for loop 2 also changes.

Cascade closed: "C" lamp lit
Cascade open and CAMF1=0: "A" lamp lit
Cascade open and CAMF1=1: "M" lamp lit

(9) Primary direct flag (PRDF)

[Flag]

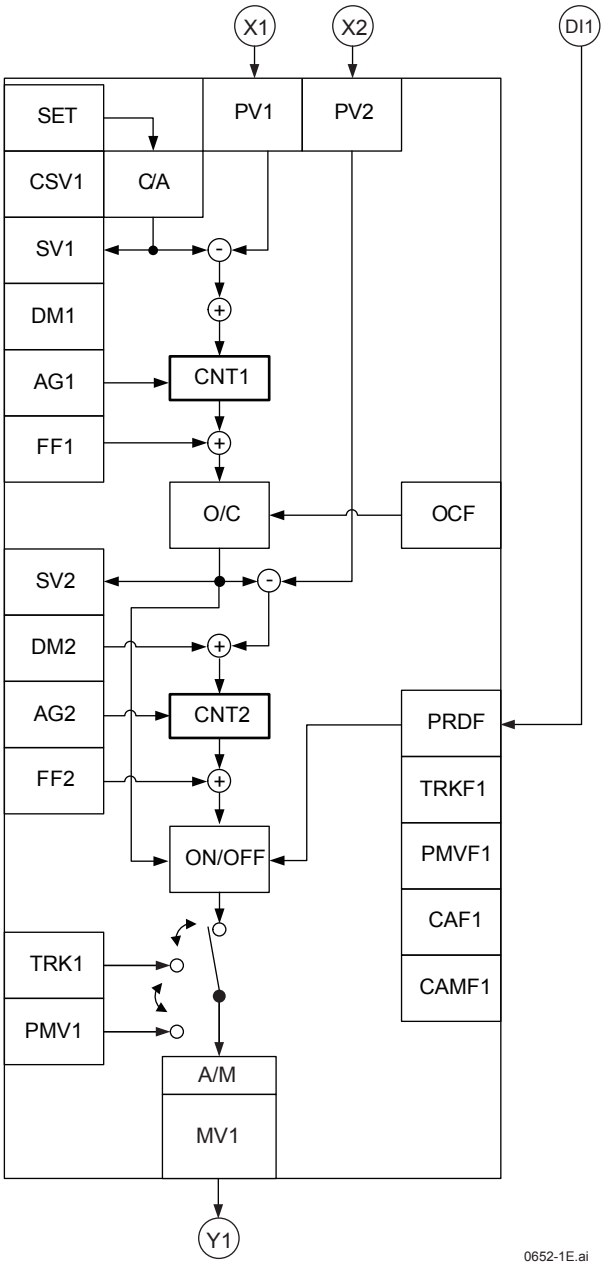
Flag	Name	Function	Setting Range	Default
PRDF	Primary direct flag	Output of loop 1 output in CSC to MV	0: Do not output 1: Output	0

[Operation]

Performing a control with the control calculation of loop 1 in the cascade control is called the primary direct method.
This method is used when the drum level is low in the drum level control of the boiler, or when an input error or the like occurs in the control loop of loop 2.
If the primary direct flag (PRDF) is set to ON (1) during closed loop operation, the output value (SV2) of loop 1 is output to the operating output value (MV).
The primary direct output is enabled only in the closed loop operation (the C lamp of loop 2 is on). The primary direct output is disabled in the open loop operation (the A and M lamps of loop 2 are on).
When the primary direct flag is set to ON (1), the output tracking function is activated to prevent a rapid change in the operating output.

[Function block overview]

The following shows a function block overview of the Primary direct.



Function Block Overview of Primary Direct

[Program example]

The following shows the program in the above figure.

- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD DI1	0/1					Reads the digital input (DI1).
ST PRDF	0/1					Stores to PRDF.
LD X1	X1	0/1				Executes the control module.
LD X2	X2	X1	0/1			
CSC	MV1	0/1				
ST Y1	MV	0/1				
END						Ends program excution.

Note

The PID parameters for the normal state (non-primary direct state) and primary direct state are different. Consider changing the PID value by using the manual setup or preset PID function, and the like.

(10) SV analog/computer flag (CCFn)

[Flags]

Flag	Name	Function	Setting Range	Default
CCF1	SV1 analog/computer flag	Selection of cascade setting signal Interlocked with setting of CMOD1 parameter	0: Analog 1: Computer	Parameter setpoint value
CCF2	SV2 analog/computer flag	Selection of cascade setting signal Interlocked with setting of CMOD2 parameter	0: Analog 1: Computer	

(11) DDC output flag (DDCFn)

[Flags]

Flag	Name	Function	Setting Range	Default
DDCF1	DDC output 1 flag	Set by DDC command on host PC	0: - 1: DDC	–
DDCF2	DDC output 2 flag			

6.2.3 Control Parameter Registers

Control parameter registers are extended registers that are used to set control parameters (e.g. proportional band, integral time, and derivative time) on the user program.

(Loop 1)

Register (data)	Control Module				Name	Computation Effective Range	Default before Execution of Program at Control Period	Data Update after Execution of Control Module	Default
	BSC 1	BSC 2	CSC	SSC					
PB1	●	–	●	●	Proportional band 1	0.001 to 9.999 (0.1 to 999.9%)	ON	Not updated	(Note 1)
TI1	●	–	●	●	Integral time 1	0.001 to 9.999 (1 to 9999 seconds)			
TD1	●	–	●	●	Derivative time 1	0 to 9.999 (0 to 9999 seconds)			
GW1	●	–	●	●	Non-linear control gap width 1	0.000 to 1.000 (0 to 100.0%)			
GG1	●	–	●	●	Non-linear control gain 1	0.000 to 1.000			
PH1	●	–	●	●	High limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.3%)			
PL1	●	–	●	●	Low limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.4%)			
HH1	●	–	●	●	High-high limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.3%)			
LL1	●	–	●	●	Low-low limit alarm setpoint for PV1	-0.063 to 1.063 (-6.3 to 106.3%)			
DL1	●	–	●	●	Alarm setpoint for deviation variable 1	0.000 to 1.063 (0.0 to 106.3%)			
VL1	●	–	●	●	Velocity alarm setpoint for PV1	0.000 to 1.063 (0.0 to 106.3%)			
VT1	●	–	●	●	Velocity alarm time setpoint for PV1	0.001 to 9.999 (1 to 9999 seconds)			
HYS1	●	–	●	●	Alarm hysteresis 1	0.000 to 0.200 (0.0 to 20.0%)			
MH1	●	–	●	●	High limit setpoint of MV1	-0.063 to 1.063 (-6.3 to 106.3%)			
ML1	●	–	●	●	Low limit setpoint of MV1	-0.063 to 1.063 (-6.3 to 106.3%)			
STM1	●	–	●	●	Sample PI sampled time 1	0.000 to 9.999 (0 to 9999 seconds)			
SWD1	●	–	●	●	Sample PI control time span 1	0.000 to 9.999 (0 to 9999 seconds)			
BD1	●	–	–	–	Batch PID deviation setting value 1	0.000 to 1.000 (0 to 100.0%)			
BB1	●	–	–	–	Batch PID bias 1	0.000 to 1.000 (0 to 100.0%)			
BL1	●	–	–	–	Batch PID lock-up width 1	0.000 to 1.000 (0 to 100.0%)			
SFA1	●	–	●	●	Adjustable setpoint filter α1	0.000 to 1.000			
SFB1	●	–	●	●	Adjustable setpoint filter β1	0.000 to 1.000			
PMV1	●	–	●	●	Preset output 1 (Note 1)	-0.063 to 1.063 (-6.3 to 106.3%)			
RB1	●	–	●	●	Reset bias 1	0.000 to 1.063 (0 to 106.3%)			
MR1	●	–	●	●	Manual reset 1	-0.063 to 1.063 (-6.3 to 106.3%)			

●: Available by read (LD) and store (ST) instructions, –: Not available
When programming by function block programming, the read (LD) instruction is the equivalent of "wiring" to the module inputs and module parameters, and the store (ST) instruction is the equivalent of "wiring" from module outputs.
Note 1: Parameter setpoint value

(Loop 2)

Register (data)	Control Module				Name	Computation Effective Range	Default before Execution of Program at Control Period	Data Update after Execution of Control Module	Default
	BSC 1	BSC 2	CSC	SSC					
PB2	–	●	●	●	Proportional band 2	0.001 to 9.999 (0.1 to 999.9%)	ON	Not updated	(Note 1)
TI2	–	●	●	●	Integral time 2	0.001 to 9.999 (1 to 9999 seconds)			
TD2	–	●	●	●	Derivative time 2	0 to 9.999 (0 to 9999 seconds)			
GW2	–	●	●	●	Non-linear control gap width 2	0.000 to 1.000 (0 to 100.0%)			
GG2	–	●	●	●	Non-linear control gain 2	0.000 to 1.000			
PH2	–	●	●	●	High limit alarm setpoint for PV2	-0.063 to 1.063 (-6.3 to 106.3%)			
PL2	–	●	●	●	Low limit alarm setpoint for PV2	-0.063 to 1.063 (-6.3 to 106.4%)			
HH2	–	●	●	●	High-high limit alarm setpoint for PV2	-0.063 to 1.063 (-6.3 to 106.3%)			
LL2	–	●	●	●	Low-low limit alarm setpoint for PV2	-0.063 to 1.063 (-6.3 to 106.3%)			
DL2	–	●	●	●	Alarm setpoint for deviation variable 2	0.000 to 1.063 (0.0 to 106.3%)			
VL2	–	●	●	●	Velocity alarm setpoint for PV2	0.000 to 1.063 (0.0 to 106.3%)			
VT2	–	●	●	●	Velocity alarm time setpoint for PV2	0.001 to 9.999 (1 to 9999 seconds)			
HYS2	–	●	●	●	Alarm hysteresis 2	0.00 to 0.200 (0.0 to 20.0%)			
MH2	–	●	●	–	High limit setpoint of MV2	-0.063 to 1.063 (-6.3 to 106.3%)			
ML2	–	●	●	–	Low limit setpoint of MV2	-0.063 to 1.063 (-6.3 to 106.3%)			
STM2	–	●	●	●	Sample PI sampled time 2	0.000 to 9.999 (0 to 9999 seconds)			
SWD2	–	●	●	●	Sample PI control time span 2	0.000 to 9.999 (0 to 9999 seconds)			
BD2	–	–	–	–	Batch PID deviation setting value 2	0.000 to 1.000 (0 to 100.0%)			
BB2	–	–	–	–	Batch PID bias 2	0.000 to 1.000 (0 to 100.0%)			
BL2	–	–	–	–	Batch PID lock-up width 2	0.000 to 1.000 (0 to 100.0%)			
SFA2	–	●	●	●	Adjustable setpoint filter α2	0.000 to 1.000			
SFB2	–	●	●	●	Adjustable setpoint filter β2	0.000 to 1.000			
PMV2	–	●	●	●	Preset output 2	-0.063 to 1.063 (-6.3 to 106.3%)			
RB2	–	●	●	●	Reset bias 2	0.000 to 1.063 (0 to 106.3%)			
MR2	–	●	●	●	Manual reset 2	-0.063 to 1.063 (-6.3 to 106.3%)			

●: Available by read (LD) and store (ST) instructions, –: Not available
When programming by function block programming, the read (LD) instruction is the equivalent of "wiring" to the module inputs and module parameters, and the store (ST) instruction is the equivalent of "wiring" from module outputs.
Note 1: Parameter setpoint value

Bumpless computation

Generally, non-continuous changes (bumps) occur when the proportional band is changed during execution of PID control. On the YS1700, however, the control operation formula has been designed to prevent bumps from occurring. For this reason, there is no need to devise measures for output bumps.

6.2.4 Self-tuning (STC) Registers

[Related registers]

Register	Explanation	Setting Range	Default
STCSW	STC stop flag	0: Not stopped, 1: Stopped	0
STCM1	STC mode designation 1 flag	(*1)	–
STCM2	STC mode designation 2 flag		–
STCLP	STC loop flag	0: Loop 1, 1: Loop 2	0
STCOD	On-demand designation flag	0: OFF, 1: ON	0

*1: These registers function on the loop selected at "STCLP".

STCM1	STCM2	STC Mode	Explanation
0	0	OFF	Function is OFF.
1	0	DISP	Only displays self-tuning.
0	1	ON	Executes self-tuning and updates PID.
1	1	ATSTUP	Executes self-tuning (startup).

[Operation]

The user program Available to read and write to all these registers.
When making changes to the registers from the user program, use the store (ST) instruction.

► Self-tuning function: "Chapter 7 Self-tuning Functions" in YS1500 Indicating Controller/YS1700 Programmable Indicating Controller User's Manual (CD-ROM)

6.2.5 Preset PID Switch Registers

[Related registers]

Register	Explanation	Setting Range	Default
PPID1	Settings are made on loop 1.	-8.000 to 8.000 (Note)	-8.000
PPID2	Settings are made on loop 2.		

Note: When the numerical value is a negative value, the current setpoint is maintained.

[Operation]

With preset PID (PPIDn), a maximum of eight sets of PID parameters predefined in the YSS1000 Setting Software for YS1000 Series or by operating the keys on the YS1700 can be set in the user program.
The table below shows operation when preset PID (PPIDn) is set.

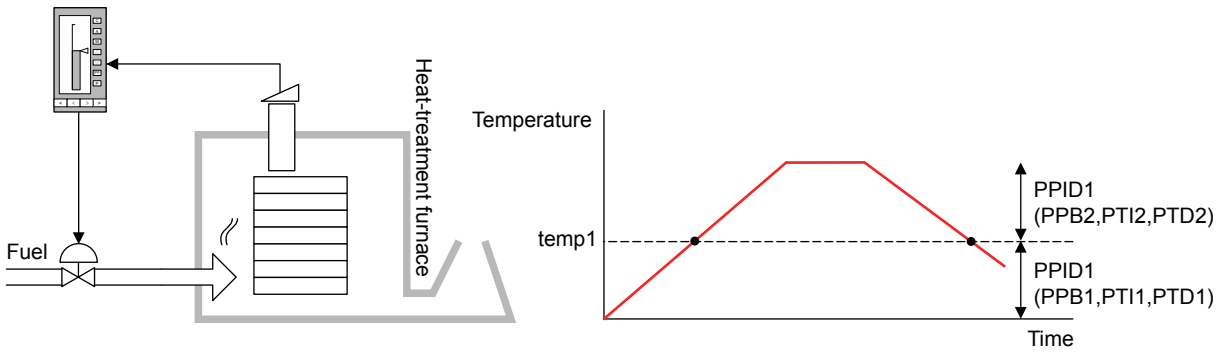
Relationship between Preset PID (PPIDn) and PID Parameters

Setpoint Value for registers (n=1, 2)	Operation (n=1, 2)
-8.000 ≤ PPIDn < 0.000	No operation is performed. (The current setpoint value is maintained.)
0.000 ≤ PPIDn < 0.100	The values of PPB1, PTI1, and PTD1 are set to PBn, TIn, TDn, respectively.
0.100 ≤ PPIDn < 0.200	The values of PPB2, PTI2, and PTD2 are set to PBn, TIn, TDn, respectively.
0.200 ≤ PPIDn < 0.300	The values of PPB3, PTI3, and PTD3 are set to PBn, TIn, TDn, respectively.
0.300 ≤ PPIDn < 0.400	The values of PPB4, PTI4, and PTD4 are set to PBn, TIn, TDn, respectively.
0.400 ≤ PPIDn < 0.500	The values of PPB5, PTI5, and PTD5 are set to PBn, TIn, TDn, respectively.
0.500 ≤ PPIDn < 0.600	The values of PPB6, PTI6, and PTD6 are set to PBn, TIn, TDn, respectively.
0.600 ≤ PPIDn < 0.700	The values of PPB7, PTI7, and PTD7 are set to PBn, TIn, TDn, respectively.
0.700 ≤ PPIDn ≤ 8.000	The values of PPB8, PTI8, and PTD8 are set to PBn, TIn, TDn, respectively.

For example, when preset PID (PPID1) is 0.250, PPB3, PTI3, and PTD3 are set to PB1, TI1, and TD1, respectively.

[Function block overview]

The following shows an example of changing the PID parameters by temperature and a function block overview.



Changing PID Parameters by Temperature

[Program example]

The above figure displays a program of using temp1 as the reference temperature, and using PPB1, PTI1, and PTD1 for PPID1 when the temperature is lower than temp1, and using PPB2, PTI2, and PTD2 for PPID2 for PPID1 when the temperature is higher than temp1.

- Programming by text programming

Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads the input value (X1).
LD K1	K1	X1				Sets the ratio value (temp1).
CMP	0/1					Compares the input value (X1) with the comparison value (temp1).
GIF @JUMP1	0/1					Go to @JUMP1 when input value (X1) ≥ comparison value (temp1).
LD P1	P1					P1=0.050
ST PPID1						Sets the PID parameters.
GO @JUMP2						Go to @JUMP2.
@JUMP1 LD P2	P2					P2=0.150
ST PPID1						Sets the PID parameters.
@JUMP2 LD X1	X1					Executes the control module.
BSC1	MV1					
ST Y1	MV1					
END						Ends program execution.

6.2.6 System Flag Registers

The system flag register indicates the system alarm status. The contents of these registers can be read by the program but not written to.

Register (flag)	Function	Signal		Explanation	Default
		0	1		
IOP	Input overrange	Normal	Error occurring	1 or more of X1 to X8 overrange (-6.3% or less or 106.3% or more)	–
OOP	Current output open	Normal	Error occurring	Current output of analog output 1 (Y1) and analog output 3 (Y3) released	–
CAL	Computation overflow	Normal	Error occurring	Computation result exceeds ±8.000.	–
IOPX01	X1 input overrange	Normal	Error occurring	Only inputs used by the user program are targeted. (Unused inputs are always "0".)	–
IOPX02	X2 input overrange	Normal	Error occurring		–
IOPX03	X3 input overrange	Normal	Error occurring		–
IOPX04	X4 input overrange	Normal	Error occurring		–
IOPX05	X5 input overrange	Normal	Error occurring		–
IOPX06	X6 input overrange	Normal	Error occurring		–
IOPX07	X7 input overrange	Normal	Error occurring		–
IOPX08	X8 input overrange	Normal	Error occurring		–
OOPY01	Y1 current output open	Normal	Error occurring	Release of output current of analog output 1	–
OOPY03	Y3 current output open	Normal	Error occurring	Release of output current of analog output 3	–
COMS01	Host system communication hardware error	Normal	Error occurring	A hardware-based error occurred during communications.	–
COMS02	Host system communication software error	Normal	Error occurring	A software-based error occurred during communications.	–
CF1	Peer-to-peer communication reception timeout flag	Normal	Error	Represents normal/error of the received data from the YS1700 machine No.1.	–
CF2	Peer-to-peer communication reception timeout flag	Normal	Error	Represents normal/error of the received data from the YS1700 machine No.2.	–
CF3	Peer-to-peer communication reception timeout flag	Normal	Error	Represents normal/error of the received data from the YS1700 machine No.3.	–
CF4	Peer-to-peer communication reception timeout flag	Normal	Error	Represents normal/error of the received data from the YS1700 machine No.4.	–
E_PTOP	Peer-to-peer communication data error	Normal	Error	Error occurs when the corresponding CF1 to CF4 of the peer-to-peer communications registers used in the user program is in error.	–
E_PROG	User program error	Normal	Error	Error occurs by the END instruction not included, machine number out of range, or an RTN error.	–
E_DATA	YSS1000 writing not completed	Normal	Error	Error occurs when writing from YSS1000 is not completed.	–

6.2.7 Operation Display Control Registers (Z Registers)

Any of the Operation Display can be switched to by the user program. This way, you can know which Operation Display is currently displayed.
To switch the display, use register Z01, and to detect the display screen, use register Z02.

Registers for Display Screen Control

Register	Name	Setting Range	Function	Default
Z01	Operation Display control register	-800.0 to 800.0%	Switches the display screen	0.0% (Note 1)
Z02	Operation Display status register	-800.0 to 800.0%	Detects the display screen	OFF

Specifications of Operation Display Control Register Z01

Setpoint Value (Note 2)	Display Switching Operation (Note 3)
Z01 < 50.0%	OFF (Display of the current display screen is maintained.)
50.0% ≤ Z01 < 150.0%	The LOOP 1 Display is displayed.
150.0% ≤ Z01 < 250.0%	The LOOP 2 Display is displayed.
250.0% ≤ Z01 < 350.0%	The TREND 1 Display is displayed.
350.0% ≤ Z01 < 450.0%	The TREND 2 Display is displayed.
450.0% ≤ Z01 < 550.0%	The ALARM Display is displayed.
550.0% ≤ Z01 < 650.0%	The DUAL 1 Display is displayed.
650.0% ≤ Z01 < 750.0%	The DUAL 2 Display is displayed.
750.0% ≤ Z01 < 850.0%	OFF (Display of the current display screen is maintained.)
850.0% ≤ Z01 < 950.0%	The METER 1 Display is displayed.
950.0% ≤ Z01 < 1050.0%	The METER 2 Display is displayed.
1050.0% ≤ Z01 < 1100.0%	The TREND 3 Display is displayed.

Specifications of Operation Display Status Register Z02

Display	Z02 Indication Value (Note 2)
LOOP 1 Display	100.0%
LOOP 2 Display	200.0%
TREND 1 Display	300.0%
TREND 2 Display	400.0%
ALARM Display	500.0%
DUAL 1 Display	600.0%
DUAL 2 Display	700.0%
METER 1 Display	900.0%
METER 2 Display	1000.0%
TREND 3 Display	1100.0%
Other than above	Less than -700.0%

Note 1: Register Z01 is initialized at each screen refresh period (always at 200 msec intervals unrelated to the control period).
Note 2: When reading or writing to registers Z01 and Z02, try to use the values preset to the K register in the YSS1000 Setting Software for YS1000 Series. Otherwise (for example, when using the value set to the P register), a deviation of within ±0.1% may occur. For example, the value "50.0%" in the table above becomes a value between 49.9% and 50.1%, and the value "100.0%" becomes a value between 99.9% and 100.1%.
Note 3: When the hide designation for a display specified by the setting made to register Z01 is OFF (parameter for displaying the CONFIG1 Operation Display), the display does not switch to that display.

Blank Page

7.1 List of Computing Module (Instruction)

This chapter explains operation of the computing modules and their applications.

Category	Instruction	Dynamic	New (*)	Text Programming	Function Block Programming			
				Instruction	Symbol	Number of Inputs	Number of Parameters	Remarks
	Read			LD (reg)	These instructions are not provided in function block programming. (These functions correspond to the operation of "wiring" when creating a program. For details, see "Chapter 3 User Program Creation Guide.")			
	Store			ST (reg)				
	Store with "Enable switch"		○	STE (reg)				
	End			END				
Basic Computations	Addition			+	+	2	0	ADDITION
	Subtraction			–	–	2	0	SUBTRACTION
	Multiplication			*	*	2	0	MULTIPLICATION
	Division			/	/	2	0	DIVISION
	Ratio		○	RATIO	RATIO	1	3	
	Square root extraction			SQT	SQT	1	0	
	Square root extraction with variable low cutoff			SQTE	SQTE	1	1	
	Absolute value			ABS	ABS	1	0	
	High selector			HSL	HSL	2	0	
	Low selector			LSL	LSL	2	0	
	High Limiter			HLM	HLM	1	1	
	Low Limiter			LLM	LLM	1	1	
	Scaling		○	SCAL	SCAL	1	3	
	Normalization		○	NORM	NORM	1	3	
	Natural logarithm		○	LN	LN	1	0	
	Common logarithm		○	LOG	LOG	1	0	
	Exponential		○	EXP	EXP	1	0	
	Power		○	PWR	PWR	1	1	
	Temperature compensation (°C)		○	TCMP1	TCMP1	2	1	
	Temperature compensation (°F)		○	TCMP2	TCMP2	2	1	
	Temperature compensation (K)		○	TCMP3	TCMP3	2	1	
	Pressure compensation (MPa)		○	PCMP1	PCMP1	2	1	
	Pressure compensation (kgf/cm^2)		○	PCMP2	PCMP2	2	1	
	Pressure compensation (psi)		○	PCMP3	PCMP3	2	1	
	Conversion from DI to BCD		○	DIBCD	DIBCD	0	2	
	Conversion from BCD to DO		○	DOBCD	DOBCD	1	2	Special to module output
	Conversion from DI to Binary		○	DIBIN	DIBIN	0	2	
	Conversion from Binary to DO		○	DOBIN	DOBIN	1	2	Special to module output
	Maximum of 2 values		○	MAX2	MAX2	2	0	
	Maximum of 3 values		○	MAX3	MAX3	3	0	
	Maximum of 4 values		○	MAX4	MAX4	4	0	
	Minimum of 2 values		○	MIN2	MIN2	2	0	
	Minimum of 3 values		○	MIN3	MIN3	3	0	
	Minimum of 4 values		○	MIN4	MIN4	4	0	
	Average of 2 values		○	AVE2	AVE2	2	0	
	Average of 3 values		○	AVE3	AVE3	3	0	
	Average of 4 values		○	AVE4	AVE4	4	0	
	Increment		○	INC	INC	1	0	
	Decrement		○	DEC	DEC	1	0	

*: ○ "New" refers to instructions newly added onto YS1700 and not YS170.
For details on the Read (LD), Store (ST), Store with "Enable switch" (STE), and the End (END) instructions, see "Chapter 12 List of Text Program Instructions."

7.1 List of Computing Module (Instruction)

Category	Instruction	Dynamic	New (*)	Text Programming	Function Block Programming			
				Instruction	Symbol	Number of Inputs	Number of Parameters	Remarks
Dynamic Operations	10-segment linearizer function	2		FXn	FXn	1	0	
	Inverse conversion of 10-segment linearizer function	2	○	IFXn	IFXn	1	0	
	Arbitrary segment linearizer function	2		GXn	GXn	1	0	
	Inverse conversion of arbitrary segment linearizer function	2	○	IGXn	IGXn	1	0	
	First order lag (second)	8		LAGm	LAGm	1	1	
	First order lag (minute)	8	○	LAGMm	LAGMm	1	1	
	Derivative (second)	2		LEDm	LEDm	1	1	
	Derivative (minute)	2	○	LEDMm	LEDMm	1	1	
	Dead time (second)	3		DEDm	DEDm	1	1	
	Dead time (minute)	3	○	DEDMm	DEDMm	1	1	
	Velocity computation (second)	3		VELm	VELm	1	1	
	Velocity computation (minute)	3	○	VELMm	VELMm	1	1	
	Velocity limiter	6		VLMm	VLMm	1	2	
	Moving average computation (second)	3		MAVm	MAVm	1	1	
	Moving average computation (minute)	3	○	MAVMm	MAVMm	1	1	
	0 to 1 change detection	8		CCDm	CCDm	1	0	
	0 to 1 change detection	8	○	UEDGm	UEDGm	1	0	
	1 to 0 change detection	8	○	DEDGm	DEDGm	1	0	
	Change detection	8	○	EDGEm	EDGEm	1	0	
	Timer (second)	8	◆	TIMm	TIMm	1	0	
	Timer (minute)	8	○	TIMMm	TIMMm	1	0	
	Time out (second)	8	○	TUPm	TUPm	1	1	
	Time out (minute)	8	○	TUPMm	TUPMm	1	1	
	Program setter (second)	2		PGMm	PGMm_A PGMm_B	1 0	2 0	Use PGMm_A and PGMm_B together as a pair.
	Program setter (minute)	2	○	PGMMm	PGMMm_A PGMMm_B	1 0	2 0	Use PGMMm_A and PGMMm_B together as a pair.
	Pulse input counter	8	◆	PICm	PICm	1	1	
	Totalizer pulse output	2		CPOm	CPOm	1	1	Special to module output
	High limit alarm	8	◆	HALm	HALm	1	2	
	Low limit alarm	8	◆	LALm	LALm	1	2	
	Square root extraction (Low cutoff point or less: Linear)	8	○	SQAm	SQAm	1	1	
	Square root extraction (Low cutoff point or less: Zero)	8	○	SQBm	SQBm	1	1	
	RS flip-flop	8	○	RSFFm	RSFFm	2	0	
	Hold timer (second)	8	○	HTIMm	HTIMm	1	1	
	Hold timer (minute)	8	○	HTIMMm	HTIMMm	1	1	
	Previous input variable	8	○	DELAYm	DELAYm	1	0	
	Hold	8	○	HOLDm	HOLDm	1	1	

*○: "New" refers to instructions newly added onto YS1700 and not YS170.
◆: Four of these instructions were used on the YS170. The number used on the YS1700 has been increased to eight.

Category	Instruction	Dynamic	New (*)	Text Programming	Function Block Programming			
				Instruction	Symbol	Number of Inputs	Number of Parameters	Remarks
Logic Computations	AND			AND	AND	2	0	
	OR			OR	OR	2	0	
	NOT			NOT	NOT	1	0	
	Exclusive OR			EOR	EOR	2	0	
	Multi-input AND		○	MAND	MAND	4	0	
	Multi-input OR		○	MOR	MOR	4	0	
	Multi-input exclusive OR		○	MEOR	MEOR	4	0	
Condition Judgments	Comparison			CMP	CMP	2	0	
	Signal switching			SW	SW	2	1	
	≥, Greater or equal		○	GE	GE	2	0	
	>, Greater than		○	GT	GT	2	0	
	≤, Less or equal		○	LE	LE	2	0	
	<, Less than		○	LT	LT	2	0	
	In range		○	INRNG	INRNG	3	0	
	Out of range		○	OUTRNG	OUTRNG	3	0	
	Multi input signals switching A		○	None	MSWA	4	1	
	Multi input signals switching B		○	None	MSWB	4	4	
	Selection from four inputs and conversion to a numerical value		○	None	BITSEL	4	0	
	Jump			GO @<label name>	None	-	-	
	Conditional jump			GIF @<label name>	None	-	-	
	Jump to the sub-program			GOSUB@<sub-program name>	SUB@n(*1)	4	0	
	Conditional jump to the sub-program			GIFSUB@<sub-program name>	IFSUB@n (*1)	4	1	
	Condition comparison jump to the sub-program		○	None	GTSUB@n (*1)	4	2	
	Jump to the sub-program (for nesting)			None	SSUB@n (*2)	4	0	
	Conditional jump to the sub-program (for nesting)			None	IFSSUB@n (*2)	4	1	
	Condition comparison jump to the sub-program (for nesting)		○	None	GTSSUB@n (*2)	4	2	
	Sub-program startup			SUB@<sub-program name>	None	-	-	
	Sub-program termination			RTN	None	-	-	
Register Moves	S register change			CHG	None	-	-	
	S register rotation			ROT	None	-	-	
Control Functions	Basic control			BSC1, BSC2	BSC1, BSC2	1	0	
	Cascade control			CSC	CSC	2	0	
	Selector control			SSC	SSC	2	0	
Other	Storage register terminal	-	-	-	Register name	1	0	
	Storage register terminal with enable switch	-	-	-	Register name	1	1	

* ○: "New" refers to instructions newly added onto YS1700 and not YS170.
*1: n=1 to 200, *2: n=201 to 256

+

Category	Basic computations		
Name of Computation	Addition	Computation Instruction Symbol	+

Explanation of Operation

[Operation Formula]
Function block programming: Addition result (J) = Value to be added (A) + value to add (B)
Text programming: S1 after computation = S2 + S1

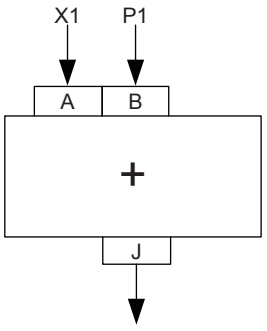
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Value to be added
	B	Value to add
Output	J	A+B

(Example)
X1: Analog input 1
P1: Variable register 1



0701E.ai

Text Programming

(Example) Analog input 1 and variable register 1 are added.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD P1	P1	X1	a	b	c	Reads the variable register 1.
+	X1+P1	a	b	c	c	Addition result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

Category	Basic computations		
Name of Computation	Subtraction	Computation Instruction Symbol	–

Explanation of Operation

[Operation Formula]
Function block programming: Subtraction result (J) = Value to be subtracted (A) - value to subtract (B)
Text programming: S1 after computation = S2 - S1

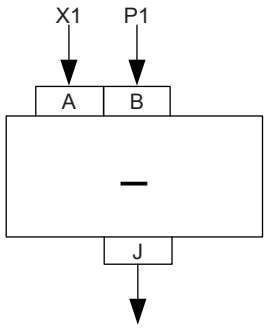
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Value to be subtracted
	B	Value to subtract
Output	J	A-B

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Variable register 1 is subtracted from analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD P1	P1	X1	a	b	c	Reads the variable register 1.
-	X1-P1	a	b	c	c	Subtraction result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

*

Category	Basic computations		
Name of Computation	Multiplication	Computation Instruction Symbol	*

Explanation of Operation

[Operation Formula]
Function block programming: Multiplication result (J) = Value to be multiplied (A) × value to multiply (B)
Text programming: S1 after computation = S2 × S1

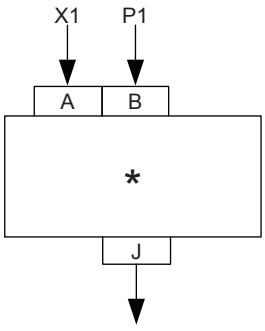
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Value to be multiplied
	B	Value to multiply
Output	J	A×B

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Variable register 1 is multiplied by analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD P1	P1	X1	a	b	c	Reads the variable register 1.
*	X1×P1	a	b	c	c	Multiplication result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

/

Category	Basic computations		
Name of Computation	Division	Computation Instruction Symbol	/

Explanation of Operation

[Operation Formula]
Function block programming: Division result (J) = Value to be divided (A) ÷ value to divide (B)
Text programming: S1 after computation = S2 ÷ S1

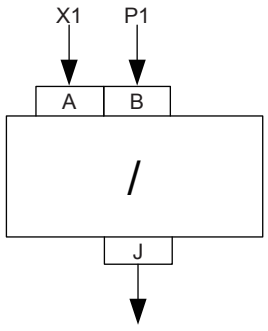
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Value to be divided
	B	Value to divide
Output	J	A÷B

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Analog input 1 is divided by variable register 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD P1	P1	X1	a	b	c	Reads the variable register 1.
/	X1÷P1	a	b	c	c	Division result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

RATIO

Category	Basic computations		
Name of Computation	Ratio	Computation Instruction Symbol	**RATIO**

Explanation of Operation

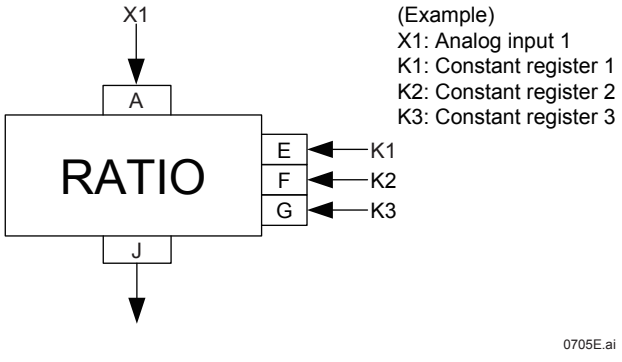
[Operation Formula]
Function block programming: Ratio computation result (J) = (A+E) × F+G
Text programming: S1 after computation = (S4+S3) × S2+S1

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Coefficient 1
	F	Coefficient 2
	G	Coefficient 3
Output Value	**J**	(A+E)×F+G



Text Programming

(Example) The ratio is set to analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD K1	K1	X1	a	b	c	Reads the constant register 1.
LD K2	K2	K1	X1	a	b	Reads the constant register 2.
LD K3	K3	K2	K1	X1	a	Reads the constant register 3.
RATIO	(X1+K1)×K2+K3	a	a	a	a	Result of ratio computation

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

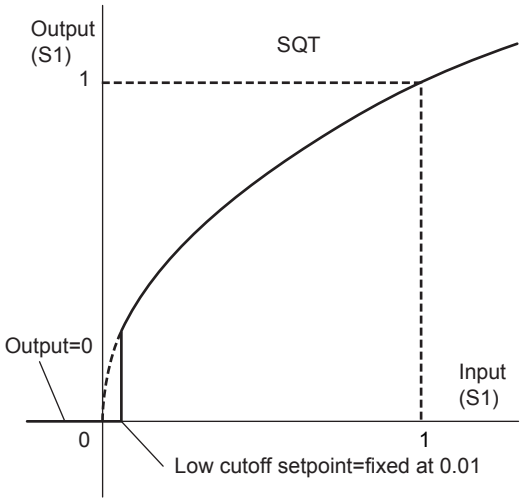
SQT

Category	Basic computations		
Name of Computation	Square root extraction	Computation Instruction Symbol	**SQT**

Explanation of Operation

The SQT instruction takes the input value (A or S1) to be "0" without it being cut at the low cutoff setpoint when it is < 0.01. There is no hysteresis at the low cutoff setpoint.
When input value (A or S1) is ≥ 0.01:
Function block programming: √A is output (J).
Text programming: √S1 is output (S1 after computation).

Computing data is a 4-byte floating point number.

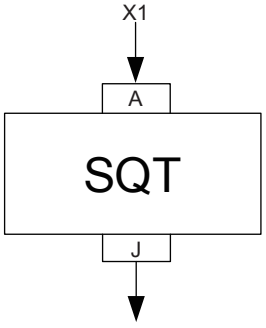


Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	√A

(Example)
X1: Analog input 1



Text Programming

(Example) Square root computation is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
SQT	√X1	a	b	c	d	Result of square root extraction after low cutoff processing

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

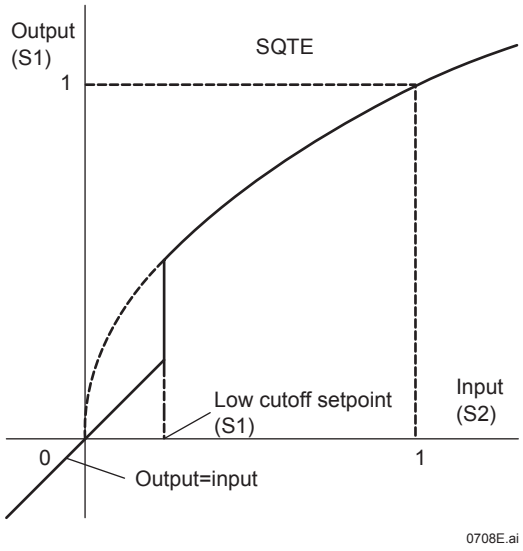
SQTE

Category	Basic computations		
Name of Computation	Square root extraction with variable low cutoff	Computation Instruction Symbol	**SQTE**

Explanation of Operation

The SQTE instruction takes the low cutoff setpoint (E or S1) to be "0" when it is < 0. There is no hysteresis at the low cutoff setpoint.
When the input value (A or S2) ≤ low cutoff setpoint (E or S1), the input value is output.
When the input value (A or S2) > low cutoff setpoint (E or S1), √A or √S2 is output (J or S1 after computation)

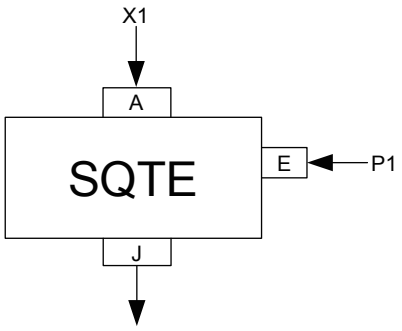
Computing data is a 4-byte floating point number.



Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Low cutoff setpoint
Output	**J**	√A lower cut by E



(Example)
X1: Analog input 1
P1: Variable register 1

When setting 10% on scale 0 to 1 of low cutoff setpoint, set "0.1"

Text Programming

(Example) Square root computation with low cutoff is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD P1	P1	X1	a	b	c	Reads the low cutoff point.
SQTE	√X1	a	b	c	c	Result of square root extraction after low cutoff processing

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

ABS

Category	Basic computations		
Name of Computation	Absolute value	Computation Instruction Symbol	**ABS**

Explanation of Operation

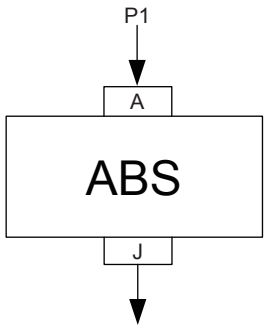
The ABS instruction outputs (J or S1 after computation) the absolute value of the input value (A or S1).

Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Absolute value of A



(Example)
P1: Variable register 1

Text Programming

(Example) Absolute value of variable register 1 is output.

Text Program	S1	S2	S3	S4	S5	Explanation
LD P1	P1	a	b	c	d	Reads the variable register 1.
ABS	|P1|	a	b	c	d	Absolute value

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

HSL

Category	Basic computations		
Name of Computation	High selector	Computation Instruction Symbol	**HSL**

Explanation of Operation

The HSL instruction outputs the following:

When input value 2 (B or S1) > input value 1 (A or S2), input value 2 (B or S1) is output (J or S1 after computation).
When input value 2 (B or S1) ≤ input value 1 (A or S2), input value 1 (A or S2) is output (J or S1 after computation).
(The values of registers S3 to S5 are pushed up.)

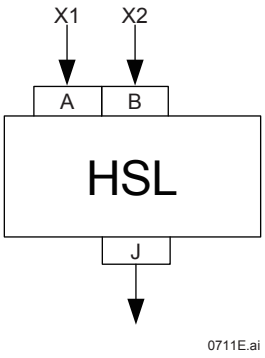
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	Larger value of A and B

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 and analog input 2 are compared, and the larger of the two values is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
HSL	When X2>X1, X2 When X2≤X1, X1	a	b	c	c	Compares the values of X1 and X2, and takes the larger value.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LSL

Category	Basic computations		
Name of Computation	Low selector	Computation Instruction Symbol	**LSL**

Explanation of Operation

The LSL instruction outputs the following:

When input value 2 (B or S1) > input value 1 (A or S2), input value 1 (A or S2) is output (J or S1 after computation).
When input value 2 (B or S1) ≤ input value 1 (A or S2), input value 2 (B or S1) is output (J or S1 after computation).
(The values of registers S3 to S5 are pushed up.)

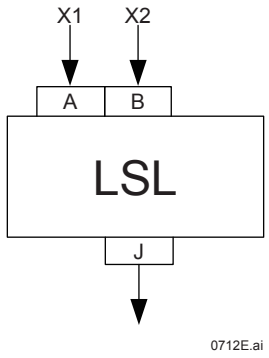
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	Smaller value of A and B

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 and analog input 2 are compared, and the smaller of the two values is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LSL	When X2>X1, X1 When X2≤X, X2	a	b	c	c	Compares the values of X1 and X2, and takes the smaller value.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

HLM

Category	Basic computations		
Name of Computation	High Limiter	Computation Instruction Symbol	**HLM**

Explanation of Operation

The HLM instruction outputs the following:

When the high limit setpoint (E or S1) > input value (A or S2), input value (A or S2) is output (J or S1 after computation).
When the high limit setpoint (E or S1) ≤ input value (A or S2), high limit setpoint (E or S1) is output (J or S1 after computation).
(The values of registers S3 to S5 are pushed up.)

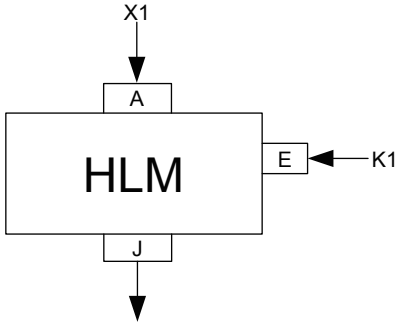
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	High limit setpoint
Output	**J**	Value limited to E or less

(Example)
X1: Analog input 1
K1: Constant register 1



Text Programming

(Example) The High Limiter is applied to analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD K1	K1	X1	a	b	c	Reads the high limit setpoint.
HLM	When K1>X1, X1 When K1≤X1, K1	a	b	c	c	Computation result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LLM

Category	Basic computations		
Name of Computation	Low Limiter	Computation Instruction Symbol	**LLM**

Explanation of Operation

The LLM instruction outputs the following:

When the low limit setpoint (E or S1) > input value (A or S2), low limit setpoint (E or S1) is output (J or S1 after computation).
When the low limit setpoint (E or S1) ≤ input value (A or S2), input value (A or S2) is output (J or S1 after computation).
(The values of registers S3 to S5 are pushed up.)

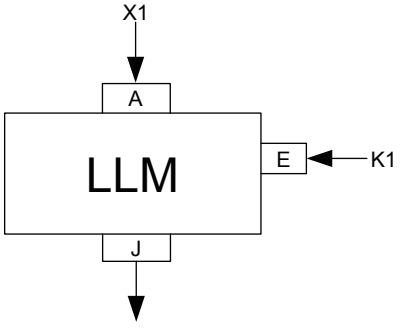
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Low limit setpoint
Output	**J**	Value limited to E or more

(Example)
X1: Analog input 1
K1: Constant register 1



Text Programming

(Example) The Low Limiter is applied to analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD K1	K1	X1	a	b	c	Reads the low limit setpoint.
LLM	When K1>X1, K1 When K1≤X1, X1	a	b	c	c	Computation result

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

SCAL

Category	Basic computations		
Name of Computation	Scaling	Computation Instruction Symbol	**SCAL**

Explanation of Operation

[Operation Formula]

Scaled value (J or S1 after computation) = {input value (A or S4) × (100% value of scale (E or S3) - 0% value of scale (F or S2))+ 0% value of scale (F or S2)}/10^n

(n=Decimal point position (0 to 4))

(S5 is pushed up three places.)

The decimal point position (S1) is handled as follows:

When S1 <0.5, 0
When 0.5 ≤ S1 <1.5, 1
When 1.5 ≤ S1 <2.5, 2
When 2.5 ≤ S1 <3.5, 3
When 3.5 ≤ S1 <4.5, 4
When 4.5 ≤ S1 , 0

The decimal point position (S1) is rounded off.

Computing data is a 4-byte floating point number.

The "computing overflow" error occurs when the computation has exceeded the computation effective range.

The computation is limited to the computation effective range.

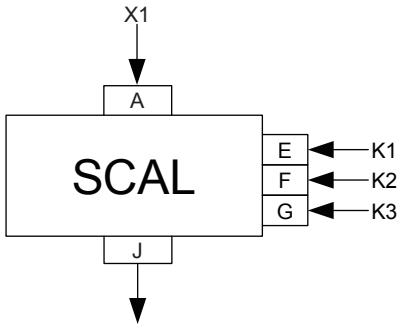
After input values are scaled by the scaling (SCAL) instruction, return to the original scale (0 to 1) by the normalization (NORM) instruction before executing the control module (BSC1, BSC2, CSC, SSC).

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	100% scale value
	F	0% scale value
	G	Decimal point position
Output	**J**	Scaled value

(Example)
X1: Analog input 1
K1: Constant register 1
K2: Constant register 2
K3: Constant register 3



Text Programming

(Example) Analog input 1 is scaled.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD K1	K1	X1	a	b	c	Reads the 100% scale value.
LD K2	K2	K1	X1	a	b	Reads the 0% scale value.
LD K3	K3	K2	K1	X1	a	Reads the decimal point position.
SCAL	(X1 × (K1 - K2) + K2) ÷ 10^{K3}	a	a	a	a	{Input value × (100% value of scale - 0% value of scale) + 0% value of scale}/10^{K3}

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

NORM

Category	Basic computations		
Name of Computation	Normalization	Computation Instruction Symbol	**NORM**

Explanation of Operation

[Operation Formula]

Normalized value (J or S1 after computation) = {(input value × 10^n) (A or S4) - 0% value of scale (F or S2)} ÷ {100% value of scale (E or S3) - 0% value of scale (F or S2)}

(n=Decimal point position (0 to 4))

(S5 is pushed up three places.)

The decimal point position (S1) is handled as follows:

When S1 <0.5, 0
When 0.5 ≤ S1 <1.5, 1
When 1.5 ≤ S1 <2.5, 2
When 2.5 ≤ S1 <3.5, 3
When 3.5 ≤ S1 <4.5, 4
When 4.5 ≤ S1 , 0

The decimal point position (S1) is rounded off.

Computing data is a 4-byte floating point number.

The "computing overflow" error occurs when the computation has exceeded the computation effective range.

The computation is limited to the computation effective range.

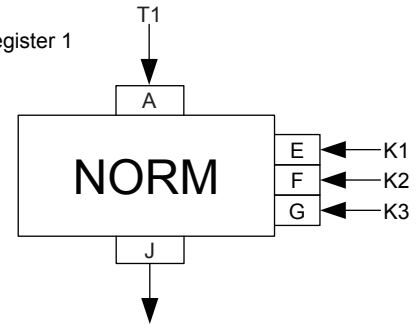
After input values are scaled by the scaling (SCAL) instruction, return to the original scale (0 to 1) by the normalization (NORM) instruction before executing the control module (BSC1, BSC2, CSC, SSC).

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	100% scale value
	F	0% scale value
	G	Decimal point position
Output	**J**	Normalized value

(Example)
T1: Temporary memory register 1
K1: Constant register 1
K2: Constant register 2
K3: Constant register 3



0718E.ai

Text Programming

(Example) Temporary memory register 1 is normalized.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
LD K1	K1	T1	a	b	c	Reads the 100% scale value.
LD K2	K2	K1	T1	a	b	Reads the 0% scale value.
LD K3	K3	K2	K1	T1	a	Reads the decimal point position.
NORM	((T1 × 10^{K3} – K2)÷ (K1–K2)	a	a	a	a	{(Input value × 10^{K3}) - 0% value of scale} ÷ (100% value of scale-0% value of scale)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LN

Category	Basic computations		
Name of Computation	Natural logarithm	Computation Instruction Symbol	**LN**

Explanation of Operation

[Operation Formula]

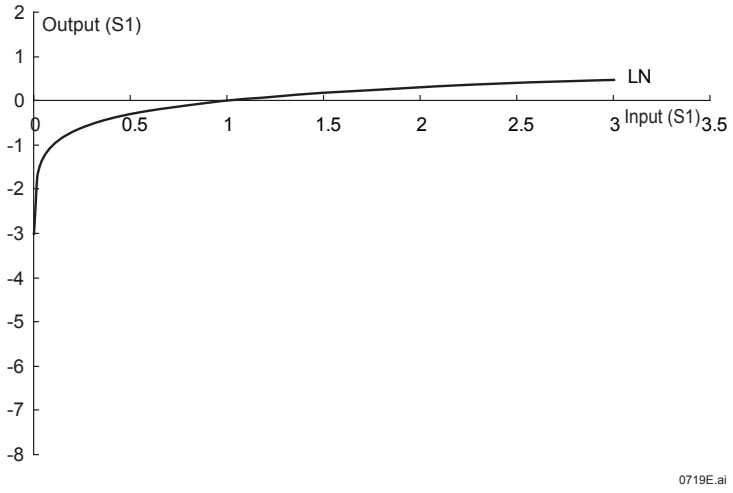
Function block programming:

Output(J) = $\log_e$(A)

Text programming:

S1 after computation = $\log_e$(S1)

The LN instruction calculates the natural logarithm taking the input value (A or S1) before computation as the exponential.
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the input value (A or S1) ≤ 0 or when the computation has exceeded the computation effective range.
When a computing overflow occurs, output remains at either J or S1.

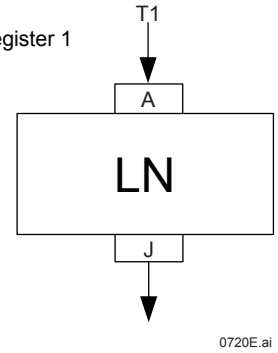


Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Natural logarithm of A

(Example)
T1: Temporary memory register 1



Text Programming

(Example) The natural logarithm is calculated with temporary memory register 1 taken as the exponential.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
LN	$\log_e$ (T1)	a	b	c	d	Natural logarithm

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LOG

Category	Basic computations		
Name of Computation	Common logarithm	Computation Instruction Symbol	**LOG**

Explanation of Operation

[Operation Formula]

Function block programming:

Output(J) = $\log_{10}$(A)

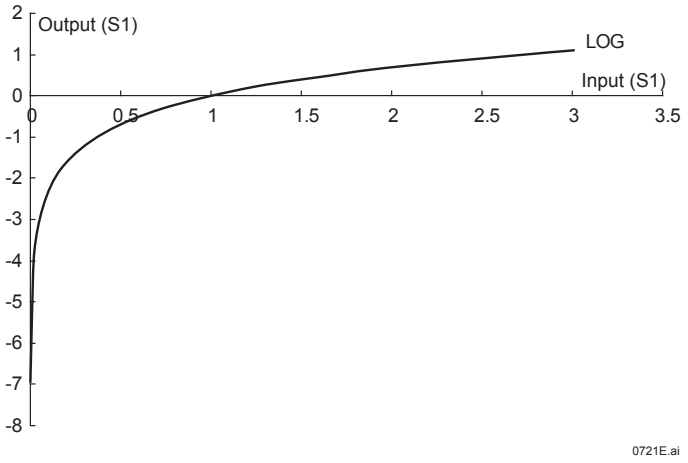
Text programming:

S1 after computation = $\log_{10}$(S1)

The LOG instruction calculates the common logarithm taking the input value (A or S1) before computation as the exponential. Computing data is a 4-byte floating point number.

The "computing overflow" error occurs when the input value (A or S1) ≤ 0 or when the computation has exceeded the computation effective range.

When a computing overflow occurs, output remains at either J or S1.

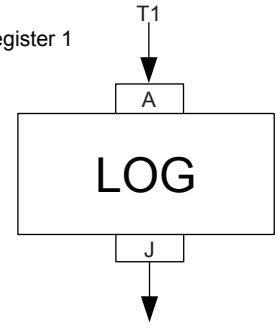


Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Common logarithm of A

(Example)
T1: Temporary memory register 1



Text Programming

(Example) The common logarithm is calculated with temporary memory register 1 taken as the exponential.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
LOG	$\log_{10}$ (T1)	a	b	c	d	Common logarithm

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

EXP

Category	Basic computations		
Name of Computation	Exponential	Computation Instruction Symbol	**EXP**

Explanation of Operation

[Operation Formula]

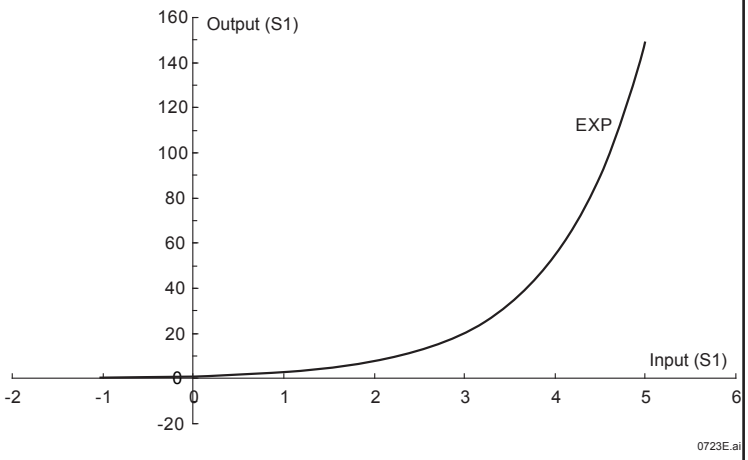
Function block programming:

Output (J) = e^{A}

Text programming:

S1 after computation = e^{S1}

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

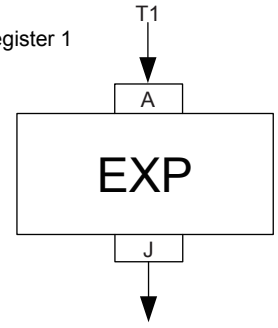


Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	e to the power of A

(Example)
T1: Temporary memory register 1



Text Programming

(Example) The logarithm of temporary memory register 1 is calculated.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
EXP	e^{T1}	a	b	c	d	e to the power of T1

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PWR

Category	Basic computations		
Name of Computation	Power	Computation Instruction Symbol	**PWR**

Explanation of Operation

[Operation Formula]
Function block programming: Output (J) = A^E
Text programming: S1 after computation = $S2^{S1}$
(After computation, S3 to S5 are pushed up.)

A computing overflow occurs in the following instances:
The base (A or S2) = 0.0 and exponential (E or S1) is ≤ 0.0.
The base < 0.0 and the exponential is not an integer value.
The computation has exceeded the computation effective range.

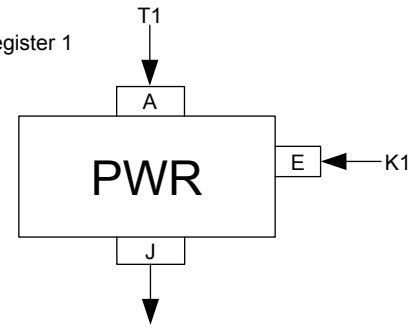
When a computing overflow occurs, output (J or S1 after computation) becomes 0.0.
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Base
	E	Exponential
Output	**J**	A to the power of E

(Example)
T1: Temporary memory register 1
K1: Constant register 1



Text Programming

(Example) Temporary memory register 1 undergoes multiplication by the power of constant register 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the base.
LD K1	K1	T1	a	b	c	Reads the exponential.
PWR	$T1^{K1}$	a	b	c	c	T1 to the power of K1

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TCMP1

Category	Basic computations		
Name of Computation	Temperature compensation (°C)	Computation Instruction Symbol	**TCMP1**

Explanation of Operation

<Before computation>
Reference temperature (°C) (E or S1)
Temperature (°C) (B or S2)
Flow (A or S3)

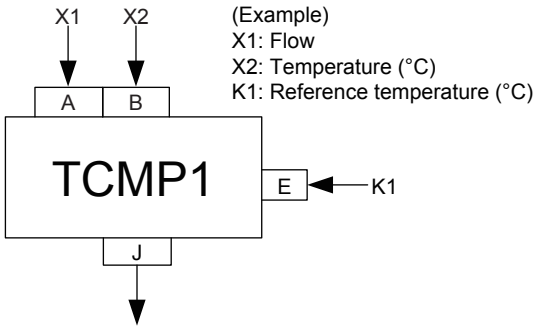
[Operation Formula]
Function block programming: Output (J) = Flow (A) × (reference temperature (E) + 273.15) ÷ (temperature (B) + 273.15)
Text programming: S1 after computation = Flow (S3) × (reference temperature (S1) + 273.15) ÷ (temperature (S2) + 273.15)
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Temperature (°C)
	E	Reference temperature (°C)
Output	**J**	Flow × (reference temperature + 273.15) ÷ (temperature + 273.15)



Text Programming

(Example) Analog input 1 (flow) undergoes temperature compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the temperature (X2).
LD K1	K1	X2	X1	a	b	Reads the reference temperature (K1).
TCMP1	X1×(K1+273.15)÷(X2+273.15)	a	b	b	b	Flow after temperature compensation (°C)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TCMP2

Category	Basic computations		
Name of Computation	Temperature compensation (°F)	Computation Instruction Symbol	**TCMP2**

Explanation of Operation

<Before computation>
Reference temperature (°F) (E or S1)
Temperature (°F) (B or S2)
Flow (A or S3)

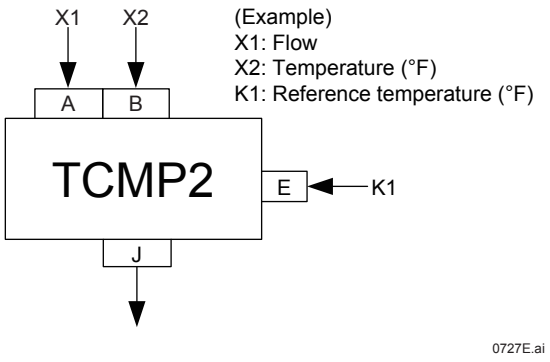
[Operation Formula]
Function block programming: Output (J) = Flow (A) × ((reference temperature (E) -32) ÷ 1.8 + 273.15) ÷ ((temperature (B) -32) ÷ 1.8 + 273.15)
Text programming: S1 after computation = Flow (S3) × ((reference temperature (S1) -32) ÷ 1.8 + 273.15) ÷ ((temperature (S2) -32) ÷ 1.8 + 273.15))
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Temperature (°F)
	E	Reference temperature (°F)
Output	**J**	Flow × ((reference temperature - 32) ÷ 1.8 + 273.15) ÷ ((temperature - 32) ÷ 1.8 + 273.15))



Text Programming

(Example) Analog input 1 (flow) undergoes temperature compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the temperature (X2)
LD K1	K1	X2	X1	a	b	Reads the reference temperature (K1)
TCMP2	X1×((K1+32)÷1.8+273.15)÷((X2+32)÷1.8+273.15)	a	b	b	b	Flow after temperature compensation (°F)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TCMP3

Category	Basic computations		
Name of Computation	Temperature compensation (K)	Computation Instruction Symbol	**TCMP3**

Explanation of Operation

<Before computation>
Reference temperature (K) (E or S1)
Temperature (K) (B or S2)
Flow (A or S3)

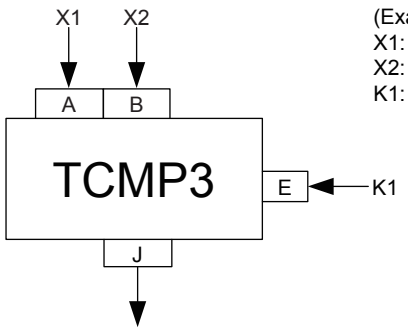
[Operation Formula]
Function block programming: Output (J) = Flow (A) × reference temperature (E) ÷ temperature (B)
Text programming: S1 after computation = Flow (S3) × reference temperature (S1) ÷ temperature (S2)
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Temperature (K)
	E	Reference temperature (K)
Output	**J**	Flow × reference temperature ÷ temperature



(Example)
X1: Flow
X2: Temperature (K)
K1: Reference temperature (K)

0728E.ai

Text Programming

(Example) Analog input 1 (flow) undergoes temperature compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the temperature (X2).
LD K1	K1	X2	X1	a	b	Reads the reference temperature (K1).
TCMP3	X1×K1÷X2	a	b	b	b	Flow after temperature compensation (K)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PCMP1

Category	Basic computations		
Name of Computation	Pressure compensation (MPa)	Computation Instruction Symbol	**PCMP1**

Explanation of Operation

<Before computation>
Reference pressure (MPa) (E or S1)
Pressure (MPa) (B or S2)
Flow (A or S3)

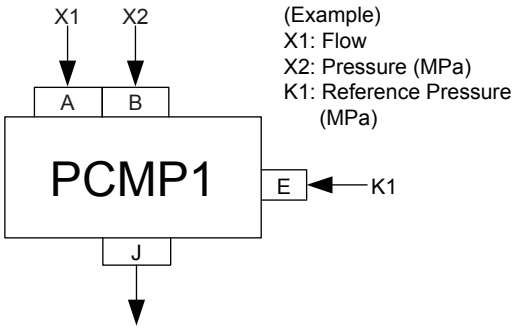
[Operation Formula]
Function block programming: Output (J) = Flow (A) × (pressure (B) + 0.101325) ÷ (reference pressure (E) + 0.101325)
Text programming: S1 after computation = Flow (S3) × (pressure (S2) + 0.101325) ÷ (reference pressure (S1) + 0.101325)
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Pressure (MPa)
	E	Reference pressure (MPa)
Output	**J**	Flow × (pressure + 0.101325) ÷ (reference pressure + 0.101325)



Text Programming

(Example) Analog input 1 (flow) undergoes pressure compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the pressure (X2).
LD K1	K1	X2	X1	a	b	Reads the reference pressure (K1).
PCMP1	X1×(X2 +0.101325)÷ (K1 +0.101325)	a	b	b	b	Flow after pressure compensation (MPa)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PCMP2

Category	Basic computations		
Name of Computation	Pressure compensation (kgf/cm^2)	Computation Instruction Symbol	**PCMP2**

Explanation of Operation

<Before computation>
Reference pressure (kgf/cm^2) (E or S1)
Pressure (kgf/cm^2) (B or S2)
Flow (A or S3)

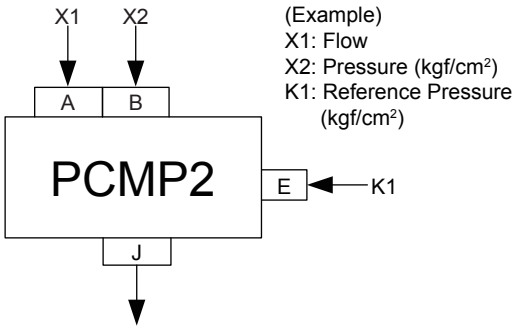
[Operation Formula]
Function block programming: Output (J) = Flow (A) × (pressure (B) + 1.03323) ÷ (reference pressure (E) + 1.03323)
Text programming: S1 after computation = Flow (S3) × (pressure (S2) + 1.03323) ÷ (reference pressure (S1) + 1.03323)
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Pressure (kgf/cm^2)
	E	Reference pressure (kgf/cm^2)
Output	**J**	Flow × (pressure + 1.03323) ÷ (reference pressure + 1.03323)



Text Programming

(Example) Analog input 1 (flow) undergoes pressure compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the pressure (X2).
LD K1	K1	X2	X1	a	b	Reads the reference pressure (K1).
PCMP2	X1×(X2+1.03323)÷(K1+1.03323)	a	b	b	b	Flow after pressure compensation (kgf/cm^2)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PCMP3

Category	Basic computations		
Name of Computation	Pressure compensation (psi)	Computation Instruction Symbol	**PCMP3**

Explanation of Operation

<Before computation>
Reference pressure (psi=Pound per square inch) (E or S1)
Pressure (psi) (B or S2)
Flow (A or S3)

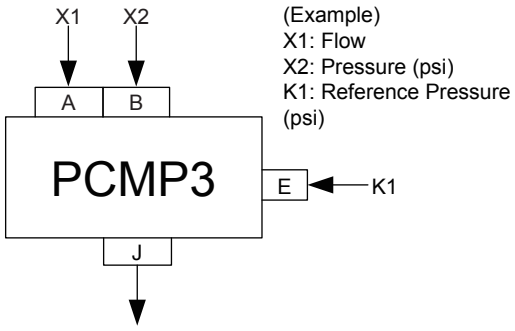
[Operation Formula]
Function block programming: Output (J) = Flow (A) × (pressure (B) + 14.6959) ÷ (reference pressure (E) + 14.6959)
Text programming: S1 after computation = Flow (S3) × (pressure (S2) + 14.6959) ÷ (reference pressure (S1) + 14.6959)
(S4 and S5 are pushed up two places.)

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Flow
	B	Pressure (psi)
	E	Reference pressure (psi)
Output	**J**	Flow × (pressure + 14.6959) ÷ (reference pressure + 14.6959)



Text Programming

(Example) Analog input 1 (flow) undergoes pressure compensation.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the flow (X1).
LD X2	X2	X1	a	b	c	Reads the pressure (X2).
LD K1	K1	X2	X1	a	b	Reads the reference pressure (K1).
PCMP3	X1×(X2+14.6959)÷(K1+14.6959)	a	b	b	b	Flow after pressure compensation (psi)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DIBCD

Category	Basic computations		
Name of Computation	Conversion from DI to BCD	Computation Instruction Symbol	**DIBCD**

Explanation of Operation

The DIBCD instruction reads the status of the number of DI points specified by the number of bits (F or S1) taking the DI specified by start DI number (E or S2) as the LSB, and outputs the result (J or S1 after computation) after converting the integer value converted to BCD as a floating number. (S3 to S5 are pushed up.)

0.5 ≤ start DI number (E or S2) < 10.5, 0.5 ≤ number of bits (F or S1) < 11.5 - (E or S2). Numbers past the decimal point are rounded to the nearest integer. Numbers outside of this range are not converted to BCD and "-1.0" is output as the result.
Numeric values are handled as follows. The first four numeric values from the DI specified by start DI number (E or S2) are the 1's digit, the next four are the 10's digit, and so forth.
When number of bits (F or S1) is not a multiple of four, the insufficient bits are handled as "0".
When the read DI status is not a BCD code, "-1.0" is output (J or S1) as the result.
Digital I/O 1 to 6 are handled as "0" when they are set as DO or are the DI7 to DI10 of the basic type (without expandable I/O).

Computing data is a 4-byte floating point number.

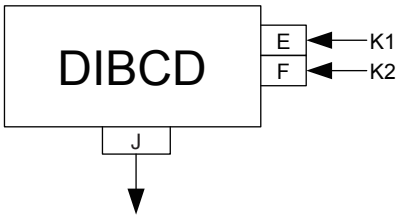
For example, when S2=1 and S1=10, DI1 to 4 are the 1's digit, DI5 to 8 are the 10's digit, and DI9 and 10 are taken to be the upper two bits and are handled as the 100's digit. In other words, the values that can be converted using all of DI1 to 10 are 0.0 to 399.0.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Parameters	**E**	Start DI number
	F	Number of bits
Output	**J**	The status of the number of DI points specified by F is read taking the DI specified by E as the LSB, and the value obtained by conversion to BCD is turned into a floating number.

(Example)
K1: Constant register 1
K2: Constant register 2



0732E.ai

Text Programming

(Example) The status of the specified digital input is read.

Text Program	S1	S2	S3	S4	S5	Explanation
LD K1	K1	a	b	c	d	Reads the start DI No.
LD K2	K2	K1	a	b	c	Reads the number of bits.
DIBCD	The DI status of the number of points specified by K2 is read taking the DI specified by K1 as the LSB, and the value obtained by conversion from BCD is turned into a floating number.	a	b	c	c	Performs conversion from BCD.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DOBCD

Category	Basic computations		
Name of Computation	Conversion from BCD to DO	Computation Instruction Symbol	**DOBCD**

Explanation of Operation

The DOBCD instruction converts the converted value (A or S3) to BCD, and outputs the results the four numerical values from DO specified by start DO number (E or S2) as the 1's digit, the next four numerical values as the 10's digit, and so forth. When conversion is performed successfully, the converted value (A or S3) is output (J or S1 after computation). (S4 and S5 are pushed up two places.)

0.5 ≤ start DO number (E or S2) < 10.5, 0.5 ≤ number of bits (F or S1) < 11.5 - (E or S2), 0.0 ≤ converted value (A or S3) < N ("N" is a value determined by F or S1, and is 399.5 when number of bits (F or S1) = 10). Numbers past the decimal point are rounded to the nearest integer. Numbers outside of this range are not output to DOn, and "-1.0" is output (J or S1).
Digital I/O 1 to 6 are not output when they are set as DI or are the DO7 to DO10 of the basic type (without expandable I/O).

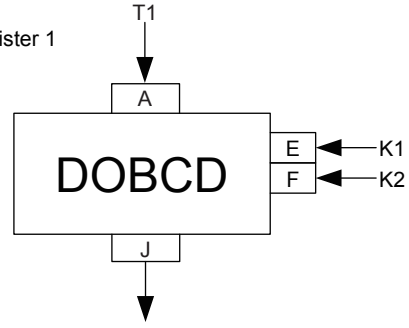
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Converted value
	E	Start DO number
	F	Number of bits
Output	**J**	Value of A

(Example)
T1: Temporary memory register 1
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example) BCD is output to the specified digital output.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the value to be converted.
LD K1	K1	T1	a	b	c	Reads the start DO number.
LD K2	K2	K1	T1	a	b	Reads the number of bits.
DOBCD	T1	a	b	b	b	Converted value

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DIBIN

Category	Basic computations		
Name of Computation	Conversion from DI to Binary	Computation Instruction Symbol	**DIBIN**

Explanation of Operation

The DIBIN instruction reads the status of the number of DI points specified by the number of bits (F or S1) taking the DI specified by start DI number (E or S2) as the LSB, and outputs the result (J or S1 after computation) after converting the integer value converted to binary as a floating number. (S3 to S5 are pushed up.)

0.5 ≤ start DI number (E or S2) < 10.5, 0.5 ≤ number of bits (F or S1) < 11.5 - (E or S2). Numbers past the decimal point are rounded to the nearest integer.
Numbers outside of this range are not converted to binary, and "-1.0" is output (J or S1).
The number of DI points specified by number of bits (F or S1) is converted to binary taking the DI specified by start DI number (E or S2) as the LSB.
Digital I/O 1 to 6 are handled as "0" when they are terminals set as DO or are the DI7 to DI10 of the basic type (without expandable I/O).

For example, when S2=1 and S1=10, the values that can be converted using all DI1 to DI10 are 0.0 to 1023.0.

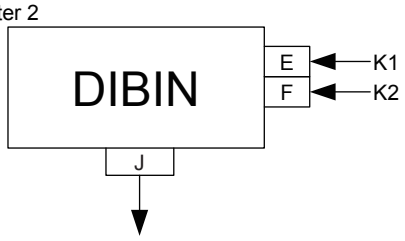
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Parameters	**E**	Start DI number
	F	Number of bits
Output	**J**	The status of the number of DI points specified by F is read taking the DI specified by E as the LSB, and the value obtained by conversion to binary is turned into a floating number.

(Example)
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example) The specified digital input is converted to binary.

Text Program	S1	S2	S3	S4	S5	Explanation
LD K1	K1	a	b	c	d	Reads the start DI No.
LD K2	K2	K1	a	b	c	Reads the number of bits.
DIBIN	The DI status of the number of points specified by K2 is read taking the DI specified by K1 as the LSB, and the value obtained by conversion to binary is turned into a floating number.	a	b	c	c	Performs conversion to binary.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DOBIN

Category	Basic computations		
Name of Computation	Conversion from Binary to DO	Computation Instruction Symbol	**DOBIN**

Explanation of Operation

The DOBIN instruction outputs the status of each bit after conversion of converted value (A or S3) to binary to number of bits (F or S1) from DO specified by start DO number (E or S2).
When conversion is performed successfully, the converted value (A or S3) is output (J or S1 after computation). (S4 and S5 are pushed up two places.)

0.5 ≤ start DO number (E or S2) < 10.5, 0.5 ≤ number of bits (F or S1) < 11.5 - (E or S2), 0.0 ≤ (A or S3 (converted value)) < (2 to the power of F or S1) - 0.5. Numbers past the decimal point are rounded to the nearest integer.
Numbers outside of this range are not output to DOn, and "-1.0" is output (J or S1).
Digital I/O 1 to 6 are not output when they are set as DI or are the DO7 to DO10 of the basic type (without expandable I/O).

For example, when E=0 and F=10 (DO1 to DO10 all are used), the range of A or S3 is 0.0 to 1023.0.

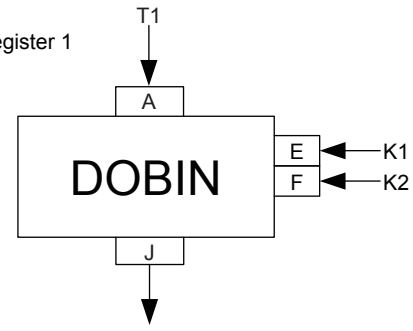
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Converted value
	E	Start DO number
	F	Number of bits
Output	**J**	Value of A

(Example)
T1: Temporary memory register 1
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example) Binary is converted to digital output.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the value to be converted.
LD K1	K1	T1	a	b	c	Reads the start DO number.
LD K2	K2	K1	T1	a	b	Reads the number of bits.
DOBIN	T1	a	b	b	b	Converted value

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAX2

Category	Basic computations		
Name of Computation	Maximum of 2 values	Computation Instruction Symbol	**MAX2**

Explanation of Operation

The MAX2 instruction outputs (J or S1 after computation) the larger value of input value 1 (A or S2) and input value 2 (B or S1).
(S3 to S5 are pushed up.)

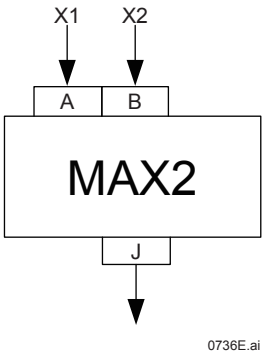
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	Maximum value of A and B

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) The larger value of analog input 1 and analog input 2 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
MAX2	X1 (when X1>X2)	a	b	c	c	Maximum of 2 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAX3

Category	Basic computations		
Name of Computation	Maximum of 3 values	Computation Instruction Symbol	**MAX3**

Explanation of Operation

X3 instruction outputs (J or S1 after computation) the largest value of input values 1 (A or S3) to input value 3 (C or S1).
(S4 and S5 are pushed up two places.)

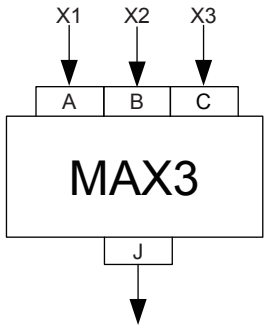
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
Output	**J**	Maximum value of A to C

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3



Text Programming

(Example) The largest value of analog input 1 to analog input 3 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
MAX3	X1 (when X1>X2>X3)	a	b	b	b	Maximum of 3 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAX4

Category	Basic computations		
Name of Computation	Maximum of 4 values	Computation Instruction Symbol	**MAX4**

Explanation of Operation

The MAX4 instruction outputs (J or S1 after computation) the largest value of input values 1 (A or S4) to input value 4 (D or S1).
(S5 is pushed up three places.)

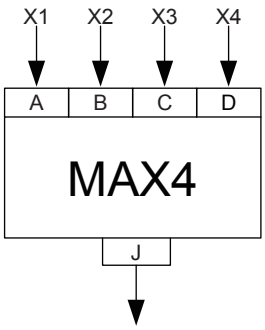
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	Maximum value of A to D

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3
X4: Analog input 4



Text Programming

(Example) The largest value of analog input 1 to analog input 4 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
LD X4	X4	X3	X2	X1	a	Reads the analog input 4.
MAX4	X1 (when X1>X2>X3>X4)	a	a	a	a	Maximum of 4 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MIN2

Category	Basic computations		
Name of Computation	Minimum of 2 values	Computation Instruction Symbol	**MIN2**

Explanation of Operation

The MIN2 instruction outputs (J or S1 after computation) the smaller value of input value 1 (A or S2) and input value 2 (B or S1).
(S3 to S5 are pushed up.)

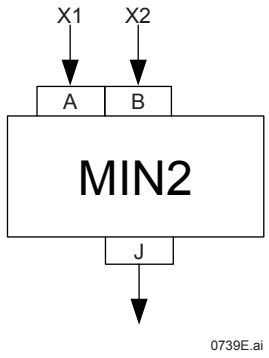
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	Minimum value of A and B

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) The smaller value of analog input 1 and analog input 2 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
MIN2	X1 (when X1<X2)	a	b	c	c	Minimum of 2 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MIN3

Category	Basic computations		
Name of Computation	Minimum of 3 values	Computation Instruction Symbol	**MIN3**

Explanation of Operation

The MIN3 instruction outputs (J or S1 after computation) the smallest value of input values 1 (A or S3) to input value 3 (C or S1).
(S4 and S5 are pushed up two places.)

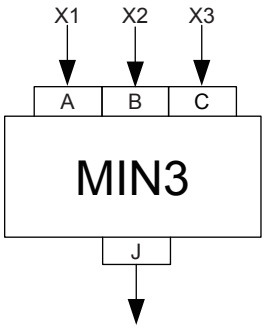
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
Output	**J**	Minimum value of A to C

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3



Text Programming

(Example) The smallest value of analog input 1 to analog input 3 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
MIN3	X1 (when X1<X2<X3)	a	b	b	b	Minimum of 3 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MIN4

Category	Basic computations		
Name of Computation	Minimum of 4 values	Computation Instruction Symbol	**MIN4**

Explanation of Operation

The MIN4 instruction outputs (J or S1 after computation) the smallest value of input values 1 (A or S4) to input value 4 (D or S1).
(S5 is pushed up three places.)

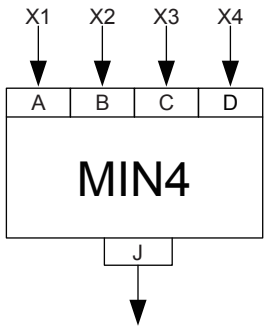
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	Minimum value of A to D

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3
X4: Analog input 4



Text Programming

(Example) The smallest value of analog input 1 to analog input 4 is selected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
LD X4	X4	X3	X2	X1	a	Reads the analog input 4.
MIN4	X1 (when X1<X2<X3<X4)	a	a	a	a	Minimum of 4 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

AVE2

Category	Basic computations		
Name of Computation	Average of 2 values	Computation Instruction Symbol	**AVE2**

Explanation of Operation

The AVE2 instruction outputs (J or S1 after computation) the average value of input value 1 (A or S2) and input value 2 (B or S1).
(S3 to S5 are pushed up.)

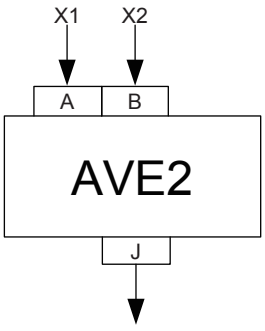
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	(A+B)÷2

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) The average value of analog input 1 and analog input 2 is calculated.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
AVE2	(X1+X2)÷2	a	b	c	c	Average of 2 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

AVE3

Category	Basic computations		
Name of Computation	Average of 3 values	Computation Instruction Symbol	**AVE3**

Explanation of Operation

The AVE3 instruction outputs (J or S1 after computation) the average value of input value 1 (A or S3) to input value 3 (C or S1).
(S4 and S5 are pushed up two places.)

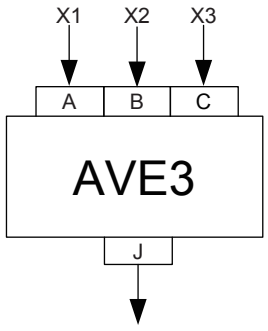
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
Output	**J**	(A+B+C)÷3

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3



Text Programming

(Example) The average value of analog input 1 to analog input 3 is calculated.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
AVE3	(X1+X2+X3)÷3	a	b	b	b	Average of 3 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

AVE4

Category	Basic computations		
Name of Computation	Average of 4 values	Computation Instruction Symbol	**AVE4**

Explanation of Operation

The AVE4 instruction outputs (J or S1 after computation) the average value of input value 1 (A or S4) to input value 4 (D or S1).
(S5 is pushed up three places.)

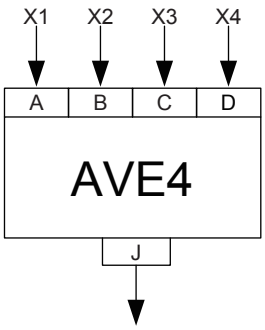
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	(A+B+C+D)÷4

(Example)
X1: Analog input 1
X2: Analog input 2
X3: Analog input 3
X4: Analog input 4



Text Programming

(Example) The average value of analog input 1 to analog input 4 is calculated.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads the analog input 1.
LD X2	X2	X1	a	b	c	Reads the analog input 2.
LD X3	X3	X2	X1	a	b	Reads the analog input 3.
LD X4	X4	X3	X2	X1	a	Reads the analog input 4.
AVE4	(X1+X2+X3+X4)÷4	a	a	a	a	Average of 4 values

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

INC

Category	Basic computations		
Name of Computation	Increment	Computation Instruction Symbol	**INC**

Explanation of Operation

The INC instruction adds "1.0" to the input value (A or S1) and outputs (J or S1 after computation) the result.
(S2 to S5 remain as they are.)

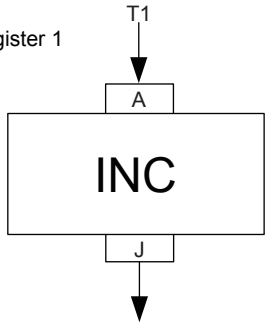
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	A+1.0

(Example)
T1: Temporary memory register 1



Text Programming

(Example) "1" is added to the temporary memory register.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
INC	T1+1.0	a	b	c	d	Increments the variable.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DEC

Category	Basic computations		
Name of Computation	Decrement	Computation Instruction Symbol	**DEC**

Explanation of Operation

The DEC instruction subtracts "1.0" from the input value (A or S1) and outputs (J or S1 after computation) the result. (S2 to S5 remain as they are.)

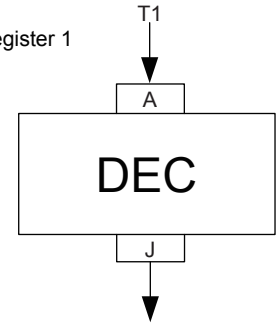
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	None	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	A–1.0

(Example)
T1: Temporary memory register 1



Text Programming

(Example) "1" is subtracted from the temporary memory register.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the temporary memory register 1.
DEC	T1–1.0	a	b	c	d	Decrements the variable

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

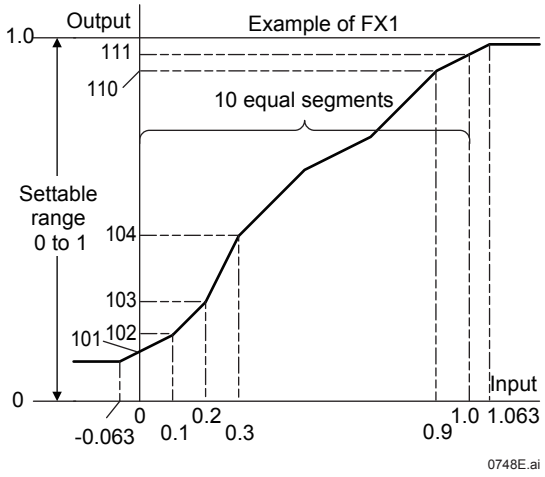
FX1, FX2

Category	Dynamic operations		
Name of Computation	10-segment linearizer function	Computation Instruction Symbol	**FX1, FX2**

Explanation of Operation

Input of 10-segment linearizer function is the input of 10 equal segments.
Set outputs to 101 to 111 of FX TABLE (in the case of FX1) and to 201 to 211 of FX TABLE (in the case of FX2) in the FX Table Setting Display or YSS1000 Setting Software for YS1000 Series.
The setting range is 0 to 1 (0 to 100%).
Input is limited at 1.063 (106.3%) and -0.063 (-6.3%).
When input is 1 (100%) or more, handling is the same as 0.9 (90%) to 1 (100%) as indicated by the line extension.
When input is smaller than 0, handling is the same as 0 (0%) to 0.1 (10%) as indicated by the line extension.
Conversion is 2-point linear interpolation.
Multiple instructions can be used in the same control period.

Computing data is a 4-byte floating point number.

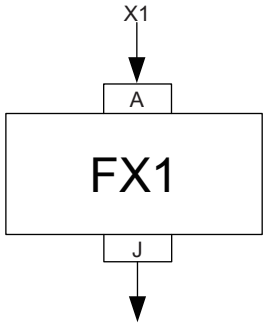


Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Input value after linear interpolation

(Example)
X1: Analog input 1



Text Programming

(Example) Linear interpolation is performed on analog input 1 in accordance with table FX1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
FX1	Input value after linear interpolation	a	b	c	d	Linear interpolation

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

IFX1, IFX2

Category	Dynamic operations		
Name of Computation	Inverse conversion of 10-segment linearizer function	Computation Instruction Symbol	**IFX1, IFX2**

Explanation of Operation

Set inputs in the FX Table Setting Display or YSS1000 Setting Software for YS1000 Series.
The divisions set in 101 to 111 (in the case of IFX1) and in 201 to 211 (in the case of IFX2) are used.
101 to 111 (201 to 211) are set in ascending order or descending order for use.
Input is limited by the value set at 101 (201) and 111 (211). At this time, outputs are 0% and 100%.
The value obtained after linear interpolation is output at the division point found by a 2-part search.
In other words, calculation is performed at the division point that was found even when 101 to 111 (201 to 211) are not in the ascending order or descending order.
Conversion is 2-point linear interpolation.
Multiple instructions can be used in the same control period.

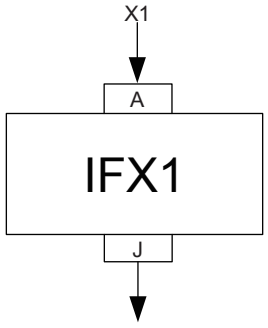
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Input value after linear interpolation

(Example)
X1: Analog input 1



Text Programming

(Example) Linear interpolation is performed on analog input 1 in accordance with table IFX1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
IFX1	Input value after linear interpolation	a	b	c	d	Linear interpolation

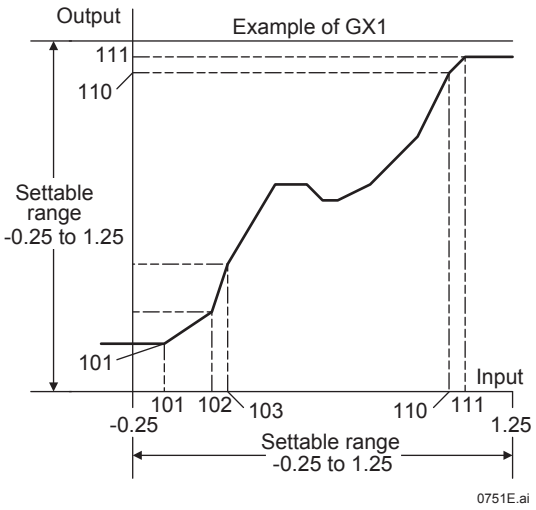
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

GX1, GX2

Category	Dynamic operations		
Name of Computation	Arbitrary segment linearizer function	Computation Instruction Symbol	**GX1, GX2**

Explanation of Operation

Set the arbitrary segment linearizer function in the GX Table Setting Display (in the case of GX1, INPUT1: 101 to 111 and OUTPUT1: 101 to 111, and in the case of GX2, INPUT2: 201 to 211 and OUTPUT2: 201 to 211), or the YSS1000 Setting Software for YS1000 Series (in the case of GX1, GXI101 to GXI111 and GXO101 to GXO111 of GX1 TABLE, and in the case of GX2, GXI201 to GXI211 and GXO201 to GXO211 of GX2 TABLE).
The setting range is -0.25 (-25.0%) to 1.25 (125.0%).
Input is limited at 101 (GXI101) and 111 (GXI111) in the case of GX1, and at 201 (GXI201) and 211 (GXI211) in the case of GX2.
When input is 101 (GXI101) or less, output 101 (GXO101) is output, and when input is 111 (GXI111) or more, output 111 (GXO111) is output. The same applies to GX2.
Conversion is 2-point linear interpolation.
Multiple instructions can be used in the same control period.



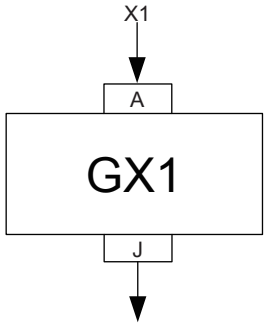
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Input value after linear interpolation

(Example)
X1: Analog input 1



Text Programming

(Example) Linear interpolation is performed on analog input 1 in accordance with table GX1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
GX1	Input value after linear interpolation	a	b	c	d	Linear interpolation

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

IGX1, IGX2

Category	Dynamic operations		
Name of Computation	Inverse conversion of arbitrary segment linearizer function	Computation Instruction Symbol	**IGX1, IGX2**

Explanation of Operation

The divisions set in the GX Table Setting Display (in the case of IGX1, OUTPUT1: 101 to 111, and in the case of IGX2, OUTPUT2: 201 to 211) or YSS1000 Setting Software for YS1000 Series (in the case of GX1, GXO101 to GXO111 of GX1 TABLE and in the case of GX2, GXO201 to GXO211 of GX2 TABLE) for use as the inputs.
Outputs are the values set in the GX Table Setting Display (in the case of IGX1, INPUT1: 101 to 111, and in the case of IGX2, INPUT2: 201 to 211) or YSS1000 Setting Software for YS1000 Series (in the case of GX1, GXI101 to GXI111 of GX1 TABLE and in the case of GX2, GXI201 to GXI211 of GX2 TABLE).

Input is limited at OUTPUT1: 101 (GXO101) and 111 (GXO111) in the case of IGX1, and at OUTPUT2: 201 (GXO201) and 211 (GXO211) in the case of IGX2.
When input is 101 (GXI101) or less, output 101 (GXO101) is output, and when input is 111 (GXI111) or more, output 111 (GXO111) is output. The same applies to IGX2.
The value obtained after linear interpolation is output at the division point found by a 2-part search. In other words, calculation is performed at the division point that was found even when outputs 101 (GXO101) to 111 (GXO111) and 201 (GXO201) to 211(GXO211) are not in the ascending order or descending order.
Conversion is 2-point linear interpolation.
Multiple instructions can be used in the same control period.

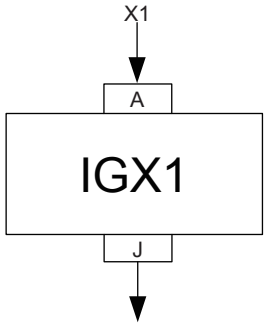
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Input value after linear interpolation

(Example)
X1: Analog input 1



Text Programming

(Example) Linear interpolation is performed on analog input 1 in accordance with table IGX1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
IGX1	Input value after linear interpolation	a	b	c	d	Linear interpolation

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

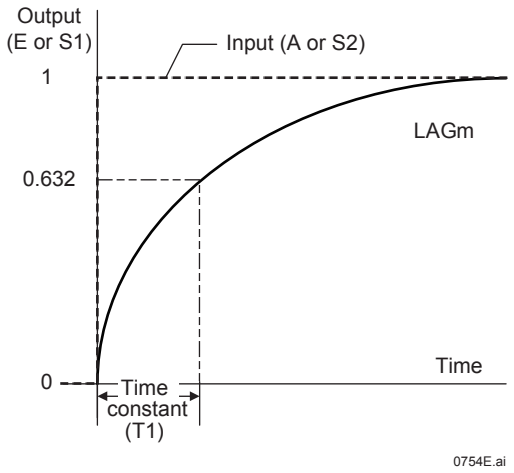
LAGm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	First order lag (second)	Computation Instruction Symbol	**LAGm (m=1 to 8)**

Explanation of Operation

Time constants 0 to 1 correspond to 0 to 100 seconds, and can be set up to 800 seconds.
The LAGm instruction is a dynamic operation. Only one instruction can be used for each machine number.

Output=Δt/(Δt + T) × (this time - previous) + previous
where,
Δt: control period
T: time constant (E or S1)
This time: input value this time (A or S2)
Previous: previous output value



0754E.ai

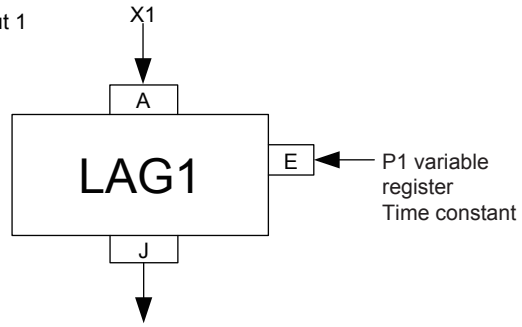
Time constants are not initialized even if they are changed.
At a cold start and when the time constant is 0, the previous value buffer is initialized by the input value this time, and the input value this time is output.
The previous value is backed up at a hot start.
Computing data is a 4-byte floating point number.
Note: The LAGm and LAGMm with the same machine number cannot be used simultaneously.
(Example) The LAG1 and LAGM1 cannot be used simultaneously.

Restriction on Number Used	One each of LAG1 to LAG8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Time constant (second)
Output	**J**	A after first order lag computation

(Example)
X1: Analog input 1



0755E.ai

Text Programming

(Example) First order lag computation (second) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Reads time constant (second).
LAG1	$(1-e^{-\frac{t}{P1}})$•X1	a	b	c	c	Input value after first order lag computation ΔT/(ΔT+T) × (this time - previous) + previous

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LAGMm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	First order lag (minute)	Computation Instruction Symbol	**LAGMm (m=1 to 8)**

Explanation of Operation

Time constants 0 to 1 correspond to 0 to 100 minutes, and can be set up to 800 minutes.
For the explanation of operation, refer to the explanation for the LAGm (m=1 to 8) instruction.

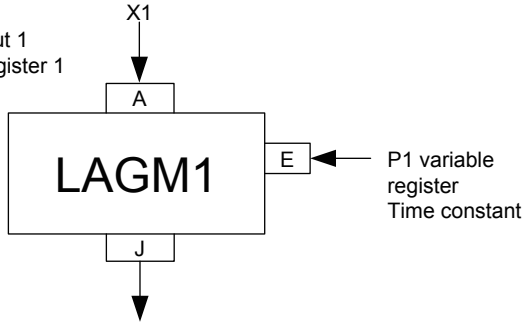
Note: The LAGMm and LAGm with the same machine number cannot be used simultaneously.
(Example) The LAG1 and LAGM1 cannot be used simultaneously.

Restriction on Number Used	One each of LAGM1 to LAGM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Time constant (minute)
Output	**J**	A after first order lag computation

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) First order lag computation (minute) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Reads time constant (minute).
LAGM1	$(1-e^{-\frac{t}{P1}})\cdot X1$	a	b	c	c	Input value after first order lag computation $\Delta T/(\Delta T+T)$ × (this time - previous) + previous

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LED1, LED2

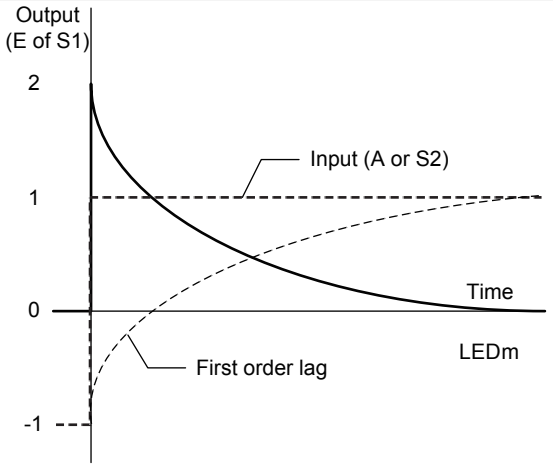
Category	Dynamic operations		
Name of Computation	Derivative (second)	Computation Instruction Symbol	**LED1, LED2**

Explanation of Operation

The LED1/LED2 instruction is a dynamic operation. Only one instruction can be used for each machine number.
Time constants 0 to 1 correspond to 0 to 100 seconds, and can be set up to 800 seconds.
This instruction is executed by incomplete derivative computation.

Output = this time - result of first order lag = this time - $\Delta t/(\Delta t + T) \times$ (this time - previous) + previous

where,
Δt: control period
T: time constant (E or S1)
This time: input value this time (A or S2)
Previous: previous output value



Time constants are not initialized even if they are changed.
At a cold start and when the time constant is 0, the previous value buffer is initialized by the input value this time, and the "0" is output.
Output is "0" if there is no change in the input.
The previous value is backed up at a hot start.

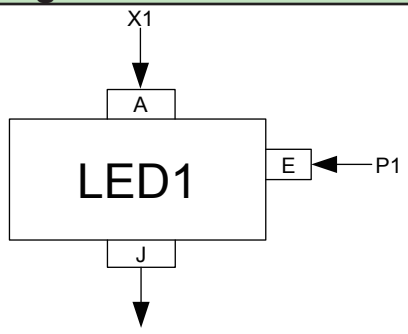
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.
Note: The LED and LEDM with the same machine number cannot be used simultaneously.
(Example) The LED1 and LEDM1 cannot be used simultaneously.

Restriction on Number Used	One each of LED1 and LED2 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Time constant (second)
Output	**J**	Input value after incomplete derivative computation of A

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Derivative computation (second) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Reads time constant (second).
LED1	$e^{-\frac{t}{P1}} \cdot \Delta X1$	a	b	c	c	Input value after incomplete derivative computation (input value this time) - (first order lag result)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LEDM1, LEDM2

Category	Dynamic operations		
Name of Computation	Derivative (minute)	Computation Instruction Symbol	**LEDM1, LEDM2**

Explanation of Operation

Time constants 0 to 1 correspond to 0 to 100 minutes, and can be set up to 800 minutes.
For the explanation of operation, refer to the explanation for the LED1 and LED2 instructions.

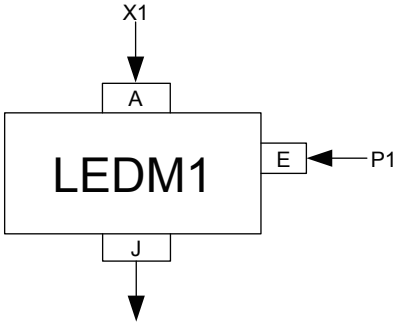
Note: The LEDM and LED with the same machine number cannot be used simultaneously.
(Example) The LED1 and LEDM1 cannot be used simultaneously.

Restriction on Number Used	One each of LEDM1 and LEDM2 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Time constant (minute)
Output	**J**	Input value after incomplete derivative computation of A

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Derivative computation (minute) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Reads time constant (minute).
LEDM1	$e^{-\frac{t}{P1}} \cdot \Delta X1$	a	b	c	c	Input value after incomplete derivative computation (input value this time) - (first order lag result)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DEDm (m=1 to 3)

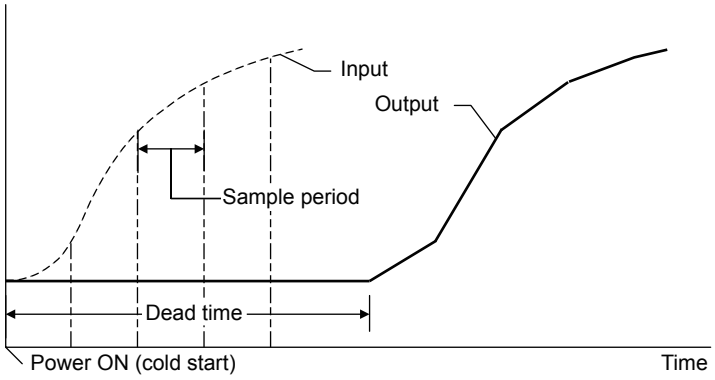
Category	Dynamic operations		
Name of Computation	Dead time (second)	Computation Instruction Symbol	**DEDm (m=1 to 3)**

Explanation of Operation

The DEDm instruction is a dynamic operation. Only one instruction can be used for each machine number.
Dead time 0 to 1 correspond to 0 to 1000 seconds, and can be set up to 8000 seconds.
The DEDm instruction stores data for each sample time to 20 buffers, and shifts the data successively to output it.
The sample period is "setting time/20." When a short time is set, all 20 buffers are not used.
When a long time is set, the data is not shifted at each control period.
At control periods when data is not shifted, the output values are interpolated.
When the power is turned ON (i.e. at a cold start), the buffers are initialized by the initial input value. Buffers are not initialized even if time constants are changed.
The buffer and time elapsed are backed up at a hot start.

Computing data is a 4-byte floating point number.

Note: The DEDm and DEDMm with the same machine number cannot be used simultaneously.
(Example) The DED1 and DEDM1 cannot be used simultaneously.

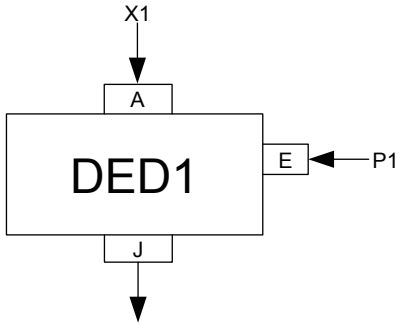


Restriction on Number Used	One each of DED1 to DED3 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Dead time (second)
Output	**J**	Value of A E-seconds previous

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Dead time computation (second) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets dead time (second).
DED1	$X1_{t - P1}$	a	b	c	c	Value of input P1 seconds previous

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DEDMm (m=1 to 3)

Category	Dynamic operations		
Name of Computation	Dead time (minute)	Computation Instruction Symbol	**DEDMm (m=1 to 3)**

Explanation of Operation

Dead time 0 to 1 correspond to 0 to 1000 minutes, and can be set up to 8000 minutes.
For the explanation of operation, refer to the explanation for the DEDm (m=1 to 3) instruction.

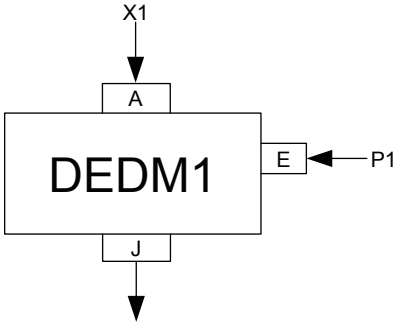
Note: The DEDMm and DEDm with the same machine number cannot be used simultaneously.
(Example) The DED1 and DEDM1 cannot be used simultaneously.

Restriction on Number Used	One each of DEDM1 to DEDM3 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Dead time (minute)
Output	**J**	Value of A E-minutes previous

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Dead time computation (minute) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets dead time (minute).
DEDM1	$X1_{t - P1}$	a	b	c	c	Value of input P1 minutes previous

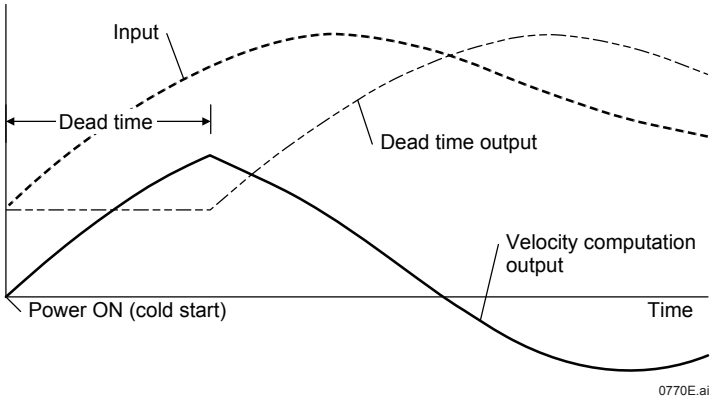
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

VELm (m=1 to 3)

Category	Dynamic operations		
Name of Computation	Velocity computation (second)	Computation Instruction Symbol	**VELm (m=1 to 3)**

Explanation of Operation

The VELm instruction is a dynamic operation.
Only one instruction can be used for each machine number.
Dead time 0 to 1 correspond to 0 to 1000 seconds, and can be set up to 8000 seconds.
The dead time is applied to subtract past input values by the amount of dead time (E or S1) from the current input value (A or S2).
As shown in the figure, the computation result is also sometimes a minus value.
The buffer and time elapsed are backed up at a hot start.



Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

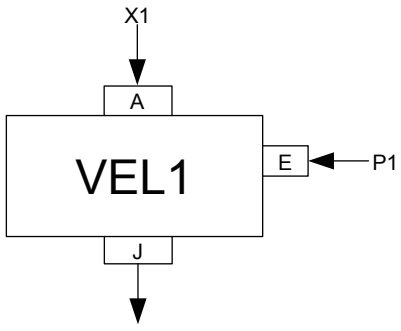
Note: The VELm and VELMm with the same machine number cannot be used simultaneously.
(Example) The VEL1 and VELM1 cannot be used simultaneously.

Restriction on Number Used	One each of VEL1 to VEL3 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Dead time (second)
Output	**J**	The value of A E-seconds previous is subtracted from the present value of A.

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Velocity computation (second) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets dead time (second).
VEL1	$X1_t$ - $X1_{t-P1}$	a	b	c	c	(Input value this time) - (dead time computation result)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

VELMm (m=1 to 3)

Category	Dynamic operations		
Name of Computation	Velocity computation (minute)	Computation Instruction Symbol	**VELMm (m=1 to 3)**

Explanation of Operation

Dead time 0 to 1 correspond to 0 to 1000 minutes, and can be set up to 8000 minutes.
For the explanation of operation, refer to the explanation for the VELm (m=1 to 3) instruction.

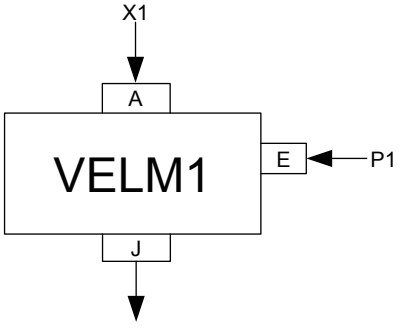
Note: The VELMm and VELm with the same machine number cannot be used simultaneously.
(Example) The VEL1 and VELM1 cannot be used simultaneously.

Restriction on Number Used	One each of VELM1 to VELM3 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
	E	Dead time (minute)
Output	**J**	The value of A E-minutes previous is subtracted from the present value of A.

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Velocity computation (minute) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets dead time (minute).
VELM1	$X1_t$ - $X1_{t-P1}$	a	b	c	c	(Input value this time) - (dead time computation result)

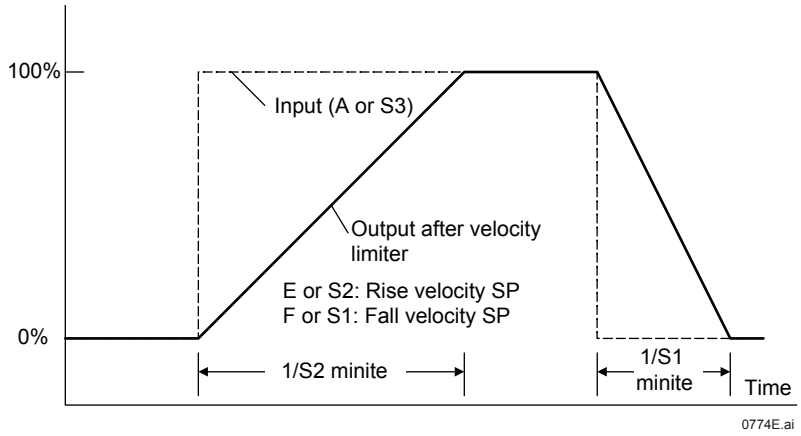
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

VLMm (m=1 to 6)

Category	Dynamic operations		
Name of Computation	Velocity limiter	Computation Instruction Symbol	**VLMm (m=1 to 6)**

Explanation of Operation

The VLMm instruction is a dynamic operation. Only one instruction can be used for each machine number.
The velocity limit values are 0 to 100%/minute for 0 to 1.
Setpoint values smaller than 0.001 (0.1%) are treated as "0."
When a value 7 (700%) or higher is set, the input value (A or S3) is output as it is without velocity restrictions being applied. (This is the "limit open feature.")
The previous value is backed up at a hot start.



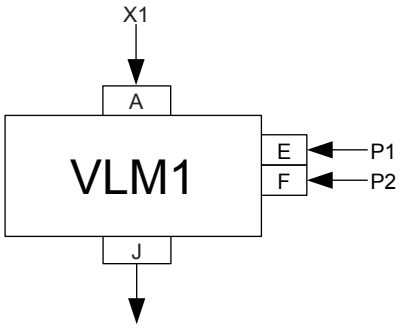
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

Restriction on Number Used	One each of VLM1 to VLM6 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Rise velocity restricted value
	F	Fall velocity restricted value
Output	**J**	Value of A whose velocity is restricted

(Example)
X1: Analog input 1
P1: Variable register 1
P2: Variable register 2



Text Programming

(Example) The velocity limiter of analog input 1 is set.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Rise velocity restricted value
LD P2	P2	P1	X1	a	b	Fall velocity restricted value
VLM1	Restricted X1	a	b	b	b	Input value whose velocity is restricted

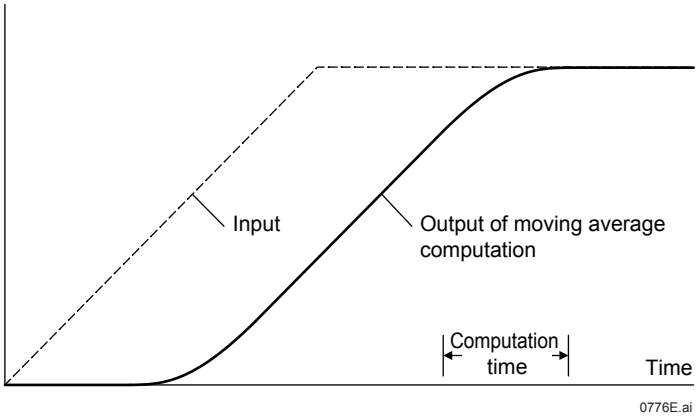
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAVm (m=1 to 3)

Category	Dynamic operations		
Name of Computation	Moving average computation (second)	Computation Instruction Symbol	**MAVm (m=1 to 3)**

Explanation of Operation

The MAVm instruction is a dynamic operation.
Only one instruction can be used for each machine number.
Computation time 0 to 1 correspond to 0 to 1000 seconds, and can be set up to 8000 seconds.
The dead time is applied to average past input values by the computation time (E or S1) from the current input value (A or S2), and output (J or S1 after computation) the result.
The buffer and time elapsed are backed up at a hot start.
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.



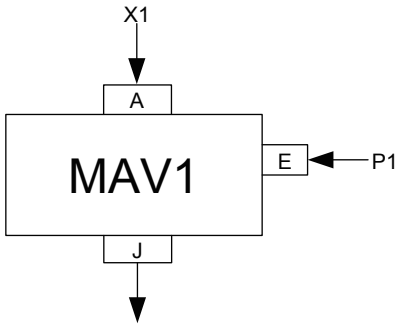
Note: The MAVm and MAVMm with the same machine number cannot be used simultaneously.
(Example) The MAV1 and MAVM1 cannot be used simultaneously.

Restriction on Number Used	One each of MAV1 to MAV3 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Computation time (second)
Output	**J**	Input value after moving average

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Moving average computation (second) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets the computation time (seconds).
MAV1	Input value after moving average	a	b	c	c	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAVMm (m=1 to 3)

Category	Dynamic operations		
Name of Computation	Moving average computation (minute)	Computation Instruction Symbol	**MAVMm (m=1 to 3)**

Explanation of Operation

Computation time 0 to 1 correspond to 0 to 1000 minutes, and can be set up to 8000 minutes.
For the explanation of operation, refer to the explanation for the MAVm (m=1 to 3) instruction.

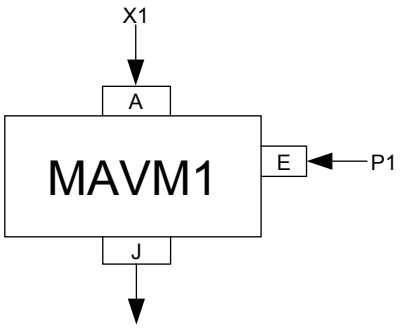
Note: The MAVMm and MAVm with the same machine number cannot be used simultaneously.
(Example) The MAV1 and MAVM1 cannot be used simultaneously.

Restriction on Number Used	One each of MAVM1 to MAVM3 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Computation time (minute)
Output	**J**	Input value after moving average

(Example)
X1: Analog input 1
P1: Variable register 1



Text Programming

(Example) Moving average computation (minute) is performed on analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD P1	P1	X1	a	b	c	Sets the computation time (minute).
MAVM1	Input value after moving average	a	b	c	c	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

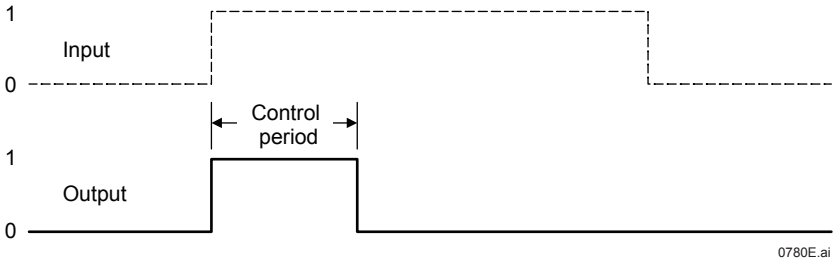
CCDm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	0 to 1 change detection	Computation Instruction Symbol	**CCDm (m=1 to 8)**

Explanation of Operation

The CCDm instruction is a dynamic operation. Only one instruction can be used for each machine number.
When the input value (A or S1) is 0 (less than 0.5) at the previous period and 1 (0.5 or more) this time, "1" is output (J or S1 after computation) for one control period. Otherwise, "0" is output (J or S1 after computation).

Computing data is a 4-byte floating point number.

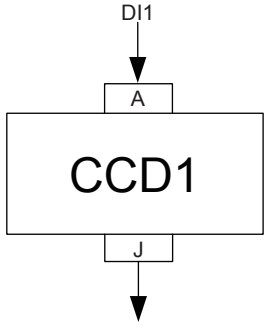


Restriction on Number Used	One each of CCD1 to CCD8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	When A<0.5 changes to A ≥0.5, 1 Otherwise, 0

(Example)
DI1: Digital input 1



Text Programming

(Example) The change in status of digital input 1 is detected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads the status change input.
CCD1	0/1	a	b	c	d	When previous DI1<0.5 and this time DI1≥0.5, 1 Otherwise, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

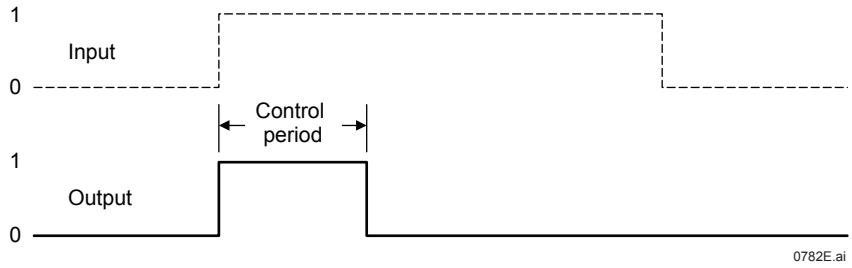
UEDGm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	0 to 1 change detection	Computation Instruction Symbol	**UEDGm (m=1 to 8)**

Explanation of Operation

Operation of the UEDGm instruction is the same as 0 to 1 change detection (CCDm) instruction. When the input value (A or S1) is 0 (less than 0.5) at the previous period and 1 (0.5 or more) this time, "1" is output (J or S1 after computation) for one control period. Otherwise, "0" is output (J or S1 after computation).

Computing data is a 4-byte floating point number.

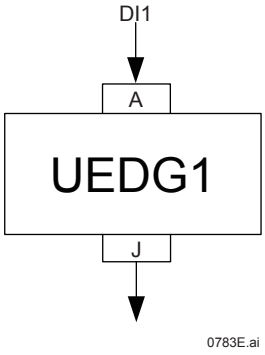


Restriction on Number Used	One each of UEDG1 to UEDG8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	When A<0.5 changes to A ≥0.5, 1 Otherwise, 0

(Example)
DI1: Digital input 1



Text Programming

(Example) The change in status of digital input 1 is detected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads the status change input.
UEDG1	0/1	a	b	c	d	When previous DI1<0.5 and this time DI1≥0.5, 1 Otherwise, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DEDGm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	1 to 0 change detection	Computation Instruction Symbol	**DEDGm (m=1 to 8)**

Explanation of Operation

The DEDGm instruction is a dynamic operation. Only one instruction can be used for each machine number.
When the input value (A or S1) is 1 (0.5 or more) at the previous period and 0 (less than 0.5) this time, "1" is output (J or S1 after computation) for one control period. Otherwise, "0" is output (J or S1 after computation).
The previous value is backed up at a hot start.

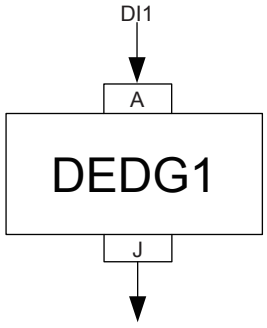
Computing data is a 4-byte floating point number.

Restriction on Number Used	One each of DEDG1 to DEDG8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	When A≥0.5 changes to A<0.5, 1 Otherwise, 0

(Example)
DI1: Digital input 1



Text Programming

(Example) The change in status of digital input 1 is detected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads the status change input.
DEDG1	1/0	a	b	c	d	1 to 0 change detection

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

EDGEm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Change detection	Computation Instruction Symbol	**EDGEm (m=1 to 8)**

Explanation of Operation

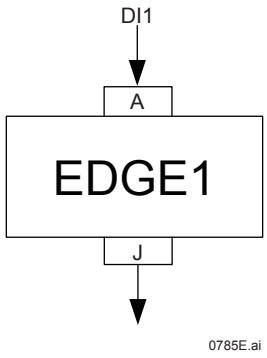
The EDGEm instruction is a dynamic operation. Only one instruction can be used for each machine number.
When the input value (A or S1) is 0 (less than 0.5) at the previous period and 1 (0.5 or more) this time, or is 1 (0.5 or more) at the previous period and 0 (less than 0.5) this time, "1" is output (J or S1 after computation) for one control period. Otherwise, "0" is output (J or S1 after computation).
The previous value is backed up at a hot start.

Computing data is a 4-byte floating point number.

Restriction on Number Used	One each of EDGE1 to EDGE8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	When A<0.5 changes to A ≥0.5 or A≥0.5 changes to A<0.5 , 1 Otherwise, 0



Text Programming

(Example) Both edges of digital input 1 are detected.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads the status change input.
EDGE1	1/0	a	b	c	d	When (previous DI1<0.5 and this time DI1≥0.5) or (previous DI1≥0.5 and this time DI1<0.5), 1 Otherwise, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TIMm (m=1 to 8)

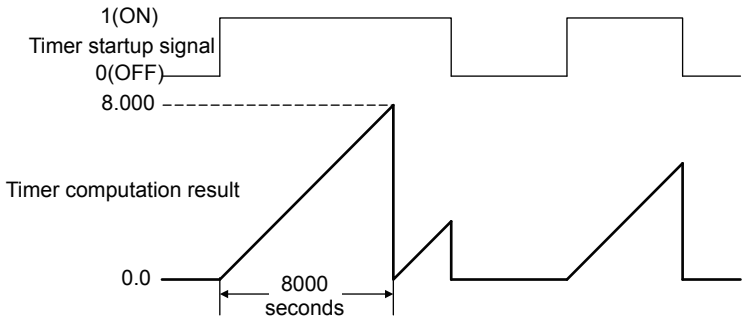
Category	Dynamic operations		
Name of Computation	Timer (second)	Computation Instruction Symbol	**TIMm (m=1 to 8)**

Explanation of Operation

The TIMm instruction is a dynamic operation. Only one instruction can be used for each machine number.
When the timer start signal (A or S1) is 0 (less than 0.5), "0" is output. When the timer start signal (A or S1) is 1 (0.5 or more), the time elapsed is output (J or S1 after computation). Time elapsed 0 to 1 is equivalent to 0 to 1000 seconds. The count is restarted from 0 seconds following 8000 seconds.
The previous value is backed up at a hot start.

Computing data is a 4-byte floating point number.

Note: The TIMm and TIMMm with the same machine number cannot be used simultaneously.
(Example) The TIM1 and TIMM1 cannot be used simultaneously.

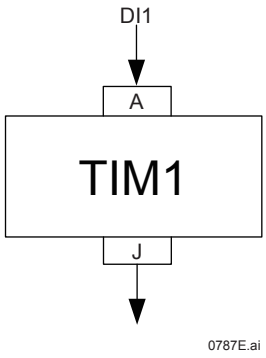


Restriction on Number Used	One each of TIM1 to TIM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Timer start signal A≥0.5; ON A<0.5; RESET
Output	**J**	Time elapsed since A turned ON (second)

(Example)
DI1: Digital input 1



Text Programming

(Example) Timer (second) operation is performed with digital input 1 taken as the trigger.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Timer ON/OFF
TIM1	Time elapsed (second)	a	b	c	d	The count is continued from 0 following 8.000 (8000 seconds).

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TIMMm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Timer (minute)	Computation Instruction Symbol	**TIMMm (m=1 to 8)**

Explanation of Operation

For the explanation of operation, refer to the explanation for the TIMm (m=1 to 8) instruction.
However, read a time unit "second" as "minute."

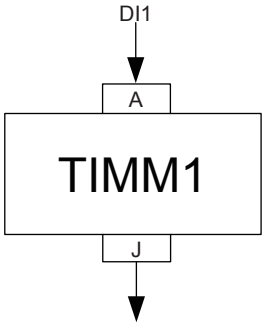
Note: The TIMMm and TIMm with the same machine number cannot be used simultaneously.
(Example) The TIM1 and TIMM1 cannot be used simultaneously.

Restriction on Number Used	One each of TIMM1 to TIMM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Timer start signal A≥0.5; ON A<0.5; RESET
Output	**J**	Time elapsed since A turned ON (minute)

(Example)
DI1: Digital input 1



Text Programming

(Example) Timer (minute) operation is performed with digital input 1 taken as the trigger.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Timer ON/OFF
TIMM1	Time elapsed (minute)	a	b	c	d	The count is continued from 0 following 8.000 (8000 minutes).

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TUPm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Time out (second)	Computation Instruction Symbol	**TUPm (m=1 to 8)**

Explanation of Operation

The internal timer is activated when timer ON/OFF (A or S2) is 0.5 or more.
When the time specified by the setting time (E or S1) has elapsed, "1" is output (J or S1 after computation) for one control period.
The setting range of the setting time (E or S1) is 0 to 4096.0 (equivalent to 4096000 seconds). The setting time is limited when it is out of range. When the setting time is "0", "1" is output at all times. (S3 to S5 are pushed up.)
Time elapsed 0 to 1 is equivalent to 0 to 1000 seconds.
The time elapsed is reset when timer ON/OFF (A or S2) is less than 0.5.
When timer ON/OFF (A or S2) is constantly 0.5 or more, the time elapsed is held at 4096.0 (4096000 seconds).
The time elapsed is backed up at a hot start.

Computing data is a 4-byte floating point number.

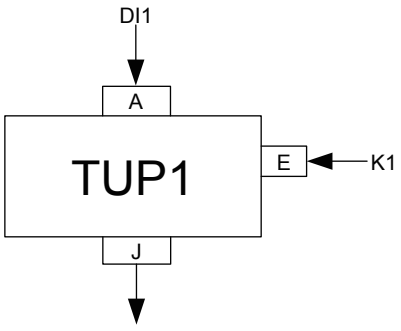
Note: The TUPm and TUPMm with the same machine number cannot be used simultaneously.
(Example) The TUP1 and TUPM1 cannot be used simultaneously.

Restriction on Number Used	One each of TUP1 to TUP8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Timer ON/OFF A≥0.5; ON A<0.5; OFF
	E	Setting time (second)
Output	**J**	1 output for only 1 period at a time out

(Example)
DI1: Digital input 1
K1: Constant register 1



0790E.ai

Text Programming

(Example) The flag is output after the setting time with digital input 1 taken as the trigger.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Timer ON/OFF
LD K1	K1	DI1	a	b	c	Setting time (second)
TUP1	"1" is output for only one control period after the preset time is reached.	a	b	c	c	Time out (second)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

TUPMm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Time out (minute)	Computation Instruction Symbol	**TUPMm (m=1 to 8)**

Explanation of Operation

For the explanation of operation, refer to the explanation for the TUPm (m=1 to 8) instruction.
However, read a time unit "second" as "minute."

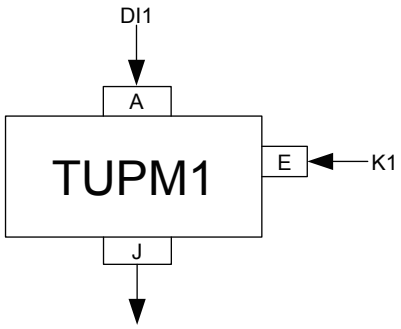
Note: The TUPMm and TUPm with the same machine number cannot be used simultaneously.
(Example) The TUP1 and TUPM1 cannot be used simultaneously.

Restriction on Number Used	One each of TUPM1 to TUPM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Timer ON/OFF A≥0.5; ON A<0.5; OFF
	E	Setting time (minute)
Output	**J**	1 output for only 1 period at a time out

(Example)
DI1: Digital input 1
K1: Constant register 1



Text Programming

(Example) The flag is output after the setting time with digital input 1 taken as the trigger.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Timer ON/OFF
LD K1	K1	DI1	a	b	c	Setting time (minute)
TUPM1	"1" is output for only one control period after the preset time is reached.	a	b	c	c	Time out (minute)

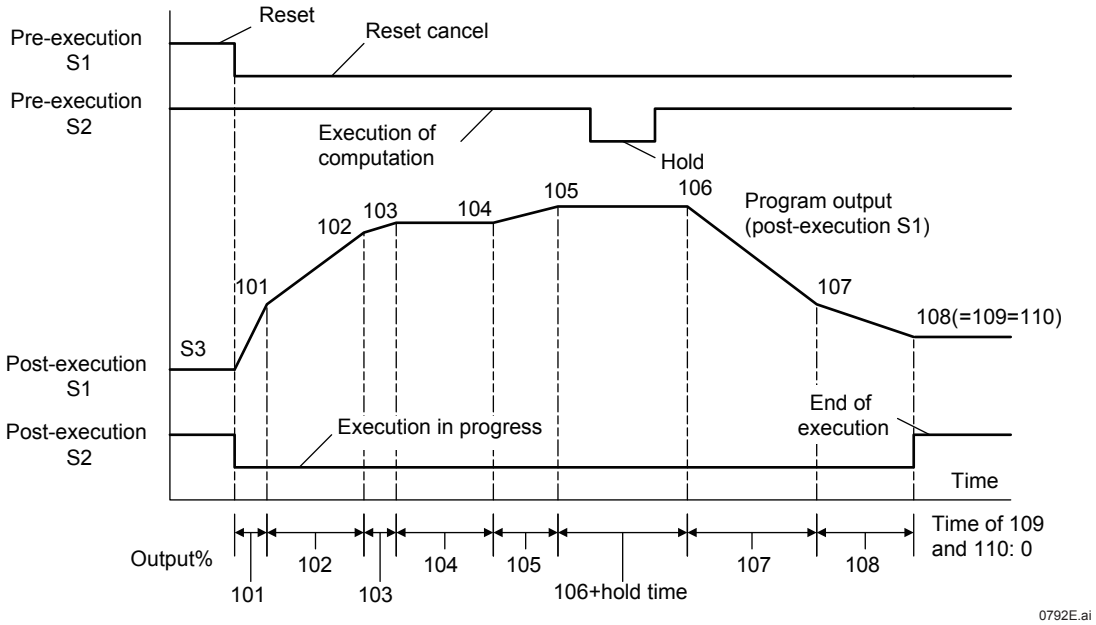
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PGMm_A, PGMm_B, PGMm (m=1, 2)

Category	Dynamic operations		
Name of Computation	Program setter (second)	Computation Instruction Symbol	**PGMm_A, PGMm_B, PGMm (m=1 to 2)**

Explanation of Operation

The PGMm_A, PGMm_B and PGMm instructions are dynamic operations. Only one instruction can be used for each machine number.
These instructions change the output value by each specified time span, and the points of the output curve are interpolated linearly.
Set the time span and output value of each point of the output curve in the Program-Setting-Unit Setting Display (in the case of PGM1, TIME1: 101 to 110 and OUTPUT1: 101 to 110, and in the case of PGM2, TIME2: 101 to 110 and OUTPUT2: 101 to 110) or in the YSS1000 Setting Software for YS1000 Series (in the case of PGM1, PGT101 to PGT110 at TIME1, and PGO101 to PGO110 at OUTPUT1, and in the case of PGM2, PGT101 to PGT110 at TIME2, and PGO101 to PGO110 at OUTPUT2). TIME and OUTPUT are limited at 0 to 9999 seconds and -0.25 to 1.25, respectively.
A single time span on the output curve is called a "segment."
When all ten points of the output curve are not used, set "0" as the time span and set output to the same value as the final used point of the output curve for unused points.
In the example in the figure below, the time span of the 9th and 10th segments is set to "0" and the output value is set to the same value as that of the 8th segment.
A reset is canceled when the reset input (S1 before execution) is less than 0.5, and is applied when the reset input is 0.5 or more.
In a reset status, the reset value (S3 before execution) is output as the program output (S1 after execution).
When calculation execution/hold (S2 before execution) is less than 0.5, a hold is applied to the output and output is held. When calculation execution/hold is 0.5 or higher, calculation is executed. In other words, computation progresses.
The reset value (S3 before execution) is limited at -0.25 to 1.25.
"0" is output as the end flag (S2 after execution) when the reset is canceled, and "1" is output when the 10th segment ends.
To restart a program once it has ended, set the reset input to 0.5 or more, and then cancel the reset (by setting the reset input to less than 0.5).
The time elapsed is backed up at a hot start.
Computing data is a 4-byte floating point number.
Note: The PGMm and PGMMm with the same machine number cannot be used simultaneously.
(Example) The PGM1 and PGMM1 cannot be used simultaneously.



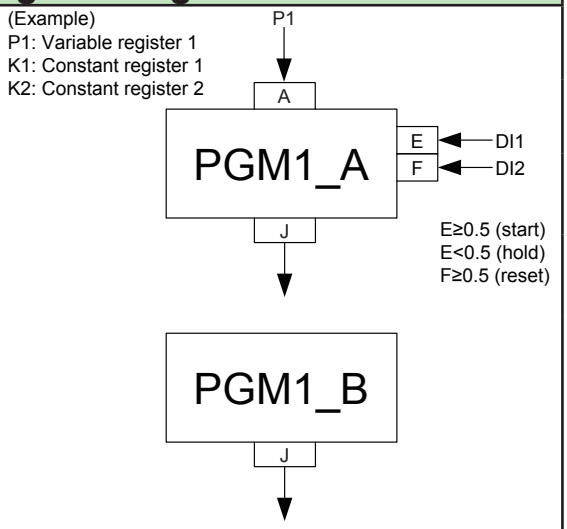
In function block programming, be sure to use PGM1_A and PGM1_B together as a pair.
Also, set PGM1_A followed by PGM1_B in the program execution sequence.
The above explanation of operation applies to both PGM2_A and PGM2_B.

Restriction on Number Used	One each of PGM1_A(B) and PGM2_A(B) to be executed during one control period	Overflow	No

PGMm_A, PGMm_B, PGMm (m=1, 2)

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Reset value
	E	Program E≥0.5 (start), E<0.5 (hold)
	F	Program F≥0.5 (reset)
Output	**J**	PGMm_A: program output value PGMm_B: end flag When time of 10th segment has elapsed, 1 At program reset signal 1 of PGMm_A, 0



Text Programming

(Example)

S Registers before Computation

Register S3: default (reset value)

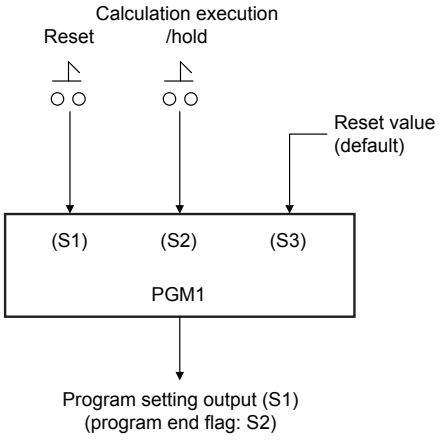
This is the output value (-0.25 to 1.25 (-25.0 to 125.0%) of the starting point in a reset status.

Register S2: calculation execution/hold

When the execution signal that is input to register S2 turns ON (signal state 0.5 or more), program setter operation advances, and when the signal turns OFF (signal state less than 0.5), output is held.

Register S1: reset

When the reset input that is input to register S1 turns ON (signal state 0.5 or more), the program setter returns to the starting point (time 01 start point), the output becomes the value of register S1 after computation of output (value of register S3) at a reset, and end flag output (value of register S2 after computation) is set to "0".



(Note) To start a program setter whose execution has ended, apply a reset (by setting S1=1), cancel the reset, and start the program setting (by setting S1=0 and S2=1).

S Registers after Computation

Register S1: output value

The setting outputs of program elapsed time points are input to register S1. When the time of time 10 is exceeded, the value of output 10 is output.

Register S2: end flag

When the time of time 10 is exceeded, data "1" is stored to register S2, and data "0" is stored by the reset input turning ON. For details, refer to the explanation of operation for the S register change (CHG) instruction.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the default.
LD DI1	DI1	T1	a	b	c	Reads the start instruction.
LD DI2	DI2	DI1	T1	a	b	Reads the reset instruction.
PGM1	Program output value	End flag (0/1)	a	b	b	Program computation When time of time 10 has elapsed, 1 At program reset signal 1 of PGMm_A, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PGMMm_A, PGMMm_B, PGMMm (m=1, 2)

Category	Dynamic operations		
Name of Computation	Program setter (minute)	Computation Instruction Symbol	**PGMMm_A, PGMMm_B, PGMMm (m=1, 2)**

Explanation of Operation

For the explanation of operation, refer to the explanation for the PGMm_A, PGMm_B and PGMm (m=1, 2) instructions.
Read a time unit "second" as "minute."

Note: The PGMMm and PGMm with the same machine number cannot be used simultaneously.
(Example) The PGM1 and PGMM1 cannot be used simultaneously.

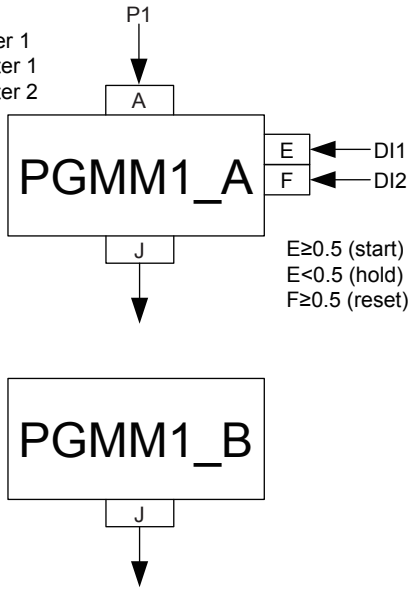
Restriction on Number Used	One each of PGMM1_A(B) to PGMM2_A(B) to be executed during one control period	Overflow	No

PGMMm_A, PGMMm_B, PGMMm (m=1, 2)

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	A	Reset value
	E	Program E≥0.5 (start), E<0.5 (hold)
	F	Program F≥0.5 (reset)
Output	J	PGMMm_A: program output value PGMMm_B: end flag When time of 10th segment has elapsed, 1 At program reset signal 1 of PGMMm_A, 0

(Example)
P1: Variable register 1
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example)
For details on the behavior of S registers before and after computation, see the explanation of operation for the PGMm (m=1, 2) instruction.

Text Program	S1	S2	S3	S4	S5	Explanation
LD T1	T1	a	b	c	d	Reads the default.
LD DI1	DI1	T1	a	b	c	Reads the start instruction.
LD DI2	DI2	DI1	T1	a	b	Reads the reset instruction.
PGMM1	Program output value	End flag (0/1)	a	b	b	Program computation When time of time 10 has elapsed, 1 At program reset signal 1 of PGMMm_A, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

PICm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Pulse input counter	Computation Instruction Symbol	**PICm (m=1 to 8)**

Explanation of Operation

The PICm instruction is a dynamic operation. Only one instruction can be used for each machine number.
This PICm instruction counts the number of times that the input value (A or S2) in a counter started state (E before instruction execution or S1≥0.5) is less than 0.5 at the previous control period and changes state to 0.5 or more at the control period this time. The ON/OFF pulse width of the input value must be at least twice the control period.
Count values 0 to 1 are equivalent to 0 to 1000 pulses, and the pulse can be counted up to 8000 pulses at which the count value is limited.
In a counter reset status, 0 count is output. 0 count is output also when the power is turned ON (i.e. at a cold start), and the count is started from the next control period.
The previous value and count value are backed up at a hot start.

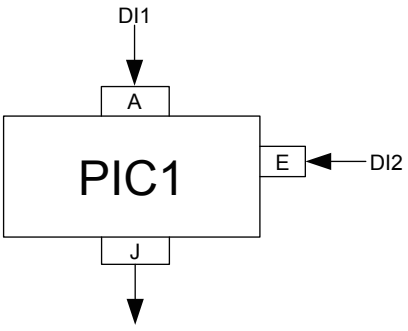
Computing data is a 4-byte floating point number.

Restriction on Number Used	One each of PIC1 to PIC8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	E≥0.5 (start) E<0.5 (reset)
Output	**J**	When E≥0.5, count output When E<0.5, 0.0

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example) The pulse input is counted.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Pulse input
LD DI2	DI2	DI1	a	b	c	Counter start/reset
PIC1	Count value	a	b	c	c	Computation result pulse count value

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

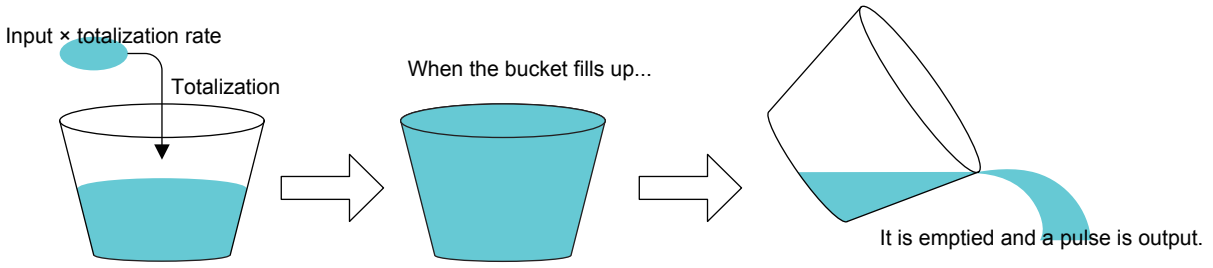
CPO1, CPO2

Category	Dynamic operations		
Name of Computation	Totalizer pulse output	Computation Instruction Symbol	**CPO1, CPO2**

Explanation of Operation

The CPO1 and CPO2 instructions are dynamic operations. Only one instruction can be used for each machine number.
The digital output (DO1, DO2) corresponding to the machine number is used. In other words, this includes processing equivalent to the ST DOm (m=1, 2) instruction.
To use the CPO1 and CPO2 operations, set parameters DIO61 and DIO52 to DO.
Totalization rate (E or S1) 0 to 1 are equivalent to 0 to 1000 pulses/hour (= pph), and can be set within the range 0.1 to 8.0.
The input value (A or S2) is limited at 0 to 4 (400%).
Totalizer pulse output is "totalization rate × input value × 1000 pph". The pulse is a 100 ms wide ON pulse.
The maximum number of output pulses is not dependent on the control period and is 5 pulses/second.
The totalization value is backed up at a hot start.

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

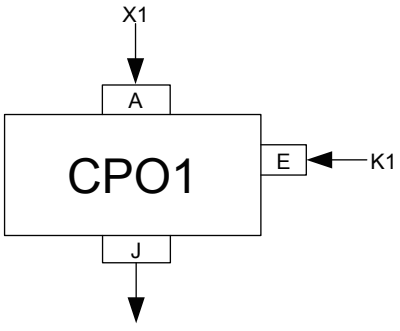


Restriction on Number Used	One each of CPO1 and CPO2 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Totalization rate
Output	**J**	Input value

(Example)
A1: Analog input 1
K1: Constant register 1



Text Programming

(Example) Analog input 1 is totalized.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD K1	K1	X1	a	b	c	Totalization rate
CPO1	Input value	a	b	c	c	Totalizer pulse output

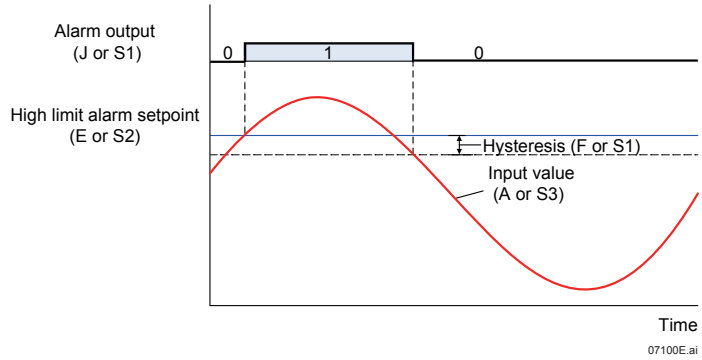
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

HALm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	High limit alarm	Computation Instruction Symbol	**HALm (m=1 to 8)**

Explanation of Operation

The HALm instruction is a dynamic operation. Only one instruction can be used for each machine number. This function is not interlocked with the alarm lamp on the controller's front panel.
The control modules (BSC1, BSC2, CSC, and SSC) are equipped with exclusive alarm functions in addition to this function.
Hysteresis (F or S1) is limited at 0 to 8.0. The high limit alarm setpoint (E or S2) and input value (A or S3) are limited at ±8.0.
If the input value reaches or exceeds the setpoint value, "1" is output (J or S1 after computation). Hysteresis is applied when the previous alarm status is entered.
The previous status for hysteresis is backed up at a hot start.
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

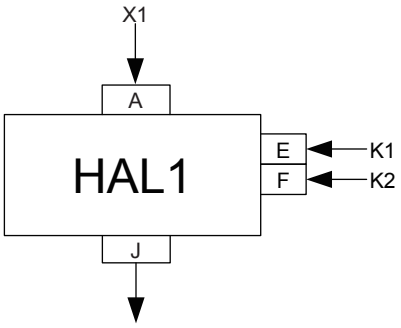


Restriction on Number Used	One each of HAL1 to HAL8 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	High limit alarm setpoint
	F	Hysteresis
Output	**J**	Alarm output

(Example)
A1: Analog input 1
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example) The high limit alarm is set to analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD K1	K1	X1	a	b	c	Reads the high limit alarm setpoint.
LD K2	K2	K1	X1	a	b	Reads the hsyteresis.
HAL1	When X1≥K1 (or K1-K2), 1 When X1<K1 (or K1-K2), 0	X1	a	b	b	Alarm output

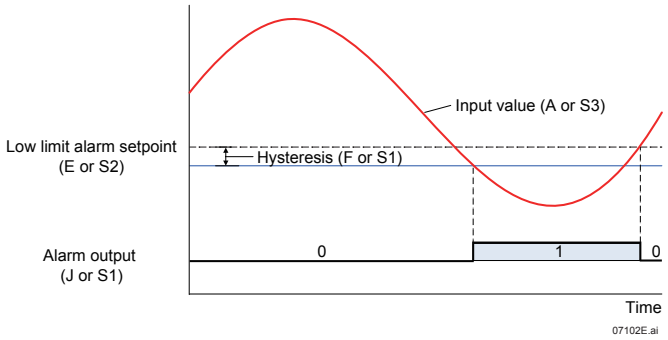
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LALm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Low limit alarm	Computation Instruction Symbol	**LALm (m=1 to 8)**

Explanation of Operation

The LALm instruction is a dynamic operation. Only one instruction can be used for each machine number. This function is not interlocked with the alarm lamp on the controller's front panel.
The control modules (BSC1, BSC2, CSC, and SSC) are equipped with exclusive alarm functions in addition to this function.
Hysteresis (F or S1) is limited at 0 to 8.0. The low limit alarm setpoint (E or S2) and input value (A or S3) are limited at ±8.0.
If the input value reaches or exceeds the setpoint value, "1" is output (J or S1 after computation). Hysteresis is applied when the previous alarm status is entered.
The previous status for hysteresis is backed up at a hot start.
Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

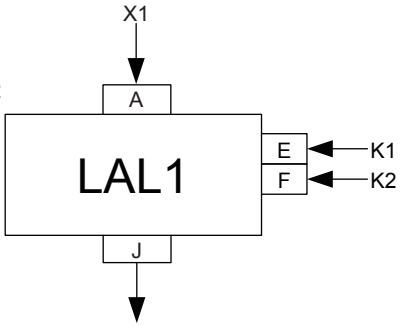


Restriction on Number Used	One each of LAL1 to LAL8 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value
	E	Low limit alarm setpoint
	F	Hysteresis
Output	**J**	Alarm output

(Example)
A1: Analog input 1
K1: Constant register 1
K2: Constant register 2



Text Programming

(Example) The low limit alarm is set to analog input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD K1	K1	X1	a	b	c	Reads the low limit alarm setpoint.
LD K2	K2	K1	X1	a	b	Reads the hsyteresis.
LAL1	When K1 (or K1+K2)≥X1, 1 When K1 (or K1+K2)<X1, 0	a	b	b	b	Alarm output

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

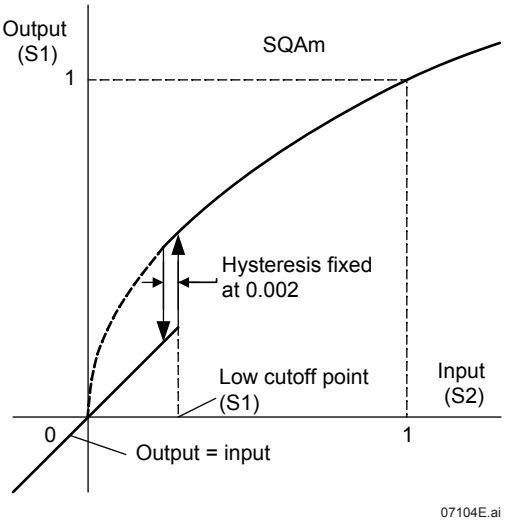
SQAm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Square root extraction (Low cutoff point or less: Linear)	Computation Instruction Symbol	**SQAm (m=1 to 8)**

Explanation of Operation

The SQAm instruction is a dynamic operation.
The low cutoff point has hysteresis of 0.002. When the previous low cutoff status is not entered, the low cutoff point becomes "low cutoff setpoint (E or S1) -0.002."
When the low cutoff point < 0, the output becomes "0."
When the input value (A or S2) ≤ low cutoff point, the input value is output (J or S1 after computation).
When the input value (A or S2) > low cutoff point, √A or √S2 is output (J or S1 after computation).
(S3 to S5 are pushed up.)
The previous low cutoff status is backed up at a hot start.

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

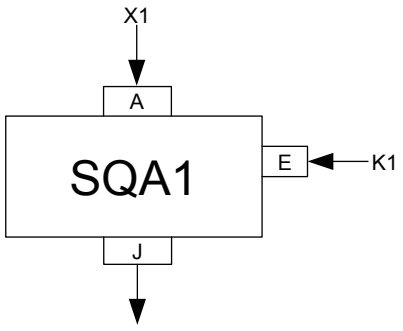


Restriction on Number Used	One each of SQA1 to SQA8 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Value to perform square root on
	E	Low cutoff point (hysteresis fixed at 0.002)
Output	**J**	√A cut at E

(Example)
A1: Analog input 1
K1: Constant register 1



Text Programming

(Example) Analog input 1 is cut at the low cutoff setpoint, and square root computation is performed on the result.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (value to perform square root on).
LD K1	K1	X1	a	b	c	Reads the low cutoff point.
SQA1	When X1>K1, √X1 When X1≤K1, X1	a	b	c	c	Square root extraction (Low cutoff point or less: Linear)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

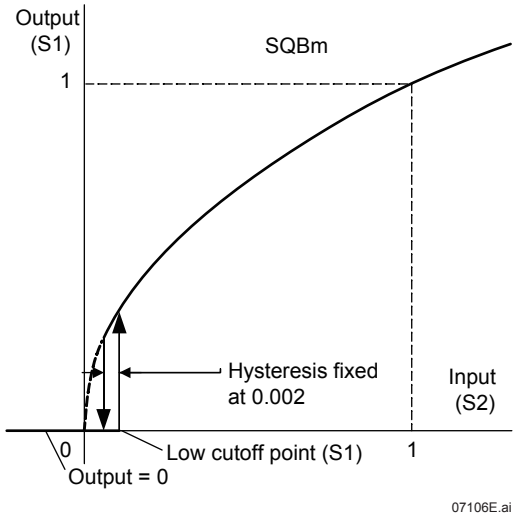
SQBm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Square root extraction (Low cutoff point or less: Zero)	Computation Instruction Symbol	**SQBm (m=1 to 8)**

Explanation of Operation

The SQBm instruction is a dynamic operation.
The low cutoff point has hysteresis of 0.002. When the previous low cutoff status is not entered, the low cutoff point becomes "low cutoff setpoint (E or S1) -0.002."
When the low cutoff point < 0, the output becomes "0."
When the input value (A or S2) ≤ low cutoff point, "0" is output (J or S1 after computation).
When the input value (A or S2) > low cutoff point, √A or √S2 is output (J or S1 after computation).
(S3 to S5 are pushed up.)
The previous low cutoff status is backed up at a hot start.

Computing data is a 4-byte floating point number.
The "computing overflow" error occurs when the computation has exceeded the computation effective range.
The computation is limited to the computation effective range.

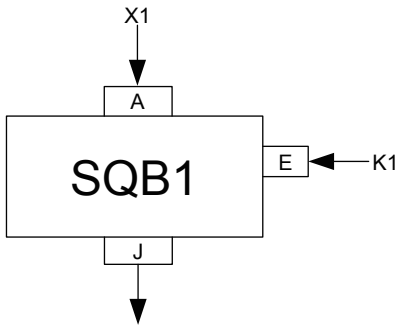


Restriction on Number Used	One each of SQB1 to SQB8 to be executed during one control period	Overflow	Yes

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Value to perform square root on
	E	Low cutoff point (hysteresis fixed at 0.002)
Output	**J**	√A cut at E

(Example)
A1: Analog input 1
K1: Constant register 1



Text Programming

(Example) Analog input 1 is cut at the low cutoff setpoint, and square root computation is performed on the result.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (value to perform square root on).
LD K1	K1	X1	a	b	c	Reads the low cutoff point.
SQB1	When X1>K1, √X1 When X1≤K1, 0	a	b	c	c	Square root extraction (Low cutoff point or less: Zero)

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

RSFFm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	RS flip-flop	Computation Instruction Symbol	**RSFFm (m=1 to 8)**

Explanation of Operation

The RSFFm instruction is a dynamic operation. Only one instruction can be used for each machine number.
The RSFFm instruction performs computation with values less than 0.5 as "0" and values 0.5 or more as "1", and outputs the result (J or S1 after computation) as either "0" or "1."
The previous output value of the RSFFm instruction is backed up at a hot start.
The result computed with the previous output value as "0" is output at a cold start.

Computing data is a 4-byte floating point number.

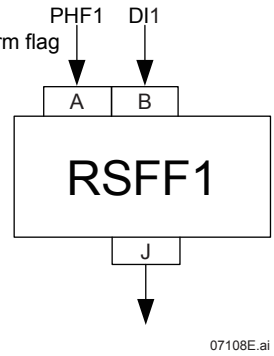
	RSFF							
S1(R) before computation	0	0	0	0	1	1	1	1
S2(S) before computation	0	0	1	1	0	0	1	1
Previous output (O')	0	1	0	1	0	1	0	1
S1(O) after computation	0	1	1	1	0	0	1	1

Restriction on Number Used	One each of RSFF1 to RSFF8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	SET
	B	RESET
Output	**J**	When SET≥0.5, 1 When SET<0.5, and RESET≥0.5, 0 Otherwise, same as previous output

(Example)
PHF1: PV1 high limit alarm flag
DI1: Digital input 1



Text Programming

(Example)

Text Program	S1	S2	S3	S4	S5	Explanation
LD PHF1	PHF1	a	b	c	d	SET
LD DI1	DI1	PHF1	a	b	c	RESET
RSFF1	When PHF1≥0.5, 1 When PHF1<0.5, DI1≥0.5, 0 Otherwise, same as previous output	a	b	c	c	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

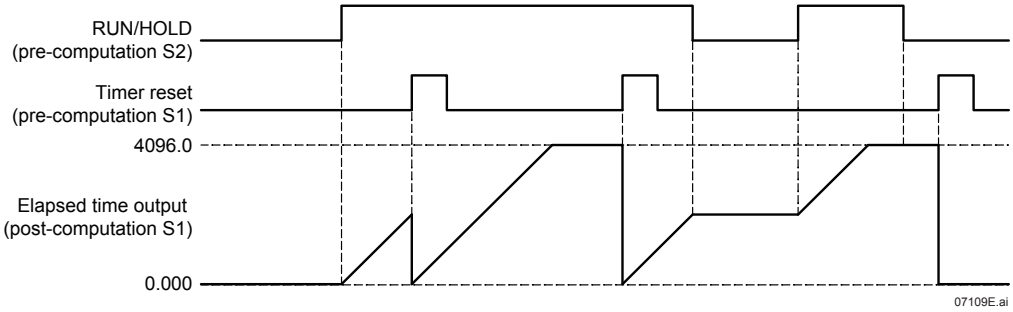
HTIMm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Hold timer (second)	Computation Instruction Symbol	**HTIMm (m=1 to 8)**

Explanation of Operation

<Before computation>
Timer reset (E or S1), timer RUN/HOLD (A or S2)
<After computation>
Time elapsed (1.0 is equivalent to 1000 seconds.) (J or S1)
(S2 to S5 are pushed up.)
The HTIMm instruction is a dynamic operation. Only one instruction can be used for each machine number.
Timer RUN when the timer RUN/HOLD (A or S2) is ≥0.5, and timer HOLD when the timer RUN/HOLD (A or S2) is <0.5.
The timer is reset (time elapsed=0) when the previous timer reset (E or S1) is <0.5 and the timer reset (E or S1) this time is ≥0.5.
The time elapsed (E or S1 after computation) is held at 4096.0 (4096000 seconds).
The previous value and time elapsed are backed up at a hot start.

Computing data is a 4-byte floating point number.



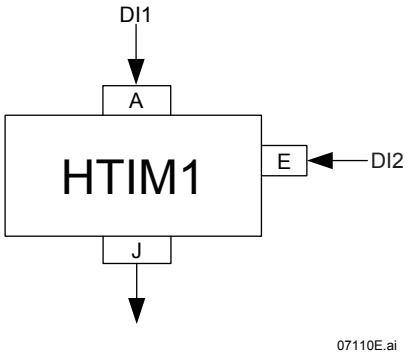
Note: The HTIMm and HTIMMm with the same machine number cannot be used simultaneously.
(Example) The HTIM1 and HTIMM1 cannot be used simultaneously.

Restriction on Number Used	One each of HTIM1 to HTIM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Timer RUN when A≥0.5 Timer HOLD when A<0.5
	E	Timer reset
Output	**J**	Time elapsed (second)

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example)

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Input for RUN/HOLD Timer RUN when DI1≥0.5, and timer HOLD when DI1<0.5
LD DI2	DI2	DI1	a	b	c	Resets the timer.
HTIM1	Time elapsed (second)	a	b	c	c	Holds at 4096.000 (4096000 seconds).

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

HTIMMm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Hold timer (minute)	Computation Instruction Symbol	**HTIMMm (m=1 to 8)**

Explanation of Operation

For the explanation of operation, refer to the explanation for the MTIMm (m=1 to 8) instruction.
However, read a time unit "second" as "minute."

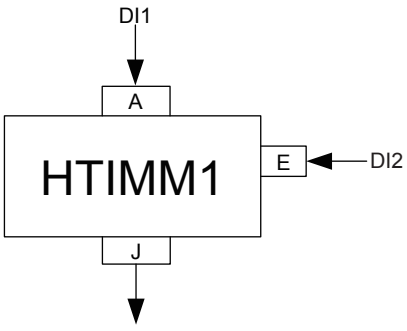
Note: The HTIMMm and HTIMm with the same machine number cannot be used simultaneously.
(Example) The HTIM1 and HTIMM1 cannot be used simultaneously.

Restriction on Number Used	One each of HTIMM1 to HTIMM8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Timer RUN when A≥0.5, Timer HOLD when A<0.5
	E	Timer reset
Output	**J**	Time elapsed (minute)

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example)

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Input for RUN/HOLD Timer RUN when DI1≥0.5, and timer HOLD when DI1<0.5
LD DI2	DI2	DI1	a	b	c	Resets the timer.
HTIMM1	Time elapsed (minute)	a	b	c	c	Holds at 4096.000 (4096000 minutes).

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

DELAYm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Previous input variable	Computation Instruction Symbol	**DELAYm (m=1 to 8)**

Explanation of Operation

The DELAY instruction outputs (J or S1 after computation) the previous input value (A or S1).
The input value (A or S1) this time is output only at the beginning of a cold start.
The previous value is backed up at a hot start.

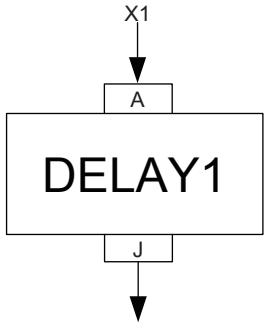
Computing data is a 4-byte floating point number.

Restriction on Number Used	One each of DELAY1 to DELAY8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
Output	**J**	Previous value of A

(Example)
X1: Analog input 1



Text Programming

(Example) Analog input 1 is delayed by one control period.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
DELAY1	Previous X1	a	b	c	d	Previous input variable

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

HOLDm (m=1 to 8)

Category	Dynamic operations		
Name of Computation	Hold	Computation Instruction Symbol	**HOLDm (m=1 to 8)**

Explanation of Operation

When hold/through (E or S1) is ≥0.5, the previous output is output (J or S1 after computation) again.
When hold/through (E or S1) is <0.5, the input (A or S2) this time is output (J or S1 after computation).
The input this time is output independently of the hold/through (E or S1) setting only at the beginning of a cold start. (S3 to S5 are pushed up.)
The previous value is backed up at a hot start.

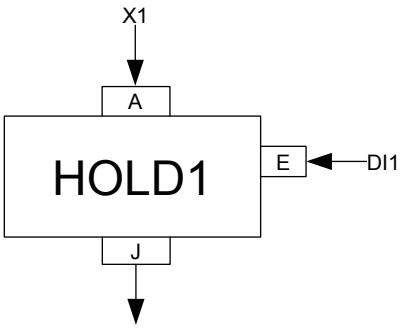
Computing data is a 4-byte floating point number.

Restriction on Number Used	One each of HOLD1 to HOLD8 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value
	E	Hold/through
Output	**J**	Previous output when E≥0.5 When E<0.5, input (A) this time is output

(Example)
X1: Analog input 1
DI1: Digital input 1



Text Programming

(Example) Analog input 1 is held or allowed through by the status of digital input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD DI1	DI1	X1	a	b	c	Hold/through
HOLD1	When DI1≥0.5, previous output When DI1<0.5, input this time (X1)	a	b	c	c	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

AND

Category	Logic computations		
Name of Computation	AND	Computation Instruction Symbol	**AND**

Explanation of Operation

The AND instruction performs computation with values less than 0.5 as "0" and values 0.5 or more as "1," and outputs the result (J or S1 after computation) as either "0" or "1."

	AND			
A (S1) before computation	0	0	1	1
B (S2) before computation	0	1	0	1
J (S1) after computation	0	0	0	1

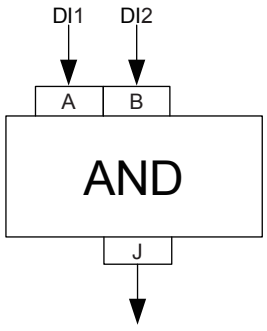
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	0 or 1
	B	0 or 1
Output	**J**	When A ≥ 0.5 and B ≥ 0.5, 1 Otherwise, 0

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example) Digital inputs 1 and 2 are ANDed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	When DI1=1
LD DI2	DI2	DI1	a	b	c	When DI2=0
AND	0	a	b	c	c	AND S1=0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

OR

Category	Logic computations		
Name of Computation	OR	Computation Instruction Symbol	**OR**

Explanation of Operation

The OR instruction performs computation with values less than 0.5 as "0" and values 0.5 or more as "1," and outputs the result (J or S1 after computation) as either "0" or "1."

	OR			
A (S1) before computation	0	0	1	1
B (S2) before computation	0	1	0	1
J (S1) after computation	0	1	1	1

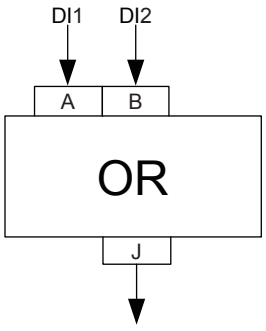
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	0 or 1
	B	0 or 1
Output	**J**	When A ≥ 0.5 or B ≥ 0.5, 1 Otherwise, 0

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example) Digital inputs 1 and 2 are ORed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	When DI1=1
LD DI2	DI2	DI1	a	b	c	When DI2=0
OR	1	a	b	c	c	OR S1=1

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

NOT

Category	Logic computations		
Name of Computation	NOT	Computation Instruction Symbol	**NOT**

Explanation of Operation

The NOT instruction performs computation with values less than 0.5 as "0" and values 0.5 or more as "1," and outputs the result (J or S1 after computation) as either "0" or "1."

	NOT	
A (S1) before computation	0	1
J (S1) after computation	1	0

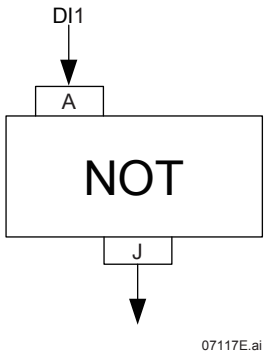
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	0 or 1
Output	**J**	When A ≥ 0.5, 0 When A < 0.5, 1

(Example)
DI1: Digital input 1



Text Programming

(Example) The status of digital input 1 is inverted.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	When DI1=1
NOT	0	a	b	c	d	NOT S1=0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

EOR

Category	Logic computations		
Name of Computation	Exclusive OR	Computation Instruction Symbol	**EOR**

Explanation of Operation

The EOR instruction performs computation with values less than 0.5 as "0" and values 0.5 or more as "1," and outputs the result (J or S1 after computation) as either "0" or "1."

	EOR			
A (S1) before computation	0	0	1	1
B (S2) before computation	0	1	0	1
J (S1) after computation	0	1	1	0

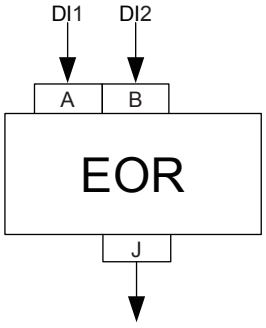
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	0 or 1
	B	0 or 1
Output	**J**	Exclusive OR of A and B (See "Explanation of Operation.")

(Example)
DI1: Digital input 1
DI2: Digital input 2



Text Programming

(Example) Digital input 1 and digital input 2 are exclusively ORed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	When DI1=1
LD DI2	DI2	DI1	a	b	c	When DI2=1
EOR	0	a	b	c	c	EOR S1=0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MAND

Category	Logic computations		
Name of Computation	Multi-input AND	Computation Instruction Symbol	**MAND**

Explanation of Operation

The MAND instruction outputs the following (J or S1 after computation):

When input value 1 (A or S4) to input value 4 (D or S1) before computation all are 0.5 or more, "1" is output.
When even one input value is less than 0.5, "0" is output.

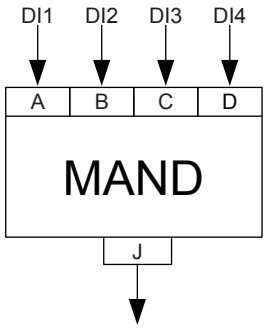
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	When A to D all are ≥ 0.5, 1 When even one of A to D is < 0.5, 0

(Example)
DI1: Digital input 1
DI2: Digital input 2
DI3: Digital input 3
DI4: Digital input 4



Text Programming

(Example) Digital inputs 1 to 4 are ANDed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads digital input 1.
LD DI2	DI2	DI1	a	b	c	Reads digital input 2.
LD DI3	DI3	DI2	DI1	a	b	Reads digital input 3.
LD DI4	DI4	DI3	DI2	DI1	a	Reads digital input 4.
MAND	0/1	a	a	a	a	When DI1 to DI4 all are ≥ 0.5, 1 When even one of DI1 to DI4 is < 0.5, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MOR

Category	Logic computations		
Name of Computation	Multi-input OR	Computation Instruction Symbol	**MOR**

Explanation of Operation

The MOR instruction outputs the following (J or S1 after computation):

When even one of input value 1 (A or S4) to input value 4 (D or S1) before computation is 0.5 or more, "1" is output.
When all input values are less than 0.5, "0" is output.

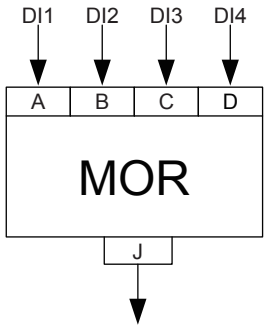
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	When even one of A to D is ≥ 0.5, 1 When A to D all are < 0.5, 0

(Example)
DI1: Digital input 1
DI2: Digital input 2
DI3: Digital input 3
DI4: Digital input 4



Text Programming

(Example) Digital inputs 1 to 4 are ORed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads digital input 1.
LD DI2	DI2	DI1	a	b	c	Reads digital input 2.
LD DI3	DI3	DI2	DI1	a	b	Reads digital input 3
LD DI4	DI4	DI3	DI2	DI1	a	Reads digital input 4.
MOR	0/1	a	a	a	a	When even one of DI1 to DI4 is ≥ 0.5, 1 When DI1 to DI4 all are < 0.5, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MEOR

Category	Logic computations		
Name of Computation	Multi-input exclusive OR	Computation Instruction Symbol	**MEOR**

Explanation of Operation

The MEOR instruction outputs the following (J or S1 after computation):

When input value 1 (A or S4) to input value 4 (D or S4) before computation all are 0.5 or more or less than 0.5, "0" is output.
Otherwise, "1" is output.

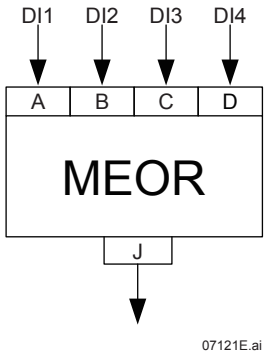
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	When A to D all are ≥ 0.5, 0 When A to D all are < 0.5, 0 Otherwise, 1

(Example)
DI1: Digital input 1
DI2: Digital input 2
DI3: Digital input 3
DI4: Digital input 4



Text Programming

(Example) Digital inputs 1 to 4 are exclusively ORed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads digital input 1.
LD DI2	DI2	DI1	a	b	c	Reads digital input 2.
LD DI3	DI3	DI2	DI1	a	b	Reads digital input 3.
LD DI4	DI4	DI3	DI2	DI1	a	Reads digital input 4.
MEOR	0/1	a	a	a	a	When DI1 to DI4 all are ≥ 0.5, 0 When DI1 to DI4 all are < 0.5, 0 Otherwise, 1

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

CMP

Category	Condition judgments		
Name of Computation	Comparison	Computation Instruction Symbol	**CMP**

Explanation of Operation

The CMP instruction is the same as the greater or equal (GE) instruction.
The CMP instruction compares the content of input value 1 (A or S2) with input value 2 (B or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When input value 1 (A or S2) ≥ input value 2 (B or S1), "1" is output.
When input value 1 (A or S2) < input value 2 (B or S1), "0" is output.

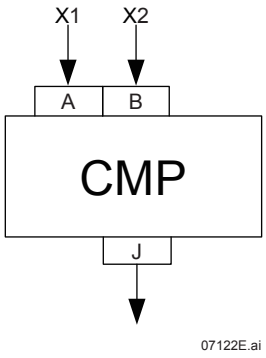
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
Output	**J**	When A < B, 0 When A ≥ B, 1

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 is compared with analog input 2, and "1" is selected if analog input 1 is larger or "0" is selected if analog input 1 is smaller.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1					Reads analog input 1.
LD X2	X2	X1				Reads analog input 2.
CMP	0/1					X2 ≤ X1, 1 X2 > X1, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

SW

Category	Condition judgments		
Name of Computation	Signal switching	Computation Instruction Symbol	**SW**

Explanation of Operation

The SW instruction outputs input value 1 (A or S3) or input value 2 (B or S2) according to the value of the switching signal (E or S1), and outputs the following (J or S1 after computation) as a result of the comparison.

When the switching signal (E or S1) < 0.5, input value 1 (A or S3) is output.
When the switching signal (E or S1) ≥ 0.5, input value 2 (B or S2) is output.

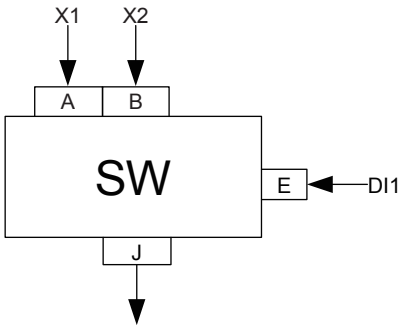
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Input value 1
	B	Input value 2
	E	Switching signal (0/1)
Output	**J**	When E ≥ 0.5, B When E < 0.5, A

(Example)
X1: Analog input 1
X2: Analog input 2
DI1: Digital input 1



Text Programming

(Example) Analog input 1 and analog input 2 are switched by digital input 1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD X2	X2	X1	a	b	c	Reads analog input 2.
LD DI1	0/1	X2	X1	a	b	Reads the switching signal.
SW	X1 or X2	a	b	b	b	When DI1 ≥ 0.5, X2 When DI1 < 0.5, X1

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

GE

Category	Condition judgments		
Name of Computation	≥, Greater or equal	Computation Instruction Symbol	**GE**

Explanation of Operation

The GE instruction is the same as the comparison (CMP) instruction.
The GE instruction compares the content of the compared value (A or S2) with the comparison value (B or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When compared value (A or S2) ≥ comparison value (B or S1), "1" is output.
When compared value (A or S2) < comparison value (B or S1), "0" is output.

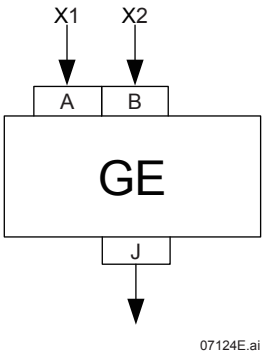
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Compared value
	B	Comparison value
Output	**J**	When A ≥ B, 1 When A < B, 0

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 is compared with analog input 2, and "1" is selected if analog input 1 is larger or "0" is selected if analog input 1 is smaller.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD X2	X2	X1	a	b	c	Reads analog input 2 (comparison value).
GE	0/1	a	b	c	c	X1 ≥ X2, 1 X1 < X2, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

GT

Category	Condition judgments		
Name of Computation	>, Greater than	Computation Instruction Symbol	GT

Explanation of Operation

The GT instruction compares the content of the compared value (A or S2) with the comparison value (B or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When compared value (A or S2) > comparison value (B or S1), "1" is output.
When compared value (A or S2) ≤ comparison value (B or S1), "0" is output.

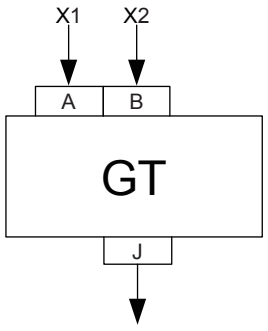
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Compared value
	B	Comparison value
Output	J	When A>B, 1 When A ≤ B, 0

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 is compared with analog input 2, and "1" is selected if analog input 1 is larger or "0" is selected if analog input 1 is smaller.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD X2	X2	X1	a	b	c	Reads analog input 2 (comparison value).
GT	0/1	a	b	c	c	X1 > X2, 1 X1 ≤ X2, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LE

Category	Condition judgments		
Name of Computation	≤, Less or equal	Computation Instruction Symbol	**LE**

Explanation of Operation

The LE instruction compares the content of the compared value (A or S2) with the comparison value (B or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When compared value (A or S2) ≤ comparison value (B or S1), "1" is output.
When compared value (A or S2) > comparison value (B or S1), "0" is output.

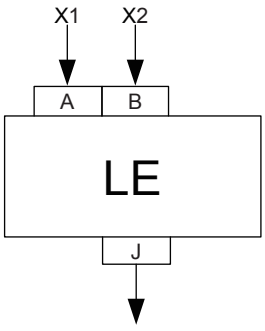
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Compared value
	B	Comparison value
Output	**J**	When A ≤ B, 1 When A > B, 0

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 is compared with analog input 2, and "1" is selected if analog input 2 is larger or "0" is selected if analog input 2 is smaller.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD X2	X2	X1	a	b	c	Reads analog input 2 (comparison value).
LE	0/1	a	b	c	c	X1 ≤ X2, 1 X1 > X2, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

LT

Category	Condition judgments		
Name of Computation	<, Less than	Computation Instruction Symbol	LT

Explanation of Operation

The LT instruction compares the content of the compared value (A or S2) with the comparison value (B or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When compared value (A or S2) < comparison value (B or S1), "1" is output.
When compared value (A or S2) ≥ comparison value (B or S1), "0" is output.

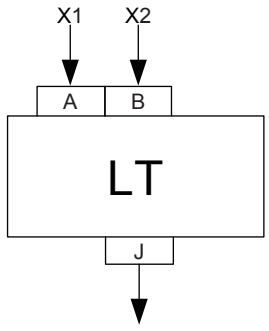
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Compared value
	B	Comparison value
Output	J	When A < B, 1 When A ≥ B, 0

(Example)
X1: Analog input 1
X2: Analog input 2



Text Programming

(Example) Analog input 1 is compared with analog input 2, and "1" is selected if analog input 2 is larger or "0" is selected if analog input 2 is smaller.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD X2	X2	X1	a	b	c	Reads analog input 2 (comparison value).
LT	0/1	a	b	c	c	X1 < X2, 1 X1 ≥ X2, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

INRNG

Category	Condition judgments		
Name of Computation	In range	Computation Instruction Symbol	**INRNG**

Explanation of Operation

The INRNG instruction compares the content of the compared value (A or S3), comparison value (small) (B or S2) and comparison value (large) (C or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When comparison value (small) (B or S2) ≤ compared value (A or S3) ≤ comparison value (large) (C or S1), "1" is output.
When compared value (A or S3) < comparison value (small) (B or S2), and comparison value (large) (C or S1) < compared value (A or S3), "0" is output.

(S3 to S5 are pushed up.)

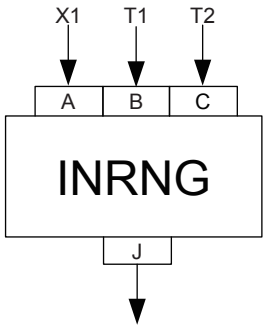
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Compared value
	B	Comparison value, small
	C	Comparison value, large
Output	**J**	When B ≤ A ≤ C, 1 When A < B and C < A, 0

(Example)
X1: Analog input 1
T1: Temporary memory register 1
T2: Temporary memory register 1



Text Programming

(Example) Analog input 1 is compared with temporary memory register 1 and temporary memory register 2 to see if it is in range.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD T1	T1	X1	a	b	c	Reads temporary memory register 1 (comparison value, small).
LD T2	T2	T1	X1	a	b	Reads temporary memory register 2 (comparison value, large).
INRNG	0/1	a	b	b	b	T1 ≤ X1 ≤ T2, 1 X1 < T1, T2 < X1, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

OUTRNG

Category	Condition judgments		
Name of Computation	Out of range	Computation Instruction Symbol	**OUTRNG**

Explanation of Operation

The OUTRNG instruction compares the content of the compared value (A or S3), comparison value (small) (B or S2) and comparison value (large) (C or S1) before execution, and outputs the following (J or S1 after computation) as a result of the comparison:

When compared value (A or S3) ≤ comparison value (small) (B or S2) or comparison value (large) (C or S1) ≤ compared value (A or S3), "1" is output.
When comparison value (small) (B or S2) < compared value (A or S3) < comparison value (large) (C or S1), "0" is output.

(S3 to S5 are pushed up.)

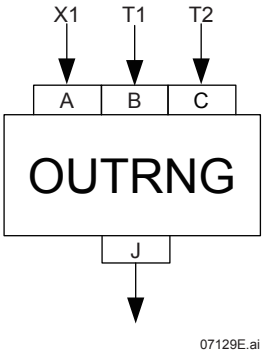
Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Compared value
	B	Comparison value, small
	C	Comparison value, large
Output	**J**	When A ≤ B and C ≤ A, 1 When B < A < C, 0

(Example)
X1: Analog input 1
T1: Temporary memory register 1
T2: Temporary memory register 1



Text Programming

(Example) Analog input 1 is compared with temporary memory register 1 and temporary memory register 2 to see if it is out of range.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1 (compared value).
LD T1	T1	X1	a	b	c	Reads temporary memory register 1 (comparison value, small).
LD T2	T2	T1	X1	a	b	Reads temporary memory register 2 (comparison value, large).
OUTRNG	0/1	a	b	b	b	X1 ≤ T1, T2 ≤ X1, 1 T1 < X1 < T2, 0

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

MSWA

Category	Condition judgments		
Name of Computation	Multi input signals switching A	Computation Instruction Symbol	**MSWA**

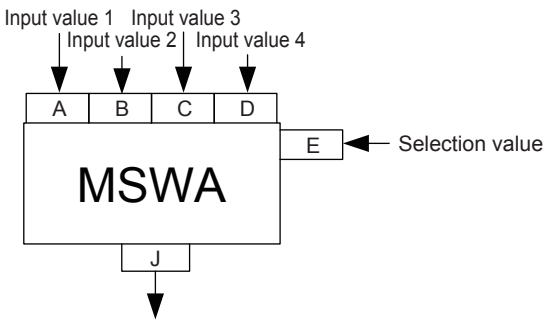
Explanation of Operation

When 1.5 ≤ Selection value E < 2.5, Input value B is output.
When 2.5 ≤ Selection value E < 3.5, Input value C is output.
When 3.5 ≤ Selection input E, Input value D is output.
Otherwise, A is output.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
	E	Selection value
Output	**J**	Selected input value



Text Programming

Used only in function block programming

MSWB

Category	Condition judgments		
Name of Computation	Multi input signals switching B	Computation Instruction Symbol	**MSWB**

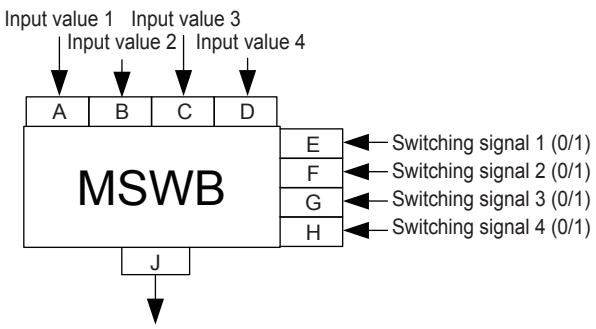
Explanation of Operation

When Input value F ≥ 0.5, Input value B is output.
When Input value G ≥ 0.5, Input value C is output.
When Input value H ≥ 0.5, Input value D is output.
Otherwise, A is output.
If multiple conditions occur simultaneously, the order of priority is A > B > C > D.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
	E	Switching signal 1 (0/1)
	F	Switching signal 2 (0/1)
	G	Switching signal 3 (0/1)
	H	Switching signal 4 (0/1)
Output	**J**	Selected input value



Text Programming

Used only in function block programming

BITSEL

Category	Condition judgments		
Name of Computation	Selection from four inputs and conversion to a numerical value	Computation Instruction Symbol	**BITSEL**

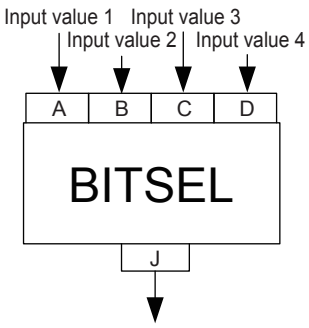
Explanation of Operation

When Input value B ≥ 0.5, 2.0 is output.
When Input value C ≥ 0.5, 3.0 is output.
When Input value D ≥ 0.5, 4.0 is output.
Otherwise, 1.0 is output.
If multiple conditions occur simultaneously, the order of priority is A > B > C > D.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Input value 1
	B	Input value 2
	C	Input value 3
	D	Input value 4
Output	**J**	Selected value



Text Programming

Used only in function block programming

GO @<label name>

Category	Condition judgments		
Name of Computation	Jump	Computation Instruction Symbol	**GO@<label name>**

Explanation of Operation

The GO instruction is used as an instruction signal, for example, to change the flow of the program.
A "label" is a text string of up to ten characters following "@" (not including "@").
Available characters are alphanumerics and "_" (underbar).

Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) A jump is performed to label name XYZ.

Text Program	S1	S2	S3	S4	S5	Explanation
GO @XYZ : : @XYZ Next computation	a a	b b	c c	d d	e e	Jumps to the program line containing the label name XYZ.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

GIF @<label name>

Category	Condition judgments		
Name of Computation	Conditional jump	Computation Instruction Symbol	**GIF @<label name>**

Explanation of Operation

The GIF instruction is used as an instruction signal, for example, to change the flow of the program.
A "label" is a text string of up to ten characters following "@" (not including "@"). Available characters are alphanumerics and "_" (underbar).
By a conditional jump instruction, a jump is performed when the content of register S1 is 0.5 or more.

Computing data is a 4-byte floating point number.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) When digital input 1 is "1," a jump is performed to label name XYZ. When digital input 1 is "0," the computation programmed to the next program line is executed.

Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	0/1	a	b	c	d	Reads digital input 1.
GIF @XYZ	a	b	c	d	d	When DI1 ≥ 0.5, jumps to the program line containing the label name XYZ. When DI1 < 0.5, executes the computation programmed to the next program line.
Next computation						
:						
@XYZ Next computation						

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

SUB@n (n=1 to 200),GOSUB @<sub-program name>

Category	Condition judgments		
Name of Computation	Jump to the sub-program	Computation Instruction Symbol	**SUB@n (n=1 to 200)** **GOSUB@<sub-program name>**

Explanation of Operation

The SUB instruction performs a jump to the subprogram from the main program.
In text programming, the line indicated by GOSUB @<sub-program name> is jumped to unconditionally.
The computation following the jump instruction is executed by the subprogram termination instruction.

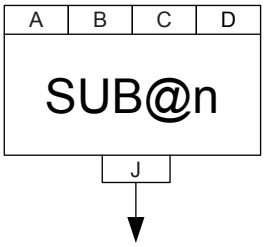
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SUB@n, IFSUB@n and GTSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 200 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
Output	**J**	Subprogram output value that is passed from the sub-program

(Example)



*A to D can be used even if registers are not registered.

07130E.ai

Text Programming

(Example) A jump is made to subprogram name XYZ after analog input 1 is read.

Main Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
GOSUB @XYZ	X1	a	b	c	d	Jumps to subprogram name XYZ.
Next computation	a	b	c	d	e	Computation to be executed after the subprogram termination instruction.

Sub Text Program	S1	S2	S3	S4	S5	Explanation
SUB @XYZ	X1	a	b	c	d	
Next computation						
RTN	a	b	c	d	e	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

IFSUB@n (n=1 to 200),GIFSUB @<sub-program name>

Category	Condition judgments		
Name of Computation	Conditional jump to the sub-program	Computation Instruction Symbol	**IFSUB@n (n=1 to 200)** **GIFSUB@<sub-program name>**

Explanation of Operation

The IFSUB instruction performs a conditional jump to the subprogram from the main program.
In text programming, the program line indicated by SUB @<sub-program name> is jumped to when there is a change in status (when S1 ≥ 0.5) by GIFSUB @<sub-program name>. (S2 to S5 are pushed up.)
The computation following the jump instruction is executed by the subprogram termination instruction.

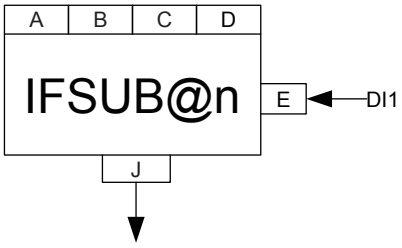
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SUB@n, IFSUB@n and GTSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 200 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
	E	0/1
Output	**J**	Subprogram output value that is passed from the subprogram when E ≥ 0.5 When E<0.5, value of A

(Example)



*A to D can be used even if registers are not registered.

07131E.ai

Text Programming

(Example) When digital input 1 is "1," a jump is performed to subprogram name XYZ. When digital input 1 is "0," the next computation is executed.

Main Text Program	S1	S2	S3	S4	S5	Explanation
LD DI1	DI1	a	b	c	d	Reads digital input 1.
GIFSUB @XYZ	DI1	b	c	d	d	When DI1 ≥ 0.5, same as GOSUB When DI1 < 0.5, the subprogram termination instruction is just executed.
Next computation	a	b	c	d	e	Computation to be executed after the subprogram termination instruction.

Sub Text Program	S1	S2	S3	S4	S5	Explanation
SUB @XYZ						
Next computation						
RTN	a	b	c	d	e	

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

GTSUB@n (n=1 to 200)

Category	Condition judgments		
Name of Computation	Condition comparison jump to the sub-program	Computation Instruction Symbol	**GTSUB@n (n=1 to 200)**

Explanation of Operation

The GTSUB instruction performs a condition comparison jump to the subprogram from the main program.

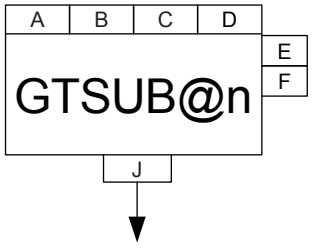
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SUB@n, IFSUB@n and GTSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 200 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
	E	Compared value
	F	Comparison value
Output	**J**	Subprogram output value that is passed from the subprogram when E ≥ F When E < F, value of A

(Example)



*A to D can be used even if registers are not registered.

07132E.ai

Text Programming

Used only in function block programming

SSUB@n (n=201 to 256)

Category	Condition judgments		
Name of Computation	Jump to the sub-program (for nesting)	Computation Instruction Symbol	**SSUB@n (n=201 to 256)**

Explanation of Operation

The SSUB instruction performs a jump to the subprogram from the main program or subprogram.

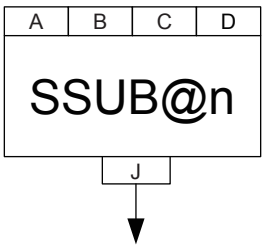
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SSUB@n, IFSSUB@n and GTSSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 56 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
Output	**J**	Subprogram output value that is passed from the subprogram

(Example)



*A to D can be used even if registers are not registered.

07133E.ai

Text Programming

Used only in function block programming

IFSSUB@n (n=201 to 256)

Category	Condition judgments		
Name of Computation	Conditional jump to the sub-program (for nesting)	Computation Instruction Symbol	**IFSSUB@n (n=201 to 256)**

Explanation of Operation

The IFSSUB instruction performs a conditional jump to the subprogram from the main program or subprogram.

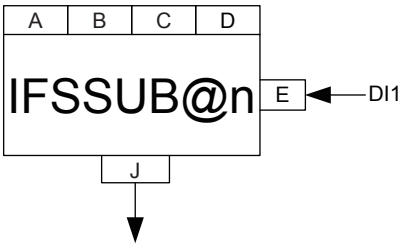
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SSUB@n, IFSSUB@n and GTSSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 56 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
	E	0/1
Output	**J**	Subprogram output value that is passed from the subprogram when E ≥ 0.5 When E < 0.5, value A

(Example)



*A to D can be used even if registers are not registered.

07134E.ai

Text Programming

Used only in function block programming

GTSSUB@n (n=201 to 256)

Category	Condition judgments		
Name of Computation	Condition comparison jump to the sub-program (for nesting)	Computation Instruction Symbol	**GTSSUB@n (n=201 to 256)**

Explanation of Operation

The GTSSUB instruction performs a condition comparison jump to the subprogram from the main program or subprogram.

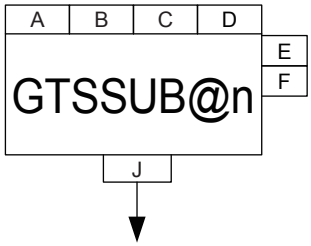
When arguments 1(A) to 4(D) are not used, "0" is output (J) in the subprogram.
The "n" of the SSUB@n, IFSSUB@n and GTSSUB@n instructions is a common number, and the same number cannot be used.

Restriction on Number Used	Total of 56 SUB, IFSUB and GTSUB instructions	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input and Parameters	**A**	Argument 1
	B	Argument 2
	C	Argument 3
	D	Argument 4
	E	Compared value
	F	Comparison value
Output	**J**	Subprogram output value that is passed from the subprogram when E ≥ F When E < F, value of A

(Example)



*A to D can be used even if registers are not registered.

07135E.ai

Text Programming

Used only in function block programming

SUB @<sub-program name>

Category	Condition judgments		
Name of Computation	Sub-program startup	Computation Instruction Symbol	**SUB @<sub-program name>**

Explanation of Operation

The SUB instruction indicates the start of the subprogram.
The subprogram begins at "SUB @<sub-program name>" and is terminated by the subprogram termination (RTN) instruction.

As the inter-loop operations of the cascade control module CSC, @CSCPR1 and @CSCPR2, are started up from CSC, the GOSUB and GIFSUB instructions are not required. Also, the GOSUB and GIFSUB instructions need not be used if "SUB @CSCPR1" and "SUB @CSCPR2" are not written in the program.
For details on the application and role of @CSCPRn, see "5.3 How to Use the Cascade (CSC) Control Module."

Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) Subprogram ABC is started.

Text Program	S1	S2	S3	S4	S5	Explanation
SUB @ABC	a	b	c	d	e	Starts subprogram ABC.
	a	b	c	d	e	Processing

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

RTN

Category	Condition judgments		
Name of Computation	Sub-program termination	Computation Instruction Symbol	**RTN**

Explanation of Operation

The RTN instruction indicates the end of the subprogram, and executes a return to the main program.

Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) Computation processing of subprogram name XYZ is terminated, and program execution returns to the main program.

Sub Text Program	S1	S2	S3	S4	S5	Explanation
SUB @XYZ	a	b	c	d	e	Starts subprogram XYZ.
:	a	b	c	d	e	
:						
RTN						Terminates the subprogram and returns to the main program.

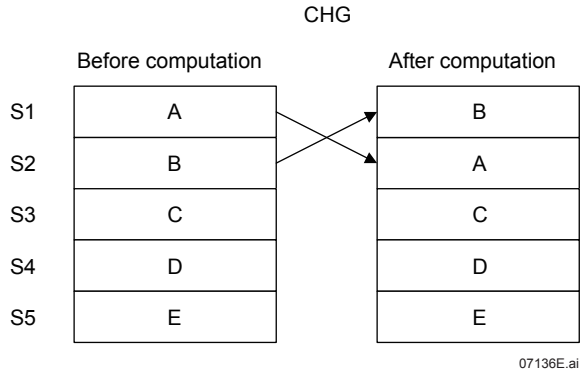
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

CHG

Category	Register moves		
Name of Computation	S register change	Computation Instruction Symbol	**CHG**

Explanation of Operation

The CHG registers changes the values of register S1 and register S2.



Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) Register S2 is changed with register S1 after register S1 is read.

Text Program	S1	S2	S3	S4	S5	Explanation
LD P1	P1	a	b	c	d	Reads the default.
LD DI1	0/1	P1	a	b	c	Sets the start status.
LD T1	0/1	0/1	P1	a	b	Resets the program if the program is ended.
PGM1	Linearizer output	0/1	a	b	b	Computes the program.
ST CSV1	Linearizer output	0/1	a	b	b	Stores the result to register CSV1.
CHG	0/1	Linearizer output	a	b	b	Changes the S register.
ST T1	0/1	Linearizer output	a	b	b	Stores the PGM end flag to T1.
Next computation						

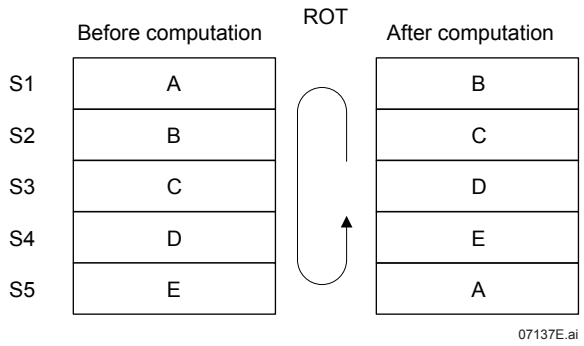
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

ROT

Category	Register moves		
Name of Computation	S register rotation	Computation Instruction Symbol	**ROT**

Explanation of Operation

The ROT instruction rotates registers S2 to S5 successively upwards, and moves S1 to S5.



Restriction on Number Used	None	Overflow	No

Function Block Programming

Used only in text programming

Text Programming

(Example) Register T containing in S1 to S5 is rotated.

Text Program	S1	S2	S3	S4	S5	Explanation
:	T1	T2	T3	T4	T5	
ROT	T2	T3	T4	T5	T1	Rotates the S register.
Next computation						

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

BSC1, BSC2

Category	Control functions		
Name of Computation	Basic control	Computation Instruction Symbol	**BSC1, BSC2**

Explanation of Operation

Function block programming : Process variable 1 (A) is input to perform the control operations of the single loop. The computation result is output to J.

Text programming : Process variable 1 (A) is input to perform the control operations of the single loop.The computation result is output to S1.

- For both inputs and outputs, 0.0 to 1.0 is the equivalent of 0 to 100%. When PV has been computed by actual quantities before this instruction, use the normalization (NORM) instruction to normalize to 0.0 to 1.0.
- Flags, such as the operation mode and parameters other than PV, are read and written when a control instruction is executed.
  These parameters and flags are handled by front panel key operations, communication, and the read and store (LD reg, ST reg) instructions.
- The BSC1 and BSC2 instructions can be used within the same program. (Example: control operation in dual-loop control)
  Other combinations are not possible.

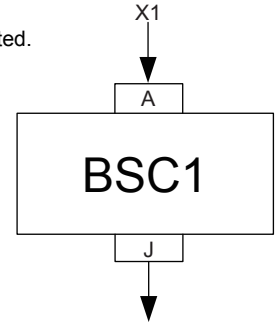
For details, see "5.1 How to Use the Basic (BSC) Control Module."

Restriction on Number Used	One each of BSC1 and BSC2 to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	**A**	Process variable 1 (PV1)
Output	**J**	Control operation result

(Example)
X1 is input, and BSC is executed.



Text Programming

(Example) Analog input 1 is input, and the computation result is output to Y1 after execution of BSC1.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
BSC1	MV1	a	b	c	d	Executes basic control. (The control operation result is stored to S1.)
ST Y1	MV1	a	b	c	d	Outputs to Y1.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

CSC

Category	Control functions		
Name of Computation	Cascade control	Computation Instruction Symbol	CSC

Explanation of Operation

Function block programming : Process variable 1 (A) and process variable 2 (B) are input to perform cascade control operation.
The computation result is output to J.

Text programming : Process variable 1 (S2) and process variable 2 (S1) are input to perform cascade ontrol operation.
The computation result is output to S1. (S3 to S5 are pushed up.)

- For both inputs and outputs, 0.0 to 1.0 is the equivalent of 0 to 100%. When PV has been computed by actual quantities before this instruction, use the normalization (NORM) instruction to normalize to 0.0 to 1.0.
- Flags, such as the operation mode and parameters other than PV, are read and written when a control instruction is executed.
  These parameters and flags are handled by front panel key operations, communication, and the read and store (LD reg, ST reg) instructions.

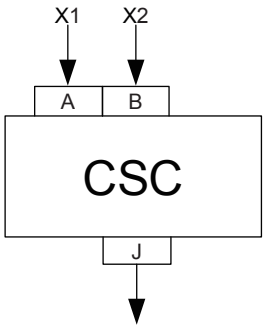
For details, see "5.3 How to Use the Cascade (CSC) Control Module."

Restriction on Number Used	One CSC to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Process variable 1 (PV1)
	B	Process variable 2 (PV2)
Output	J	Control operation result

(Example)
X1 and X2 are input,
and CSC is executed.



Text Programming

(Example) Analog input 1 and analog input 2 are input, and the computation result is output to Y1 after execution of CSC.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD X2	X2	X1	a	b	c	Reads analog input 2.
CSC	MV1	a	b	c	c	Executes cascade control. (The control operation result is stored to S1.)
ST Y1	MV1	a	b	c	c	Outputs to Y1.

► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

SSC

Category	Control functions		
Name of Computation	Selector control	Computation Instruction Symbol	SSC

Explanation of Operation

Function block programming : Process variable 1 (A) and process variable 2 (B) are input to perform selector control operation.
The computation result is output to J.

Text programming : Process variable 1 (S2) and process variable 2 (S1) are input to perform selector control operation.
The computation result is output to S1. (S3 to S5 are pushed up.)

- For both inputs and outputs, 0.0 to 1.0 is the equivalent of 0 to 100%. When PV has been computed by actual quantities before this instruction, use the normalization (NORM) instruction to normalize to 0.0 to 1.0.
- Flags, such as the operation mode and parameters other than PV, are read and written when a control instruction is executed.
  These parameters and flags are handled by front panel key operations, communication, and the read and store (LD reg, ST reg) instructions.

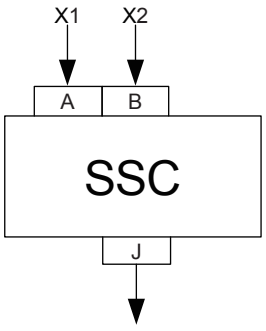
For details, see "5.4 How to Use the Selector (SSC) Control Module."

Restriction on Number Used	One SSC to be executed during one control period	Overflow	No

Function Block Programming

Computing Module	Symbol	Explanation
Input	A	Process variable 1 (PV1)
	B	Process variable 2 (PV2)
Output	J	Control operation result

(Example)
X1 and X2 are input,
and SSC is executed.



Text Programming

(Example) Analog input 1 and analog input 2 are input, and the computation result is output to Y1 after execution of SSC.

Text Program	S1	S2	S3	S4	S5	Explanation
LD X1	X1	a	b	c	d	Reads analog input 1.
LD X2	X2	X1	a	b	c	Reads analog input 2.
SSC	MV1	a	b	c	c	Executes selector control. (The control operation result is stored to S1.)
ST Y1	MV1	a	b	c	c	Outputs to Y1.

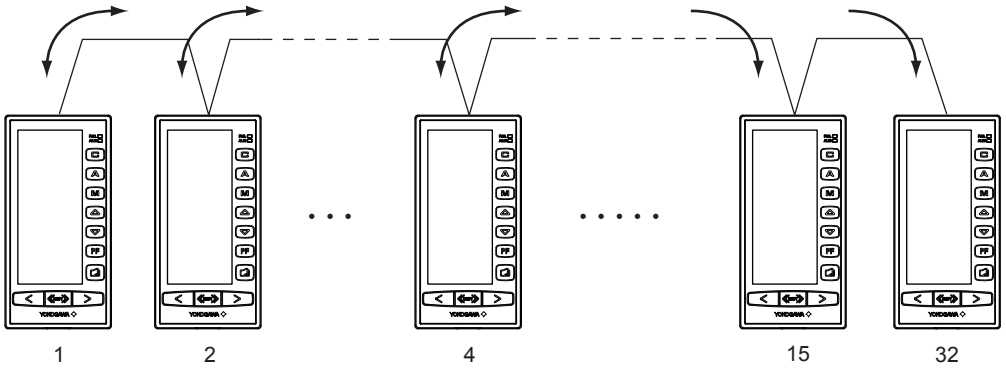
► Operation of S register before and after execution of instruction: "Chapter 12 List of Text Program Instructions"

Blank Page

8.1 Overview of Function

Peer-to-peer communication can be used on YS1700 (optional code: /A31).
Peer-to-peer communication enables up to 32 YS1700s to be connected. Of these 32 YS1700s, four units can send four analog data and 16 digital data, and receive 16 analog data and 64 digital data. The remaining 28 units can only receive 16 analog data and 64 digital data. The user can send and receive data simply by reading data from peer-to-peer communication registers (data reception) or writing data to peer-to-peer communication registers (data transmission) by the user program on the YS1700 without being aware that communications is being performed.

Controller Nos.1 to 4 can transmit and receive data. Controller Nos.5 to 32 can only receive data.



Max. 32 units

0801E.ai

Specifications of Peer-to-peer Communications

Item	Specifications
Interface	RS-485, 2-wire type
Transmission control procedure	Asynchronous, non-procedural, half-duplex
Baud rate	19200 bps
Connection method	Parallel connection
Number of connected units	Max. 32 (4 transmitting/receiving controllers, 28 receiving-only controllers
Maximum connection distance	1200 m
Amount of data transmitted	(4 analog data + 16 digital data) per transmitting/receiving controller
Amount of data received	16 analog data + 64 digital data
Transmitted data update period	200 ms (asynchronous with user program execution period)
Communication fail detection time	2 seconds

8.2 Peer-to-peer Communication Settings

8.2.1 Setting Peer-to-peer Communication, Communication Address and Terminating Resistor

Setting Display

Engineering Display

0802E.ai

Operation Display > SHIFT+M keys (to the Tuning Menu Display) > SHIFT+M keys (to the Engineering Menu Display) > [COMM] software key (Communication Setting Display)

Setting Details

Parameter	Name	Setting Range	Factory Default
PSL	RS-485 protocol selection	PCL: PC link communication PCLSUM: PC link communication (with checksum) MODASC: Modbus communication (ASCII) MODRTU: Modbus communication (RTU) YS: YS protocol P-to-P: Peer-to-peer communication	MODRTU
ADRS	RS-485 communication address	1 to 4: Transmitting/receiving controllers 5 to 99: Data receiving controllers only (Note)	1
TRMR	RS-485 communication terminating resistor ON/OFF	OFF, ON	OFF

Note: Do not set the same communication address to different controllers.

Description

- ○ RS-485 protocol selection
  Set "P-to-P" to the controllers that are made to perform peer-to-peer communication.
- ○ RS-485 communication address
  Set any communication address between 1 to 4 to controllers that transmit and receive data. Set a unique address to each controller. Do not set the same address to two or more controllers.
  Set any communication address within the range 5 to 32 to controllers that only receive data. Set a unique address to each controller. Do not set the same address to two or more controllers.
- ○ RS-485 communication terminating resistor ON/OFF
  Set the terminating resistor on the last controller on the network to ON.

► For wiring: see "Wiring for the Serial Communication Interface (Optional Code /A31)" in YS1500 Indicating Controller/YS1700 Programmable Indicating Controller Operation Guide.

8.2.2 Peer-to-peer Communication Registers

Data that can be transferred by peer-to-peer communication is analog data and digital data. Digital data is in either of two states, ON (1) or OFF (0), depending on the rules of the user program.
Transmitted/received data can be used in the user program via peer-to-peer communication registers. The following table shows the peer-to-peer communication registers and the read/write operations performed on these registers by the user program.
*Registers are floating point numbers (single-precision real numbers).

Peer-to-peer Communication Registers

Register Name	Name	Explanation	Data Type
CXn	Peer-to-peer communication analog input register	n: 01 to 04 Data received from communication address 1 n: 05 to 08 Data received from communication address 2 n: 09 to 12 Data received from communication address 3 n: 13 to 16 Data received from communication address 4	Floating point number (single-precision real number)
CYn	Peer-to-peer communication analog output register	n: 01 to 04 Data transmitted to other controllers	Floating point number (single-precision real number)
CIn	Peer-to-peer communication digital input register	n: 01 to 16 Data received from communication address 1 n: 17 to 32 Data received from communication address 2 n: 33 to 48 Data received from communication address 3 n: 49 to 64 Data received from communication address 4	Digital data (0, 1)
COn	Peer-to-peer communication digital output register	n: 01 to 16 Data transmitted to other controllers	Digital data (0, 1)
CFn	Reception timeout flag	n: 01 to 04 Indicates the status (normal/error) of the data received from communication address n.	Status data (0: normal, 1: error)

Read/Write Instruction Symbols in User Program

Instruction Symbol	Instruction
LD CXn	Reads CXn to computation register (S1).
LD CIn	Reads CIn to computation register (S1).
LD CFn	Reads CFn to computation register (S1).
ST CYn	Stores the data of computation register (S1) to CYn.
ST COn	Stores the data of computation register (S1) to COn.

8.2.3 Processing at Communication Failure

This item describes processing on YS1700 when peer-to-peer communication fails.

Causes of Communication Failure and Processing

Item	Cause of Failure	Processing on Receiving Controller	Processing on Transmitting Controller
1	Broken communication line Receiving controller communication card malfunction	The receiving controller holds the previously received peer-to-peer communication input data. If the error continues for two seconds, the reception timeout flag changes state to 1 (error).	An error cannot be detected. When the transmitting controller receives data, it detects an error on the transmitting controller as the receiving controller.
2	The user program is being downloaded or uploaded, or parameters are being setup on the transmitting controller.	Same as above	Functions are stopped.
3	The user program is being downloaded or uploaded, or parameters are being setup on the receiving controller.	Functions are stopped. Note, however, that even if functions are stopped, peer-to-peer communication input data is received normally, and stored to registers CX and CI.	An error cannot be detected. When the transmitting controller receives data, it detects an error on the transmitting controller as the receiving controller.
4	Failure of transmitting controller	The receiving controller holds the previously received peer-to-peer communication input data. If the error continues for two seconds, the reception timeout flag changes state to 1 (error).	Failure
5	Power failure on transmitting controller	The receiving controller holds the previously received peer-to-peer communication input data. If the error continues for two seconds, the reception timeout flag changes state to 1 (error).	A power failure has occurred. For details on processing during a power failure, see "8.2.4 Processing at Power Failure."
6	Power failure on receiving controller	A power failure has occurred. For details on processing during a power failure, see "8.2.4 Processing at Power Failure."	An error cannot be detected. When the transmitting controller receives data, it detects an error on the transmitting controller as the receiving controller.
7	Communication error (parity error, framing error)	The receiving controller holds the previously received peer-to-peer communication input data. If the error continues for two seconds, the reception timeout flag changes state to 1 (error).	An error cannot be detected. When the transmitting controller receives data, it detects an error on the transmitting controller as the receiving controller.

8.2.4 Processing at Power Failure

This item describes processing when a YS1700 controller is recovered from a power failure during peer-to-peer communication.

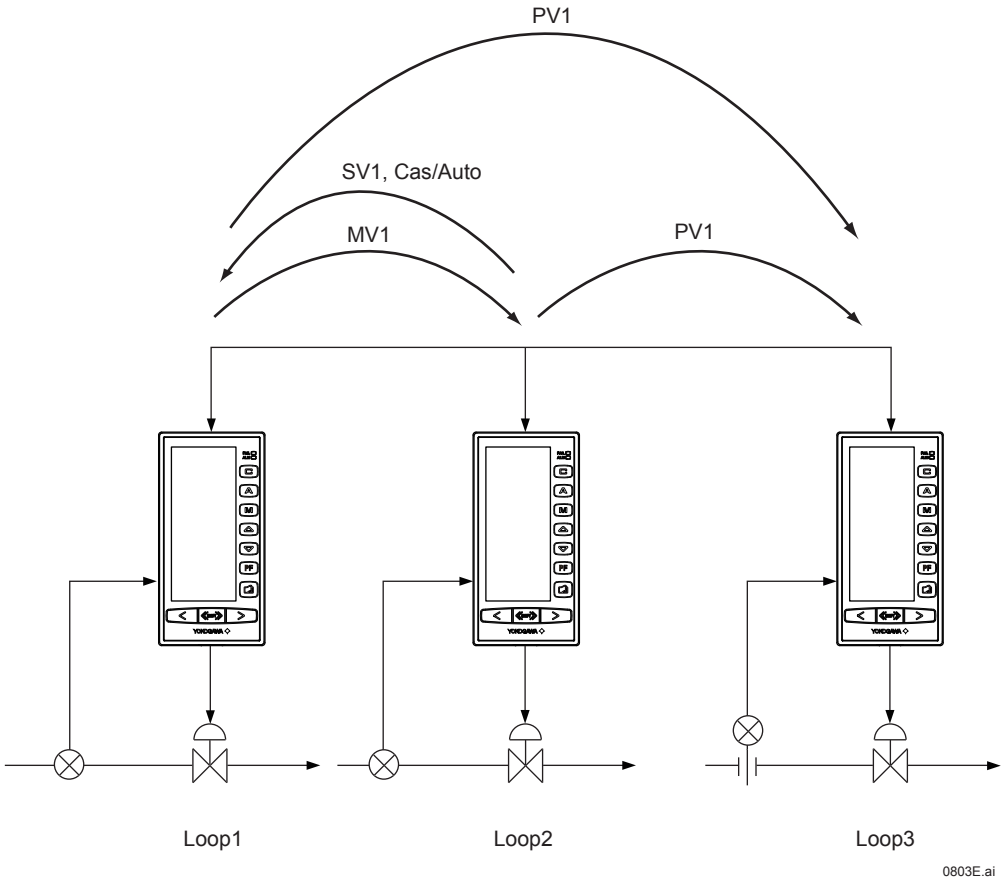
(1) Cold start
The values of registers CX, CY, CI, and CO start from 0%. When the transmitting controller or user program writes data to these registers, that data becomes valid. At a start, the state of the reception timeout flag (CFn) is 1 (error). However, when communication is recovered, it changes to 0 (normal).

(2) Hot start
The values received before the power failure are held as the values of registers CX, CY, CI, and CO. When the transmitting controller or user program writes data to these registers, that data becomes valid. At a start, the state of the reception timeout flag (CFn) is 1 (error). However, when communication is recovered, it changes to 0 (normal).

8.3 Example of Peer-to-peer Communication

Let us assume a peer-to-peer communication system such as follows:

- The system is configured by three loops, loop 1, loop 2 and loop 3.
- Loop 1 requires the setpoint (SV1) and Cas/Auto status (CAF1 flag) of loop 2.
- Loop 2 requires the manipulated output variable (MV1) of loop 1.
- Loop 3 requires the process variables (PV1) of loops 1 and 2.



Example of Peer-to-peer Communication

Assignment of communication address

As loops 1 and 2 require transmission and reception, set the communication address of these loops to 1 and 2, respectively. Loop 3 only receives data, so set its communication address to 5. As there are few controllers in this case, 3 (the communication address of a transmitting/receiving controller) also can be assigned to loop 3.

Assignment of communication registers

Assign data to the peer-to-peer communication registers of each controller as shown in the following table.

Example of Assigning Peer-to-peer Communication Registers on a Transmitting Controller

		Loop1	Loop2
Peer-to-peer communication analog output register	CY01	Process variable (PV1)	Process variable (PV1)
	CY02	Manipulated output variable (MV1)	Setpoint value (SV1)
	CY03	Unused	C/A status (CAF1)
	CY04	Unused	Unused

Example of Assigning Peer-to-peer Communication Registers on a Transmitting Controller

		Content of Register
Peer-to-peer communication analog input register	CX01	CY01 of communication address 1 controller = process variable (PV1) of loop 1 controller
	CX02	CY02 of communication address 1 controller = manipulated output variable (MV1) of loop 1 controller
	CX03	CY03 of communication address 1 controller=unused
	CX04	CY04 of communication address 1 controller=unused
	CX05	CY01 of communication address 2 controller = process variable (PV1) of loop 2 controller
	CX06	CY02 of communication address 2 controller = setpoint value (SV1) of loop 2 controller
	CX07	CY03 of communication address 2 controller = C/A status (CAF1) of loop 2 controller
	CX08	CY04 of communication address 2 controller=unused
Reception timeout flag	CF01	Judges a normal communication or communication error from the communication address 1 controller.
	CF02	Judges a normal communication or communication error from the communication address 2 controller.

Example of User Program

• Programming by text programming

(1) Example of loop 1 user program

```
    LD          CF02          ;Reception timeout flag
    GIF         @ERR          ;Jump to @ERR at receive error
;<<Processing when peer-to-peer communication data is normal>>
    LD          CX07          ;Read Cas/Auto status of loop 2
    GIF         @LOOP-CAS     ;Jump to @LOOP-CAS in Cas mode
    ..............
    ..............
    LD          CX06          ;Read SV1 of loop 2
    ST          MV1           ;
@LOOP-CAS
    ..............
;<<Creation of send data>>
    LD          PV1
    ST          CY01          ;Write process variable 1 (PV1) to send register CY01
    LD          MV1
    ST          CY02          ;Write manipulated output variable 1 (MV1) to send register
                              CY02
    ..............
;<<Processing when peer-to-peer communication data is in error>>
@ERR
    ..............
    ..............
```

(2) Example of loop 2 user program

```
    LD          CF01          ;Reception timeout flag
    GIF         @ERR          ;Jump to @ERR at receive error
;<<Processing when peer-to-peer communication data is normal>>
    LD          CX02          ;Read manipulated output variable of loop 1
    ST          CSV1          ;Write to cascade setting value
    ..............
    ..............
;<<Creation of send data>>
    LD          PV1
    ST          CY01          ;Write process variable 1 (PV1) to send register CY01
    LD          SV1
    ST          CY02          ;Write setpoint value 1 (SV1) to send register CY02
    LD          CAF1
    ST          CY03          ;Write Cas/Auto mode 1 (CAF1) to send register CY03
    ..............
;<<Processing when peer-to-peer communication data is in error>>
@ERR
    ..............
    ..............
```

(3) Example of loop 3 user program

```
    LD          CX01          ;Read PV1 of loop 1
    ST          T01           ;Write to temporary memory register 1
    LD          CF01          ;Read reception timeout flag 1
    GIF         @ERR          ;Jump to @ERR at input data error
;<<Processing when loop 1 peer-to-peer communication data is normal>>
    LD          CX05          ;Read PV1 of loop 2
    ST          T02           ;Write to temporary memory register 2
    LD          CF02          ;Read reception timeout flag 2
    GIF         @ERR          ;Jump to @ERR at input data error
;<<Processing when loop 2 peer-to-peer communication data is normal>>
    .............
    .............
;<<Use T01 and T02 to perform compensation operation, etc.>>
    .............
    .............
;<<Processing when peer-to-peer communication data is in error>>
@ERR
    .............
    .............
```

This chapter mainly describes how to maintain user programs and how to use the Check Support Function.

9.1 Maintenance of User Programs

WARNING

When performing maintenance on user programs, remove the YS1700 from the instrumentation panel to set it to an offline status. Do not perform maintenance on users programs with the YS1700 in an online status (plant control status).

Operation of I/O signals during user program maintenance

The YS1700 enters a "functions stopped status" and "STOP" appears at the top right of the LCD display when the user program is uploaded or downloaded. When this happens, I/O signals, internal registers and the operation mode enter the statuses shown in the following table.

Signal/Data	Status during Maintenance	Remarks
Operation mode	M mode	
Setpoint values SV1, SV2	Value just before stop is held.	
Manipulated output variables MV1, MV2	Value just before stop is held.	
Analog outputs 1 to 4	Value just before stop is held.	
Analog output registers Y1 to Y4	Value just before stop is held.	
Digital outputs 1 to 10	Value just before stop is held.	
Digital output registers DO1 to DO10	0	
PF key status	OFF	
PF key status register LP01	0	
Temporary memory registers T01 to T60	0	
Output registers Y5, Y6	0	
Digital output registers DO11 to DO32	0	
Dynamic operation	Reset	First order lag, dead time, etc.

9.2 Using Check Support Function

The Check Support Function is available for the YSS1000 version R1.03 or higher.

WARNING

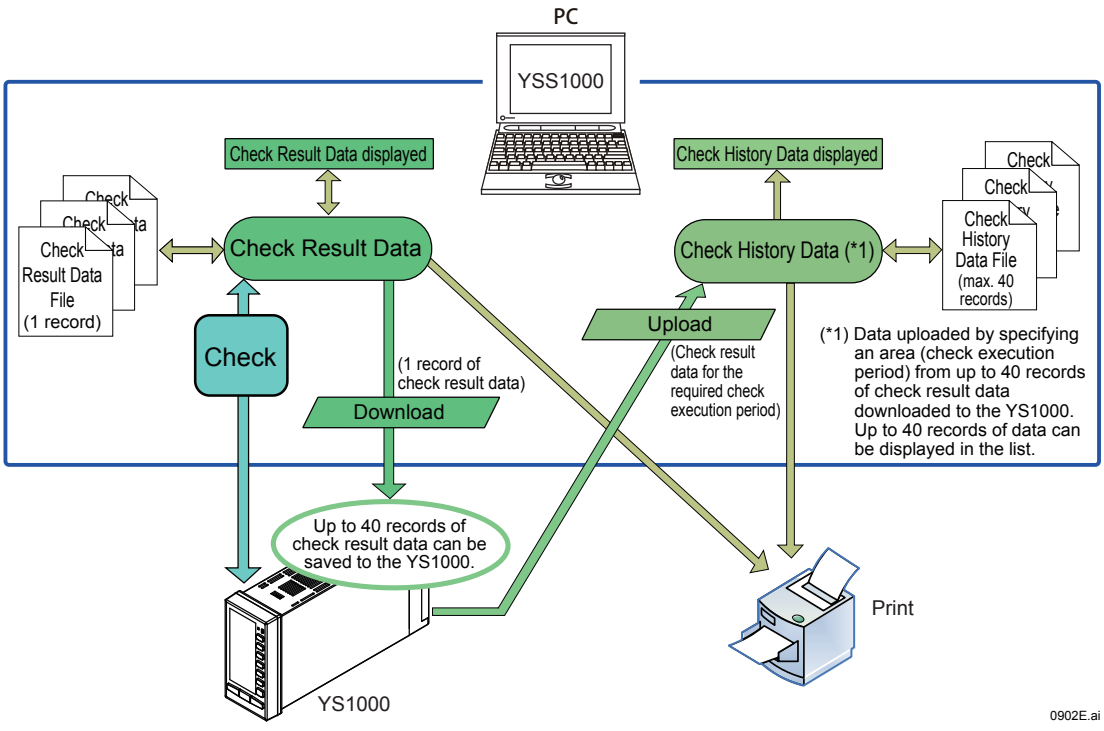
- The Check Support Function is intended for maintenance personnel, who have practical experience with the maintenance of instrumentation and control equipment, and who have professional knowledge of operation and monitoring tasks with controllers.
- When checking the YS1000 using the Check Support Function of YSS1000, be sure to perform a check while the YS1000 is in the case or housing in order to meet safety requirements.
- When removing the YS1000 main unit from the device or control panel, be sure to shut off the power supply to the YS1000 main unit and then remove the main unit while it is in the case or housing.
- Be sure to use instruments that are calibrated to perform a check.
- Be sure to perform a check in the operating environment at an ambient temperature of 23°C ± 2°C.

CAUTION

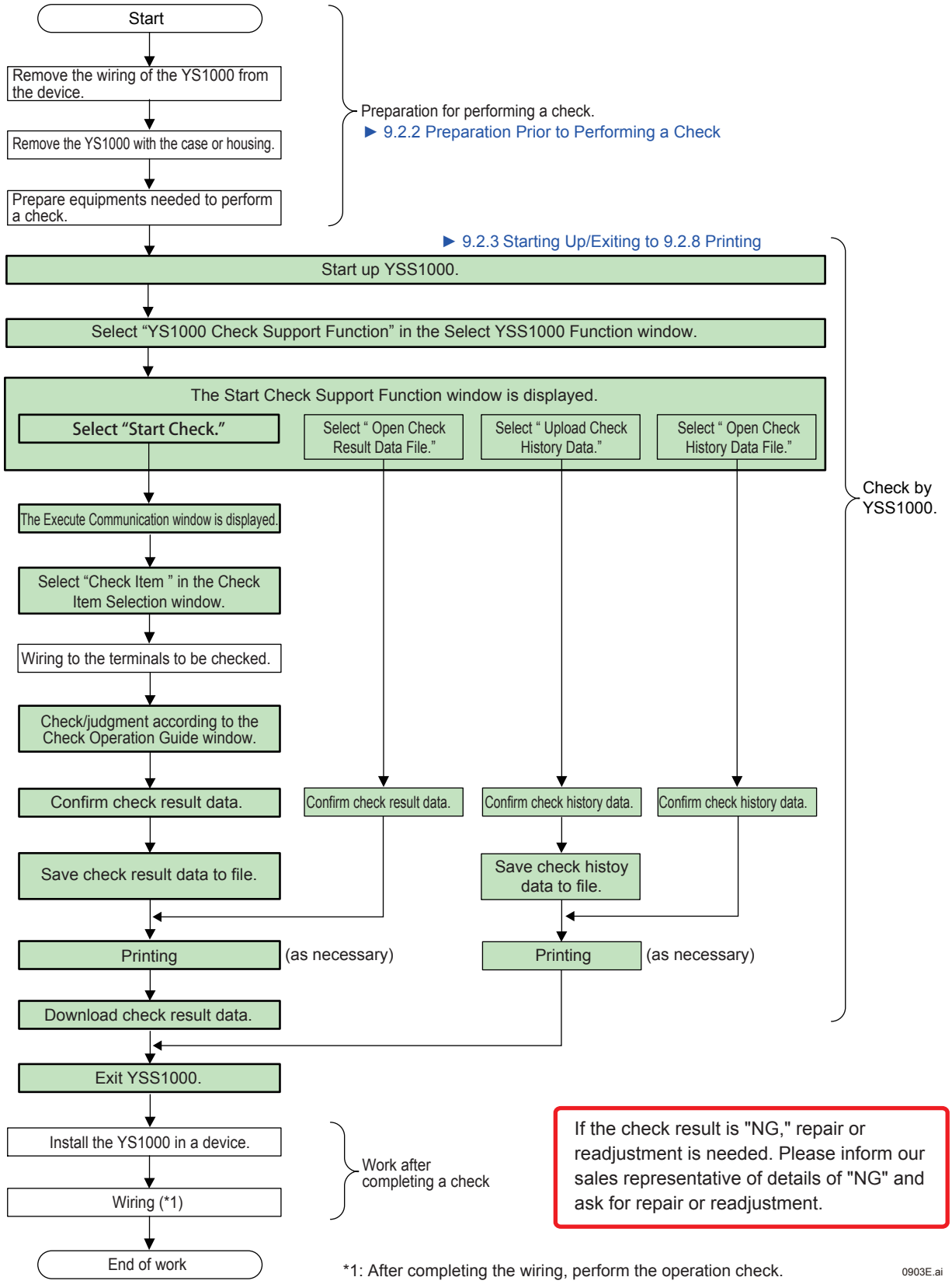
- Before powering off the YS1000, be sure to save the set data of parameters and user programs. After finishing a check, make sure that the set data of parameters and user programs are identical to those before performing the check.
- If the check result is "NG," repair or readjustment is needed. Please inform our sales representative of details of "NG" and ask for repair or readjustment.

Note

To use the Check Support Function, Adobe Reader version 8.1 or higher needs to be installed on your PC.



Check Flow



Note

For installation and wiring, see "Installation and Wiring" in the respective Operation Guides.

9.2.1 Applicable Models

The Check Support Function can be used for the YS1700, YS1500, YS1310, YS1350, and YS1360.

Note

The Check Support Function can be used only for program communication (connecting by the YS1000 exclusive cable). It cannot be used for RS-485 communication and Ethernet communication.

9.2.2 Preparation Prior to Performing a Check

1. Prepare the following instruments which are needed to perform a check.

Name	Requirements specification	Accuracy specification	Recommended instrument	Number of Units
Digital Multimeter (DMM)	Shall be capable of measuring the current, voltage, and resistance	Voltage: ±0.01% Current: ±0.05% Resistance: ±0.01%	Yokogawa 7561 or equivalent (*1)	1
Standard DC Voltage/Current Generator (GOS)	• Shall be capable of outputting the voltage • Shall be capable of sinking a current of 60 mA in the SINK mode	Voltage: ±0.01% Current: ±0.03%	Yokogawa 7651 or equivalent	1

*1: When checking the current output, be sure to use a measuring instrument with sufficient current measurement accuracy, or measure the voltage of a shunt resistor (e.g. 250 Ω) with the voltage measuring instrument and convert it to current.
Shunt resistor X010-250-1 (Yokogawa): 250 Ω, accuracy ±0.1%, using M4 screws

Note 1: Use a DMM and GOS that have been calibrated.
Note 2: When using a calibrator (for example, Yokogawa Meters & Instruments CA150) instead of a DMM and GOS, the check results should be considered as reference values because the accuracy specification is not satisfied.

2. Check to make sure that the power supply to the YS1000 main unit is shut off.

3. Remove the wiring of the YS1000.

► "Installation and Wiring" - "Wiring" of the respective Operation Guides

4. Remove the YS1000 from the device or control panel while the unit is in the case or housing.

► "General Specifications" - "General Safety Standards" of the respective User's Manuals (CD-ROM)

5. Connect the YS1000 and the PC.
For connection, use the connector in the swing-up panel of the YS1000.

► "1.2 Connecting the YS1000 to a PC and Setup Parameters" - "Connecting by the YS1000 exclusive cable" of this document

Note

- Check to make sure that the YSS1000 (R1.03 or higher) is installed on the PC.
- The warm up time shall be at least 1 minute after the power is turned on while the wiring for the Check Support Function is completed. It shall be at least 3 minutes when the optional code /A0☐ direct input is implemented. As for the detalis of the warm up time of DMM and GOS, refer to the respective specifications of the instruments.

Notes during Check and YS1000 Window Displays

CAUTION

During a check, do not perform any YS1000 key operations other than those of the operating instructions in the Check Operation Guide window.

Note

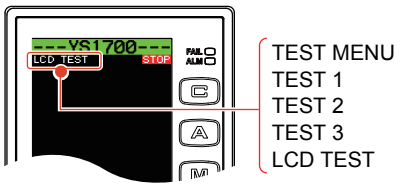
During a check, the control status (RUN, STOP, TEST1, TEST2) cannot be changed.

YS1000 window display during a check

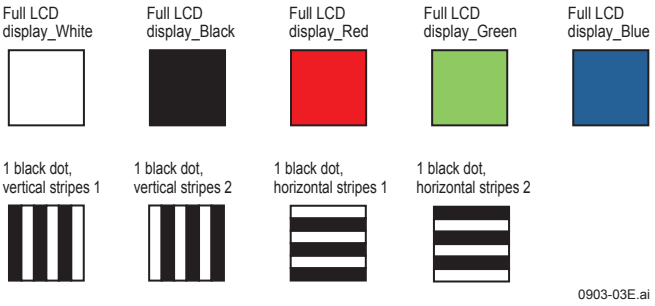
The YS1000 switches to the check mode (*1) according to the check item.
The YS1000 LCD display and LED lamp switch to the display mode for performing a check.

*1: The YS1000 window display in the check mode is any of the following.

Any of the following window titles: "TEST MENU," "TEST 1," TEST 2," "TEST 3," or "LCD TEST" is displayed.



A display pattern is displayed during the checking of the LCD .



9.2.3 Starting Up/Exiting

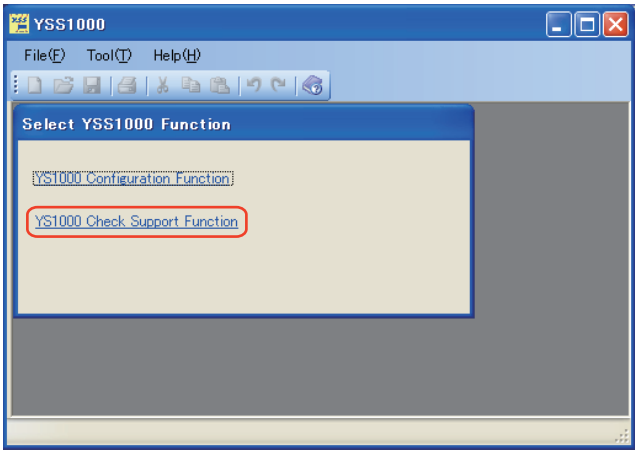
(1) Start Up

Note

To start the Check Support Function, the YS1000 Configuration Function needs to be exited.

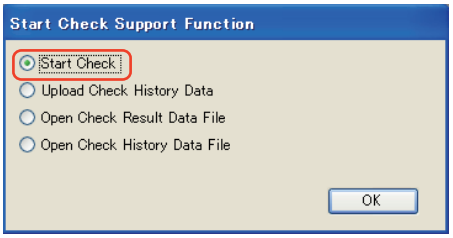
Operation

1. Click Windows **Start>All Programs>YS1000 Series>YSS1000**.



0904E.ai

2. Click **YS1000 Check Support Function** or **Tool>YS1000 Check Support Function** from the menu to display the Start Check Support Function window.

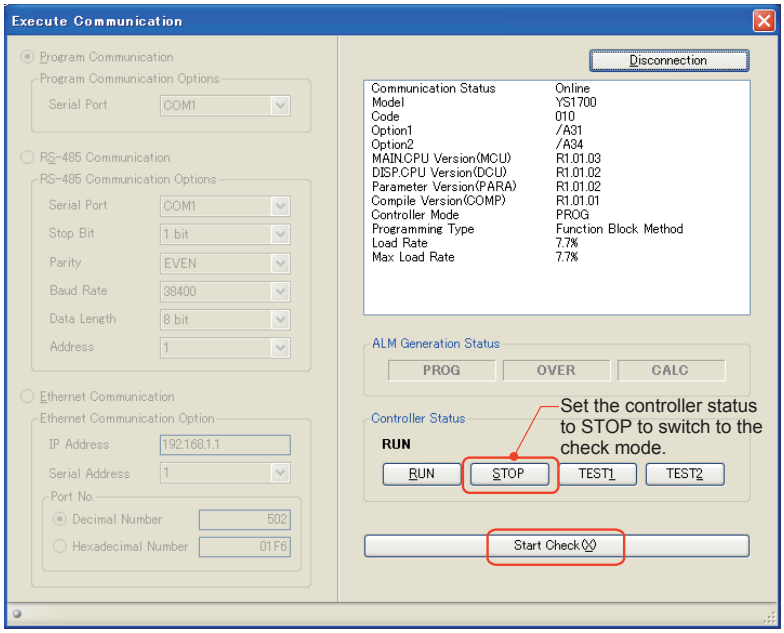


0905E.ai

►Upload Check History Data:
9.2.7 Download/Upload/Delete Data

►Open Check Result Data File, Open Check History Data File:
9.2.6 Opening Data File/Saving

3. Select **Start Check**, and click **OK** to display the Execute Communication window.



0906E.ai

4. To switch to the check mode, click **STOP** to set the controller status to the operation STOP status.

5. Click **Start Check**.

6. The following message is displayed (only for the first check).

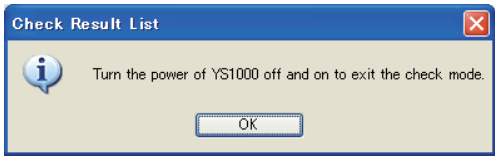
7. Click **OK**.

   The three windows (Check Item Selection, Check Operation Guide, and Check Result List windows) are displayed.

(2) Exit

Operation

1. Click **File**>**Exit** from the menu or  at the top right of the window.



0909E.ai

2. Click **OK**.

   YSS1000 is exited.

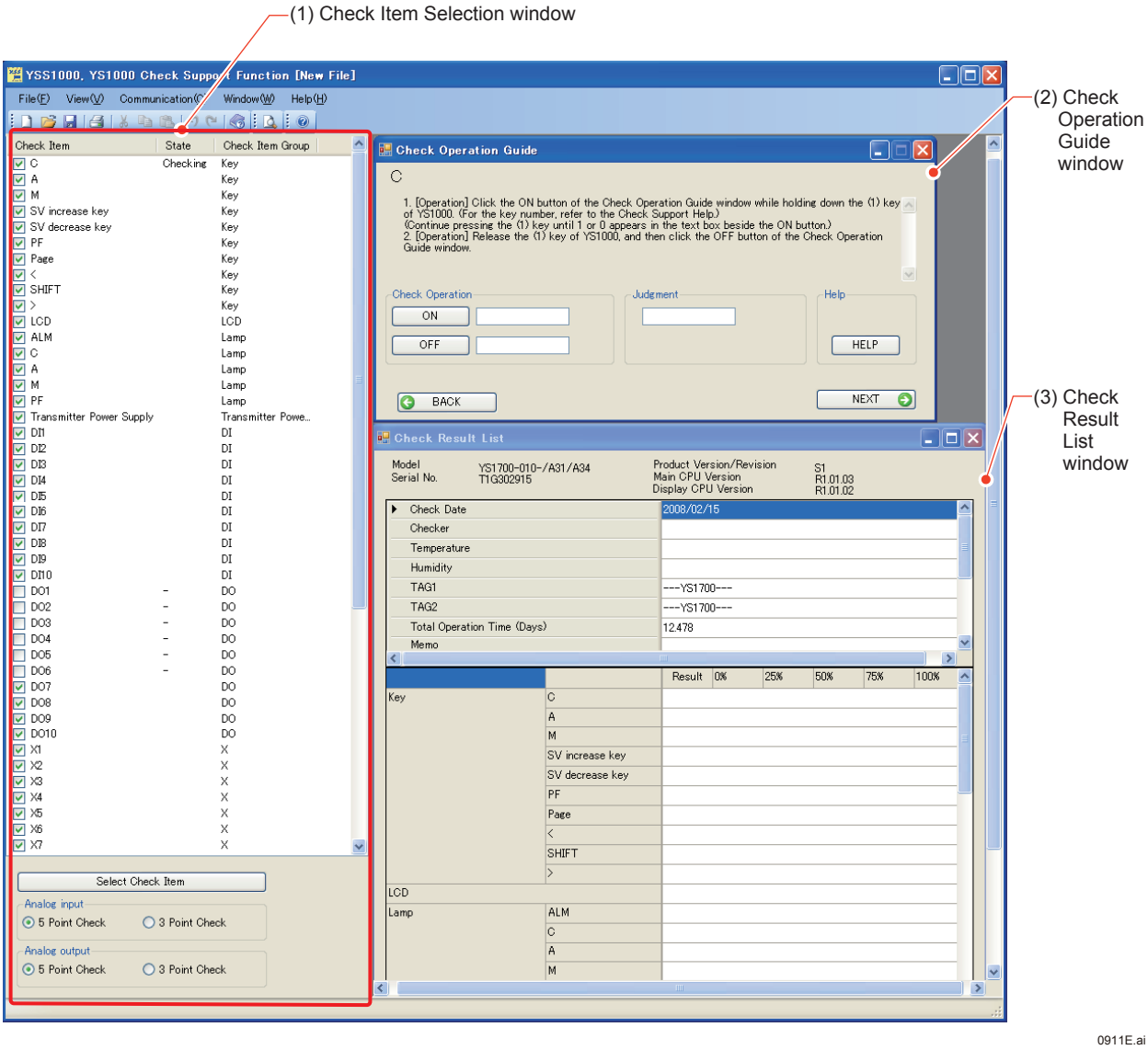
Note

To switch to the normal mode, turn the power of the YS1000 off and on after the check is finished. When using the YS1000 Configuration Functions, check to make sure that the YS1000 is in the normal mode. If the YS1000 is in the check mode, the communication function of the YS1000 Configuration Functions does not work correctly.

9.2.4 Description of Each Window

Windows Displayed during a Check

The following three windows are simultaneously displayed during a check.

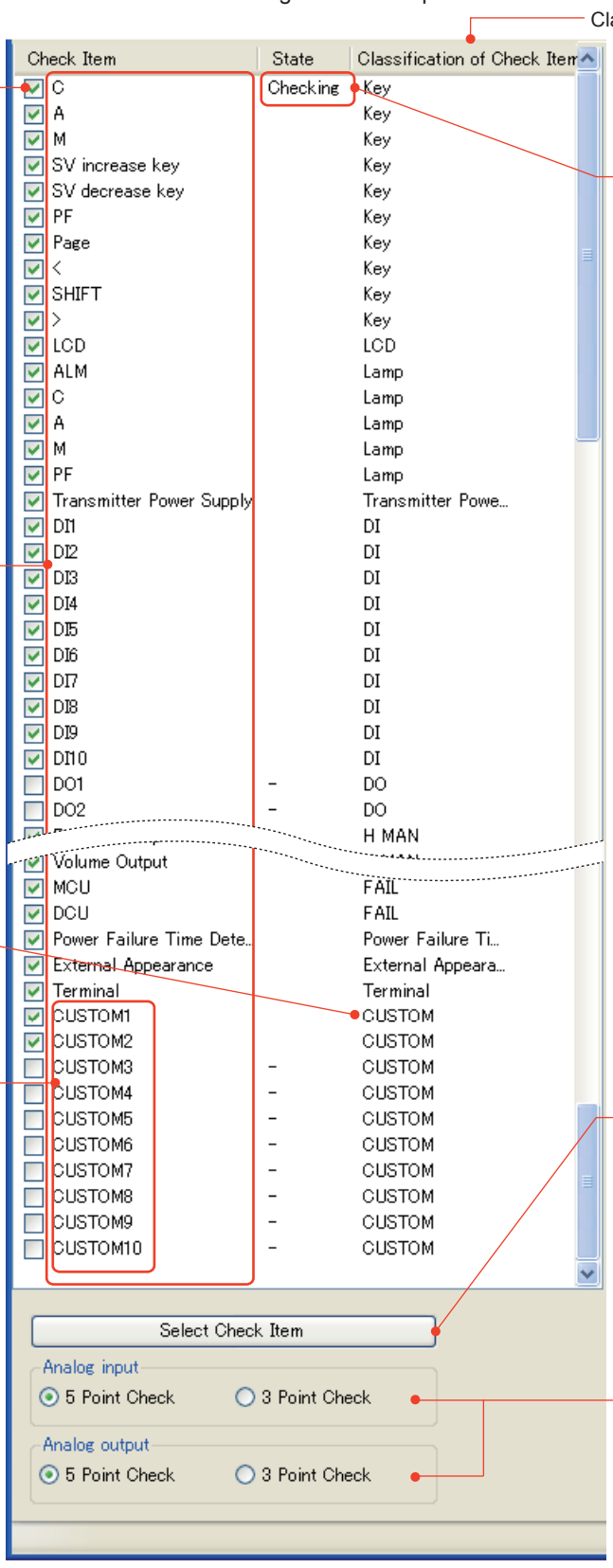


0911E.ai

Note: For details on the windows (1) to (3) above, refer to the descriptions on and after the following page.

(1) Check Item Selection window

This window allows for selecting the items to perform checks.



Checkbox
Add checks for the items to perform a check.
This function allows for choosing not to perform a check of the unused I/O terminals and functions.
(If the check is cleared from an item, that item is excluded from the check items.)
When performing a check for the first time, checkmarks are displayed for all items that can be checked (except for the CUSTOM items.)
When performing a check for the second time and so on, checkmarks are displayed for items that were selected to perform checks the last time.

Check Item
Check items vary depending on the model. See the Check Item List on the following page.

Custom check
These check items can be registered by the user. Up to 10 check items can be added.

Text box
Custom check items with a checkmark can be named. Up to 10 single-byte characters can be used.
►"9.2.5 Performing Check" - "(15) Custom check"

Note

If the check item name of a custom check that has been used is changed, the check history data of that changed custom check is deleted during downloading from the past check history data saved under the check item name before the change.

Status display
The check status, "checking" or "checked," is displayed in the status column. If the item is not checked, the column is blank.
"—" is displayed for Items without a checkmark.

CAUTION

If the checkmark is cleared from the check item that was "checked," the check result for that check item is deleted from the Check Result List window.

Select Check Item button
A check proceeds in the order of the check items with checkmarks. However, a check can be performed again anytime by clicking a check item and pressing the Select Check Item button.

Radio buttons
Three-Point Check or Five-Point Check can be selected for the calibration points for the analog input check and analog output check.

0912E.ai

Check Item List

	Check Item	YS1700 -01□	YS1700 other than YS1700-01□	YS1500 -0□□	YS1310 -0□□	YS1350 -0□□	YS1360 -0□□	Remarks
Key	C							
	A							
	M							
	SV increase key							
	SV decrease key							
	PF							
	Page							
	<		①		—	—		
	SHIFT							
	>				—	—		
LCD								
Lamp	ALM							
	C				—			
	A				—	—	—	
	M				—	①		
	PF				—	—	—	
Transmitter Power Supply								
DI	DI1							
	DI2				—			
	DI3		②		—	—	—	
	DI4				—	—	—	
	DI5				—	—	—	
	DI6				—	—	—	
	DI7		—	—	—	—	—	
	DI8	①	—	—	—	—	—	
	DI9		—	—	—	—	—	
	DI10		—	—	—	—	—	
DO	DO1							
	DO2							
	DO3		③		①	—	—	
	DO4							
	DO5					—	—	
	DO6					—	—	
	DO7		—	—	—	—	—	
	DO8		—	—	—	—	—	
	DO9		—	—	—	—	—	
	DO10		—	—	—	—	—	
X	X1							
	X2							
	X3				—	—	—	
	X4				—	—	—	
	X5			—	—	—	—	
	X6		—	—	—	—	—	
	X7		—	—	—	—	—	
	X8		—	—	—	—	—	
Y	Y1				—	—	①	
	Y2				—			
	Y3				—	—	—	
	Y4		—	—	—	—	—	
H MAN	Balance				—	—		Disabled when the optional code /NHM is selected
	Volume				—	—		
FAIL	MCU							The MV operation is not checked for the YS1310 and YS1350
	DCU							
Power Failure Time Detection								
External Appearance			①					
Terminal								
CUSTOM	CUSTOM1							
	CUSTOM2							
	CUSTOM3							
	CUSTOM4							
	CUSTOM5							
	CUSTOM6							
	CUSTOM7							
	CUSTOM8							
	CUSTOM9							
	CUSTOM10							

① A check is possible.

② A check of the digital input is possible only when DI is set for the DI/DO specification parameter (DIO16 to DIO61).

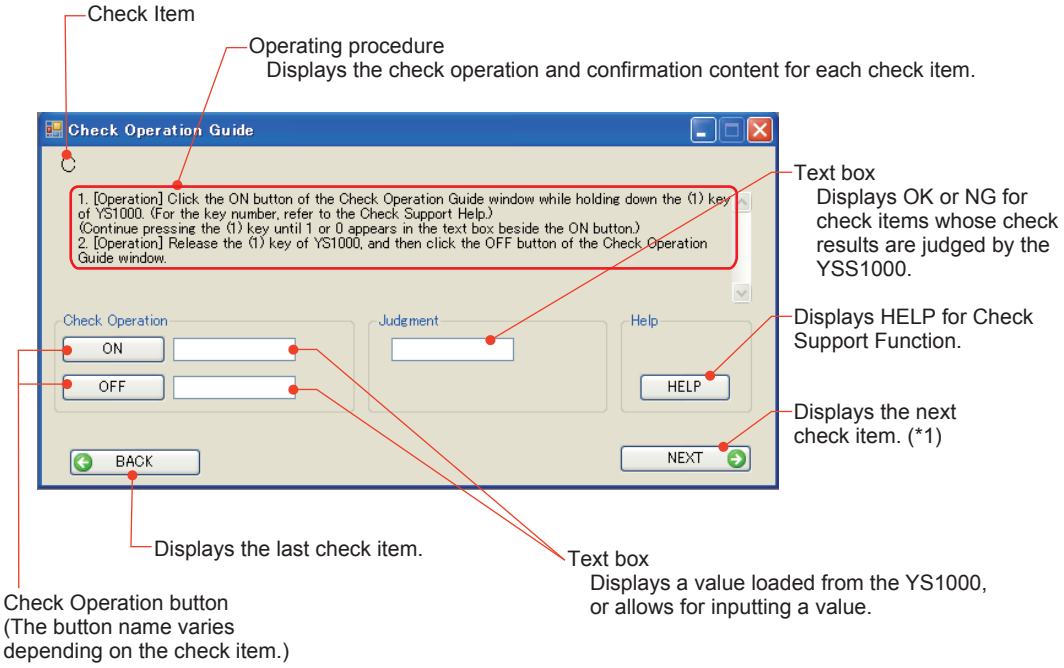
③ A check of the digital output is possible only when DO is set for the DI/DO specification parameter (DIO16 to DIO61).

— There is no check item.

(2) Check Operation Guide window

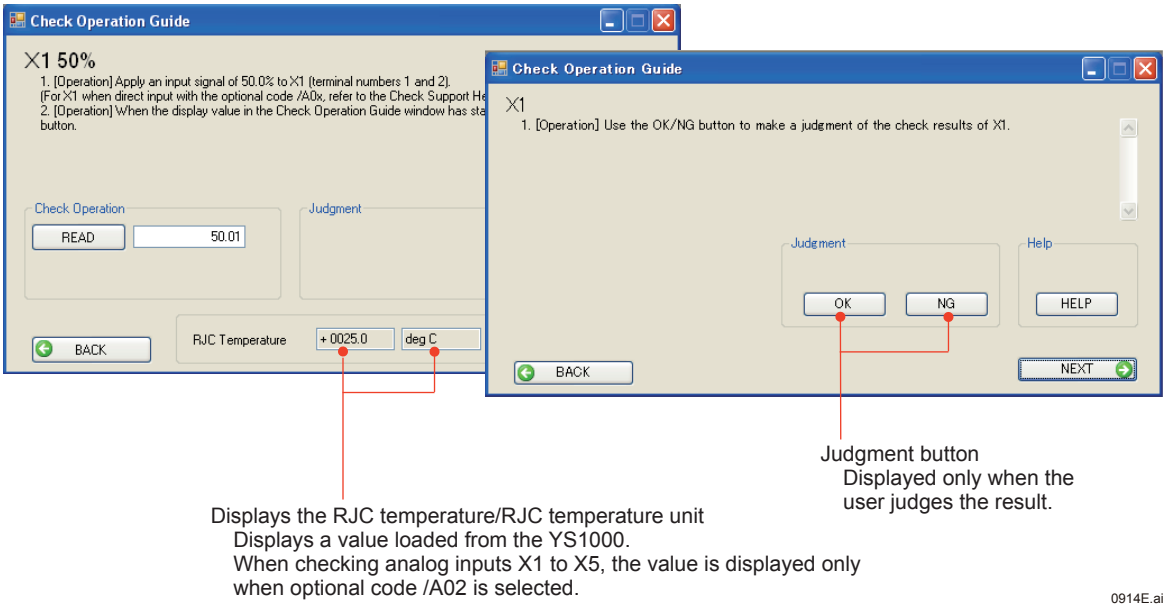
This window displays the operation of each check and the confirmation content. The display content varies depending on the check item.

<Window example for the YSS1000 to judge the check result>



0913E.ai

<Window example for the user to judge the check result>

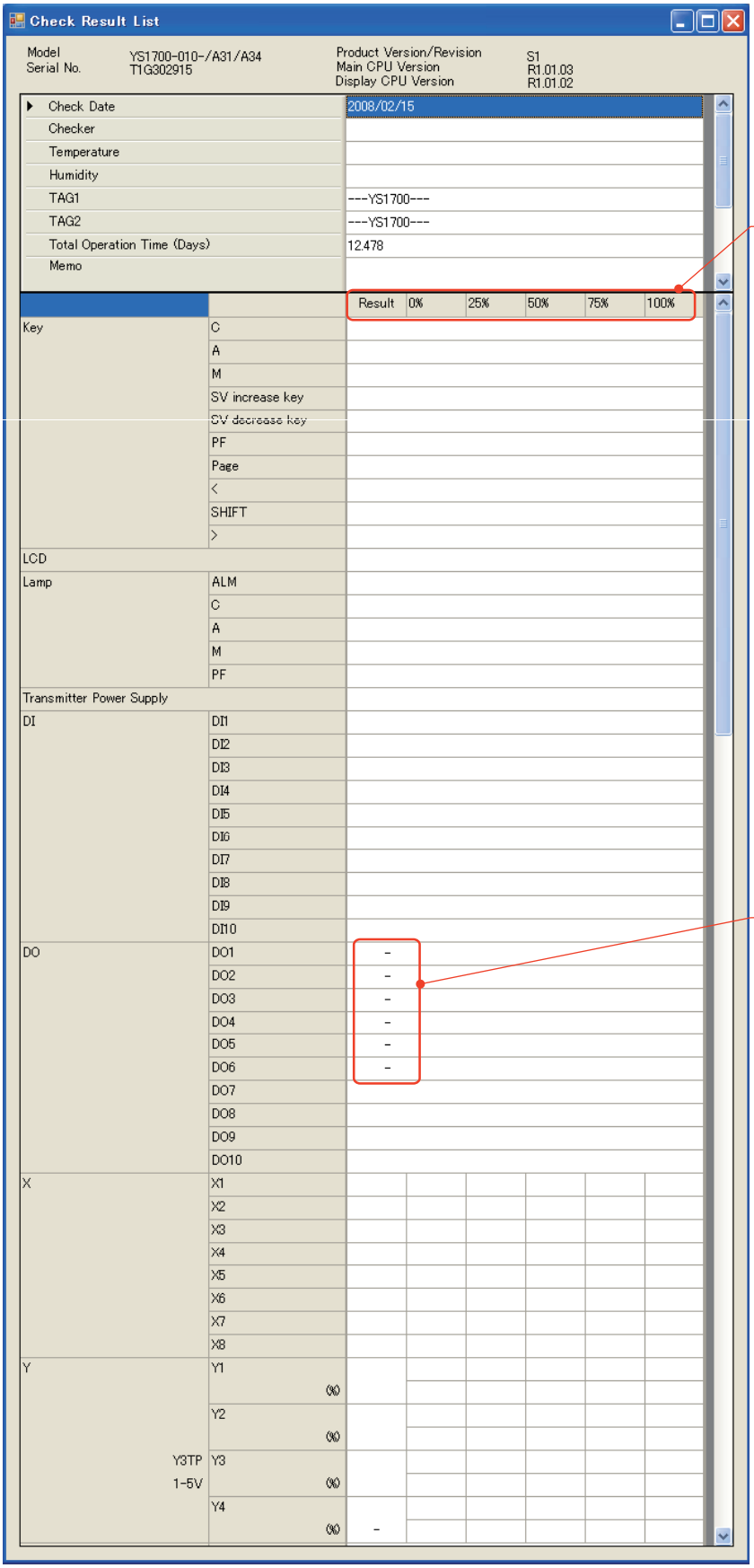


0914E.ai

*1: Clicking **NEXT** on the Check Operation Guide window displays a check item in the order of check items with checkmarks on the Check Item Selection window.

(3) Check Result List window

This window displays the check result.



For checking analog inputs (X1 to X8) and analog outputs (Y1 to Y4)

"—" is displayed in the Result field for items that are excluded from the check items. (The display items on the window are the same regardless of the selected check items and model.)

0915E.ai

<Data Types in the Check Result List>

① Data loaded from the YS1000
② Text and value input by the user
③ OK or NG selected by the user
④ OK or NG judged by the YSS1000
⑤ Value input on the Check Operation Guide window
⑥ Value input on the Check Item Selection window
⑦ %-value calculated by the YSS1000
: Data saved to the YS1000

Model	①
Serial No.	
Product Version/Revision	
MAIN.CPU Version	
DISP.CPU Version	

Check Date	②	Character restriction: Up to 10 single-byte characters (YYYY/MM/DD)
Checker		Character restriction: Up to 23 single-byte characters
Temperature		Character restriction: Up to 23 single-byte characters
Humidity		Character restriction: Up to 23 single-byte characters
TAG1	①	
TAG2		
Total Operation Time (Days) (*1)		
Memo (*2)	②	Character restriction: Up to 500 single-byte characters

		Result	0%	25%	50%	75%	100%
Key	C	④					
	A						
	M						
	SV increase key						
	SV decrease key						
	PF						
	Page						
	<						
	SHIFT						
	>						
LCD							
Lamp	ALM						
	C						
	A	③					
	M						
	PF						
Transmitter Power Supply							
DI	DI1 to DI10	④					
DO	DO1 to DO10						
X	X1 to X8		①	①	①	①	①
Y	Y1 (%)		⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦
	Y2 (%)		⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦
Y3TP (*3) ①	Y3 (%)	③	⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦
	Y4 (%)		⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦	⑤ ⑦
H MAN	Balance Lamp						
	Volume Output						
FAIL	MCU						
	DCU						
Power Failure Time Detection		④					
External Appearance							
Terminal		③					
CUSTOM1 to CUSTOM10	⑥						

*1: The unit of total operation time is "day."

*2: Input a note of anything you have to remember.
If the input characters exceed the line length, the excess characters are not printed. If that happens, press the Shift and Enter keys to add a line break. Also confirm the printed image with the print preview function.

*3: Only displayed when PROG is set in the YS1700 controller mode selection parameter (CTL).
4-20 mA or 1-5 V is displayed depending on the setting of the analog output 3 current/voltage switching parameter (Y3TP).

0916E.ai

9.2.5 Performing Check

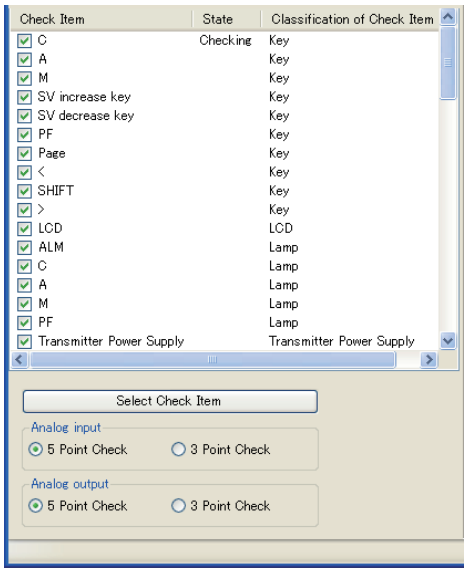
Perform a check following the Operation Procedure displayed on the Check Operation Guide window. See the **Description** below for the wiring method and judgment criteria necessary for performing a check, which is also displayed in HELP on the Check Operation Guide window.

Note

Do not unplug the connection cable during a check. Also, do not power off the YS1000, except when instructed to do so in the operating instructions for the FAIL and Power Failure Time Detection check.

(1) Selecting the items to be checked

Select the items to be checked on the Check Item Selection window.
When performing a check for the first time, checkmarks are displayed in the checkboxes of all check items that can be checked (except for the CUSTOM items). Clear the checkmarks of unused I/O terminals and functions to skip performing a check for them.
When performing a check for the second time and so on, the checkmarks are displayed for the check items that were checked the last time.
The check proceeds in the order of the check items with checkmarks.



0917E.ai

Recheck method

When a "check item" to be rechecked is clicked and the Select Check Item button is pressed, a check can be performed again.
Note: To perform a recheck, be sure to click the "check item." Do not click the checkmark.

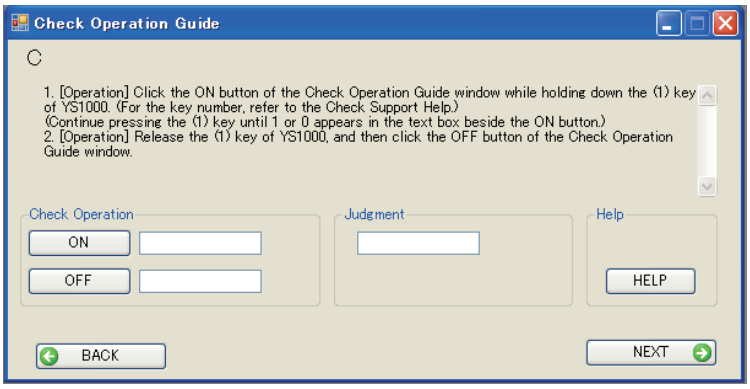
CAUTION

If the checkmark is cleared from the check item that was "checked," the check result for that check item is deleted from the Check Result List window.

► Check Item Selection window: "9.2.4 Description of Each Window" - "(1) Check Item Selection window"

(2) Key check

Check Operation Guide window



0918E.ai

Operation

1. [Operation] Click the **ON** button of the Check Operation Guide window while holding down the (1) key of YS1000. (For the key number, see the **Description** below or HELP on the Check Operatin Guide window.)

(Continue pressing the (1) key until 1 or 0 appears in the text box beside the **ON** button.)

2. [Operation] Release the (1) key of YS1000, and then click the **OFF** button of the Check Operation Guide window.

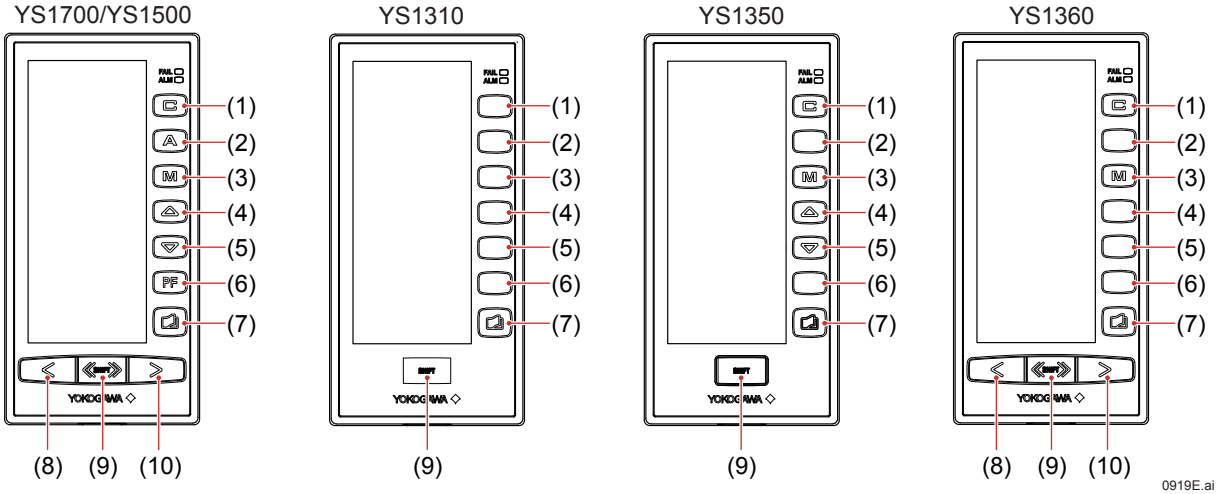
1 or 0 appears in the text box beside the OFF button.

3. Based on the results in Steps 1 and 2, "OK" or "NG" is displayed in the Judgment text box. At the same time, "OK" or "NG" is displayed in the Result field in the Check Result List.

4. Click the **NEXT** button to check the next key.

Description

The following shows the relationship between the model names and key numbers.



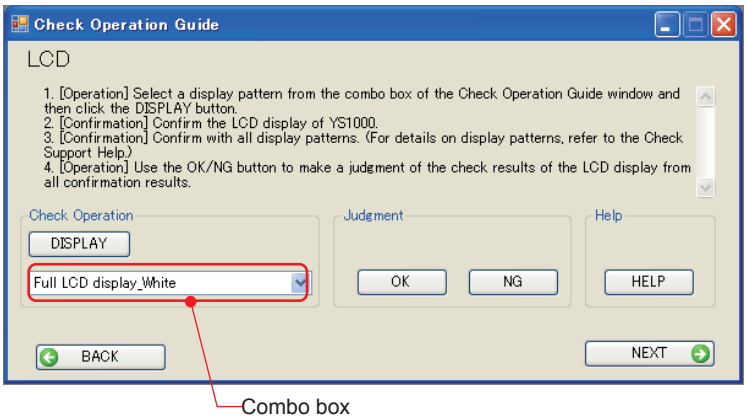
0919E.ai

Note

Since the characters on keys vary depending on the model, keys in the Check Operation Guide are indicated with numbers allocated to the keys as shown above.
Keys of the YS1310, YS1350, and YS1360 are also used as software keys even though they have no characters on them, so all keys are checked.

(3) LCD check

Check Operation Guide window



0920E.ai

Operation

1. [Operation] Select a **display pattern** from the combo box of the Check Operation Guide window and then click the **DISPLAY** button.

2. [Confirmation] Confirm the LCD display of YS1000.

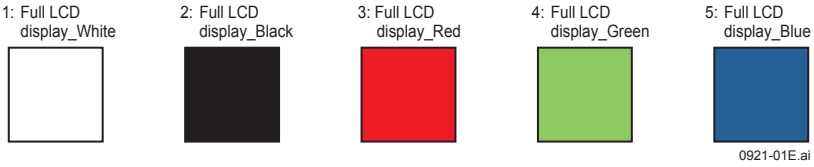
3. [Confirmation] Confirm with all display patterns.

4. [Operation] Use the **OK**/**NG** button to make a judgment of the check results of the LCD display from all confirmation results.
   "OK" or "NG" is displayed in the Result field in the Check Result List.

Description

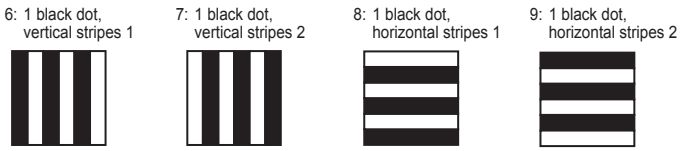
The figure below shows the display patterns to be checked.

When checking the display patterns 1 to 5:
Confirm that the LCD display color is changed to White, Black, Red, Green and Blue when clicking the **DISPLAY** button.



0921-01E.ai

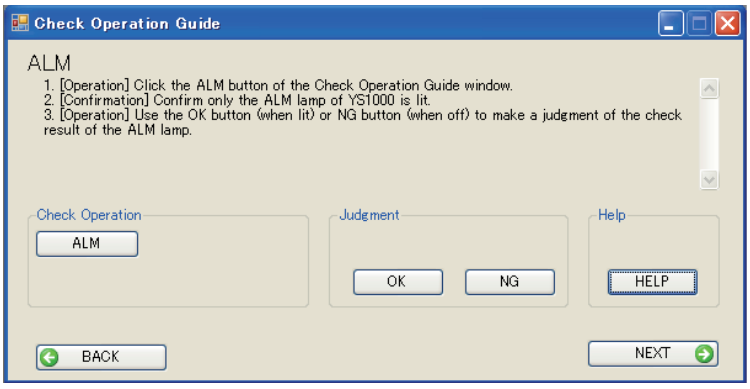
When checking the display patterns 6 to 9:
Confirm that the LCD display is normal when clicking the **DISPLAY** button. (Check whether the width of the stripes is the same, the stripes are straight lines, etc.)



0921-02E.ai

(4) LED lamp check

Check Operation Guide window



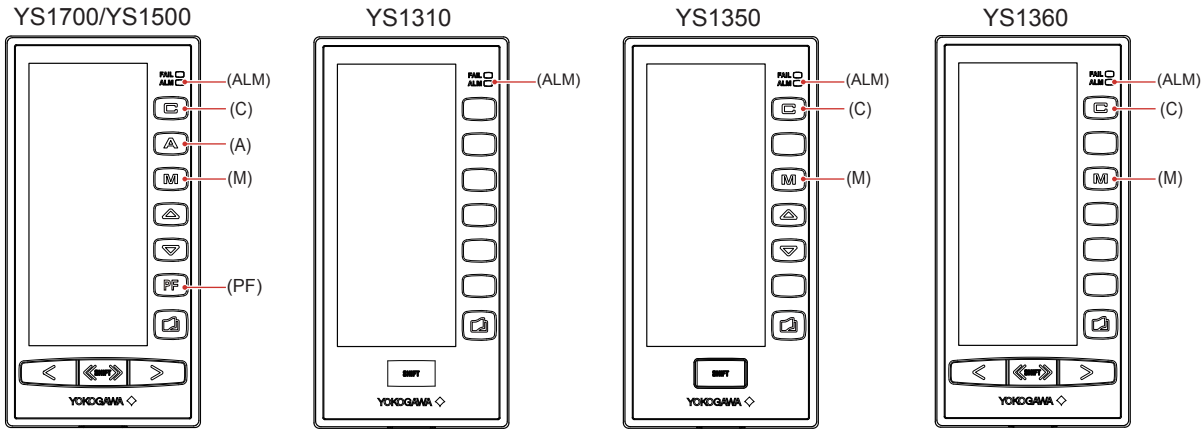
0922E.ai

Operation

1. [Operation] Click the **ALM** button of the Check Operation Guide window.
2. [Confirmation] Confirm only the ALM lamp of YS1000 is lit.
3. [Operation] Use the **OK** button (when lit) or **NG** button (when off) to make a judgment of the check result of the ALM lamp.
   "OK" or "NG" is displayed in the Result field in the Check Result List.
4. Click the **NEXT** button to check the next LED lamp.

Description

The following shows the relationship between the model names and LED lamps to be checked.

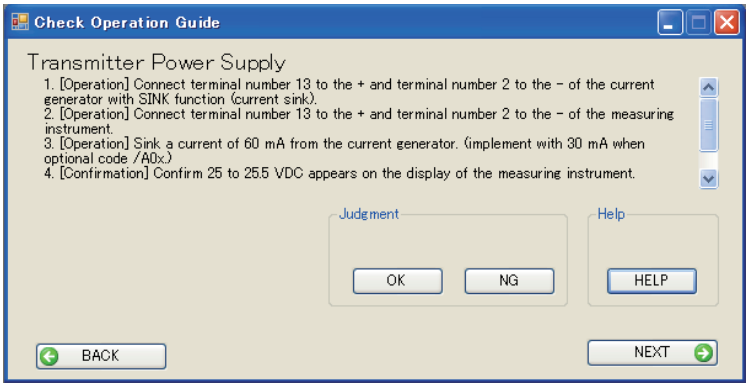


0923E.ai

(5) Transmitter power supply check

Perform the wiring and check in accordance with the **Description**.

Check Operation Guide window



0924E.ai

Operation

1. [Operation] Connect terminal number 13 to the + and terminal number 2 to the - of the current generator with SINK function (current sink).

2. [Operation] Connect terminal number 13 to the + and terminal number 2 to the - of the measuring instrument.

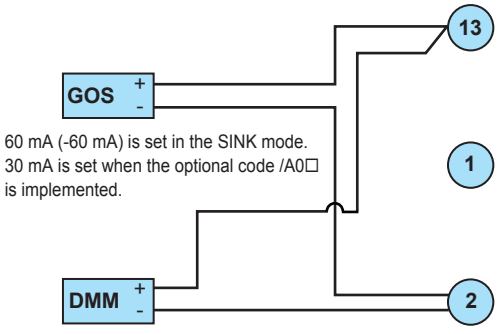
3. [Operation] Sink a current of 60 mA from the current generator. (Implement with 30 mA when optional code /A0☐.)
   Terminal number 13
   + terminal of analog input to which transmitter signal is connected.

4. [Confirmation] Confirm 25 to 25.5 V DC appears on the display of the measuring instrument.

5. [Operation] Use the **OK**/**NG** button to make a judgment of the check results of the transmitter power supply.
   "OK" or "NG" is displayed in the Result field in the Check Result List.

Description

The following shows the wiring diagram of the transmitter power supply.



0925E.ai

Judgment criteria for the transmitter power supply voltage

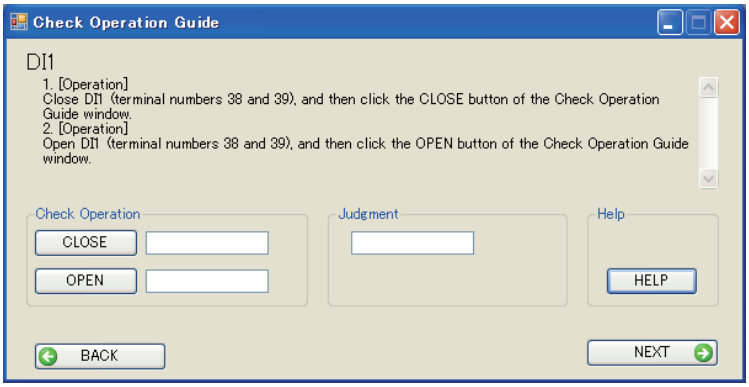
GOS set value	Judgment Criteria (DMM read value (V DC))
SINK mode 60 mA (-60 mA)	25 to 25.5

When the optional code /A0☐ is implemented, perform a check at 30 mA in the SINK mode.

(6) Digital input check

Perform the wiring and check in accordance with the **Description**.

Check Operation Guide window



0926E.ai

Note: Digital input terminals that are not selected on the Check Item Selection window are not checked.

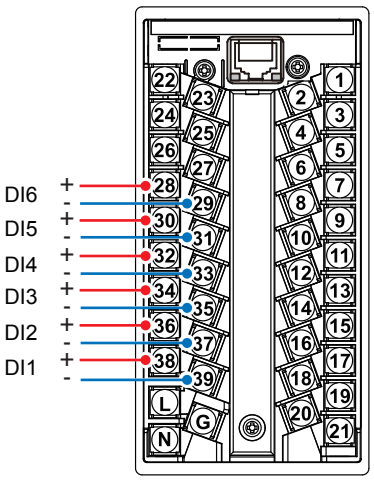
Operation

1. [Operation] Close DI1 (terminal numbers 38 and 39), and then click the **CLOSE** button of the Check Operation Guide window.
   Data of DI1 loaded from the YS1000 is displayed in the text box next to the **CLOSE** button.
2. [Operation] Open DI1 (terminal numbers 38 and 39), and then click the **OPEN** button of the Check Operation Guide window.
   Data of DI1 loaded from the YS1000 is displayed in the text box next to the **OPEN** button.
3. "OK" or "NG" is displayed in the Judgment text box. "OK" or "NG" is displayed in the Result field in the Check Result List at the same time.
4. Click the **NEXT** button to check the next digital input terminals.

Description

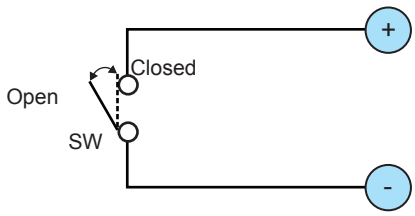
<Digital input check>

The following shows the wiring diagram for checking the digital input.



0927E.ai

Digital input check (DI1 to DI6)



0928E.ai

The following shows the model names and terminals to be checked.

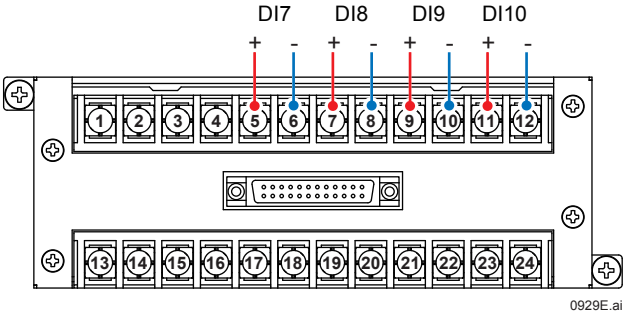
Model	DI1	DI2	DI3	DI4	DI5	DI6
YS1700	✓	✓	✓	✓	✓	✓
YS1500	✓	✓	✓	✓	✓	✓
YS1310	✓					
YS1350	✓	✓				
YS1360	✓	✓				

When DO is selected for the DI/DO selection for the digital I/O terminals, this check does not need to be performed.

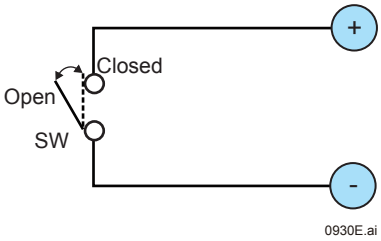
<Expandable I/O digital input check>

This can be performed only for the YS1700-01☐.

The following shows the YS010 (expandable I/O terminal) wiring diagram.



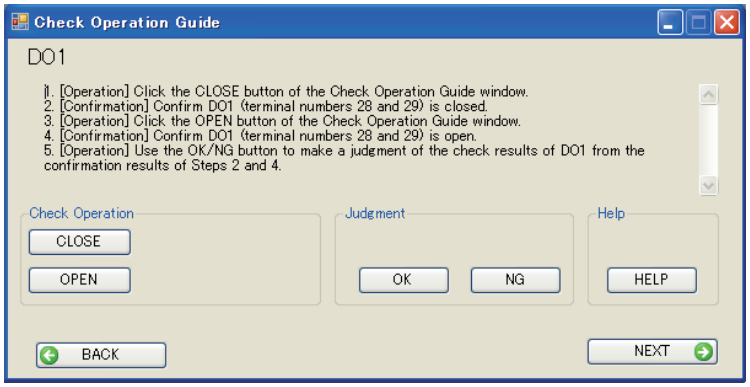
Digital input check (DI7 to DI10)



(7) Digital output check

Perform the wiring and check in accordance with the **Description**.

Check Operation Guide window



0931E.ai

Note: Digital output terminals that are not selected on the Check Item Selection window are not checked.

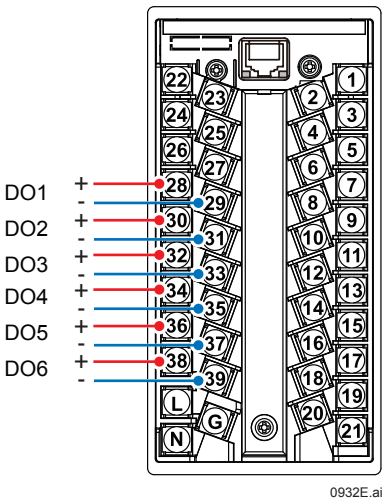
Operation

1. [Operation] Click the **CLOSE** button of the Check Operation Guide window.
2. [Confirmation] Confirm DO1 (terminal numbers 28 and 29) is closed.
3. [Operation] Click the **OPEN** button of the Check Operation Guide window.
4. [Confirmation] Confirm DO1 (terminal numbers 28 and 29) is open.
5. [Operation] Use the **OK/NG** button to make a judgment of the check results of DO1 from the confirmation results of Steps 2 and 4.
   "OK" or "NG" is displayed in the Result field in the Check Result List.
6. Click the **NEXT** button to check the next digital output terminals.

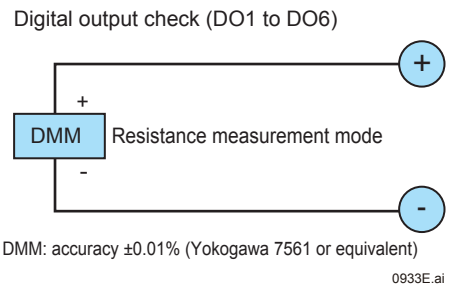
Description

<Digital output check>

The following shows the wiring diagram for checking the digital output.



0932E.ai



DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)

0933E.ai

Digital output judgment criteria

DO state	Judgment Criteria (DMM read value)
Closed	10 Ω or less
Open	Open state

The following shows the model names and terminals to be checked.

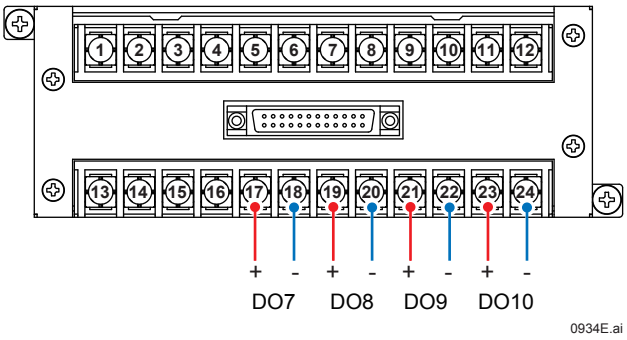
Model	DO1	DO2	DO3	DO4	DO5	DO6
YS1700	✓	✓	✓	✓	✓	✓
YS1500	✓	✓	✓	✓	✓	✓
YS1310	✓	✓	✓	✓	✓	✓
YS1350	✓	✓		✓		
YS1360	✓	✓		✓		

When DI is selected for the DI/DO selection for the digital I/O terminals, this check does not need to be performed.

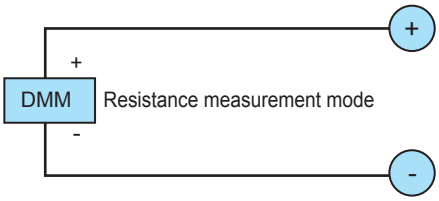
<Expandable I/O digital output check>

This can be performed only for the YS1700-01☐.

The following shows the YS010 (expandable I/O terminal) wiring diagram.



Digital output check (DO7 to DO10)



DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)

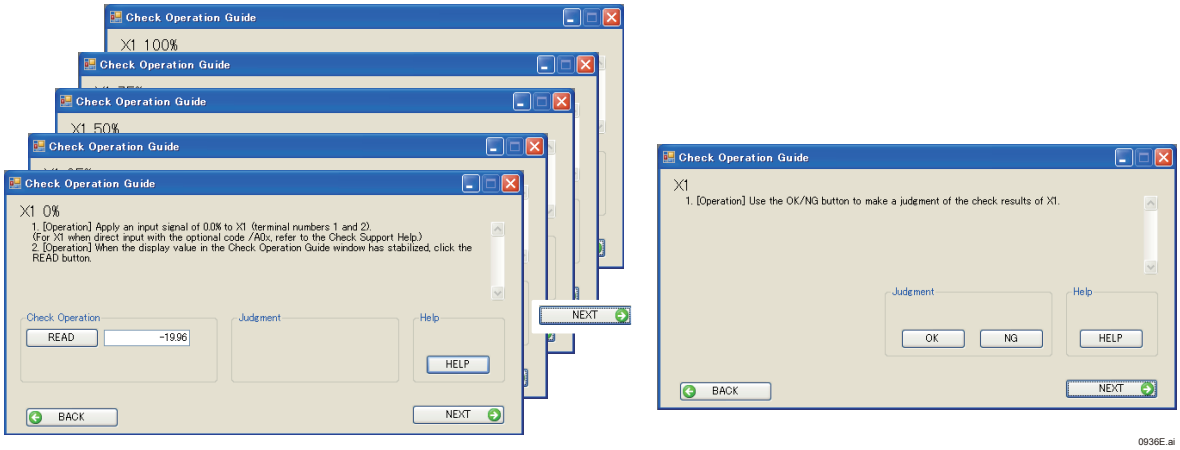
Digital output judgment criteria

DO state	Judgment Criteria (DMM read value)
Closed	10 Ω or less
Open	Open state

(8) Analog input check

Perform the wiring and check in accordance with the **Description**.

Check Operation Guide window



Note: Analog input terminals that are not selected on the Check Item Selection window and 25% and 75% calibration points in the case of selecting a three-point check are not checked.

Operation

1. [Operation] Apply an input signal of 0.0% to X1 (terminal numbers 1 and 2).

(For X1 when direct input with the optional code /A0☐, see the **Description** below or HELP on the Check Operation Guide window.)

2. [Operation] When the display value in the Check Operation Guide window has stabilized, click the **READ** button.

A value is displayed in the 0% field in the Check Result List.

3. Click the **NEXT** button to measure the following calibration points.

5 Point Check: 0%, 25%, 50%, 75%, 100%

3 Point Check: 0%, 50%, 100%

4. [Operation] Use the **OK**/**NG** button to make a judgment of the check results of X1.

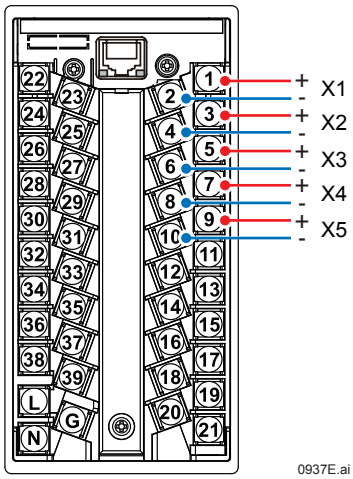
"OK" or "NG" is displayed in the Result field in the Check Result List.

5. Click the **NEXT** button to check the next analog input terminals.

Description

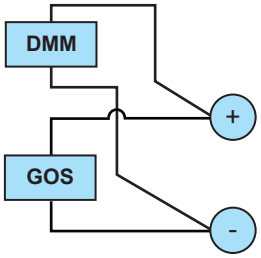
<Analog voltage input check>

The following shows the wiring diagram for checking the analog voltage input.
When the optional code /A0☐ is implemented, the analog input check that requires the connection of direct input shall be performed in accordance with <Direct input check> on page 9-25.



0937E.ai

Voltage input check
(analog input 1 to 5 (X1 to X5))



DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)
GOS: accuracy ±0.01% (Yokogawa 7651 or equivalent)

0938E.ai

The following shows the model names and terminals to be checked.

Model	X1	X2	X3	X4	X5
YS1700	✓	✓	✓	✓	✓
YS1500	✓	✓	✓	✓	
YS1310	✓	✓			
YS1350	✓	✓			
YS1360	✓	✓			

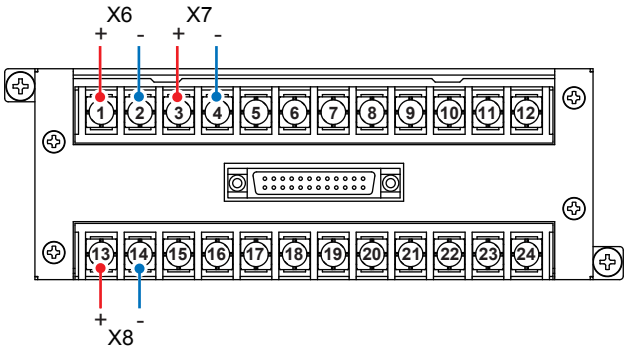
Input voltage and judgment criteria

Calibration point (%)	Input voltage (DMM read value)	Judgment criteria
		Xn (%) value (n = 1 to 5)
0	1.0000 ±0.0010 V DC	0.00 ±0.10
25	2.0000 ±0.0010 V DC	25.00 ±0.10
50	3.0000 ±0.0010 V DC	50.00 ±0.10
75	4.0000 ±0.0010 V DC	75.00 ±0.10
100	5.0000 ±0.0010 V DC	100.00 ±0.10

<Expandable I/O analog voltage input check>

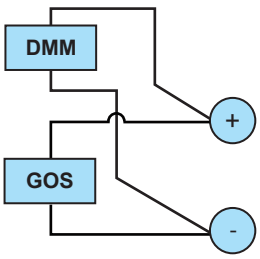
This can be performed only for the YS1700-01☐.

The following shows the YS010 (expandable I/O terminal) wiring diagram.



0939E.ai

Voltage input check (analog input 6 to 8 (X6 to X8))



DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)
GOS: accuracy ±0.01% (Yokogawa 7651 or equivalent)

0940E.ai

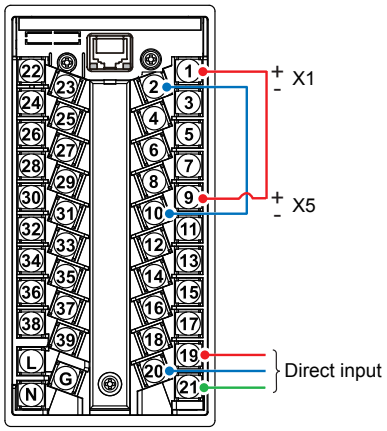
Input voltage and judgment criteria

Calibration point (%)	Input voltage (DMM read value)	Judgment criteria
		Xn (%) value (n = 6 to 8)
0	1.0000 ±0.0010 V DC	0.00 ±0.10
25	2.0000 ±0.0010 V DC	25.00 ±0.10
50	3.0000 ±0.0010 V DC	50.00 ±0.10
75	4.0000 ±0.0010 V DC	75.00 ±0.10
100	5.0000 ±0.0010 V DC	100.00 ±0.10

<Direct input check>

When wiring the direct inputs for the optional codes /A01 to /A08, perform the wiring in accodance with "Direct Input Terminals" in "Installation and Wiring" of the respective Operations Guides.
When using X5 in conjunction with the direct input in the YS1700 programmable mode, X5 does not need to be wired.
When checking the direct input, the voltage input check for the relevant terminal (X1 terminal in the figure below) shall also be performed.
Simulated input corresponding to the input range of the respective direct inputs shall be performed for the direct input terminals.

Wiring diagram for the case where the X1 input is a direct input.



Input settings and judgment criteria

Calibration point (%)	Input settings						Judgment criteria
	/A01 (mV input)	/A04 (Potentiometer input)	/A05 (Isolator)	/A06 (2-line transmitter input (insulated)	/A07 (2-line transmitter input) (Not insulated)	/A08 (Frequency input)	Xn read value of DMM (%) (n = 1 to 5)
0	0% input value of the range		1V	SINK 4mA (-4mA)		0% input value of the range	0.0 ±0.5
25	25% input value of the range		2V	SINK 8mA (-8mA)		25% input value of the range	25.0 ±0.5
50	50% input value of the range		3V	SINK 12mA (-12mA)		50% input value of the range	50.0 ±0.5
75	75% input value of the range		4V	SINK 16mA (-16mA)		75% input value of the range	75.0 ±0.5
100	100% input value of the range		5V	SINK 20mA (-16mA)		100% input value of the range	100.0 ±0.5

Calibration point (%)	Input settings /A02 (Thermocouple input)	Judgment criteria Xn read value (%) (n = 1 to 5)
0	0% input value of the range (*1)	0.00 ±0.50 (*2)
25	25% input value of the range (*1)	25.00 ±0.50 (*2)
50	50% input value of the range (*1)	50.00 ±0.50 (*2)
75	75% input value of the range (*1)	75.00 ±0.50 (*2)
100	100% input value of the range (*1)	100.00 ±0.50 (*2)

*1: Perform an input after subtracting the thermal electromotive force of the RJC sensor temperature.
*2: ±0.5% or 2 x |Accuracy of the direct input card (*3)|+0.1%
*3: Accuracy of the direct input card = ±0.2% of the span or input conversion±20 μV, whichever is larger

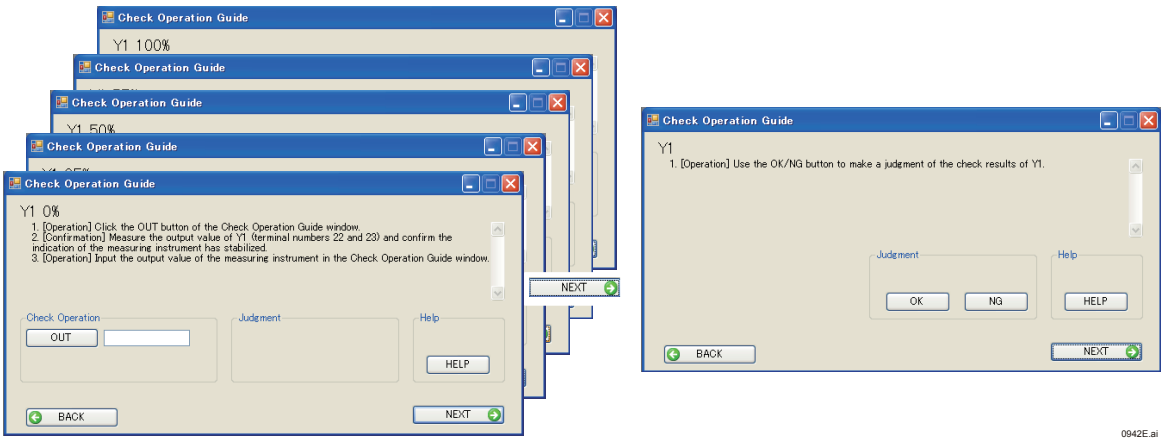
Calibration point (%)	Input settings /A03 (Resistance-temperature detector input)	Judgment criteria Xn read value (%) (n = 1 to 5)
0	0% input value of the range	0.00 ±0.50 (*4)
25	25% input value of the range	25.00 ±0.50 (*4)
50	50% input value of the range	50.00 ±0.50 (*4)
75	75% input value of the range	75.00 ±0.50 (*4)
100	100% input value of the range	100.00 ±0.50 (*4)

*4: ±0.5% or 2 x |Accuracy of the direct input card (*5)|+0.1%
*5: Accuracy of the direct input card = ±0.2% or ±0.2°C of the span, whichever is larger

(9) Analog output check

Perform the wiring and check in accordance with the **Description**.

Check Operation Guide window



Note: Analog output terminals that are not selected on the Check Item Selection window and 25% and 75% calibration points in the case of selecting a three-point check are not checked.

Operation

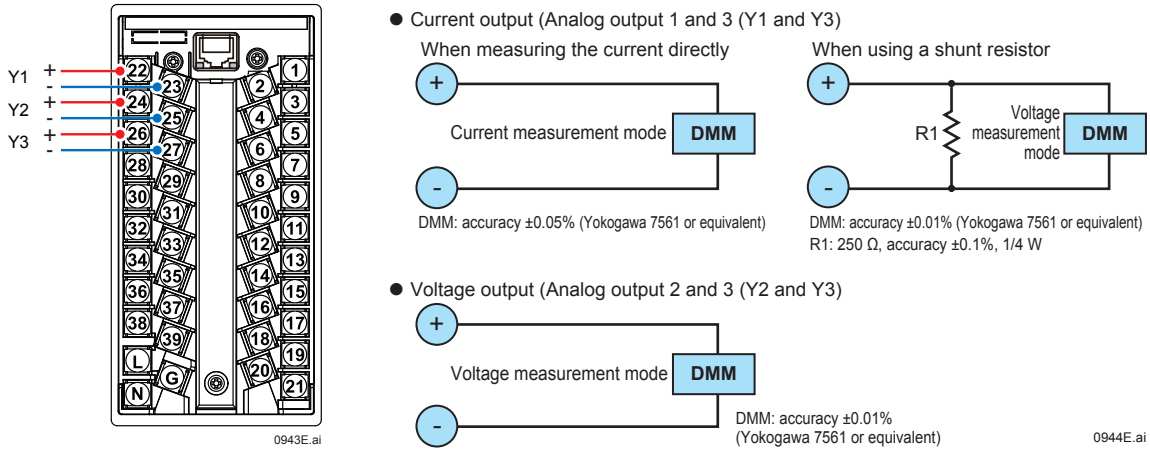
1. [Operation] Click the **OUT** button of the Check Operation Guide window.
2. [Confirmation] Measure the output value of Y1 (terminal numbers 22 and 23) and confirm the indication of the measuring instrument has stabilized.
3. [Operation] Input the output value of the measuring instrument in the text box of the Check Operation Guide window.
4. Click the **NEXT** button. (At this point, the output value that was input in Step 3 and percentage value that was measured based on the output value are displayed in the Check Result List.)

   Measure the following calibration points.
   5 Point Check: 0%, 25%, 50%, 75%, 100%
   3 Point Check: 0%, 50%, 100%
5. [Operation] Use the **OK**/**NG** button to make a judgment of the check results of Y1. "OK" or "NG" is displayed in the Result field in the Check Result List.
6. Click the **NEXT** button to check the next analog output terminals.

Description

<Analog (current/voltage) output check>

The following shows the wiring diagram for checking the analog (current/voltage) output.



The following shows the model names and terminals to be checked.

Model	Y1	Y2	Y3
YS1700	✓	✓	✓(*1)
YS1500	✓	✓	✓
YS1310			
YS1350		✓	
YS1360	✓	✓	

*1: Y3 of the YS1700 in the programmable mode shall be checked in accordance with Analog output 3 current/voltage switching parameter (Y3TP) setting (4 to 20 mA or 1 to 5 V).

Current output and judgment criteria (when measuring the current with DMM)

Calibration point (%)	Current output (mA DC)	Judgment criteria
		DMM read value (mA DC)
0	4.000	4.000 ±0.032
25	8.000	8.000 ±0.032
50	12.000	12.000 ±0.032
75	16.000	16.000 ±0.032
100	20.000	20.000 ±0.032

Current output and judgment criteria (when measuring the current using a shunt resistor)
I = V/R shall be used for the conversion to the current value.
The judgment criteria shall include the shunt resistance accuracy (±0.1% = 0.016 mA DC).

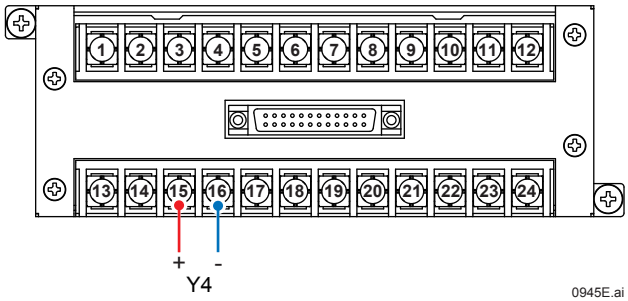
Calibration point (%)	Current output (mA DC)	Judgment criteria
		DMM read value (mA DC)
0	4.000	4.000 ±0.048
25	8.000	8.000 ±0.048
50	12.000	12.000 ±0.048
75	16.000	16.000 ±0.048
100	20.000	20.000 ±0.048

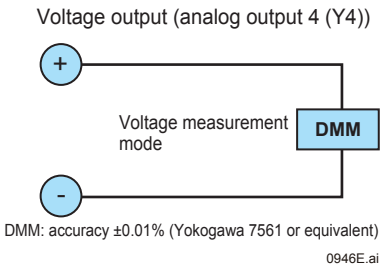
Voltage output and judgment criteria

Calibration point (%)	Voltage output (V DC)	Judgment criteria
		DMM read value (V DC)
0	1.0000	1.0000 ±0.0040
25	2.0000	2.0000 ±0.0040
50	3.0000	3.0000 ±0.0040
75	4.0000	4.0000 ±0.0040
100	5.0000	5.0000 ±0.0040

<Expandable I/O analog voltage output check>

This can be performed only for the YS1700-01☐.
The following shows the YS010 (expandable I/O terminal) wiring diagram.





Voltage output and judgment criteria

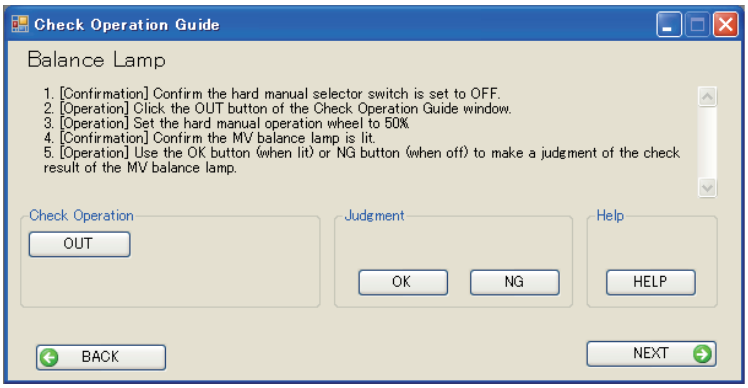
Calibration point (%)	Voltage output (V DC)	Judgment criteria
		DMM read value (V DC)
0	1.0000	1.0000 ±0.0080
25	2.0000	2.0000 ±0.0080
50	3.0000	3.0000 ±0.0080
75	4.0000	4.0000 ±0.0080
100	5.0000	5.0000 ±0.0080

(10) Hard manual (H MAN) check

A hard manual (H MAN) check includes the MV balance lamp check and volume output check.

<Balance lamp check>

Check Operation Guide window



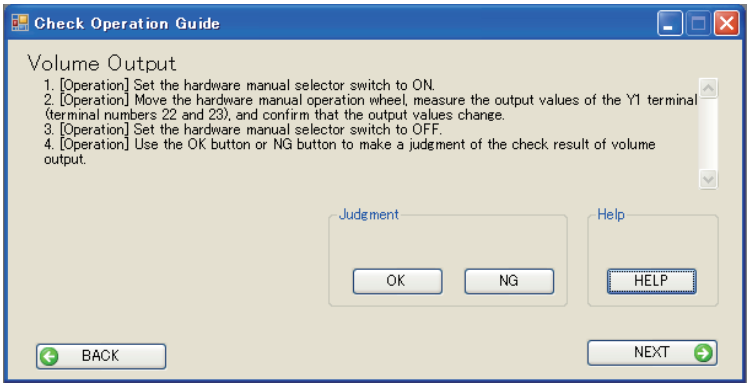
0947E.ai

Operation

1. [Confirmation] Confirm the hard manual selector switch is set to OFF.
2. [Operation] Click the **OUT** button of the Check Operation Guide window.
3. [Operation] Set the hard manual operation wheel to 50%.
4. [Confirmation] Confirm the MV balance lamp is lit.
5. [Operation] Use the **OK** button (when lit) or **NG** button (when off) to make a judgment of the check result of the MV balance lamp.
   "OK" or "NG" is displayed in the Result field in the Check Result List.
6. Click the **NEXT** button to check the volume output.

<Volume output check>

Check Operation Guide window



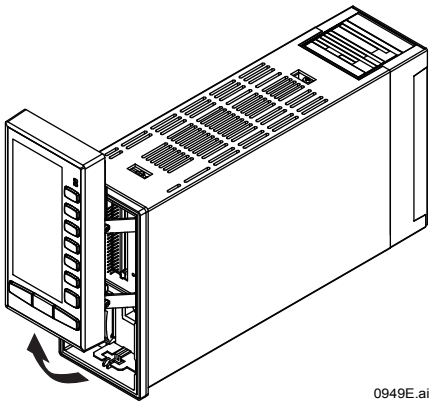
0948E.ai

Operation

1. [Operation] Set the hard manual selector switch to ON.
2. [Operation] Move the hard manual operation wheel, measure the output values of the Y1 terminal (terminal numbers 22 and 23), and confirm that the output values change.
3. [Operation] Set the hard manual selector switch to OFF.
4. [Operation] Use the **OK**/**NG** button to make a judgment of the check result of volume output.
   "OK" or "NG" is displayed in the Result field in the Check Result List.

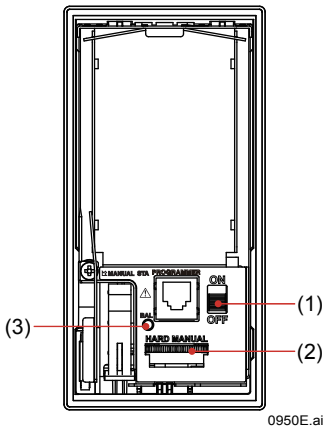
Description

Swing up the front panel as shown in the figure below.



0949E.ai

The following shows the operation parts subject to the hard manual check.



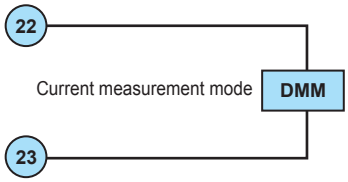
0950E.ai

(1) Hard manual selector switch
ON: Hard manual output on
OFF: Hard manual output off
(2) Hard manual operation wheel
(3) MV balance lamp: Lights up when the Y1 output value and hard manual output value match.

Connect the Y1 terminal in the following way to check the hard manual operation wheel function.

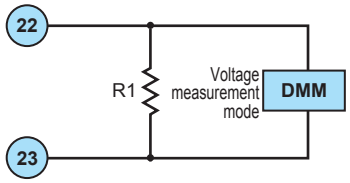
● Current output (analog output 1 (Y1))

When measuring the current directly



DMM: accuracy ±0.05% (Yokogawa 7561 or equivalent)

When using a shunt resistor



DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)
R1: 250 Ω, accuracy ±0.1%, 1/4 W

0951E.ai

Position of the hard manual operation wheel and Y1 output

Hard manual operation wheel	Y1 output	Judgment criteria
		DMM read value
0%	Approx. 4 mA DC	Approx. 4 mA DC
50%	Approx. 12 mA DC	Approx. 12 mA DC
100%	Approx. 20 mA DC	Approx. 20 mA DC

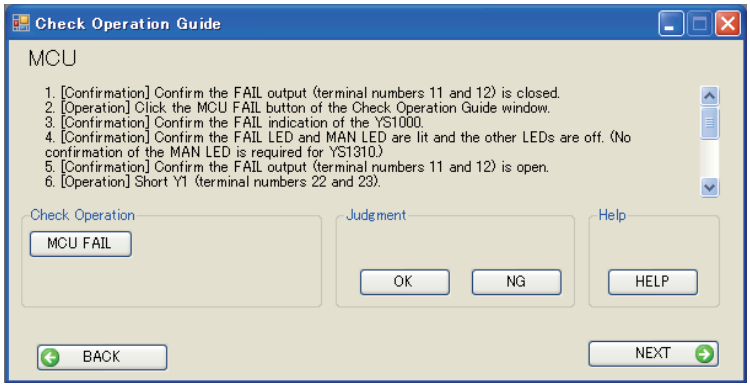
When using a shunt resistor, perform conversion to the current value using I=V/R.

(11) FAIL check

A FAIL check includes the MCUFAIL check and DCUFAIL check.

<MCUFAIL check>

Check Operation Guide window



0952E.ai

Operation

1. [Confirmation] Confirm the FAIL output (terminal numbers 11 and 12) is closed.

2. [Operation] Click the **MCUFAIL** button of the Check Operation Guide window.

3. [Confirmation] Confirm the FAIL indication of the YS1000.

4. [Confirmation] Confirm the FAIL LED and MAN LED are lit and the other LEDs are off. (No confirmation of the MAN LED is required for YS1310.)

5. [Confirmation] Confirm the FAIL output (terminal numbers 11 and 12) is open.

6. [Operation] Short Y1 (terminal numbers 22 and 23).

7. [Confirmation] Press the MV increase and decrease keys and confirm the MV bar indication increases and decreases. (No confirmation of the MV increase/decrease key operation is required for YS1310 and YS1350.)

8. [Operation] Turn off the power of YS1000, and then turn the power back on after 10 seconds has elapsed.

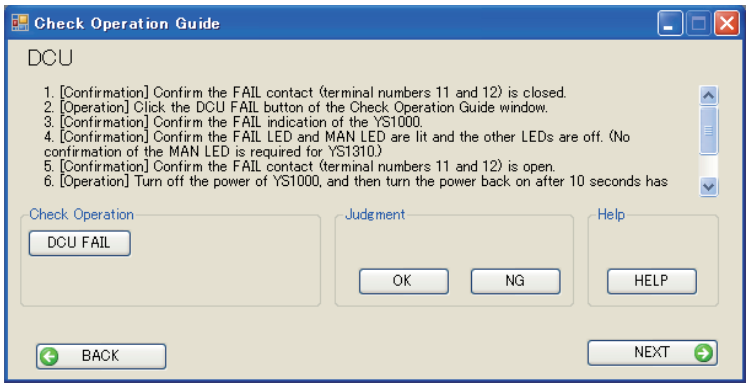
9. [Operation] Open Y1 (terminal numbers 22 and 23).

10. [Operation] Use the **OK**/**NG** button to make a judgment of the check results of MCU from the confirmation results of Steps 1, 3, 4, 5, and 7.
"OK" or "NG" is displayed in the Result field in the Check Result List.

11. Click the **NEXT** button to check the DCUFAIL.

<DCUFAIL check>

Check Operation Guide window



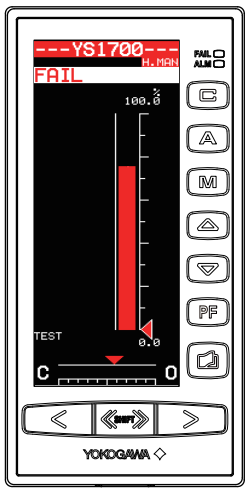
0953E.ai

Operation

1. [Confirmation] Confirm the FAIL contact (terminal numbers 11 and 12) is closed.
2. [Operation] Click the **DCUFAIL** button of the Check Operation Guide window.
3. [Confirmation] Confirm the FAIL indication of the YS1000.
4. [Confirmation] Confirm the FAIL LED and MAN LED are lit and the other LEDs are off. (No confirmation of the MAN LED is required for YS1310.)
5. [Confirmation] Confirm the FAIL contact (terminal numbers 11 and 12) is open.
6. [Operation] Turn off the power of YS1000, and then turn the power back on after 10 seconds has elapsed.
7. [Operation] Based on the results in Steps 1, 3, 4, and 5, judge the DCU check result with the **OK/NG** button.
   "OK" or "NG" is displayed in the Result field in the Check Result List.

Description

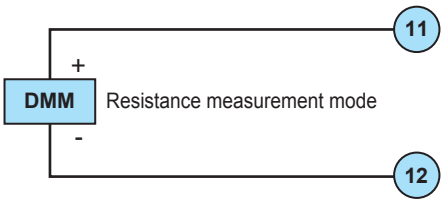
The following shows a FAIL screen example.



0954E.ai

The following shows the way to measure the FAIL output.

FAIL output check



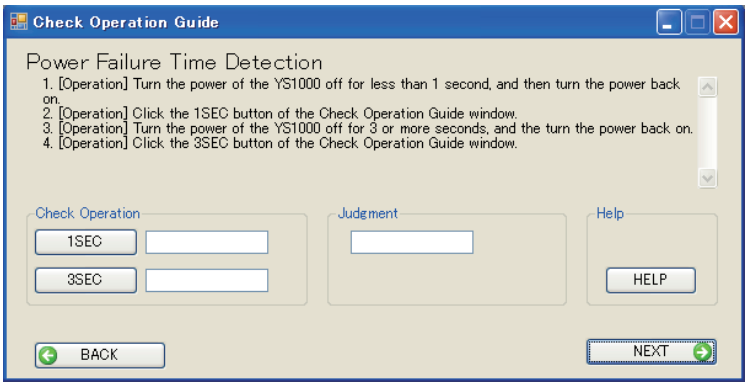
DMM: accuracy ±0.01% (Yokogawa 7561 or equivalent)

FAIL output state	Judgment criteria (DMM read value)
FAIL	Open state
Normal	10 Ω or less

0955E.ai

(12) Power failure time detection check

Check Operation Guide window



0956E.ai

Operation

1. [Operation] Turn the power of the YS1000 off for less than 1 second, and then turn the power back on.

2. [Operation] Click the **1SEC** button of the Check Operation Guide window.

   "OK" or "NG" is displayed in the text box beside the **1SEC** button of the Check Operation Guide window.

3. [Operation] Turn the power of the YS1000 off for 3 or more seconds, and then turn the power back on.

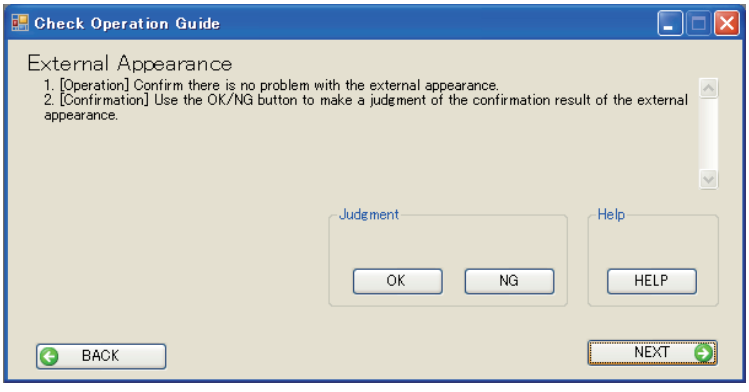
4. [Operation] Click the **3SEC** button of the Check Operation Guide window.

   "OK" or "NG" is displayed in the text box beside the **3SEC** button of the Check Operation Guide window.

   If both judgment results for the less-than-1-second power failure and the 3-second-or-more power failure are OK, "OK" is displayed in the Result field in the Check Result List.

(13) External appearance check

Check Operation Guide window



0957E.ai

Operation

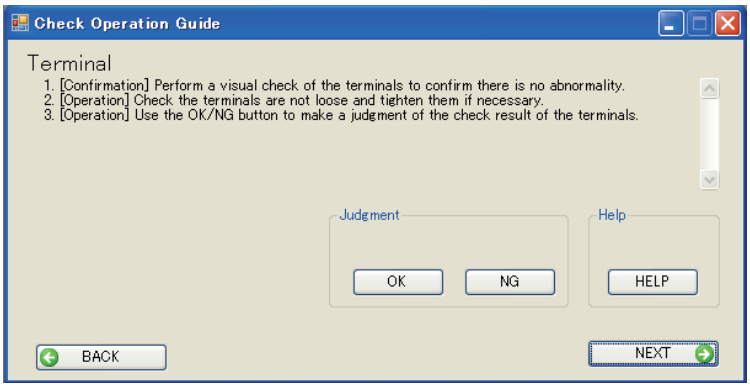
1. [Operation] Confirm there is no problem with the external appearance.

2. [Confirmation] Use the **OK**/**NG** button to make a judgment of the confirmation result of the external appearance.

   "OK" or "NG" is displayed in the Result field in the Check Result List.

(14) Terminal check

Check Operation Guide window



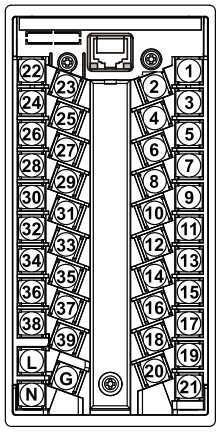
0958E.ai

Operation

1. [Confirmation] Perform a visual check of the terminals to confirm there is no abnormality.
2. [Operation] Check the terminals are not loose and tighten them if necessary.
3. [Operation] Use the **OK**/**NG** button to make a judgment of the check result of the terminals.
   "OK" or "NG" is displayed in the Result field in the Check Result List.

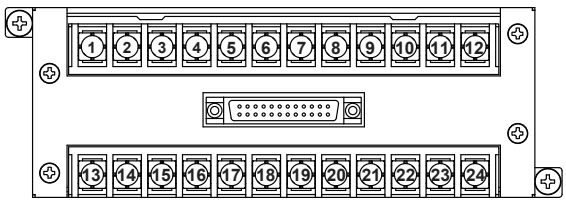
Description

After installation and wiring, make sure that the wires are connected correctly in accordance with "Installation and Wiring" items of the respective Operations Guides. Perform tasks such as retightening as needed.
The following shows the rear panel terminal diagram.



0959E.ai

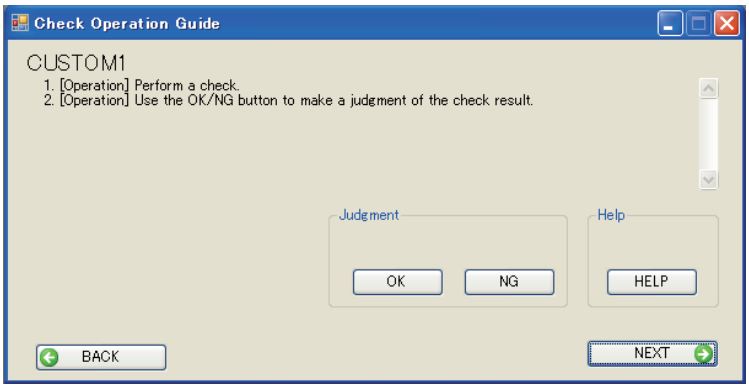
In the case of the YS010 (expandable I/O terminal), the check can be performed only for the YS1700-01☐.
The following shows the YS010 (expandable I/O terminal) wiring diagram.



0960.ai

(15) Custom check

Check Operation Guide window



0961E.ai

Operation

1. [Operation] Perform a check.

2. [Operation] Use the **OK**/**NG** button to make a judgment of the check result.
"OK" or "NG" is displayed in the Result field in the Check Result List.

Description

The custom check function enables the user to register items other than existing check items as records.
Up to 10 check items can be added.
To add an item, add a checkmark to the checkbox for the desired item on the Check Item Selection window.

To change the name of a check item, select CUSTOMn* in the text box and enter a check item name.
Up to 10 single-byte characters can be entered.
*: n = 1 to 10

Note

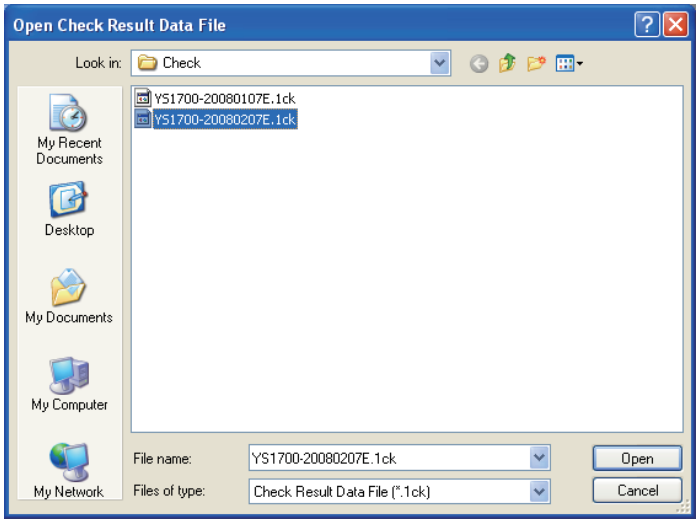
If the check item name of a custom check that has been used is changed, the check history data of that changed custom check is deleted during downloading from the past check history data saved under the check item name before the change.

9.2.6 Opening Data File/Saving

(1) Open check result data files

Operation

1. Click **File**>**Open Check Result Data File** from the menu or  on the toolber or select **Open Check Result Data File** in the Start Check Support Function window, and click **OK** to display the Open Check Result Data File window.



0962E.ai

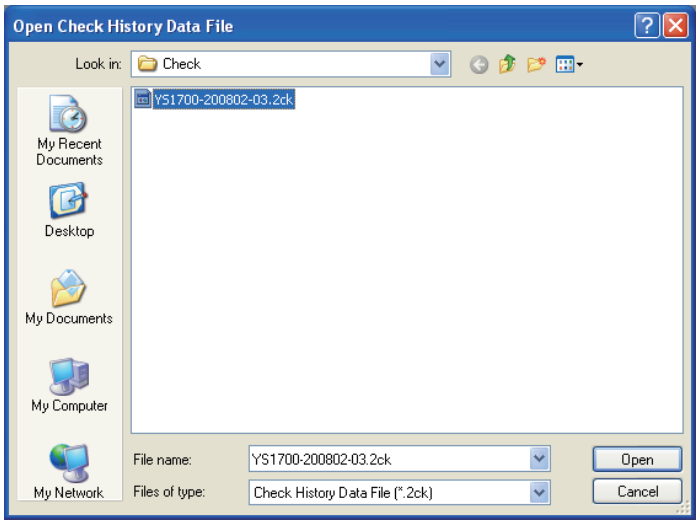
Note

- The file type is ".1ck."
- The Temperature, Humidity, and Memo data stored in the check result data file can be edited.
- When you want to open the check result data file, end the check. (The three windows Check Item Selection, Check Operation Guide, and Check Result List close.)
  After the check finishes, turn the power of the YS1000 off and on to switch to the normal mode.

(2) Open check history data files

Operation

1. Click **File**>**Open Check History Data File** from the menu or select **Open Check History Data File** in the Start Check Support Function window, and click **OK** to display the Open Check History Data File window.



0963E.ai

Note

- The file type is ".2ck."
- When you want to open the check history data file, end the check. (The three windows Check Item Selection, Check Operation Guide, and Check Result List close.)
  After the check finishes, turn the power of the YS1000 off and on to switch to the normal mode.

(3) Closing files

Operation

1. Click **File**>**Close** from the menu to close the currently active window. To save a file you are working on, use Save As.

(4) Overwriting files

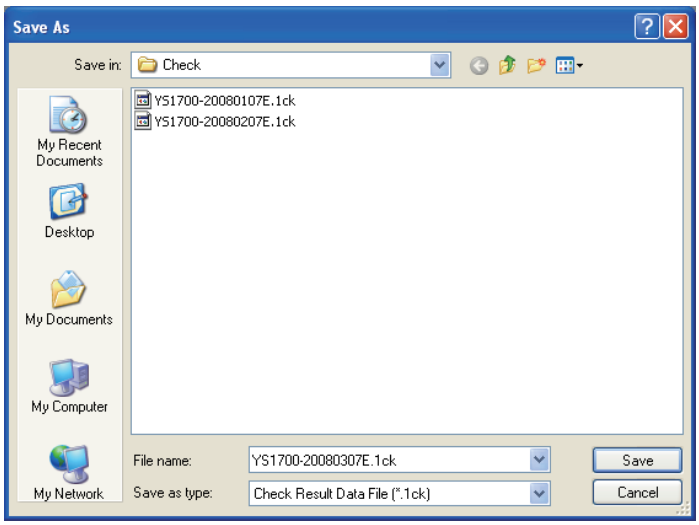
Operation

1. When checking or finishing a check, Click **File**>**Save** from the menu or  on the toolbar. The data you are currently working on is saved.

(5) Saving files under a different name

Operation

1. When checking or finishing a check, Click **File**>**Save As** from the menu to display the Save As window.



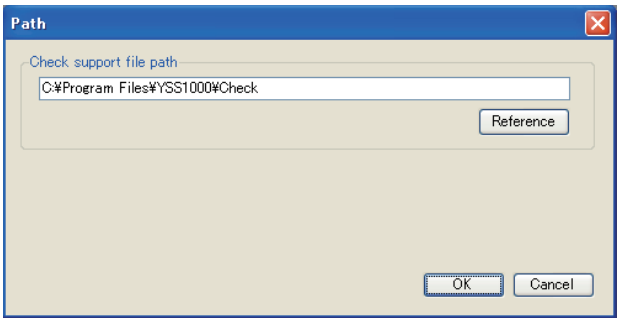
0964E.ai

2. Enter the new name for the file, and click **Save**.

(6) Setting path fot saving files

Operation

1. Make sure that only the Start Check Support Function window is displayed. Click **File**>**Environmental Setting**>**Path** from the menu to display the Path window.



0965E.ai

2. Set the path for check result data files or check history data files, and click **OK**.

Description

The paths for check result data files and check history data files are set as shown in the table below.
(Factory Defaults)

File	Extension	Storage Folder	
		Widows 2000/XP	Windows Vista Business
Check result data file	.1ck	C:\Program Files\YSS1000\Check\	C:\Users\<Username>\Documents\YSS1000\Check\
Check history data file	.2ck	C:\Program Files\YSS1000\Check\	C:\Users\<Username>\Documents\YSS1000\Check\

CAUTION

When using Windows Vista, do not set a path including the Program Files folder. If a path including the Program Files folder is set, the YSS1000 will not work correctly.

(7) Communication logs output function

See "2.11.11 Creating Communication Logs."

9.2.7 Download/Upload/Delete Data

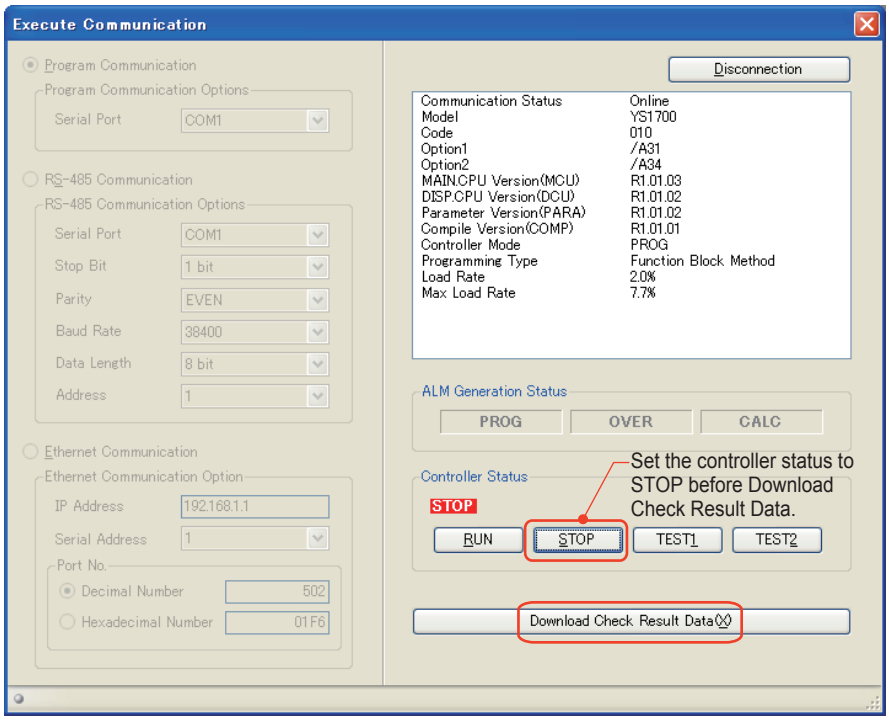
Note

Do not unplug the connection cable or power off the YS1000 during downloading, deleting, or uploading.

(1) Download check result data

Operation

1. Click **Communication>Download Check Result Data** from the menu to display the Execute Communication window.

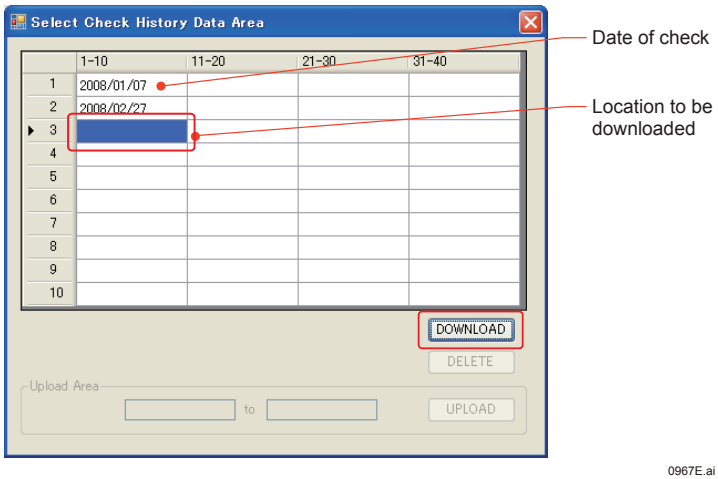


0966E.ai

2. Click **Download Check Result Data** to display the Select Check History Data Area window.

The check result data is downloaded to the location under the data with the latest check date.

If the data with the latest check date is in the No. 40 location, the data is downloaded to the No. 1 location.



0967E.ai

3. Click **DOWNLOAD**.

Description

- Check history data loaded from the YS1000 is displayed with the check date on the Select Check History Data Area window.
- If you try to close the Execute Communication window when a communication error occurs during downloading, the following message is displayed.
  The past check history data was deleted from the YS1000, because the check result data could not be downloaded properly. If you close the Execute Communication window, the check history data will no longer be able to be recovered. The check history data can be recovered if you click the DOWNLOAD button of the check result data to execute the download. Do you want to return to the Execute Communication window?
  If you want to return to the Execute Communication window to download again, click **Yes**.
  Clicking **No** deletes all check history data of the YS1000.

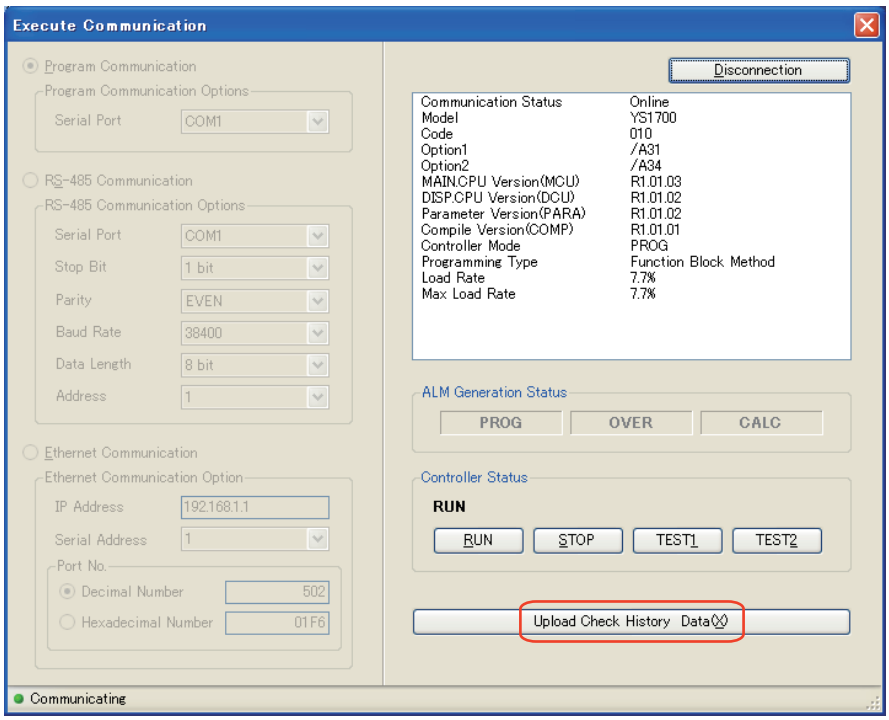
Note

If the check item name of a custom check that has been used is changed, the check history data of that changed custom check is deleted during downloading from the past check history data saved under the check item name before the change.

(2) Upload check history data

Operation

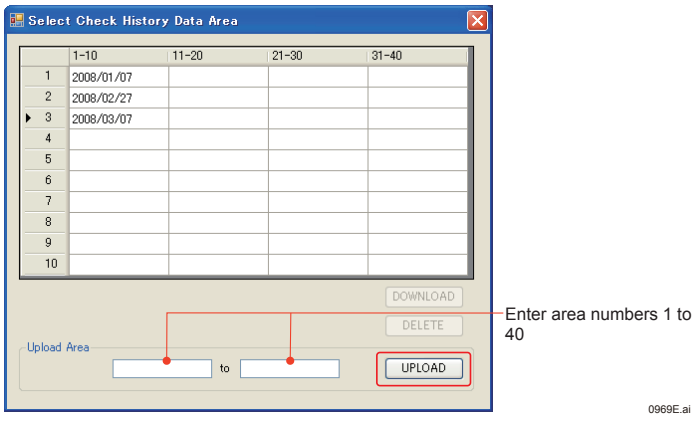
1. Click **Communication>Upload Check History Data** from the menu or select **Upload Check History Data** in the Start Check Support Function window, and click **OK** to display the Execute Communication window.



0968E.ai

2. Click **Upload Check History Data** to display the Select Check History Data Area window.

3. In the Upload Area, enter an area number to be displayed as check history data. Area numbers 1 to 40 can be entered.



0969E.ai

4. Click **UPLOAD**.

   The Check History Data window is displayed.

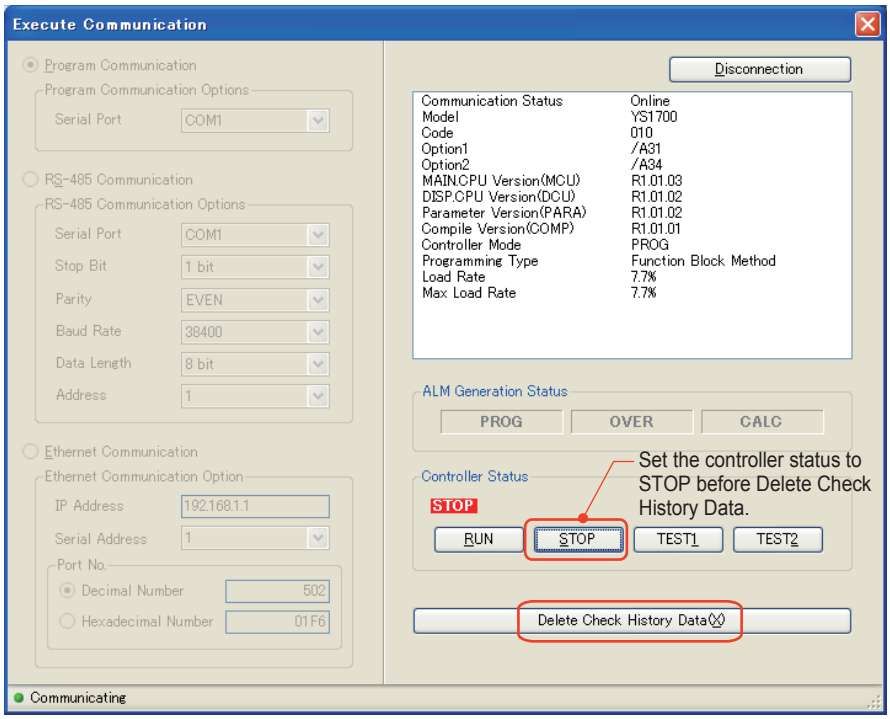
► "(4) Check history data window" on page 9-44.

(3) Delete check history data

The latest check history data can be deleted.

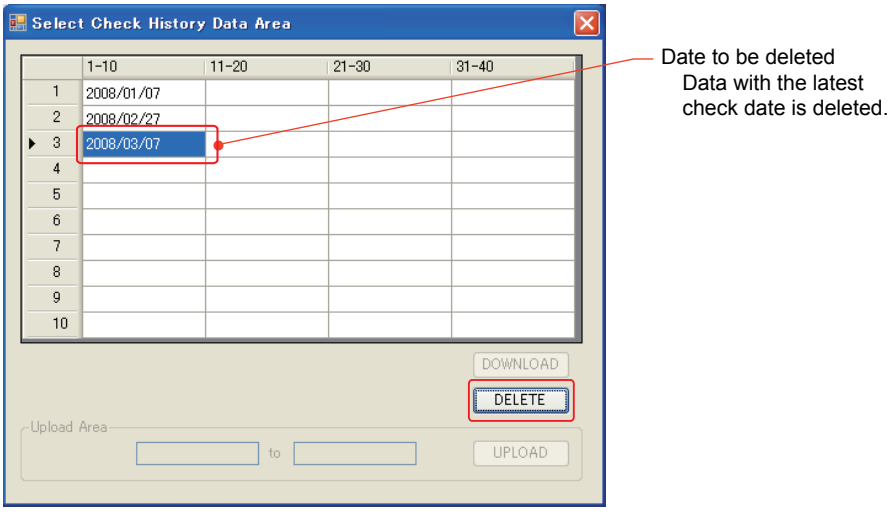
Operation

1. Click **Communication**>**Delete Check History Data** from the menu to display the Excecute Communication window.



0970E.ai

2. Click **Delete Check History Data** to display the Select Check History Data Area window.



0971E.ai

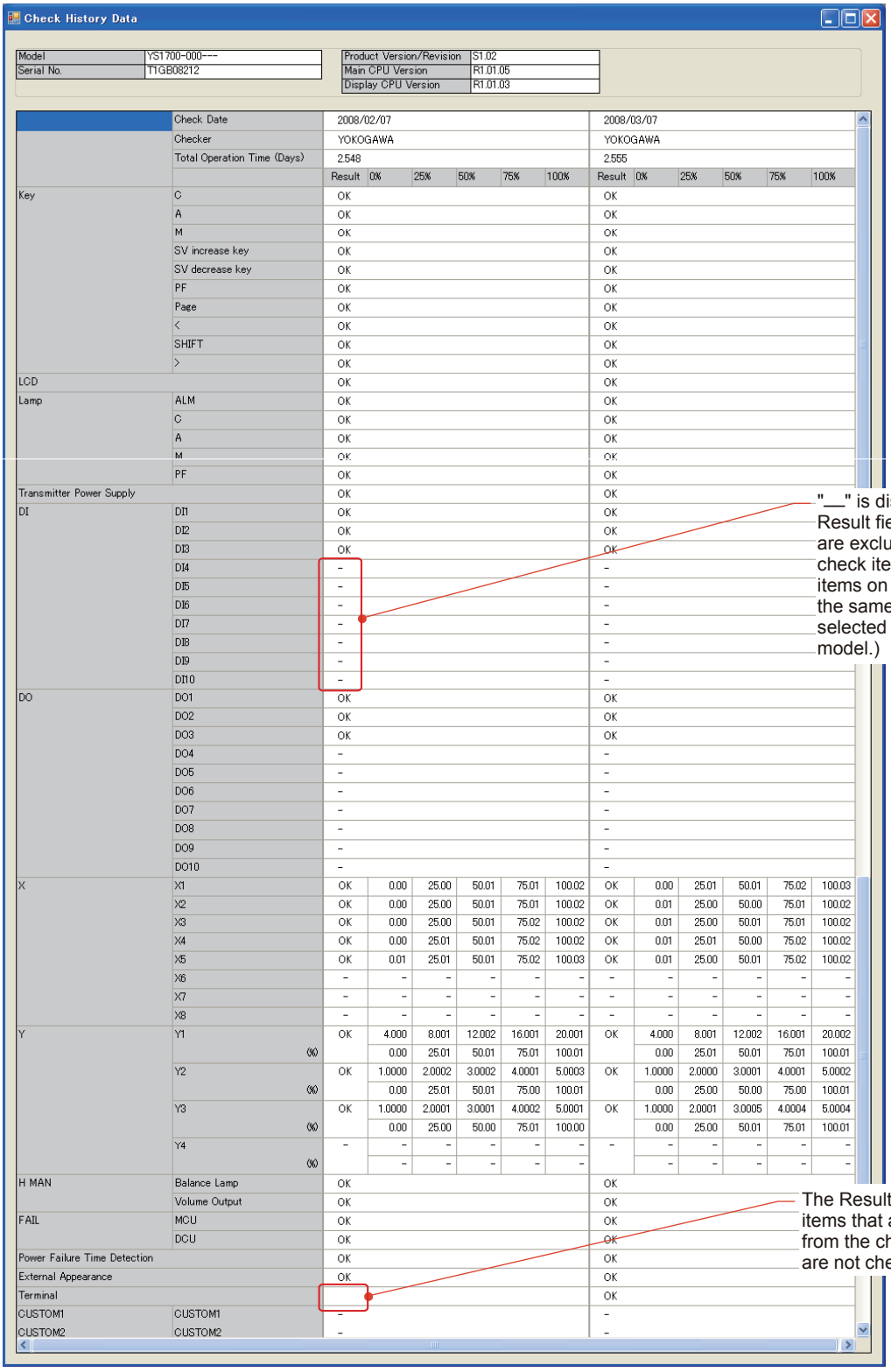
3. Click **DELETE**.

Description

- If you try to close the Execute Communication window when a communication error occurs during deletion, the following message is displayed.
  All check history data was deleted from the YS1000, because the check history data could not be deleted properly. If you close the Execute Communication window, the check history data will no longer be able to be recovered. The check history data can be recovered if you click the DELETE button of the check history data to execute deletion. Do you want to return to the Execute Communication window?
  If you want to return to the Execute Communication window to delete data again, click **Yes**.
  Clicking **No** deletes all check history data of the YS1000.

(4) Check history data window

Up to 40 records of check history data uploaded from the YS1000 can be displayed.
Data can only be displayed and cannot be changed on the Check History Data window.



"—" is displayed in the Result field for items that are excluded from the check items. (The display items on the window are the same regardless of the selected check items and model.)

The Result field is blank for items that are selected from the check items but are not checked.

0972E.ai

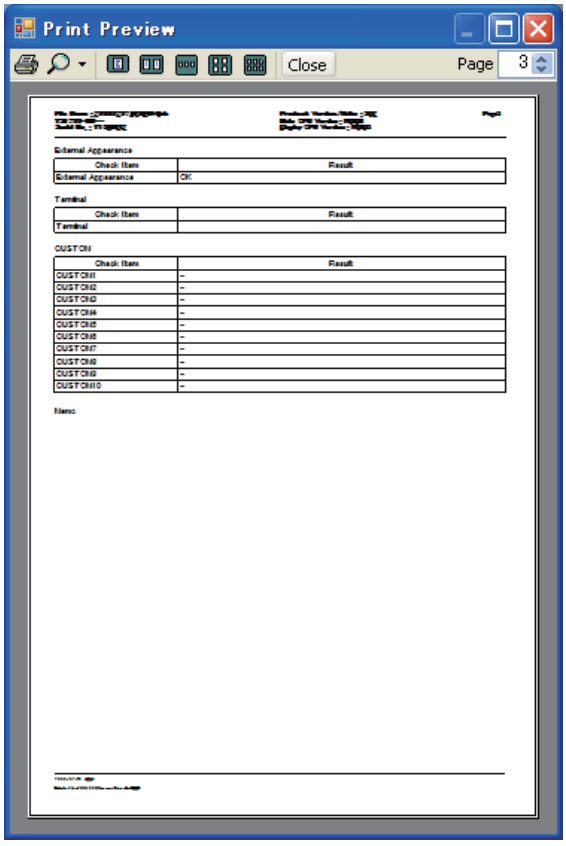
9.2.8 Printing

Operation

1. Check **File**>**Print** from the menu or  on the toolber to display the Print window.
2. Click **OK**.
3. When printing is completed, click  at the top right of the window.

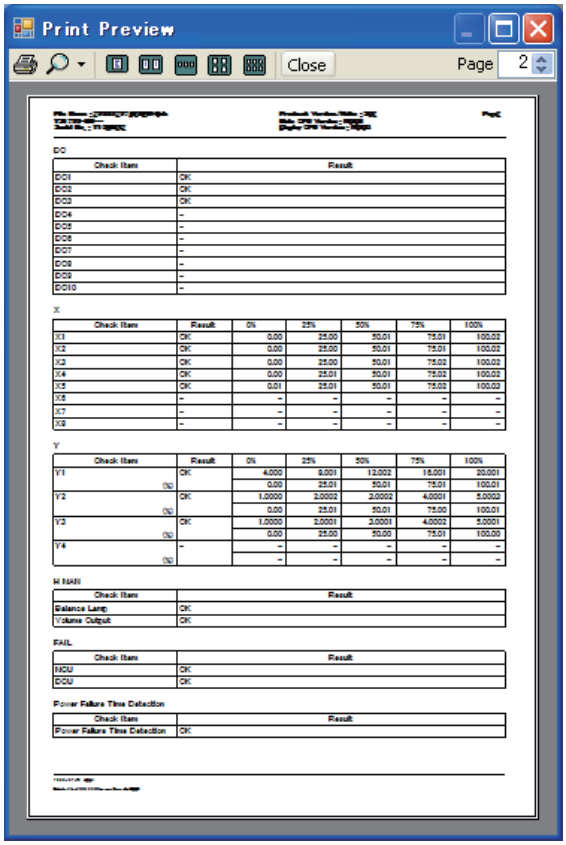
Description

The following are samples of how data is printed out.



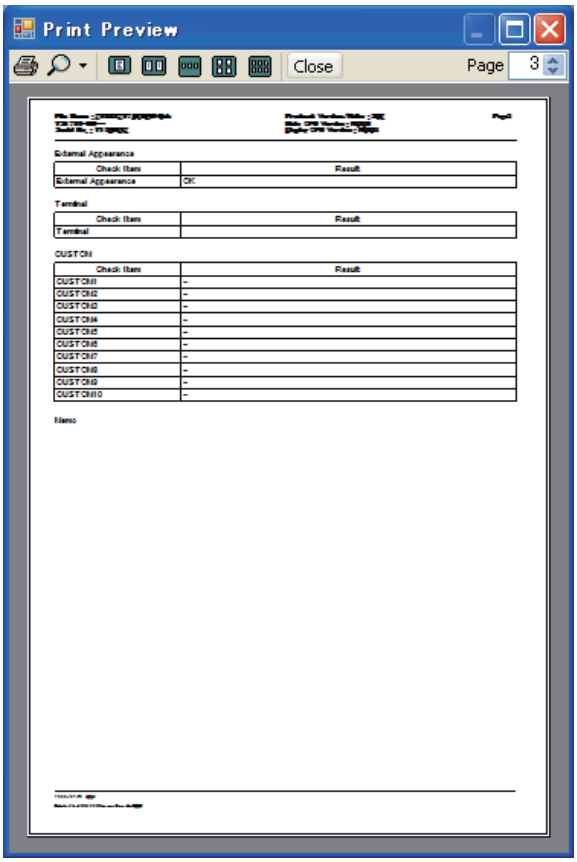
0973-01E.ai

Page 1



0973-02E.ai

Page 2



0973-03E.ai

Page 3

10.1 Sample Program

Design a user program bearing in mind the content of this manual up to Chapter 9 and paying attention to the following items.

(1)Keep the program as simple as possible!
Auxiliary functions can be added on easily later on.
Initially, focus on the main target functions to create a simple and clear program.

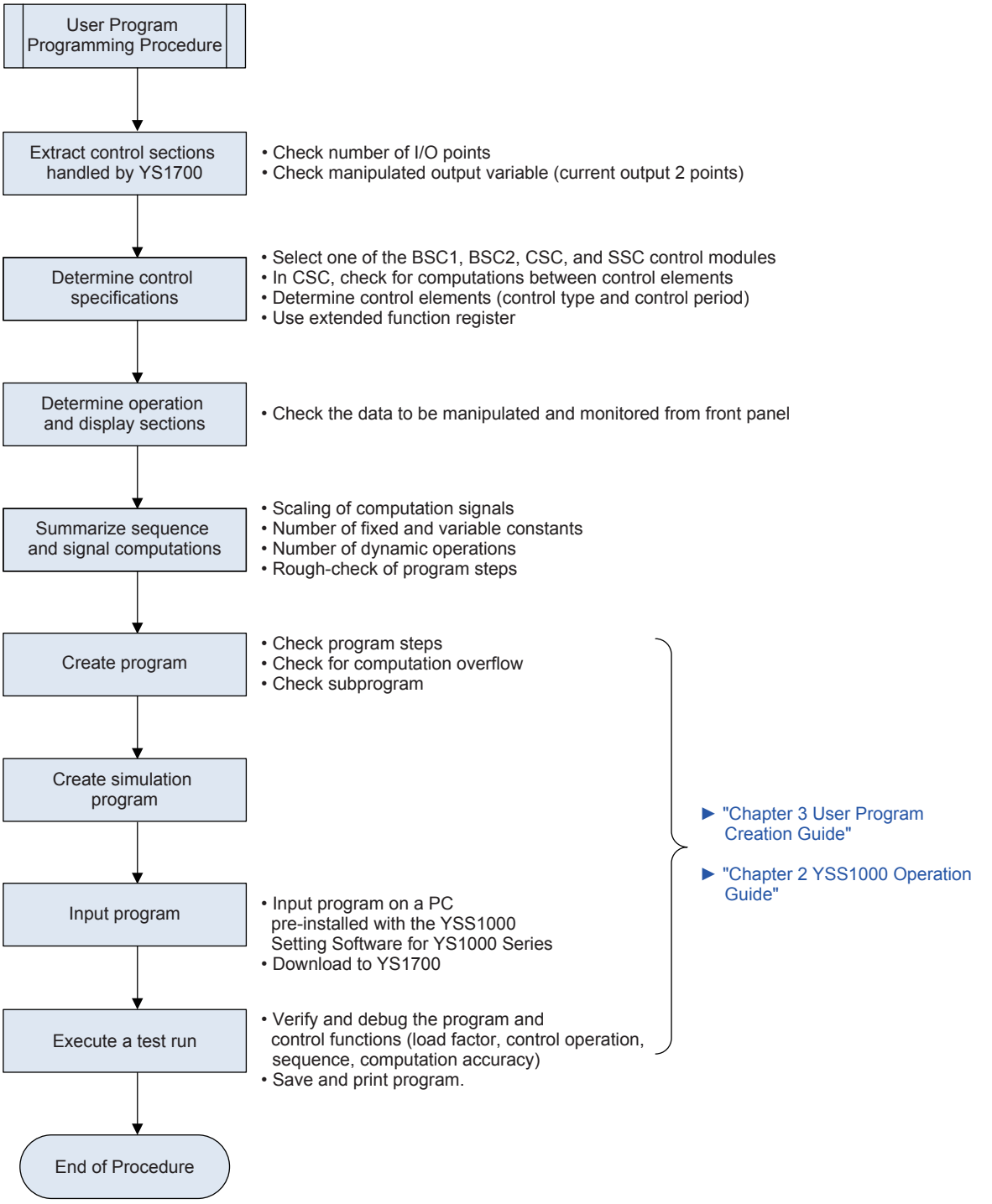
(2)Create a program that anyone can understand!
There is no need to create a well-organized program without wasteful steps. If anything, programs tend to be hard-to-understand.
Create a program that is easily comprehensible to everyone. For this reason, program as many comments as possible. They will come in very handy later on when reviewing the program.

(3)Use as many previous examples and precedents as possible!
This is a shortcut to making a less troublesome and accurate program.
Use the examples in this manual.

(4)Remember that an equal amount of effort is required for high-grade control!
Why not try using sample-and-hold PI control instead of Smith dead time compensation control?
Wouldn't batch PID control be sufficient instead of using complex batch startup control?
Always, work towards simple methods.

(5)Save programs you have made on CD or other storage media, and store them in a safe place. Do not forget to print them!
This is useful for managing the YS1700 later on, and anyone can use them.

10.2 Programming Procedure



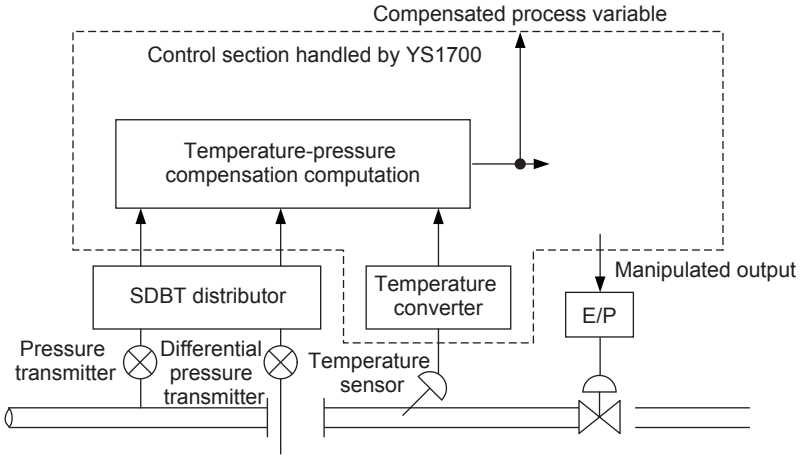
User Program Programming Procedure

10.3 Example (flow control with temperature-pressure compensation of an ideal gas flow)

The following shows temperature-pressure compensation of gas flow as an example of control accompanied by computation of the process variable. This explanation follows the order of the procedure for programming this control shown on the previous page.
The key point here is how to calculate fixed constant values and scaling in arithmetic computations, in particular.

(1)Extraction of control sections handled by YS1700

Flow signals from a differential pressure transmitter are compensated by pressure and temperature as shown in the figure below. The input signals from each sensor are input to the YS1700 as data after having been converted to 1 to 5 V DC by a distributor or converter. The YS1700 computes the data by the three inputs (differential pressure, pressure and temperature), performs control using the compensation values as the process variable signals, and outputs the manipulated output variable as 4 to 20 mA. The compensated process variable is output as 1 to 5 V DC.



Flow Control with Temperature-Pressure Compensation

(2)Determining the control module and control type

This is an example of simple PID control.

- Control module; basic control (BSC1)
- Control type; standard PID (CNT1: PID)

(3)Summarizing arithmetic computations

Temperature-pressure compensation for an ideal gas flow is generally expressed by the following formula:

$$Qx = \sqrt{\frac{Pf \bullet Tn}{Pn \bullet Tf} \bullet \Delta P}$$

1003E.ai

where,

ΔP : Differential pressure
Qx : Gas flow after conversion to a standard state
Pf : Operating pressure of gas indicated as absolute pressure
Pn: Orifice design reference pressure indicated as absolute pressure
Tf : Operating temperature of gas indicated as absolute temperature
Tn: Orifice design reference temperature indicated as absolute temperature

(4)Normalization of computation

The physical quantity of the compensation formula is converted to the normalizing signal (0 to 1) of the YS1700.

Pf =Pspan∗X2+Pmin	X2: pressure signal (0 to 1)
Tf =Tspan∗X5+Tmin	X5: temperature signal (0 to 1)
Qx =Qspan∗Y2	Y2: compensation flow signal (0 to 1)
ΔP =Pspan∗X1	X1: differential pressure signal (0 to 1)

For simplification, these variables are substituted into the formula.

$$Q_{span} \cdot Y2 = \sqrt{\left(\frac{P_{span}}{P_n} \cdot X2 + \frac{P_{min}}{P_n}\right) \Big/ \left(\frac{T_{span}}{T_n} \cdot X5 + \frac{T_{min}}{T_n}\right) \cdot \Delta P_{span} \cdot X1}$$

1004E.ai

The compensation operation formula can be simplified as follows:

$$Y2 = \sqrt{\frac{K2 \cdot X2 + K4}{K3 \cdot X5 + K5} \cdot X1}$$

1005E.ai

Qx andΔP of the orifice are often designed so that normally Y2=√X1 in a design reference status, and scaling of √ΔPspan/Qspan is not required.

Temperature-pressure compensation computation - design conditions

Orifice design reference pressure	:	Pn=0.6MPa
Orifice design reference temperature	:	Tn=300ºC
Pressure transmitter range	:	0 to 1MPa
Temperature converter range	:	0 to 500ºC

The values of Kn are calculated by substituting these conditions.

$$K2 = \frac{\text{Pressure transmitter span}}{\text{Orifice design reference pressure (absolute pressure)}} = \frac{1.0}{0.6+0.1013}$$

$$= 1.426\ (142.6\%)$$

$$K4 = \frac{\text{Minimum scale of pressure transmitter (absolute pressure)}}{\text{Orifice design reference pressure (absolute pressure)}} = \frac{0+0.1013}{0.6+0.1013}$$

$$= 0.144\ (14.4\%)$$

$$K3 = \frac{\text{Temperature converter span}}{\text{Orifice design reference temperature (absolute temperature)}} = \frac{500}{300+273.2}$$

$$= 0.872\ (87.2\%)$$

$$K5 = \frac{\text{Minimum scale of temperature transmitter (absolute temperature)}}{\text{Orifice design reference temperature (absolute temperature)}} = \frac{0+273.2}{300+273.2}$$

$$= 0.477\ (47.7\%)$$

1006E.ai

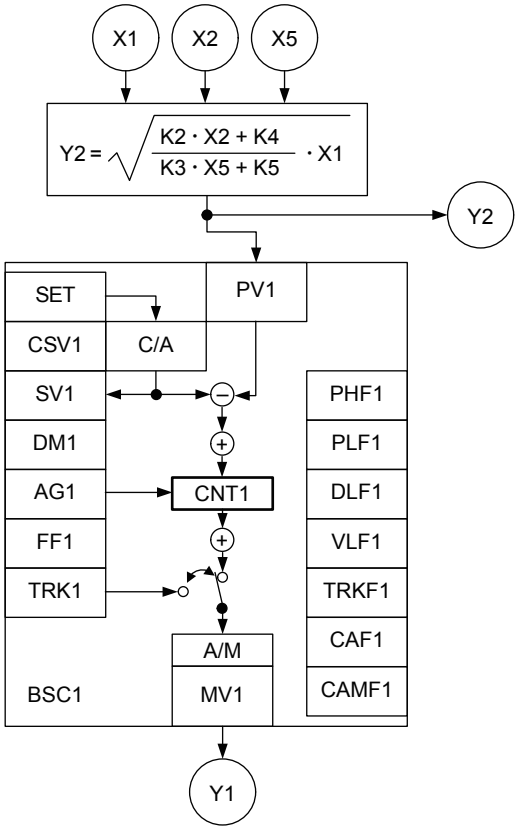
If we substitute these constants into the formula:

$$Y2 = \sqrt{\frac{1.426 \cdot X2 + 0.144}{0.872 \cdot X5 + 0.477}\ X1}$$

1007E.ai

(5)Summarizing function blocks

In this example, a single formula and control module can be summarized as a function block. The figure below shows a flow sheet with I/O terminals attached.



Flow Sheet

(6)Filling in the engineering table

The following shows an example of how to fill in an engineering table with I/O and constant assignments and indications for displaying engineering units on the front panel.

Engineering Table

		Description		0%	100%
Analog inputs	X1	Differential pressure	mmH_2O	0	3200
	X2	Pressure	MPa	0	1.000
	X5	Temperature	ºC	0	500.0
Analog outputs	Y1	Manipulated output variable	%	0	100.0
	Y2	Flow rate×	$10m^3/h$	0	800.0

		Constants	Description
Fixed constants	K2	1.426	
	K3	0.872	
	K4	0.144	
	K5	0.477	

(7)Making a program sheet

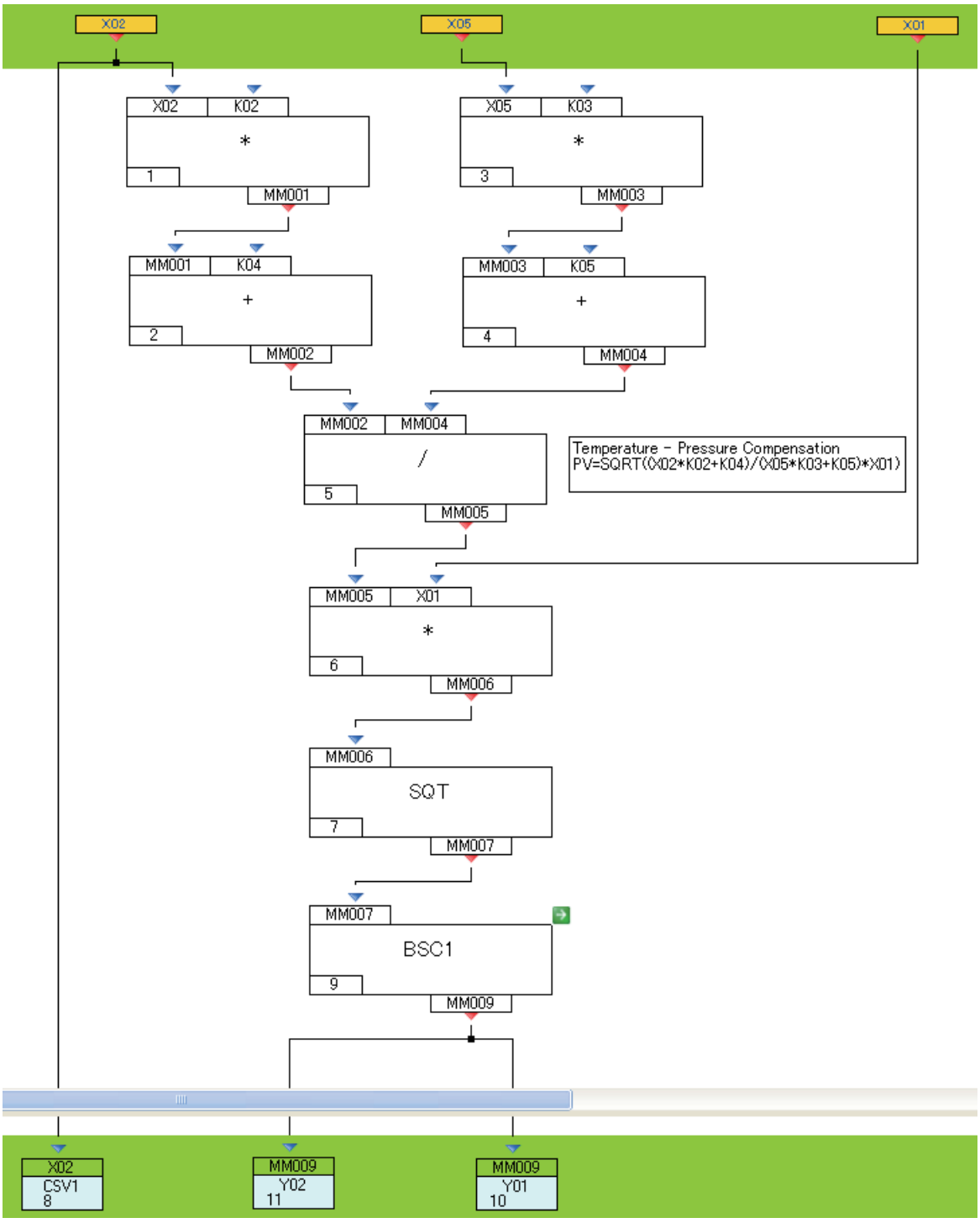
The following shows an example of a program sheet. Enter comments at the "Explanation" column. The computation registers indicate what is input as a result of executing each step.

There is no need to enter the computation register items in the program sheet. An explanation of the program is given following the program sheet.

Step	Program	S1	S2	S3	S4	S5	Explanation
1	LD X2	X2					Reads the pressure signal.
2	LD K2	K2	X2				Reads the constant. K2=1.426
3	*	K2*X2					
4	LD K4	K4	K2*X2				Reads the constant. K4=0.144
5	+	a					Pressure compensation term: a=K2*X2+K4
6	LD X5	X5	a				Reads the temperature number.
7	LD K3	K3	X5	a			Reads the constant. K3=0.872
8	*	K3*X5	a				
9	LD K5	K5	K3*X5	a			Reads the constant. K5=0.477
10	+	b	a				Temperature compensation term: b=K3*X5+K5
11	/	a/b					Computation of pressure/ temperature terms
12	LD X1	X1	a/b				Reads the differential pressure signal.
13	*	C					$C = \frac{K2 \cdot X2 + K4}{K3 \cdot X5 + K5} \cdot X1$ 1008-02E.ai
14	SQT	√C					Computes the temperature-pressure compensation signal.
15	ST Y2	√C					Outputs the compensation signal.
16	BSC1	MV1					Execute control.
17	ST Y1	MV1					Manipulated output variable
18	END						Ends program excution.

- Steps 1 to 5
  These steps comprise the computation for calculating the pressure compensation value. S1 and S2 can be used to perform this computation continuously.
- Steps 6 to 10
  The temperature compensation value is calculated in the same way.
- Step 11
  The pressure and temperature compensation values are calculated. The pressure compensation value can be computed continuously as it is saved to a computation register.
- Steps 12 to 14
  The flow input is multiplied by the temperature-pressure compensation value, and the result undergoes square root extraction.
- Step 15
  The computation result is stored to output register (Y2).
- Step 16
  The basic control module and control type perform PID control computation taking the compensated flow stored in register S1 as the PV value, and the manipulated output variable (MV) is input to register S1.
- Step 17
  The manipulated output variable is stored to output register (Y1).
- Step 18
  This indicates the end of the user program. Program execution returns to Step 1, and 4 to 20 mA and 1 to 5 V are actually output to output registers Y1 and Y2, respectively.

(8)Function block programming



1008-01E.ai

10.4 Summary of Programming Precautions

The sections so far in this chapter have explained programming of the user program while referring to the example program. This section summarizes and explains the precautions to follow in programming.

(1)Execution of modules

Modules, such as the basic control module, are executed after data is stored to the computation registers (S1 to S5) by the LD instruction. The Appendix lists tables of text program instructions. In these tables, data is input to computation registers before instruction execution.

(2)Computations containing time functions

Attention is required to the following modules that contain time-related functions, such as first order lag, dead time and timers:

1) Do not program the same instruction twice or more as these are dynamic operations. (excluding linearizer functions) Even if a subprogram contains a dynamic operation, do not use a total of two or more instructions in both the main and subprograms.
2) Be sure to execute at each control period.
   The computation result differs when computations are performed intermittently by timer functions, for example.

(3)Jump instructions

The GO and GIF instructions enable jumps to a specified label name.
Create the program to prevent endless loops from occurring by jumping to a step before one containing a jump instruction. The control period overflow error will occur if a program that might result in an endless loop is downloaded and run on the YS1700. It is also difficult to discover mistakes in such programs. So, in principle, jump to steps following a step containing a jump instruction.
The GIF instruction executes a jump judgment using 0/1 data (e.g. DI input). Note, however, that the 0/1 data of register S1 that was used for the judgment is cleared after the computation.

(4)Control module

Each type of control module (BSC1, BSC2, CSC, SSC) can be used only once. BSC1 and BSC2, however, can be used simultaneously.

(5)Connecting to the communication system

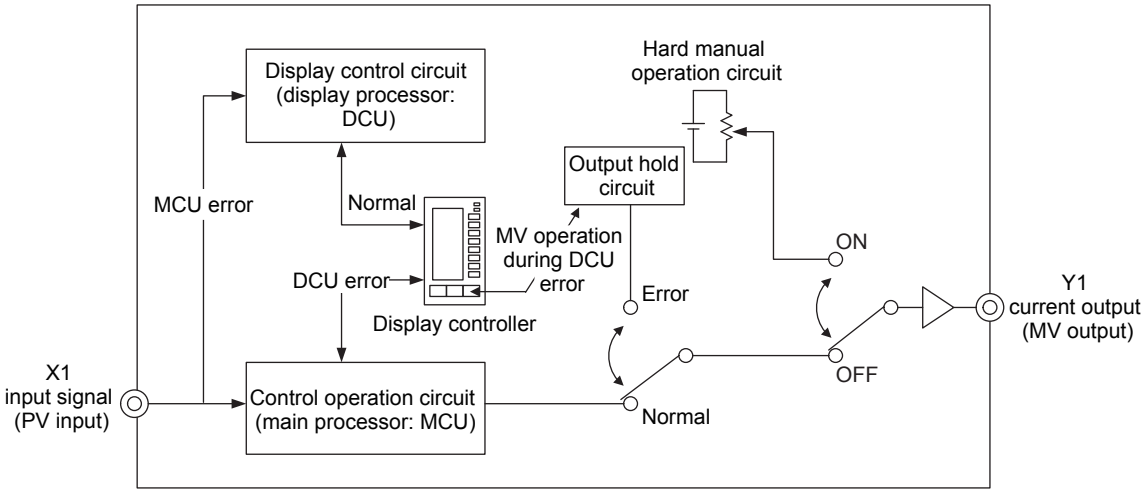
Variable constants P1 and P2 and communications outputs Y4, Y5 and Y6 are associated with DCS communication.
Receive registers CX1 to CX16 and CI1 to CI64, and send registers CY1 to CY4 and CO1 to CO16 are associated with peer-to-peer communication.

(6)Assignment of signals factoring in controller error

The YS1700 can indicate and manipulate one I/O signal point each even if the control operation circuit stops.
In this case, signal assignment is as follows when the controller (digital circuit) is in error as shown below.

PV display on a bar graph : X1 signal input is displayed.
Current manipulated signal (Y1) : Can be operated by manual operation keys and manipulated by analog backup output wheel.

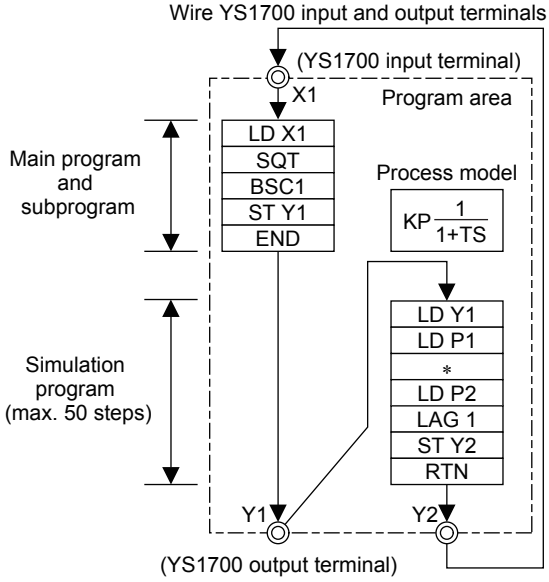
So, program important signals to X1 and Y1.



Backup Function

10.5 Making a Simulation Program

The YS1700 has program area for up to 50 steps for simulation program use in addition to program area for the main and subprograms. For example, you can create a pseudo plant by writing to the simulation program area the program of a process module assumed for the plant. Simulation programs are written in the same way as subprograms. That is, "SUB label name" must be included with @SIMPR as its label name. The difference with a main program is that the command for calling the subprogram, "GOSUB @SIMPR," is not required. Simulation programs can be executed only in the test run mode. Also, the GO and GIF instructions cannot be used in the simulation program.
The figure below shows an example of a simulation program.



[Program Example]

Program	S1	S2	S3
LD X1	X1		
SQT	$\sqrt{X1}$		
BSC1	MV1		
ST Y1	MV1		
END			
SUB @SIMPR			
LD Y1	Y1		
LD P1	P1	Y1	
*	P1×Y1		
LD P2	P2	P1×Y1	
LAG 1	$\left(1-e^{-\frac{1}{P2}}\right)$ × (P1×Y1)		
ST Y2	$\left(1-e^{-\frac{1}{P2}}\right)$ × (P1×Y1)		
RTN			

1010E.ai

Positioning of Simulation Program

11.1 YS1500/YS1700 Worksheet

YS1000 Sereis		Doc. No.		P. /
WORKSHEET	☐ **YS1500** ☐ **YS1700**	Order No.	Sec. / Loop / Item	
		Serial No.		
Customer		Model and Suffix		
Plant		Tag No.		

					CUSTOMER		REP.		ENGINEER		
					DR.	CH.	DR.	CH.	DR.	CH.	
REV.	n	REMARKS	DATE	REV. BY							

11.2 YS1700 Program Sheet

YS1000 Sereis ☐ **YS1700**	Doc. No.			P. /
PROGRAM SHEET	Order No.	Sec.	Loop	Item
	Serial No.			
Customer	Model and Suffix			
Plant	Tag No.			

Line No.	Step	Label	Program and Comments	S1	S2	S3	S4	S5
1								
2								
3								
4								
5								
6								
7								
8								
9								
0								
1								
2								
3								
4								
5								
6								
7								
8								
9								
0								
1								
2								
3								
4								
5								
6								
7								
8								
9								
0								

Note:

					CUSTOMER		REP.		ENGINEER		
					DR.	CH.	DR.	CH.	DR.	CH.	
REV.	n	REMARKS	DATE	REV. BY							

YOKOGAWA

WS 01B08K01-02
1st Edition: 2007. 03.26
1101E.ai

11.3 YS1310/YS1350/YS1360 Worksheet

YS1000 Sereis WORKSHEET	☐ YS1310 ☐ YS1350 ☐ YS1360	Doc. No.		P. /
		Order No.	Sec. / Loop / Item	
		Serial no.		
Customer		Model and Suffix		
Plant		Tag No.		

					CUSTOMER		REP.		ENGINEER		
					DR.	CH.	DR.	CH.	DR.	CH.	
REV.	n	REMARKS	DATE	REV. BY							

YOKOGAWA

WS 01B08K01-03
1st Edition: 2007. 03.26
1103E.ai

11.4 Parameter Sheet for SLPC-YS1700 Data Conversion

Use a parameter sheet when converting the data of SLPC to YS1700.

Read the side panel and switches on SLPC, write the parameters on the parameter sheet, and then set the parameters in YSS1000.

SLPC-YS1700 Parameter Sheet (1/8)

MODEL	SLPC	TAG		SERIAL	

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Tiltle	Parameter	Range	Unit	Default	Setting value	
BATCH	BD1	0.0 to 100.0	%	0.0		ENG. MENU1	SMPL& BATCH	BD1	0.0 to 100.0	%	0.0		
	BB1		%	0.0				BB1		%	0.0		
	BL1		%	0.0				BL1		%	0.0		
	BD2	0.0 to 100.0	%	0.0				*1	*1				*1
	BB2		%	0.0									
	BL2		%	0.0									
SAMPLE	ST1	0 to 9999	s	0				STM1	0 to 9999	s	0		
	SW1		s	0				SWD1		s	0		
	ST2	0 to 9999	s	0				STM2	0 to 9999	s	0		
	SW2		s	0				SWD2		s	0		
NON-LINEAR	GW1	0.0 to 100.0	%	0.0		TUNING MENU	PID1	GW1	0.0 to 100.0	%	0.0		
	GG1	0.000 to 1.000	-	1.000				GG1	0.000 to 1.000	-	1.000		
	GW2	0.0 to 100.0	%	0.0			PID2	GW2	0.0 to 100.0	%	0.0		
	GG2	0.000 to 1.000	-	1.000				GG2	0.000 to 1.000	-	1.000		
	F01	0.0 to 100.0	%	0.0		ENG. MENU2	FX TABLE	FXO101	0.000 to 1.000	-	0.000		*2
	F02		%	10.0				FXO102		-	0.100		
	F03		%	20.0				FXO103		-	0.200		
	F04		%	30.0				FXO104		-	0.300		
	F05		%	40.0				FXO105		-	0.400		
	F06		%	50.0				FXO106		-	0.500		
	F07		%	60.0				FXO107		-	0.600		
	F08		%	70.0				FXO108		-	0.700		
	F09		%	80.0				FXO109		-	0.800		
	F10		%	90.0				FXO110		-	0.900		
	F11		%	100.0				FXO111		-	1.000		

EU: Engineering Unit

*1: SLPC*A only. Do not set this as is because the specifications differ from YS1700.

*2: Divide the setting value of SLPC by 100 and calculate to three digits after the decimal point.

SLPC-YS1700 Parameter Sheet (2/8)

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
NON-LINEAR	G01	0.0 to 100.0	%	0.0		ENG. MENU2	FX TABLE	FXO201	0.000 to 1.000	-	0.000		*1
	G02		%	10.0				FXO202		-	0.100		
	G03		%	20.0				FXO203		-	0.200		
	G04		%	30.0				FXO204		-	0.300		
	G05		%	40.0				FXO205		-	0.400		
	G06		%	50.0				FXO206		-	0.500		
	G07		%	60.0				FXO207		-	0.600		
	G08		%	70.0				FXO208		-	0.700		
	G09		%	80.0				FXO209		-	0.800		
	G10		%	90.0				FXO210		-	0.900		
	G11		%	100.0				FXO211		-	1.000		
	H01	-25.0 to 125.0	%	0.0		ENG. MENU2	GX1 TABLE	GXI101	-0.250 to 1.250	-	0.000		*1,*2
	H02		%	10.0				GXI102		-	0.100		
	H03		%	20.0				GXI103		-	0.200		
	H04		%	30.0				GXI104		-	0.300		
	H05		%	40.0				GXI105		-	0.400		
	H06		%	50.0				GXI106		-	0.500		
	H07		%	60.0				GXI107		-	0.600		
	H08		%	70.0				GXI108		-	0.700		
	H09		%	80.0				GXI109		-	0.800		
	H10		%	90.0				GXI110		-	0.900		
	H11		%	100.0				GXI111		-	1.000		
	I01	-25.0 to 125.0	%	0.0				GXO101	-0.250 to 1.250	-	0.000		
	I02		%	10.0				GXO102		-	0.100		
	I03		%	20.0				GXO103		-	0.200		
	I04		%	30.0				GXO104		-	0.300		
	I05		%	40.0				GXO105		-	0.400		
	I06		%	50.0				GXO106		-	0.500		
	I07		%	60.0				GXO107		-	0.600		
	I08		%	70.0				GXO108		-	0.700		
	I09		%	80.0				GXO109		-	0.800		
	I10		%	90.0				GXO110		-	0.900		
	I11		%	100.0				GXO111		-	1.000		

EU: Engineering Unit

*1: Divide the setting value of SLPC by 100 and calculate to three digits after the decimal point.

*2: SLPC*A only.

SLPC-YS1700 Parameter Sheet (3/8)

SLPC						YS1700							Remarks
Keyvoard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
NON-LINEAR	L01	-25.0 to 125.0	%	0.0		ENG. MENU2	GX2 TABLE	GXI201	-0.250 to 1.250	-	0.000		*1,*2
	L02		%	10.0				GXI202		-	0.100		
	L03		%	20.0				GXI203		-	0.200		
	L04		%	30.0				GXI204		-	0.300		
	L05		%	40.0				GXI205		-	0.400		
	L06		%	50.0				GXI206		-	0.500		
	L07		%	60.0				GXI207		-	0.600		
	L08		%	70.0				GXI208		-	0.700		
	L09		%	80.0				GXI209		-	0.800		
	L10		%	90.0				GXI210		-	0.900		
	L11		%	100.0				GXI211		-	1.000		
	M01	-25.0 to 125.0	%	0.0				GXO201	-0.250 to 1.250	-	0.000		
	M02		%	10.0				GXO202		-	0.100		
	M03		%	20.0				GXO203		-	0.200		
	M04		%	30.0				GXO204		-	0.300		
	M05		%	40.0				GXO205		-	0.400		
	M06		%	50.0				GXO206		-	0.500		
	M07		%	60.0				GXO207		-	0.600		
	M08		%	70.0				GXO208		-	0.700		
	M09		%	80.0				GXO209		-	0.800		
	M10		%	90.0				GXO210		-	0.900		
	M11		%	100.0				GXO211		-	1.000		

EU: Engineering Unit

*1: Divide the setting value of SLPC by 100 and calculate to three digits after the decimal point.

*2: SLPC*A only.

SLPC-YS1700 Parameter Sheet (4/8)

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
PN	P01	*1	EU	0.0		TUNING MENU	P&T REG	P01	-99999 to 99999	-	0.0		Set the SLPC setting data as is in YSS1000.
	P02		EU	0.0				P02		-	0.0		
	P03		EU	0.0				P03		-	0.0		
	P04		EU	0.0				P04		-	0.0		
	P05		EU	0.0				P05		-	0.0		
	P06		EU	0.0				P06		-	0.0		
	P07		EU	0.0				P07		-	0.0		
	P08		EU	0.0				P08		-	0.0		
	P09	-800.0 to 800.0	%	0.0				P09	-99999 to 99999	-	0.0		Set the SLPC setting data as is in YSS1000.
	P10		%	0.0				P10		-	0.0		
	P11		%	0.0				P11		-	0.0		
	P12		%	0.0				P12		-	0.0		
	P13		%	0.0				P13		-	0.0		
	P14		%	0.0				P14		-	0.0		
	P15		%	0.0				P15		-	0.0		
	P16		%	0.0				P16		-	0.0		
	P20	0 to 9999	s	0		ENG. MENU3	PGM1 SET	PGT101	0 to 9999	-	0		The unit is treated as a second. *2
	P21		s	0				PGT102		-	0		
	P22		s	0				PGT103		-	0		
	P23		s	0				PGT104		-	0		
	P24		s	0				PGT105		-	0		
	P25		s	0				PGT106		-	0		
	P26		s	0				PGT107		-	0		
	P27		s	0				PGT108		-	0		
	P28		s	0				PGT109		-	0		
	P29		s	0				PGT110		-	0		
	P30	-25.0 to 125.0	%	0.0				PGO101	-0.250 to 1.250	-	0.000		*2,*3
	P31		%	0.0				PGO102		-	0.000		
	P32		%	0.0				PGO103		-	0.000		
	P33		%	0.0				PGO104		-	0.000		
	P34		%	0.0				PGO105		-	0.000		
	P35		%	0.0				PGO106		-	0.000		
	P36		%	0.0				PGO107		-	0.000		
	P37		%	0.0				PGO108		-	0.000		
	P38		%	0.0				PGO109		-	0.000		
	P39		%	0.0				PGO110		-	0.000		

EU: Engineering Unit

*1: Value ranging from -800.0 to 800.0% of the engineering unit set with PSHn, PSLn, and PSDn (n=1 to 8).

*2: SLPC*A only.

*3: Divide the setting value of SLPC by 100 and calculate to three digits after the decimal point.

SLPC-YS1700 Parameter Sheet (5/8)

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
PN	PX1	0.000 to 1.000	-	0.000		TUNING MENU	PID1	SFA1	0.000 to 1.000	-	0.000		*9, *10
	PY1		-	0.000				SFB1		-	0.000		
	PZ1	0.000 to 1.000	-	0.000				-					*11
	PX2	0.000 to 1.000	-	0.000			PID2	SFA2	0.000 to 1.000	-	0.000		*9, *10
	PY2		-	0.000				SFB2		-	0.000		
	PZ2	0.000 to 1.000	-	0.000				-					*11
PHPL	PH1	*1	EU	106.3			PID1	PH1	*5	EU	106.3		*12, *14
	PL1		EU	-6.3				PL1		EU	-6.3		*12, *15
	PH2	*2	EU	106.3			PID2	PH2	*6	EU	106.3		*13, *14
	PL2		EU	-6.3				PL2		EU	-6.3		*13, *15
DLVLVT	DL1	*3	EU	100.0			PID1	DL1	*7	EU	0.0		*12, *16
	VL1		EU	100.0				VL1		EU	0.0		
	VT1	1 to 9999	s	1				VT1	1 to 9999	s	1		
	DL2	*4	EU	100.0			PID2	DL2	*8	EU	0.0		*13, *16
	VL2		EU	100.0				VL2		EU	0.0		
	VT2	1 to 9999	s	1				VT2	1 to 9999	s	1		

EU: Engineering Unit
*1: Engineering unit of -6.3 to 106.3% set with HI1, LO1, and DP1.
*2: Engineering unit of -6.3 to 106.3% set with HI2, LO2, and DP2.
*3: Engineering unit of 0.0 to 100.0% set with HI1, LO1, and DP1.
*4: Engineering unit of 0.0 to 100.0% set with HI2, LO2, and DP2.
*5: Engineering unit of -6.3 to 106.3% set with SCH1, SCLO1, and SCDP1.
*6: Engineering unit of -6.3 to 106.3% set with SCH2, SCLO2, and SCDP2.
*7: Engineering unit of 0.0 to 106.3% set with SCH1, SCLO1, and SCDP1.
*8: Engineering unit of 0.0 to 106.3% set with SCH2, SCLO2, and SCDP2.
*9: SLPC*A only.
*10: There is no applicable parameter in SLPC-x40*E.
*11: There is no applicable parameter.
*12: Set SCH1, SCL1, and SCDP1 first.
*13: Set SCH2, SCL2, and SCDP2 first.
*14: The alarm of YS1700 does not work with a setting equivalent to 106.3%.
*15: The alarm of YS1700 does not work with a setting equivalent to -6.3%.
*16: The alarm of YS1700 does not work with a setting equivalent to 0.0%.

SLPC-YS1700 Parameter Sheet (6/8) *Only for SLPC-x81*E.

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
MVMHML	MH1	-6.3 to 106.3	%	106.3		TUNING MENU	PID1	MH1	-6.3 to 106.3	%	106.3		
	ML1		%	-6.3				ML1		%	-6.3		
	MH2	-6.3 to 106.3	%	106.3			PID2	MH2	-6.3 to 106.3	%	106.3		
	ML2		%	-6.3				ML2		%	-6.3		
PID *1	STC	OFF/0/1/2	-	OFF			STC1	STC	OFF/DISP/ON/ATSTUP	-	OFF		OFF:OFF, 0:DISP, 1:ON, 2:ATSTUP
	PB1	6.3 to 999.9	%	999.9			PID1	PB1	0.1 to 999.9	%	999.9		
	TI1	1 to 9999	s	1000				TI1	1 to 9999	s	1000		
	TD1	0 to 9999	s	0				TD1	0 to 9999	s	0		When TD1=1 for SLPC, set TD1=0 for YS1700.
	IP1	0/1	-	0			STC1	IP1	STATIC/DYNAM	-	STATIC		0:SATIC, 1:DYNAM
	TR1	4 to 9999	s	300				TR1	4 to 9999	s	300		
	NB1	*2	EU	0.0				NB1	*4	EU	0.0		*6
	OS1	0/1/2/3	-	2				OS1	ZERO/MIN/MED/MAX	-	MED		0:ZERO, 1:MIN, 2:MED, 3:MAX
	MI1	0.0 to 20.0	%	5.0				MI1	0.0 to 20.0	%	5.0		
	R01	6.3 to 999.9	%	999.9				PMX1	2.0 to 999.9	%	999.9		
	R02	6.3 to 999.9	%	6.3				PMN1	2.0 to 999.9	%	2.0		
	R03	1 to 9999	s	9999				IMX1	1 to 9999	s	9999		
	R04	1 to 9999	s	1				IMN1	1 to 9999	s	1		
	R05	0 to 9999	s	2000				DMX1	0 to 9999	s	2000		
	PB2	6.3 to 999.9	s	999.9			PID2	PB2	0.1 to 999.9	%	999.9		
	TI2	1 to 9999	s	1000				TI2	1 to 9999	s	1000		
	TD2	0 to 9999	s	0				TD2	0 to 9999	s	0		When TD2=1 for SLPC, set TD2=0 for YS1700.
	IP2	0/1	-	0			STC2	IP2	STATIC/DYNAM	-	STATIC		0:SATIC, 1:DYNAM
	TR2	4 to 9999	s	300				TR2	4 to 9999	s	300		
	NB2	*3	EU	0.0				NB2	*5	EU	0.0		*7
	OS2	0/1/2/3	-	2				OS2	ZERO/MIN/MED/MAX	-	MED		0:ZERO, 1:MIN, 2:MED, 3:MAX
	MI2	0.0 to 20.0	%	5.0				MI2	0.0 to 20.0	%	5.0		
	R06	6.3 to 999.9	%	999.9				PMX2	2.0 to 999.9	%	999.9		
	R07	6.3 to 999.9	%	6.3				PMN2	2.0 to 999.9	%	2.0		
	R08	1 to 9999	s	9999				IMX2	1 to 9999	s	9999		
	R09	1 to 9999	s	1				IMN2	1 to 9999	s	1		
	R10	0 to 9999	s	2000				DMX2	0 to 9999	s	2000		

EU: Engineering Unit
*1: SLPC-x81*E only.
*2: Engineering unit of 0.0 to 20.0% set with HI1, LO1, and DP1.
*3: Engineering unit of 0.0 to 20.0% set with HI2, LO2, and DP2.
*4: Engineering unit of 0.0 to 20.0% set with SCH1, SCL1, and SCDP1.
*5: Engineering unit of 0.0 to 20.0% set with SCH2, SCL2, and SCDP2.
*6: Set SCH1, SCL1, and SCDP1 first.
*7: Set SCH2, SCL2, and SCDP2 first.

SLPC-YS1700 Parameter Sheet (7/8) *Only for SLPC-x81*E.

SLPC						YS1700							Remarks
Keybaord	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
PVSVDV	SV1	*1	EU	-6.3		TUNING	PID1	SV1	*3	EU	0.0		*5
	SV2	*2	EU	-6.3		MENU	PID2	SV2	*4	EU	0.0		*6
SCALE	HI1	-9999 to 9999	-	1000				SCH1	-80000 to 80000	-	1000		
	LO1		-	0				SCL1		-	0		
	DP1	1 to 4	-	3			CONFIG2	SCDP1	#.#### to #####	-	####.#		1:##.###, 2:###.##, 3:####.#, 4:#####
	HI2	-9999 to 9999	-	1000				SCH2	-80000 to 80000	-	1000		
	LO2		-	0				SCL2		-	0		
	DP2	1 to 4	-	3				SCDP2	#.#### to #####	-	####.#		1:##.###, 2:###.##, 3:####.#, 4:#####
MODE	MODE1	0/1	-	0		ENG.	CONFIG1	START	AUT to COLD	-	M-COLD		0:M-COLD, 1:AUT
	MODE2	0/1/2	-	0		MENU1	CONFIG2	CMOD1	-/CAS/CMP	-	-		0:-, 1:CAS, 2:CMP
	MODE3	0/1	-	0				-					*7
	MODE4	0/1	-	0			CONFIG2	BMOD1	BUM/BUA	-	BUM		0:BUM, 1:BUA
	MODE5	0/1	-	0				-					*7
SW	ACTION1	DIR/RVS	-	RVS			CONFIG2	ACT1	DIR/RVS	-	RVS		
Action switch	ACTION2	DIR/RVS	-	RVS				ACT2	DIR/RVS	-	RVS		

EU: Engineering Unit
*1: Engineering unit of -6.3 to 106.3% set with HI1, LO1, and DP1.
*2: Engineering unit of -6.3 to 106.3% set with HI2, LO2, and DP2.
*3: Engineering unit of -6.3 to 106.3% set with SCH1, SCL1, and SCDP1.
*4: Engineering unit of -6.3 to 106.3% set with SCH2, SCL2, and SCDP2.
*5: Set SCH1, SCL1, and SCDP1 first.
*6: Set SCH2, SCL2, and SCDP2 first.
*7: There is no applicable parameter.

SLPC-YS1700 Parameter Sheet (8/8) *Exclude SLPC-x81*E.

SLPC						YS1700							Remarks
Keyboard	Parameter	Range	Unit	Default	Setting value	Menu	Title	Parameter	Range	Unit	Default	Setting value	
MVMHML	MH1	-6.3 to 106.3	%	106.3		TUNING MENU	PID1	MH1	-6.3 to 106.3	%	106.3		
	ML1		%	-6.3				ML1		%	-6.3		
	MH2	-6.3 to 106.3	%	106.3			PID2	MH2	-6.3 to 106.3	%	106.3		
	ML2		%	-6.3				ML2		%	-6.3		
PID *1	PB1	6.3 to 999.9	%	999.9			PID1	PB1	0.1 to 999.9	%	999.9		
	TI1	1 to 9999	s	1000				TI1	1 to 9999	s	1000		
	TD1	0 to 9999	s	0				TD1	0 to 9999	s	0		When TD1=1 for SLPC, set TD1=0 for YS1700.
	PB2	6.3 to 999.9	%	999.9			PID2	PB2	0.1 to 999.9	%	999.9		
	TI2	1 to 9999	s	1000				TI2	1 to 9999	s	1000		
	TD2	0 to 9999	s	0				TD2	0 to 9999	s	0		When TD2=1 for SLPC, set TD2=0 for YS1700.
PVSVDV	SV1	*2	EU	-6.3			PID1	SV1	*4	EU	0.0		*6
	SV2	*3	EU	-6.3			PID2	SV2	*5	EU	0.0		*7
SCALE	HI1	-9999 to 9999	-	1000		ENG. MENU1	CONFIG2	SCH1	-80000 to 80000	-	1000		
	LO1		-	0				SCL1		-	0		
	DP1	1 to 4	-	3				SCDP1	#.#### to #####	-	####.#		1:##.###, 2:###.##, 3:####.#, 4:#####
	HI2	-9999 to 9999	-	1000				SCH2	-80000 to 80000	-	1000		
	LO2		-	0				SCL2		-	0		
	DP2	1 to 4	-	3				SCDP2	#.#### to #####	-	####.#		1:##.###, 2:###.##, 3:####.#, 4:#####
MODE	MODE1	0/1	-	0		ENG. MENU1	CONFIG1	START	AUT to COLD	-	M-COLD		0:M-COLD, 1:AUT
	MODE2	0/1/2	-	0			CONFIG2	CMOD1	-/CAS/CMP	-	-		0:-, 1:CAS, 2:CMP
	MODE3	0/1	-	0				-					*8
	MODE4	0/1	-	0			CONFIG2	BMOD1	BUM/BUA	-	BUM		0:BUM, 1:BUA
	MODE5	0/1	-	0				-					*8
SW	ACTION1	DIR/RVS	-	RVS			CONFIG2	ACT1	DIR/RVS	-	RVS		
Action switch	ACTION2	DIR/RVS	-	RVS				ACT2	DIR/RVS	-	RVS		

EU: Engineering Unit

*1: When SLPC-x70*E, the STC parameter cannot be used because the STC method is different. Readjust the parameter on YS1700.

*2: Engineering unit of -6.3 to 106.3% set with HI1, LO1, and DP1.

*3: Engineering unit of -6.3 to 106.3% set with HI2, LO2, and DP2.

*4: Engineering unit of -6.3 to 106.3% set with SCH1, SCL1, and SCDP1.

*5: Engineering unit of -6.3 to 106.3% set with SCH2, SCL2, and SCDP2.

*6: Set SCH1, SCL1, and SCDP1 first.

*7: Set SCH2, SCL2, and SCDP2 first.

*8:There is no applicable parameter.

12.1 List of Text Program Instructions

The following tables list computation instructions and the action of the S registers before and after execution of these computation instructions.
"m" in the instruction is the machine number in dynamic operation. These computations can be used only once per single machine number in a single control period as time and previous value buffer are used. Program operation will not be normal if computations are programmed two or more times.
The number of times that "n" in instructions can be used is not restricted.

Note

- A, B, C, D and E in computation registers indicates pre-stored data.
- The following registers can be used by the read (LD reg) instruction:
  - Control data registers (6.2.1)
  - Control flag registers (6.2.2)
  - Control parameter registers (6.2.3)
  - Self-tuning (STC) registers (6.2.4)
  - Preset PID switch registers (6.2.5)
  - System flag registers (6.2.6)
  - Operation display control registers (6.2.7)
  - Peer-to-peer communication registers (8.2.2)
- The following registers can be used by the store (ST reg) instruction:
  - Control data registers (6.2.1)
  - Control flag registers (6.2.2)
  - Control parameter registers (6.2.3)
  - Self-tuning (STC) registers (6.2.4)
  - Preset PID switch registers (6.2.5)
  - Operation display control registers Z01 (6.2.7)
  - Peer-to-peer communication registers CYn, COn (8.2.2)

Computation Instruction List

Instruction	Explanation			Before Instruction Execution					After Instruction Execution					Overflow	Backup
				S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
LD Xn	n=1 to 8	Reads Xn to S1.	Xn: Analog input register	A	B	C	D	E	Xn	A	B	C	D	No	No
LD Yn	n=1 to 6	Reads Yn to S1.	Yn: Analog output register	A	B	C	D	E	Yn	A	B	C	D	No	No
LD DIn	n=1 to 10	Reads DIn to S1.	DIn: Digital input register	A	B	C	D	E	DIn	A	B	C	D	No	No
LD DOn	n=1 to 32	Reads DOn to S1.	DOn: Digital output register	A	B	C	D	E	DOn	A	B	C	D	No	No
LD KYn	n=1	Reads KYn to S1.	KYn: key register n=1: PF key	A	B	C	D	E	KYn	A	B	C	D	No	No
LD LPn	n=1 to 7	Reads LPn to S1.	LPn: LED lamp register n=1: PF, 2: C1, 3: A1, 4: M1, 5: C2, 6: A2, 7: M2	A	B	C	D	E	LPn	A	B	C	D	No	No
LD Kn	n=1 to 100	Reads Kn to S1.	Kn: Constant register	A	B	C	D	E	Kn	A	B	C	D	No	No
LD Pn	n=1 to 30	Reads Pn to S1.	Pn: Variable register	A	B	C	D	E	Pn	A	B	C	D	No	No
LD Tn	n=1 to 60	Reads Tn to S1.	Tn: Temporary memory register	A	B	C	D	E	Tn	A	B	C	D	No	No
LD CFL		Reads CFL to S1.	CFL: Backlight ON/OFF flag	A	B	C	D	E	CFL	A	B	C	D	No	No
LD reg		Reads reg to S1.	reg: Control register, flag, etc.	A	B	C	D	E	reg	A	B	C	D	No	No
LD CXn	n=1 to 16	Reads CXn to S1.	CXn: Peer-to-peer communication analog input register	A	B	C	D	E	CXn	A	B	C	D	No	No
LD CYn	n=1 to 4	Reads CYn to S1.	CYn: Peer-to-peer communication analog output register	A	B	C	D	E	CYn	A	B	C	D	No	No
LD CIn	n=1 to 16	Reads CIn to S1.	CIn: Peer-to-peer communication status input register	A	B	C	D	E	CIn	A	B	C	D	No	No
LD COn	n=1 to 16	Reads COn to S1.	COn: Peer-to-peer communication status output register	A	B	C	D	E	COn	A	B	C	D	No	No
LD CFn	n=1 to 4	Reads CFn to S1.	CFn: Read receive time out flag for peer-to-peer communication	A	B	C	D	E	CFn	A	B	C	D	No	No
LD Zn	n=1 to 2	Reads Zn to S1.	Zn: Display number Z2 is the currently displayed display number.	A	B	C	D	E	Zn	A	B	C	D	No	No
ST f0n	n=0 to 1	Stores S1 to f0n.	f0n: EXMF1, EXAF1	A	B	C	D	E	A	B	C	D	E	No	No
ST CYn	n=1 to 4	Stores S1 to CYn.		A	B	C	D	E	A	B	C	D	E	No	No
ST COn	n=1 to 16	Stores S1 to COn.		A	B	C	D	E	A	B	C	D	E	No	No
ST Zn	n=1 to 2	Stores S1 to Zn.	Zn: Display number Z1 writes the number of the desired display.	A	B	C	D	E	A	B	C	D	E	No	No
ST Xn	n=1 to 8	Stores S1 to Xn.		A	B	C	D	E	A	B	C	D	E	No	No
ST Yn	n=1 to 6	Stores S1 to Yn.		A	B	C	D	E	A	B	C	D	E	No	Yes
ST DIn	n=1 to 10	Stores S1 to DIn.		A	B	C	D	E	A	B	C	D	E	No	No

Instruction	Explanation			Before Instruction Execution					After Instruction Execution					Overflow	Backup
				S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
ST DOn	n=1 to 32	Stores S1 to DOn.		A	B	C	D	E	A	B	C	D	E	No	Yes
ST LPn	n=1	Stores S1 to LPn.		A	B	C	D	E	A	B	C	D	E	No	No
ST Pn	n=1 to 30	Stores S1 to Pn.		A	B	C	D	E	A	B	C	D	E	No	Yes
ST Tn	n=1 to 60	Stores S1 to Tn.		A	B	C	D	E	A	B	C	D	E	No	Yes
ST CFL		Stores S1 to CFL.		A	B	C	D	E	A	B	C	D	E	No	No
ST reg		Stores S1 to reg.	reg: Control register, flag, etc.	A	B	C	D	E	A	B	C	D	E	No	*1
STE xx		Store with "Enable switch"	All xx correspond to same xx as ST instruction.	Enable switch	A	B	C	D	When switch ≥ 0.5, A is stored to xx When switch < 0.5, A is not stored to xx	B	C	D	D	No	*2
END		End of computation		A	B	C	D	E	A	B	C	D	E	No	No
+		Addition		A	B	C	D	E	B+A	C	D	E	E	Yes	No
-		Subtraction		A	B	C	D	E	B-A	C	D	E	E	Yes	No
*		Multiplication		A	B	C	D	E	B×A	C	D	E	E	Yes	No
/		Division		A	B	C	D	E	B÷A	C	D	E	E	Yes	No
RATIO		Ratio computation		A	B	C	D	E	(D+C)×B+A	E	E	E	E	Yes	No
SQT		Square root extraction	Lot cutoff when input < 1%. 0 is output.	A	B	C	D	E	√A	B	C	D	E	No	No
SQTE		Square root extraction with variable low cutoff	Linear at low cutoff point or lower No hysteresis at low cutoff point	Low cutoff setpoint	A	B	C	D	√A when S2 > S1 A when S2 ≤ S1	B	C	D	D	No	No
ABS		Absolute value		A	B	C	D	E	|A|	B	C	D	E	No	No
HSL		High selector		A	B	C	D	E	S1 when S1>S2 S2 when S1≤S2	C	D	E	E	No	No
LSL		Low selector		A	B	C	D	E	S2 when S1>S2 S1 when S1≤S2	C	D	E	E	No	No
HLM		High Limiter		High limit setpoint	Input value	A	B	C	Input value when S1>S2 High limit when S1≤S2	A	B	C	C	No	No
LLM		Low Limiter		Low limit setpoint	Input value	A	B	C	Low limit when S1>S2 Input value when S1≤S2	A	B	C	C	No	No
SCAL		Scaling		Decimal point position	0% scale value	100% scale value	Input	A	(Input value x (100% scale value - 0% scale value) + 0% scale value)/10^n	A	A	A	A	Yes	No
NORM		Normalization		Decimal point position	0% scale value	100% scale value	Input	A	((input value x 10^n) - 0% scale value)/(100% scale value - 0% scale value)	A	A	A	A	Yes	No
LN		Natural logarithm		A	B	C	D	E	ln (A)	B	C	D	E	Yes	No
LOG		Common logarithm		A	B	C	D	E	log (A)	B	C	D	E	Yes	No
EXP		Exponential		A	B	C	D	E	Ath power of e	B	C	D	E	Yes	No
PWR		Power		A	B	C	D	E	Ath power of B	C	D	E	E	Yes	No
TCMP1		Temperature compensation (°C)		Reference temperature (°C)	Temperature (°C)	Flow	A	B	Flow x (ref. temperature + 273.15)/(temperature + 273.15)	A	B	B	B	Yes	No

Instruction	Explanation	Before Instruction Execution					After Instruction Execution					Overflow	Backup
		S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
TCMP2	Temperature compensation (°F)	Reference temperature (°F)	Temperature (°F)	Flow	A	B	Flow x ((ref. temperature - 32) / 18 + 273.15) / ((temperature - 32) / 18 + 273.15)	A	B	B	B	Yes	No
TCMP3	Temperature compensation (K)	Reference temperature (K)	Temperature (K)	Flow	A	B	Flow x reference temperature/ temperature	A	B	B	B	Yes	No
PCMP1	Pressure compensation (MPa)	Reference pressure (MPa)	Pressure (MPa)	Flow	A	B	Flow x (pressure + 0.101325)/(reference pressure + 0.101325)	A	B	B	B	Yes	No
PCMP2	Pressure compensation (kgf/cm^2)	Reference pressure (kgf/cm^2)	Pressure (kgf/cm^2)	Flow	A	B	Flow x (pressure + 1.03323) / (reference pressure + 1.03323)	A	B	B	B	Yes	No
PCMP3	Pressure compensation (psi)	Reference pressure (psi)	Pressure (psi)	Flow	A	B	Flow x (pressure + 14.6959) / (reference pressure + 14.6959)	A	B	B	B	Yes	No
DIBCD	Conversion from DI to BCD	Number of bits	Start DI number	A	B	C	To convert a DI input into a BCD value, the DI value specified by S2 is assigned to the least significant bit of the BCD value. Then the DI value specified by S1 is converted and assigned to the remaining bits of the BCD value. The BCD value thus determined is in turn converted into a numerical floating point value.	A	B	C	C	No	No
DOBCD	Conversion from BCD to DO	Number of bits	Start DO number	Converted value	A	B	Converted value	A	B	B	B	No	No
DIBIN	Conversion from DI to Binary	Number of bits	Start DI number	A	B	C	To convert a DI input into a binary value, the DI value specified by S2 is assigned to the least significant bit of the binary value. Then the DI value specified by S1 is converted and assigned to the remaining bits of the binary value. The binary value thus determined is in turn converted into a numerical floating point value.	A	B	C	C	No	No
DOBIN	Conversion from Binary to DO	Number of bits	Start DO number	Converted value	A	B	Converted value	A	B	B	B	No	No
MAX2	Maximum of 2 values	A	B	C	D	E	Max. value of A to B	C	D	E	E	No	No
MAX3	Maximum of 3 values	A	B	C	D	E	Max. value of A to C	D	E	E	E	No	No
MAX4	Maximum of 4 values	A	B	C	D	E	Max. value of A to D	E	E	E	E	No	No
MIN2	Minimum of 2 values	A	B	C	D	E	Min. value of A to B	C	D	E	E	No	No
MIN3	Minimum of 3 values	A	B	C	D	E	Min. value of A to C	D	E	E	E	No	No
MIN4	Minimum of 4 values	A	B	C	D	E	Min. value of A to D	E	E	E	E	No	No
AVE2	Average of 2 values	A	B	C	D	E	(A+B)÷2	C	D	E	E	Yes	No
AVE3	Average of 3 values	A	B	C	D	E	(A+B+C)÷3	D	E	E	E	Yes	No
AVE4	Average of 4 values	A	B	C	D	E	(A+B+C+D)÷4	E	E	E	E	Yes	No
INC	Increment	A	B	C	D	E	A+1.0	B	C	D	E	Yes	No
DEC	Decrement	A	B	C	D	E	A-1.0	B	C	D	E	Yes	No

Instruction	Explanation		Before Instruction Execution					After Instruction Execution					Overflow	Backup
			S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
FXn	n=1 to 2	10-segment linearizer function	Input value	A	B	C	D	Input value after linear interpolation	A	B	C	D	No	No
IFXn	n=1 to 2	Inverse conversion of 10-segment linearizer function	Input value	A	B	C	D	Input value after linear interpolation	A	B	C	D	No	No
GXn	n=1 to 2	Arbitrary linearizer function	Input value	A	B	C	D	Input value after linear interpolation	A	B	C	D	No	No
IGXn	n=1 to 2	Inverse conversion of arbitrary segment linearizer function	Input value	A	B	C	D	Input value after linear interpolation	A	B	C	D	No	No
LAGm	m=1 to 8	First order lag (second)	Time constant	Input value	A	B	C	Input value after first order lag computation $\Delta T/(\Delta T + T) \times$ (this time - previous) + previous	A	B	C	C	No	Yes
LAGMm	m=1 to 8	First order lag (minute)	Time constant	Input value	A	B	C	Input value after first order lag computation $\Delta T/(\Delta T + T) \times$ (this time - previous) + previous	A	B	C	C	No	Yes
LEDm	m=1 to 2	Derivative (second)	Time constant	Input value	A	B	C	Input value after incomplete derivative computation (input value this time) - (first order lag result)	A	B	C	C	Yes	Yes
LEDMm	m=1 to 2	Derivative (minute)	Time constant	Input value	A	B	C	Input value after incomplete derivative computation (input value this time) - (first order lag result)	A	B	C	C	Yes	Yes
DEDm	m=1 to 3	Dead time (second)	Dead time	Input value	A	B	C	Past input value	A	B	C	C	No	Yes
DEDMm	m=1 to 3	Dead time (minute)	Dead time	Input value	A	B	C	Past input value	A	B	C	C	No	Yes
VELm	m=1 to 3	Velocity computation (second)	Dead time	Input value	A	B	C	(input value this time) - (dead time computation result)	A	B	C	C	Yes	Yes
VELMm	m=1 to 3	Velocity computation (minute)	Dead time	Input value	A	B	C	(input value this time) - (dead time computation result)	A	B	C	C	Yes	Yes
VLMm	m=1 to 6	Velocity limiter	Fall velocity restricted value	Rise velocity restricted value	Input value	A	B	Input value whose velocity is restricted	A	B	B	B	Yes	Yes
MAVm	m=1 to 3	Moving average computation (second)	Computation time	Input value	A	B	C	Input value after moving average	A	B	C	C	Yes	Yes
MAVMm	m=1 to 3	Moving average computation (minute)	Computation time	Input value	A	B	C	Input value after moving average	A	B	C	C	Yes	Yes
CCDm	m=1 to 8	0 to 1 change detection	Input value	A	B	C	D	1 when S1 this time < 0.5 and S1 this time ≥ 0.5 Otherwise 0	A	B	C	D	No	Yes
UEDGm	m=1 to 8	0 to 1 change detection	Input value	A	B	C	D	1 when previous S1 < 0.5 and S1 this time ≥ 0.5, otherwise 0	A	B	C	D	No	Yes
DEDGm	m=1 to 8	1 to 0 change detection	Input value	A	B	C	D	1 when previous S1 ≥ 0.5 and S1 this time < 0.5, otherwise 0	A	B	C	D	No	Yes
EDGEm	m=1 to 8	Change detection	Input value	A	B	C	D	1 when (previous S1 < 0.5 and S1 this time ≥ 0.5) or (previous S1≥ 0.5 and S1 this time < 0.5), otherwise 0	A	B	C	D	No	Yes

Instruction	Explanation			Before Instruction Execution					After Instruction Execution					Overflow	Backup
				S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
TIMm	m=1 to 8	Timer (second)		Timer ON/OFF	A	B	C	D	Time elapsed Count continues from 0 when 8.0 (8000 seconds) is reached.	A	B	C	D	No	Yes
TIMMm	m=1 to 8	Timer (minute)		Timer ON/OFF	A	B	C	D	Time elapsed Count continues from 0 when 8.0 (8000 minutes) is reached.	A	B	C	D	No	Yes
TUPm	m=1 to 8	Time out (second)		Setting time (second)	Timer ON/OFF	A	B	C	1 output for 1 period at a time out	A	B	C	C	No	Yes
TUPMm	m=1 to 8	Time out (minute)		Setting time (minute)	Timer ON/OFF	A	B	C	1 output for 1 period at a time out	A	B	C	C	No	Yes
PGMm	m=1 to 2	Program setter (second)		Reset input	Calculation execution/hold	Reset value	A	B	Program output value	End flag	A	B	B	No	Yes
PGMMm	m=1 to 2	Program setter (minute)		Reset input	Calculation execution/hold	Reset value	A	B	Program output value	End flag	A	B	B	No	Yes
PICm	m=1 to 8	Pulse input counter		Counter start/ reset	Input value	A	B	C	Count value	A	B	C	C	No	Yes
CPOm	m=1 to 2	Totalizer pulse output		Totalization rate	Input value	A	B	C	Input value	A	B	C	C	Yes	Yes
HALm	m=1 to 8	High limit alarm		Hysteresis	Alarm setpoint	Input value	A	B	1 when S3≥S2(or S2-S1) 0 when S3<S2(or S2-S1)	Input	A	B	B	Yes	Yes
LALm	m=1 to 8	Low limit alarm		Hysteresis	Alarm setpoint	Input value	A	B	1 when S2(or S2+S1)≥S3 0 when S2(or S2+S1)<S3	Input	A	B	B	Yes	Yes
SQAm	m=1 to 8	Square root extraction	Linear at low cutoff point or lower 0.2% hysteresis at low cutoff point	Low cutoff setpoint	A	B	C	D	√A when S2>S1 A when S2≤S1	B	C	D	D	Yes	Yes
SQBm	m=1 to 8	Square root extraction	0 at low cutoff point or lower 0.2% hysteresis at low cutoff point	Low cutoff setpoint	A	B	C	D	√A when S2>S1 0 when S2≤S1	B	C	D	D	Yes	Yes
RSFFm	m=1 to 8	RS flip-flop		RESET	SET	A	B	C	1 when SET≥0.5 0 when SET<0.5, RESET≥0.5 Otherwise, same as previous output	A	B	C	C	No	Yes
HTIMm	m=1 to 8	Hold timer (second)		Timer reset	Timer RUN/HOLD	A	B	C	Time elapsed, hold at 4096.000 (4096000 seconds)	A	B	C	C	No	Yes
HTIMMm	m=1 to 8	Hold timer (minute)		Timer reset	Timer RUN/HOLD	A	B	C	Time elapsed, hold at 4096.000 (4096000 minutes)	A	B	C	C	No	Yes
DELAYm	m=1 to 8	Previous input variable		A	B	C	D	E	Previous A	B	C	D	E	No	Yes
HOLDm	m=1 to 8	Hold		HOLD/ THROUGH	Input	A	B	C	Previous output when S1 ≥ 0.5, input this time when S1 < 0.5	A	B	C	C	No	Yes
AND		AND		A	B	C	D	E	1 when S1≥0.5 and S2≥0.5 0 when S1<0.5 or S2<0.50	C	D	E	E	No	No
OR		OR		A	B	C	D	E	1 when S1≥0.5 or S2≥0.5 0 when S1<0.5 and S2<0.5	C	D	E	E	No	No
NOT		NOT		A	B	C	D	E	0 when S1≥0.5 1 when S1<0.5	B	C	D	E	No	No
EOR		Exclusive OR		A	B	C	D	E	0 when S1, S2≥0.5 or S1, S2<0.5 1 when only one condition is 0.5 or more	C	D	E	E	No	No
MAND		Multi-input AND		A	B	C	D	E	1 when A to D are all ≥ 0.5, 0 when even one of A to D < 0.5	E	E	E	E	No	No
MOR		Multi-input OR		A	B	C	D	E	1 when even one of A to D is ≥ 0.5, 0 when A to D are all < 0.5	E	E	E	E	No	No
MEOR		Multi-input exclusive OR		A	B	C	D	E	0 when A to D are all ≥ 0.5 or A to D are all < 0.5, otherwise 1	E	E	E	E	No	No

Instruction	Explanation		Before Instruction Execution					After Instruction Execution					Overflow	Backup
			S1	S2	S3	S4	S5	S1	S2	S3	S4	S5		
CMP		Comparison	A	B	C	D	E	1 when S1≤S2 0 when S1>S2	B	C	D	E	No	No
SW		Signal switching	0/1	A	B	C	D	A when S1≥0.5 B when S1<0.5	C	D	D	D	No	No
GE		≥, Greater or equal	A (comparison value)	B (compared value)	C	D	E	1 when B ≥ A, 0 when B < A	B	C	D	E	No	No
GT		>, Greater than	A (comparison value)	B (compared value)	C	D	E	1 when B > A, 0 when B ≤ A	B	C	D	E	No	No
LE		≤, Less or equal	A (comparison value)	B (compared value)	C	D	E	1 when B ≤ A, 0 when B > A	B	C	D	E	No	No
LT		<, Less than	A (comparison value)	B (compared value)	C	D	E	1 when B < A, 0 when B ≥ A	B	C	D	E	No	No
INRNG		In range	A (comparison value, large)	B (comparison value, small)	C (compared value)	D	E	1 when B ≤ C ≤ A, 0 when C < B, A < C	C	D	E	E	No	No
OUTRNG		Out of range	A (comparison value, large)	B (comparison value, small)	C (compared value)	D	E	1 when C ≤ B, A ≤ C, 0 when B < C < A	C	D	E	E	No	No
GO @<label>		Jump	A	B	C	D	E	A	B	C	D	E	No	No
GIF @<label>		Conditional jump	0/1	A	B	C	D	A	B	C	D	D	No	No
GOSUB @<sub-program name>		Jump to sub-program	A	B	C	D	E	A	B	C	D	E	No	No
GIFSUB @<sub-program name>		Sub-program conditional branch	0/1	A	B	C	D	A	B	C	D	D	No	No
SUB @<sub-program name>		Sub-program startup	A	B	C	D	E	A	B	C	D	E	No	No
RTN		Sub-program termination	A	B	C	D	E	A	B	C	D	E	No	No
CHG		S register change	A	B	C	D	E	B	A	C	D	E	No	No
ROT		S register rotation	A	B	C	D	E	B	C	D	E	A	No	No
BSCm	m=1 to 2	Basic control	PV	A	B	C	D	Control output	A	B	C	D	No	No
CSC		Cascade control	PV2	PV1	A	B	C	Control output	A	B	C	C	No	No
SSC		Selector control	PV2	PV1	A	B	C	Control output	A	B	C	C	No	No

1: YES/NO depends on a register.
2: STE xx is backed up in the same way as ST xx.

Overflow: The overflow existence of a computation result is denoted.
Backup: The backup existence of a computation result (last value) at the time of power failure is denoted.

Blank Page

Index

Symbol

Numeric

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Y

Z

Revision Information

- Title : YSS1000 Setting Software for YS1000 Series/YS1700 Programmable Function User's Manual
- Manual No. : IM 01B08K01-02E

Mar. 2007/1st Edition
Newly published

Mar. 2007/2nd Edition
Error correction

Sep. 2007/3rd Edition
Addition of old model data conversion function and components function, functional improvement, and error correction

Mar. 2008/4th Edition
Addition of Check Support Function, functional improvement, and error correction

Aug. 2009/5th Edition
Error correction

■ Written by Yokogawa Electric Corporation
■ Published by Yokogawa Electric Corporation
2-9-32 Nakacho, Musashino-shi, Tokyo 180-8750, JAPAN

Blank Page

YOKOGAWA

YOKOGAWA ELECTRIC CORPORATION

Headquarters
2-9-32, Nakacho, Musashino-shi, Tokyo, 180-8750 JAPAN

Branch Sales Offices
Nagoya, Osaka, Hiroshima, Fukuoka, Sendai, Ichihara, Toyota, Kanazawa, and Kitakyusyu.

YOKOGAWA CORPORATION OF AMERICA

2 Dart Road, Newnan, Georgia 30265-1094, U.S.A.
Phone : 1-800-888-6400
Fax : 1-770-254-0928

YOKOGAWA EUROPE B. V.

Euroweg 2, 3825 HD Amersfoort, THE NETHERLANDS
Phone : 31-88-464-1000 Fax : 31-88-464-1111

Branch Sales Offices / Wien (Austria), Zaventem (Belgium), Ratingen (Germany), Madrid (Spain), Runcorn (United Kingdom), Milano (Italy), Cinisello Balsamo (Italy), Velizy-Villacoublay (France), Budapest (Hungary), Stockholm (Sweden), Sola (Norway), Warszawa (Poland), Vila Nova de Gaia (Portugal), Bucharest (Romania), Dublin (Ireland)

YOKOGAWA AMERICA DO SUL LTDA.

Praca Acapulco, 31 - Santo Amaro. Sao Paulo/SP - BRAZIL
Phone : 55-11-5681-2400 Fax : 55-11-5681-4434

YOKOGAWA ENGINEERING ASIA PTE. LTD.

5 Bedok South Road, 469270 SINGAPORE
Phone : 65-6241-9933 Fax : 65-6241-2606

YOKOGAWA ELECTRIC KOREA CO., LTD.

14-1, Yangpyongdong-4Ga, Youngdeungpo-Gu, Seoul, 150-866 KOREA
Phone : 82-2-2628-6000 Fax : 82-2-2628-6400

YOKOGAWA AUSTRALIA PTY. LTD.

Tower A, 112-118 Talavera Road, Macquarie Park,
N.S.W.2113, AUSTRALIA
Phone : 61-2-8870-1100 Fax : 61-2-8870-1111

YOKOGAWA INDIA LTD.

Plot No.96 Electronic City Complex, Hosur Road, Bangalore 560100, INDIA
Phone : 91-80-4158-6000 Fax : 91-80-2852-0625

YOKOGAWA CHINA CO., LTD.

3F TowerD Cartelo Crocodile Building
No.568 West Tianshan Road, Shanghai 200335, CHINA
Phone : 86-21-62396262 Fax : 86-21-62387866

Sep. '09

Printed in Korea