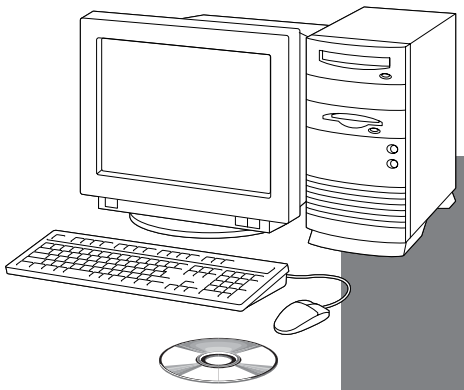
YAMATAKE

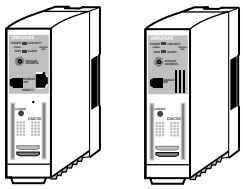
No. CP-SP-1130E

Module Type Controller DMC50

User's Manual

Function Block Reference





Thank you for purchasing the Module Type Controller DMC50.
This manual contains information for ensuring correct use of the DMC50.
This manual should be read by those who design a control system using the DMC50.
Be sure to keep this manual nearby for handy reference.

Yamatake Corporation

RESTRICTIONS ON USE

This product has been designed, developed and manufactured for general-purpose application in machinery and equipment.
Accordingly, when used in applications outlined below, special care should be taken to implement a fail-safe and/or redundant design concept as well as a periodic maintenance program.

- Safety devices for plant worker protection
- Start/stop control devices for transportation and material handling machines
- Aeronautical/aerospace machines
- Control devices for nuclear reactors

Never use this product in applications where human safety may be put at risk.

REQUEST

Ensure that this User's Manual is handed over to the user before the product is used.

Copying or duplicating this User's Manual in part or in whole is forbidden. The information and specifications in this User's Manual are subject to change without notice.

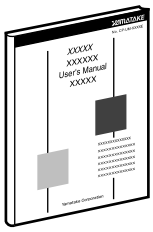
Considerable effort has been made to ensure that this User's Manual is free from inaccuracies and omissions.

If you should find any inaccuracies or omissions, please contact Yamatake Corporation.

In no event is Yamatake Corporation liable to anyone for any indirect, special or consequential damages as a result of using this product.

The Role of This Manual

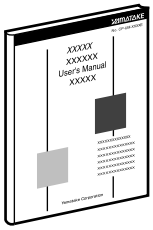
Seven different manuals in total are available for DMC50. Read each manual according to your specific requirements. Following is a brief outline of each of the manuals.
To obtain manuals, contact Yamatake Corporation or your Yamatake dealer.



Module Type Controller DMC50 User's Manual "Installation and Configuration" **Manual No. CP-SP-1139E**

Thoroughly read this manual before designing or manufacturing the equipment hardware using DMC50.

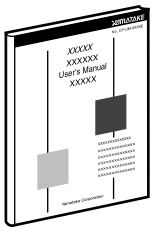
This manual consists of two parts, one for the control module and the other for the communication module. It describes the installation, wiring, specifications, and hardware troubleshooting of those controller.



Module Type Controller DMC50 User's Manual "QuickStart" **Manual No. CP-SP-1092E**

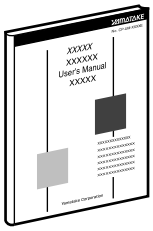
The user who operates DMC50 for the first time must read this manual thoroughly. This manual is intended to help the user understand the overview of the controller operation and basic operating procedures.

The manual describes the operation with various examples. Read this manual while using the smart loader package.



Module Type Controller DMC50 User's Manual "Communications Connection" **Manual No. CP-SP-1093E**

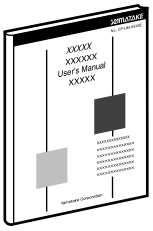
The user who uses the communication facilities of DMC50 must read this manual thoroughly. This manual describes the communication facilities of this controller, such as CPL communication and Ethernet communication.



Module Type Controller DMC50 User's Manual "Function Block Reference" **Manual No. CP-SP-1130E**

This Manual.
Read this manual when the user designs a control system most suitable for the user's application by utilizing DMC50.

This manual describes the specifications of ISaGRAF functions and function blocks essential to design a desired control system.

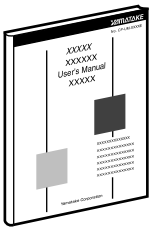


Module Type Controller DMC50 User's Manual " Application Developer's Guide" **Manual No. CP-SP-1134E**

This manual describes how to write practical application programs for the DMC50. The person in charge of programming is encouraged to read this manual thoroughly.

This manual describes the pattern function blocks, utilization of ISaGRAF, and application examples.

This manual is intended for the user who has already read the separate manual "QuickStart" and "Function Block Reference" thoroughly to fully understand their contents.



Smart Loader Package SLP-D50 for Module Type Controller DMC50
User's Manual **Manual No. CP-SP-1122E**

This manual describes the operations and features of the smart loader package SLP-D50 for DMC50 as well as its installation into a personal computer. It also writes about important points collaborating with ISaGRAF to build an application for the controller.

SLP-D50J50 Installation Guide **Manual No. CP-UM-5259E**

This manual describes how to install the smart loader package SLP-D50 for the DMC50 into a personal computer.

Organization of This User's Manual

This manual is organized as follows:

Chapter 1. CALCULATION PARAMETERS AND CALCULATION MONITOR PARAMETERS

This chapter describes the setup items and the contents of the Calculation Parameters and Calculation Monitor Parameters.

Chapter 2. STANDARD OPERATORS / FUNCTIONS / FUNCTION BLOCKS

This chapter describes each of the standard operators, functions, and function blocks. The index list by functionality is provided on pages 2-1 to 2-7. This index list should be used when the user searches for standard operators, function, or function block by functionality.

Appendix

This appendix describes the data types and auto-tuning that the user should refer to when using a function block.

Contents

Appendix 1. ISaGRAF VARIABLES AND PARAMETER DATA TYPES

Appendix 2. AUTO TUNING

Conventions Used in This Manual

The following conventions are used in this manual:

Handling Precaution : Handling Precautions indicate items that the user should pay attention to when handling the DMC50.

☞ : This indicates the item or page that the user is requested to refer to.

Note : Notes indicate useful information that the user might benefit by knowing.

(1), (2), (3) : The numbers with the parenthesis indicate steps in a sequence or indicate corresponding parts in an explanation.

Terms Used in This Manual

The following terms are used in this manual:
In addition to the following terms, terms related to the ISaGRAF are used without prior permission:
For details about terms related to the ISaGRAF,
☞refer to the User's Guide, ISaGRAF version 3.4.

Application: Application (code) is the execution form of a project for the programmable controller (DMC50 control module). Normally, an application consists of an application program and Parameters.

Application program:
Application program is an aggregate of the programs to be executed in one module (programming units written according to the language specifications IEC61131-3 are combined). An application program is created using ISaGRAF.

Controller: A module of the DMC50 is referred to as controller in this manual.

CTRL module: CTRL module is a control module of the module type controller DMC50.

DMC50: DMC50 is a module type controller. This DMC50 is an all-in-one controller, that is, I/O is built in the module. Additionally, a combination of one or more control module (CH200, CH400, CS200, CS400) with one communication module (ME200, MR200) makes it possible to communicate with external systems.

FB: FB is an abbreviation of the ISaGRAF function block.

FUNC: FUNC is an abbreviation of the ISaGRAF function.

Function: Function is a calculation block without internal state, which is used for the programming in ISaGRAF.

Function block:
Function block is a calculation block with internal state, which is used for the programming in ISaGRAF.

Parameter: Parameter is the general name of various settings that affect the operation status of the controller and monitor data showing the operation status. Parameters are classified into System Parameters, Calculation Parameters, Program Pattern Parameters, User-defined Parameters, System Monitor Parameters, and Calculation Monitor Parameters. (The Parameter type definition of the User- defined Parameter is expressed as User-defined type in this manual and SLP-D50.) Parameters are set and monitored using SLP-D50. Note that ISaGRAF uses the term "parameters" with other meanings. (Arguments of FB and FUNC, and I/O board parameters)

Program: Program is a programming unit and is executed at the top level of the ISaGRAF execution cycle.

Project: Project is an entity that manages the settings related to one controller. Actually a project is a directory. The last part of the absolute path of the directory is the project name. Application program source files and Parameter settings are stored under this directory.

Type label: Type label is the label name of a Parameter type shown in the tree view of the "Project" window.

Chapter 1. CALCULATION PARAMETERS AND CALCULATION MONITOR PARAMETERS

1 - 1 Parameter Lists

● Calculation Parameters

Type label	Description
PID_A Options	This setup is necessary to use the PID_A (standard PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method are changed.
PID_A Constants	This setup is necessary to use the PID_A (standard PID control) function block. The constants necessary for the control calculation, such as PID constant and output limit values are set and changed.
PID_CAS Options	This setup is necessary to use the PID_CAS (cascade PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method are changed.
PID_CAS Constants	This setup is necessary to use the PID_CAS (cascade PID control) function block. The constants necessary for the control calculation, such as PID constant and output limit values are set and changed. The constants are individually set on the master and slave sides.
Ra_PID Options	This setup is necessary to use the Ra_PID (RationaLOOP PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method are changed.
Ra_PID Constants	This setup is necessary to use the Ra_PID (RationaLOOP PID control) function block. The constants necessary for the control calculation, such as PID constant and output limit values are set and changed.
UP_PID Options	This setup is necessary to use the UP_PID (use-point PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method are changed.
UP_PID Constants	This setup is necessary to use the UP_PID (use-point PID control) function block. The constants necessary for the control calculation, such as PID constant and output limit values are set and changed.
TBL/TBR Setup	This setup is necessary to use the TBL (linearization table lookup) and TBR (linearization table reverse lookup) function blocks. The points of the linearization table are set and changed.

● Calculation Monitor Parameters

Type label	Description
PID_A Monitor	The status, such as PV, SP, MV, or mode can be checked when using the PID_A (standard PID control) function block. However, the values, such as SP cannot be set or changed.
PID_CAS Monitor	The status, such as PV, SP, MV, or mode can be checked when using the PID_CAS (cascade PID control) function block. However, the values, such as SP cannot be set or changed.
Ra_PID Monitor	The status, such as PV, SP, MV, or mode can be checked when using the Ra_PID (RationaLOOP PID control) function block. However, the values, such as SP cannot be set or changed.
UP_PID Monitor	The status, such as PV, SP, MV, or mode can be checked when using the UP_PID (use-point PID control) function block. However, the values, such as SP cannot be set or changed.

● User-defined Parameters

The user freely defines Parameters and can use them.
For details about how to use the User-defined Parameters,
☞refer to the Smart Loader Package SLP-D50 for Module Type Controller DMC50 User's Manual, CP-SP-1122E.

● Program Pattern Parameters

These Parameters are necessary Parameters when using a pattern generation function block. For details about how to use the Program Pattern Parameters,
☞refer to the Module Type Controller DMC50 User's Manual "Application Developer's Guide", CP-SP-1134E.

1 - 2 Calculation Parameters

■ PID_A Options

This setup is necessary to use the PID_A (standard PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method can be changed.

Parameter type ID: 201h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Control Action	0: Reverse Action 1: Direct Action	0	0		DINT
2	PV Range Min	–99999.9 to +99999.9	0.0	0	data for PID operation are calculated based on these MIN/MAX values of a range.	REAL
3	PV Range Max	–99999.9 to +99999.9	1000.0	0		
4	Initialization on SP changes	0: Automatic 1: Initialized on SP Changes 2: Not initialized	0	0		DINT
5	Initial MV Value	–10.0 to +110.0%	0.0%	0		REAL
6	MV Rate-of-change limit	0.0 to 100.0%	0.0%	0		
7	AUTO/MANUAL Transfer	0: Bump-less transfer 1: Preset manual value	0	0		DINT
8	Preset Manual Value	–10 to +110.0%	0.0%	0	This setting is valid when item ID 7 [AUTO/MANUAL transfer] is set at " setting 1 (Preset manual value)".	REAL
9	Smart Tuning	0: No smart tuning 1: Overshoot suppression 2: Overshoot suppression+	0	0		DINT
10	Auto Tuning Method	0: Standard AT 1: AT for overshoot suppression 2: NNAT 3: AT for regulatory control	0	0		
11	2 Degrees of Freedom PID	0: 2 degrees of freedom PID control disabled. 1: 2 degrees of freedom PID control enabled.	0	0		BOOL
12	Dead Band	0.0 to 10.0%FS	0.0%FS	0		REAL

● Item ID = 1 [Control Action]

This setting is used to specify a control action direction.

0	Reverse action
1	Direct action

The reverse action means that the MV value is decreased as the PV (control value) is increased.
The direct action means that the MV value is increased as the PV is increased.
Generally, the reverse action is selected for the heating control while the direct action is selected for the cooling operation.

● Item ID = 2 [PV Range Min]

● Item ID = 3 [PV Range Max]

Set the PV range in relevant industrial unit.
Full scale of the following percentage data is determined by these MIN/MAX values:

- Proportional band of PID_A calculation and proportional band for disturbance rejection
- Dead band

● **Item ID = 4 [Initialization on SP Changes]**

Normally, this setting item is used at "factory setting".
This setting is used to set the PID control initialization method if any of the following conditions is satisfied:

- The SP input parameter is changed.
- The PID input parameter (group ID of PID_A Options) is changed.

This setting works to prevent the PID control output from being excessive if the SP is changed. If the setting is set at "0" (Automatic), it is judged automatically whether to initialize or not by monitoring the amount of absolute deviation of the PV value before changing the SP.

Note

If any of the following conditions is satisfied, an optimal PID control initialization is performed internally regardless of this setting:
- The function block is initially executed (including power ON).
- The previous ISaGRAF mode is a mode other than the real-time mode (RT).
- The previous E_OK output parameter equals "ERR".
- The PARA input parameter (group ID of PID_A Options) is changed.
- The AT process is completed or stopped.
- The cycle time is changed.
- The control action (reverse action or direct action) is changed.
- The control mode is changed from MANUAL to AUTO.

Handling Precaution

If the value that is changed at each execution cycle (RAMP SP* or RSP*) is used for the SP input, set this setting item to "setting 2 (Not initialized)" to prevent unnecessary initialization of the PID control.
*: The RAMP SP is SP that the value is ramp-changed by the ISaGRAF program as the time elapses. Additionally, the RSP is SP based on the analog input.

● **Item ID = 5 [Initial MV Value]**

If any of the following conditions is satisfied, the PID control is started from this initial MV value:
- The function block is initially executed (including power ON).
- The previous ISaGRAF mode is a mode other than the real-time mode (RT).
- The previous E_OK output parameter equals "ERR".
- The PARA input parameter (group ID of PID_A Options) is changed.
- The AT process is completed or stopped.
- The cycle time is changed.
- The control action (reverse action or direct action) is changed.

● **Item ID = 6 [MV Rate-of-change Limit]**

Normally, this setting item is used at "factory setting".
This setting item works only in the AUTO mode.
Changes in MV value output at every control cycle (stored in the cycle timing setting) are limited to the set value or less.
As this value is smaller, the allowable change is getting smaller.
Additionally, if this setting is set at "0.0", this means "No-limit".
If the rapid change of the MV value may adversely affect the actuators, 0.0 should not be specified.

● **Item ID = 7 [AUTO/MANUAL Transfer]**

Select an MV value (MV_IN input parameter) when the control mode is changed from AUTO to MANUAL.

0	Bump-less transfer The MV value at the time of transfer is retained.
1	Preset manual value The set preset manual value is output.

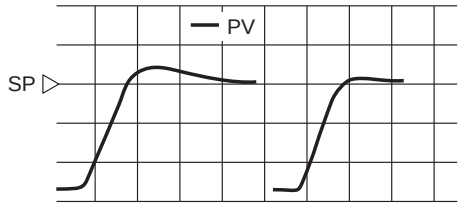
When the mode is returned from the MANUAL mode to the AUTO mode, the control calculation is started from the MV value (MV_IN input parameter) at that time.

● **Item ID = 9 [Smart Tuning]**

This smart tuning capability suppresses the overshoot in the reverse action and the undershoot in the direct action.
Two functionalities are provided on the smart turning capability, One is "Overshoot suppression" to perform the suppression control itself, and another is "Self-learning brake value" to automatically readjust the amount of suppression.

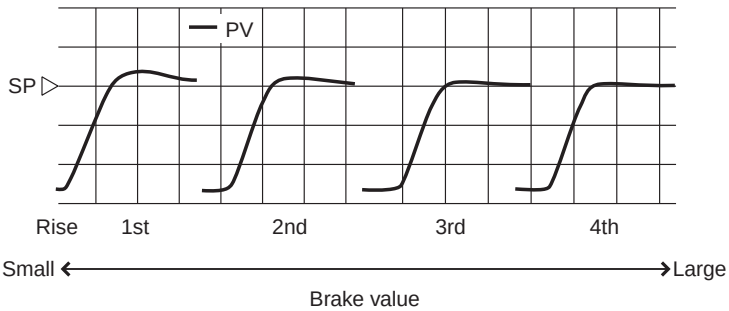
0	No smart tuning
1	Overshoot suppression The overshoot is suppressed with the brake value fixed.
2	Overshoot suppression + (Overshoot suppression with self-learned brake value) The overshoot becomes suppressed by readjusting the brake value automatically.

”Overshoot suppression” functionality uses the Brake value of the PID_A Constants to suppress the overshoot. (As the brake value becomes larger, stronger overshoot suppression effect is obtained.)



No smart tuning. Smart tuning enabled.

”Self-learning brake value” functionality reviews the Brake value every rise (for reverse action) or fall (for direct action). The overshoot becomes suppressed by readjusting the brake value automatically. This review is performed only in the direction that the brake value becomes large.



● **Item ID = 10 [Auto Tuning method]**

Select a calculation method for the auto tuning control.

0	Standard AT Calculation method for general process.
1	AT for overshoot suppression Calculation method for process that overshoot occurs easily.
2	NNAT This calculation method is intended for process that is wider than assumed by methods 1 and 2, and is neural-net processed. Use of the smart tuning (brake) is required for this calculation method.
3	AT for regulatory control Calculation method for process having only regulatory control without SP changes.

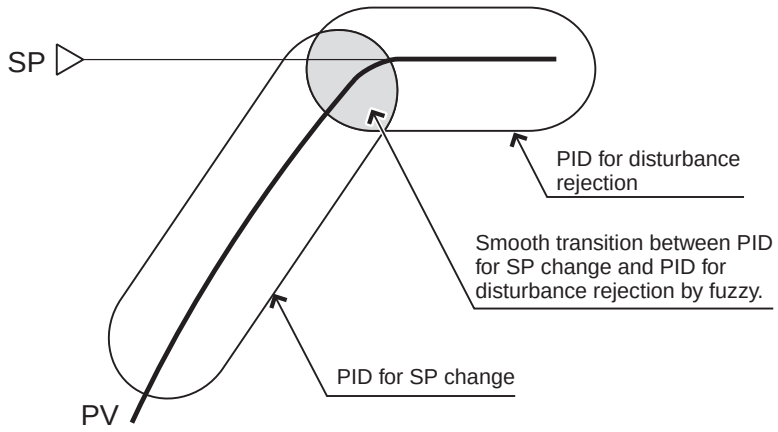
Note

The PID constants to be updated include the PID constants for disturbance rejections.

For details about the auto tuning operation diagram, stop conditions, and cautions, ☞refer to Appendix 2, Auto Tuning.

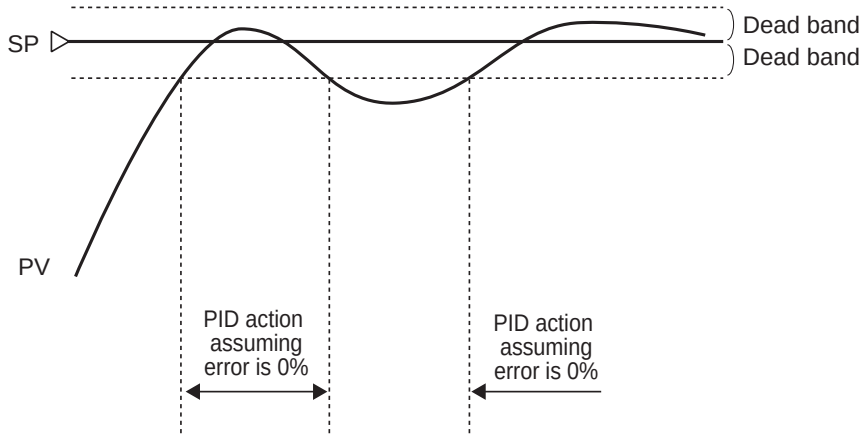
● **Item ID = 11 [2 Degrees of Freedom PID]**

"2 degrees of freedom PID control" capability is used to improve the rejection ability to the disturbance at steady state conditions without sacrificing the SP follow-up characteristics.
That is, the PID constants optimal for the SP change and those optimal for the disturbance rejection are set individually. The controller determines which PID constants should be used, and automatically change them.
When changing the PID constants, the fuzzy rule is used to prevent the control from fluctuating. The PID constants for SP change and those for disturbance rejection are automatically set by the auto tuning. However, the settings can also be changed manually.



● **Item ID = 12 [Dead Band]**

Normally, this setting item is used at "factory setting".
The dead band setting forces the PID control to operate with the deviation of 0% if the absolute value of the deviation (%FS) is the set value or less.
When the deviation vibrates a lot at around 0% because PV includes a large noise or because the actuator valve has hysteresis, this setting is used to prevent such vibration from affecting the control.
For example, when the PV range is 0 to 400˚C and Dead Band is 1.0%FS, PID action assuming the error is 0% is performed if the deviation is within ±4.0˚C.



■ PID_A Constants

This setup is necessary to use the PID_A (standard PID control) function block. The constants necessary for the control calculation, such as PID constants and output limit values can be set and changed.

Parameter type ID: 202h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Proportional Band	0.1 to 1000.0%	100.0%	0		REAL
2	Integral Time	0.00 to 6000.00s	0.00 s	0		
3	Derivative Time	0.00 to 6000.00s	0.00 s	0		
4	Output Low Limit for I-Action	–1000.0 to +1000.0%	0.0%	0		
5	Output High Limit for I-Action	–1000.0 to +1000.0%	100.0%	0		
6	Output Low Limit	–1000.0 to +1000.0%	0.0%	0	The limits do not function in the MANUAL mode.	
7	Output High Limit	–1000.0 to +1000.0%	100.0%	0		
8	Manual Reset	0.0 to 100.0%	50.0%	0	The setting is effective only when the integral time is 0.00 sec.	
9	Brake	0.0 to 30.0	0.0	0	The setting is effective only when using the smart tuning capability.	
10	P-band for Disturbance Rejection	0.1 to 1000.0%	100.0%	0	The setting is effective only when using the "2 degrees of freedom PID control" capability.	
11	I-time for Disturbance Rejection	1.00 to 6000.00s	120.00 s	0		
12	D-time for Disturbance Rejection	0.00 to 6000.00s	0.00 s	0		

- **Item ID = 1 [Proportional Band]**
- **Item ID = 2 [Integral Time]**
- **Item ID = 3 [Derivative Time]**

  • These are the "Proportional band (P)", "Integral time (I)", and "Derivative time (D)" settings of the control constants. When using the auto tuning, these settings are automatically set corresponding to the characteristics of the controlled process. If optimal values are already known or if it is difficult that the auto tuning becomes effective, make the settings manually.
  • When using the "2 degrees of freedom PID control" capability, these settings become the control constants that are used when changing the SP.
  • Full scale range of the P operation are determined by [PV Range Min/ PV Range Max] of the PID_A Options.
  • When the integral time is set at "0", it is treated as no-integral operation.

- **Item ID = 4 [Output Low Limit for I-Action]**
- **Item ID = 5 [Output High Limit for I-Action]**

  Normally, these setting items are used at "factory setting".
  It is possible to limit the integral action only, besides [Output High Limit/Output Low Limit]. If the MV value reaches the high or low limit, the integral action will not function.

- **Item ID = 6 [Output Low Limit]**
- **Item ID = 7 [Output High Limit]**

These settings are used to control the high and low limits of the MV value. The limits work even when the auto tuning is being operated. However, the limits do not work in the MANUAL mode.

Handling Precaution

Even though the values stored in [Output High Limit/Output Low Limit] are changed, the values stored in [Output High Limit for I-action/Output Low Limit for I-action] are not changed automatically. The values stored in [Output High Limit for I-action/Output Low Limit for I-action] are not affected by the changed values in [Output High Limit/Output Low Limit]. Therefore, the values stored in [Output High Limit for I-action/Output Low Limit for I-action] need to be changed when necessary.

- **Item ID = 8 [Manual Reset]**

This setting is used when the integral time is set at "0". This setting is used to remove the offset that occurs during proportional action (without integral action). Assign an MV value suitable for deviation "0".

- **Item ID = 9 [Brake]**

This setting is needed when using the [Smart Tuning].
As the brake value is made larger, the effect of the overshoot suppression becomes higher, but the rise time becomes longer. When using the brake value self-learning functionality, this value is updated automatically according to the learning results.

- **Item ID = 10 [P-band for Disturbance Rejection]**
- **Item ID = 11 [I-time for Disturbance Rejection]**
- **Item ID = 12 [D-time for Disturbance Rejection]**

• These settings are needed when using the "2 degrees of freedom PID control" capability.

• These are the "Proportional band (P)", "Integral time (I)", and "Derivative time (D)" settings of the control constants for disturbance rejection. When using the auto tuning, these settings are automatically set corresponding to the characteristics of the controlled process.

• When “Integral Time” is set at "0", the control for the SP change (at rise) and disturbance response are performed without integral action, regardless of the setting of the I-time for Disturbance Rejection.

■ PID_CAS Options

This setup is necessary to use the PID_CAS (cascade PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method can be changed.

Parameter type ID: 211h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Control Action	0 to 3	0	0	0: Reverse action for master and slave 1: Direct action for master and reverse action for slave 2: Reverse action for master and direct action for slave 3: Direct action for master and slave	DINT
2	PV Range Min (master)	–99999.9 to +99999.9	0.0	0	The settings are used for the range calculation of the master PID control.	REAL
3	PV Range Max (master)	–99999.9 to +99999.9	1000.0	0		
4	PV Range Min (slave)	–99999.9 to +99999.9	0.0	0	The settings are used for the range calculation of the slave PID control.	
5	PV Range Max (slave)	–99999.9 to +99999.9	1000.0	0		
6	Initialization on SPchanges (master)	0: Automatic 1: Initialized on SP changes 2: Not initialized (method 1) 3: Not initialized (method 2)	0	0		DINT
7	Initialization on SP Changes (slave)	0: Automatic 1: Initialized on SP changes 2: Not initialized 3: For extension (reserved)	0	0	The setting is effective only in the LOCAL mode.	
8	Initial MV Value (master)	–10.0 to +110.0%	0.0%	0	Initial MV value for master PID control.	REAL
9	Initial MV Value (slave)	–10.0 to +110.0%	0.0%	0	Initial MV value for slave PID control.	
10	MV Rate-of-Change Limit (master)	0.0 to 100.0%	0.0%	0		
11	MV Rate-of-ChangeLimit (slave)	0.0 to 100.0%	0.0%	0		
12	AUTO/MANUAL Transfer	0:Bump-less transfer 1:Preset manual value	0	0	AUTO/MANUAL operation acts on the slave.	DINT
13	Preset Manual Value	–10.0 to +110.0%	0.0%	0	This setting is effective only when item ID 12 [AUTO/MANUAL transfer] is set at "setting 1 (Preset manual value)".	REAL
14	MV in READY Mode	–10.0 to +110.0%	0.0%	0		
15	Auto Tuning Method (master)	0: Standard AT 1: AT for overshoot suppression 2: For extension (reserved) 3: AT for regulatory control	0	0		DINT
16	Auto Tuning Method (slave)	0: Standard AT 1: AT for overshoot suppression 2: For extension (reserved) 3: AT for regulatory control	0	0		
17	2 Degrees of Freedom PID	0 to 3	0	0	0: Disabled 1: Enabled for master only 2: Enabled for slave only 3: Enabled for master and slave	
18	Dead Band (master)	0.0 to 10.0%FS	0.0%FS	0	The setting is intended for the master PID control insensitive to a range of errors.	REAL
19	Dead Band (slave)	0.0 to 10.0%FS	0.0%FS	0	The setting is intended for the slave PID control insensitive to a range of errors.	

● Item ID = 6 [Initialization on SP Changes (master)]

Normally, this setting item is used at "factory setting".
This setting is used to set the PID control initialization method for the time when any of the following conditions is satisfied:

- The M_SP input parameter (SP on the master) is changed.
- The M_PID input parameter (group ID of PID_CAS Constants) is changed.
- The SP mode is changed from LOCAL to REMOTE.

This setting works to prevent the PID control output from being excessive if the SP on the master is changed. If the setting is set at "0" (Automatic), it is judged automatically whether to initialize or not, by monitoring the amount of absolute deviation of the PV value before the SP is changed.

Note

If any of the following conditions is satisfied, an optimal PID control initialization on the master is performed internally, regardless of this setting:

- The function block is initially executed (including power ON).
- The previous ISaGRAF mode is a mode other than the real-time mode (RT).
- The previous E_OK output parameter equals "ERR".
- The execution mode is changed from READY to RUN.
- The PARA input parameter (group ID of PID_CAS Options) is changed.
- The auto tuning on the master is completed or stopped.
- The cycle time is changed.
- The control action (reverse action or direct action) is changed.
- The control mode is changed from MANUAL mode to AUTO mode.

Handling Precaution

- If the value that is changed at each execution cycle (RAMP SP* or RSP*) is used for the M_SP input (SP on the master), set this setting to "setting 2 (Not initialized (method 1))" to prevent unnecessary initialization.
  * The RAMP SP is SP that the value is ramp-changed by the ISaGRAF program as the time elapses. Additionally, the RSP is SP based on the analog input.
- If the SP mode is changed from LOCAL to REMOTE in a situation that a deviation occurs in the master, "setting 1 (Initialized on SP changes)" performs the initialization so that bump occurs in the MV value when compared to "setting 0 (Automatic)" and "setting 2 (Not initialized (method 1))" : initialization by considering the immediate response. To change the MV value more smoothly, select "setting 0" or "setting 2".
- "Setting 3 (Not initialized (method 2)" is intended for extension. Normally, this setting must not be used.

● **Item ID = 7 [Initialization on SP Changes (slave)]**

Normally, this setting item is used at "factory setting".
This setting is used to set the PID control initialization method for the time when any of the following conditions is satisfied:

- The S_LSP input parameter (LSP on the slave) is changed in the LOCAL mode.
- The S_PID input parameter (group ID of PID_CAS Constants) is changed in the LOCAL mode.
- The SP mode is changed from REMOTE to LOCAL.

This feature prevents the PID control output from being excessive when the LSP on the slave is changed in the LOCAL mode. If the setting is set at "0" (Automatic), it is judged automatically whether to initialize or not by monitoring the amount of absolute deviation of the PV value before the SP is changed.

Note

- In the REMOTE mode, an optimal initialization is performed internally regardless of this setting.
- If any of the following conditions is satisfied, an optimal PID control initialization on the slave is performed internally regardless of this setting:
  - The function block is initially executed (including power ON).
  - The previous ISaGRAF mode is a mode other than the real-time mode (RT).
  - The previous E_OK output parameter equals "ERR".
  - The execution mode is changed from READY to RUN.
  - The PARA input parameter (group ID of PID_CAS Options) is changed.
  - The AT process on the slave is completed or stopped.
  - The cycle time is changed.
  - The control action (reverse action or direct action) is changed.
  - The control mode is changed from MANUAL to AUTO.

Handling Precaution

- If the value that is changed at each execution cycle (RAMP SP* or RSP*) is used for the S_LSP input (LSP on the slave) in the LOCAL mode, set this setting item to "setting 2 (Not initialized)" to prevent unnecessary initialization.
  * The RAMP SP is SP that the value is ramp-changed by the ISaGRAF program as the time elapses. Additionally, the RSP is SP based on the analog input.
- "Setting 3" is intended for extension. Normally, this setting must not be used.

- **Item ID = 15 [Auto Tuning Method (master)]**
- **Item ID = 16 [Auto Tuning Method (slave)]**

A calculation method for the auto tuning control is selected.

0	Standard AT Calculation method for general process.
1	AT for overshoot suppression Calculation method for process that overshoot occurs easily.
2	For extension Do not use this setting.
3	AT for regulatory control Calculation method for process requiring only regulatory control without SP changes.

Note

- The PID constants to be updated include the PID constants for disturbance rejection.
- For details about how to operate the auto tuning,
  refer to Auto tuning of PID_CAS (cascade PID) control (on page 2-85).

For details about the auto tuning operation diagram, stop conditions, and cautions,
refer to Appendix 2, Auto Tuning.

- **Detailed description of the other items**

The descriptions of other items are the same as those of the PID_A Options (on page 1-2).

■ PID_CAS Constants

This setup is necessary to use the PID_CAS (cascade PID control) function block. The constants necessary for the control calculation, such as PID constants and output limit values can be set and changed.

Parameter type ID: 212h Group ID: 001 to FFFh (master)
Parameter type ID: 213h Group ID: 001 to FFFh (slave)

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Proportional Band	0.1 to 1000.0%	100.0%	0		REAL
2	Integral Time	0.00 to 6000.00s	0.00s	0		
3	Derivative Time	0.00 to 6000.00s	0.00s	0		
4	Output Low Limit for I-Action	–1000.0 to +1000.0%	0.0%	0		
5	Output High Limit for I-Action	–1000.0 to +1000.0%	100.0%	0		
6	Output Low Limit	–1000.0 to +1000.0%	0.0%	0	The limits do not work in the MANUAL mode.	
7	Output High Limit	–1000.0 to +1000.0%	100.0%	0		
8	Manual Reset	0.0 to 100.0%	50.0%	0	The setting is effective only when the integral time is 0.00 sec.	
9	P-Band for Disturbance Rejection	0.1 to 1000.0%	100.0%	0	The setting is effective only when using the "2 degrees of freedom PID control" capability.	
10	I-time for Disturbance Rejection	1.00 to 6000.00s	120.00s	0		
11	D-time for Disturbance Rejection	0.00 to 6000.00s	0.00s	0		

These setting items are individually set on the master and slave.

● Detailed description of the items

The descriptions of the items are the same as those of the PID_A Constants. For details,

☞refer to PID_A constants on page 1-7.

■ Ra_PID Options

This setup is necessary to use the Ra_PID (RationaLOOP PID control) function block. The settings necessary for the control action, such as control operation, PV range, and auto-tuning method can be changed.

Parameter type ID: 234h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Ra-PID Mode	1: PID 2: Ra-PID 3: Ra-PID++	2	0		DINT
2	Control Action	0: Reverse action 1: Direct action	0	0		
3	CLAFT Self S	0: Stop 1: Start	0	1		
4	CLAFT Self H	0: Stop 1: Start	0	1		
5	Self S Tuning Phase	0: Ramp-up and down 1: Ramp-up only	0	1		
6	Self S Initialization Method	0: Reset 1: Merge 2: Initialization A 3: Initialization B	0	1		
7	PV Range Min	–99999.9 to +99999.9	0.0	0	The settings determines the full scale range of the PID control.	REAL
8	PV Range Max	–99999.9 to +99999.9	1000.0	0		
9	Initial MV Value	–10.0 to +110.0%	0.0%	0		
10	MV Rate-of-change Limit	0.0 to 100.0%	0.0%	0		
11	AUTO/MANUAL Transfer	0: Bump-less transfer 1: Preset manual value	0	0		DINT
12	Preset Manual Value	–10 to +110.0%	0.0%	0	This setting is effective only when item ID 11 [AUTO/MANUAL Transfer] is set at "setting 1 (Preset manual value)".	REAL
13	Dead Band	0.0 to 10.0%FS	0.0%FS	0		
14	CLAFT AT Pb Fine Tuning	0.0 to 10.0	1.0	0		
15	CLAFT AT Ti Fine Tuning	0.0 to 10.0	1.0	0		
16	CLAFT AT Td Fine Tuning	0.0 to 10.0	1.0	0		
17	CLAFT Self S Pb Fine Tuning	0.0 to 10.0	1.0	1		
18	CLAFT Self S Ti Fine Tuning	0.0 to 10.0	1.0	1		
19	CLAFT Self S Td Fine Tuning	0.0 to 10.0	1.0	1		
20	Just-FiTTER Settling Bound	0.0 to 100.0%FS	0.1%FS	0		
21	CLAFT Self S Step Size	0.0 to 100.0%FS	10.0%FS	1		
22	CLAFT Self S Settling Bound	0.0 to 100.0%FS	1.0%FS	1		
23	CLAFT Self H Settling Bound	0.0 to 100.0%FS	1.0%FS	1		

● Item ID = 1 [Ra-PID Mode]

Normally, this setting item is used at "factory setting".
Select an optimal control method.

1	PID The control equivalent to the PID_A control is performed.
2	Ra_PID The PID control applicable to the high precision control is performed.
3	Ra_PID++ In addition to the PID control applicable to the high precision control, the Just-FiTTER reset function (reinforcement of the overshoot suppression effect) is also executed when using the Just-FiTTER feature.

● **Item ID = 3 [CLAFT Self S]**

This setting is used to switch between the CLAFT Self S feature start and stop.

0	Stop CLAFT Self S feature Conventional PID control
1	Start CLAFT Self S feature Self-tuning PID control

While the CLAFT Self S feature is being executed, the value is automatically changed in a range of 2 to 6 corresponding to the progress of the self-tuning PID constants.
"3" indicates the condition that the self tuning can be executed.
As the self-tuning PID constants is progressed, the value is updated like 4 → 5 → 6.
After a series of self-tuning PID constants steps has been completed, the value is returned to "3".
The value that the user should specify is either "0 (Stop)" or "1 (Start)".

Note

CLAFT Self S feature

Use of CLAFT, a self-tuning capability originally developed by Yamatake Corporation, makes it possible to automatically adjust the PID constants during the step response to a set point (SP) change, ensuring excellent step response.

● **Item ID = 4 [CLAFT Self H]**

This setting is used to switch between the CLAFT Self H feature start and stop.

0	Stop CLAFT Self H feature Conventional PID control
1	Start CLAFT Self H feature Self-tuning PID control

While the CLAFT Self H feature is being executed, the value is automatically changed in a range of 2 to 6 corresponding to the progress of the self-tuning PID constants.
"3" indicates the condition that the self tuning can be executed.
As the self-tuning PID constants is progressed, the value is updated like 4 → 5 → 6.
After a series of self-tuning PID constants steps has been completed, the value is returned to "3".
The value that the user should specify is either "0 (Stop)" or "1 (Start)".

Note

CLAFT Self H feature

Use of CLAFT, a self-tuning capability originally developed by Yamatake Corporation, makes it possible to automatically adjust the PID constants if hunching occurs. This suppresses the hunching.

● **Item ID = 5 [Self S Tuning Phase]**

To make the CLAFT Self S feature not start when the PV value is decreasing, set this setting to "setting 1 (Ramp-up only)".

0	Ramp-up and down The PID constants are automatically corrected at ramp-up and ramp-down during execution of the CLAFT Self S feature.
1	Ramp-up only The PID constants are automatically corrected only at ramp-up during execution of the CLAFT Self S feature.

The setting "1" is used for the process where the characteristics of the ramp-up may greatly vary from those of the ramp-down and where the controllability is not required at ramp-down.

- **Item ID = 6 [Self S Initialization Method]**

This setting is used how to alter the PID constants (Reset, Merge, initialization A(B)) when the CLAFT Self S feature, a self tuner for PID constants, is executed.

0	Reset CLAFT Self S calculates the PID constants without consideration of the previously set PID constants. After that, the setting value of "Self S Initialization Method" is automatically changed to "setting 1 (Merge)"
1	Merge CLAFT Self S calculates the PID constants with consideration of the previously set PID constants. Actually, the PID constants are continued to be gradually altered.
2	Initialization A When the CLAFT self S feature is started, the PID constants are automatically converted into the standard numeric values (default values) and the tuning is performed by means of the "setting 0 (Reset)" method. If inappropriate values are stored in the PID constants, results of self-tuning PID constants with use of them may not be good. In such a case, this method might be effective.
3	Initialization B When the CLAFT self S feature is started, the PID constants are automatically converted into the values (special default values) corresponding to the dead time system, and the tuning is performed by means of the "setting 0 (Reset)" method. In a system that the dead time of the controlled process is larger than the time constant, results of self-tuning PID constants may not be good. In such a case, this method might be effective.

- **Item ID = 7 [PV Range Min]**
- **Item ID = 8 [PV Range Max]**

Set the PV range in relevant industrial unit.

Full scale range of the following percentage data is determined by these MIN/MAX values:

- Proportional Band of Ra_PID control
- Just-FiTTER Settling Bound
- CLAFT Self S Step Size
- CLAFT Self S Settling Bound
- CLAFT Self H Settling Bound

- **Item ID = 9 [Initial MV Value]**

If any of the following conditions is satisfied, the PID control is started from this initial MV value:

- The function block is initially executed (including power ON).
- The previous ISaGRAF mode is a mode other than the real-time mode (RT).
- The previous E_OK output parameter equals "ERR".
- The PARA input parameter (group ID of Ra_ PID Options) is changed.
- The AT process is completed or stopped.
- The cycle time is changed.
- The control action (reverse action or direct action) is changed.

- **Item ID = 10 [MV Rate-of-change Limit]**

  Normally, this setting item is used at "factory setting".
  This setting item works only in the AUTO mode.
  Changes in MV value output at every control cycle (stored in the cycle timing setting) are limited to the set value or less.
  As this value is smaller, the allowable change is getting smaller.
  Additionally, if this setting is set at "0.0", this becomes "No-limit".
  If the rapid change of the MV value may adversely affect the actuators, 0.0 should not be specified.

- **Item ID = 11 [AUTO/MANUAL Transfer]**

  Select an MV value when the control mode is changed from AUTO to MANUAL.

0	Bump-less transfer The MV value at the time of transfer is retained.
1	Preset manual value The preset manual value is output.

  When the mode is returned from the MANUAL mode to the AUTO mode, the control calculation is started from the MV value at that time.

- **Item ID = 13 [Dead Band]**

  Normally, this setting item is used at "factory setting".
  The dead band setting forces the PID control to operate with the deviation of 0% if the absolute value of the deviation (%FS) is the set value or less. When the deviation vibrates at around 0% because PV includes a large noise or because the actuator valve has hysteresis, this setting is used to prevent such vibration from affecting the control.
  For example, when the PV range setting is 0 to 400˚C and 1.0%FS is specified as this value, PID action assuming error is 0% is performed while the deviation is within ±4.0˚C.

- **Item ID = 14 [CLAFT AT Pb Fine Tuning]**
- **Item ID = 15 [CLAFT AT Ti Fine Tuning]**
- **Item ID = 16 [CLAFT AT Td Fine Tuning]**

  Normally, this setting item is used at "factory setting".
  Set the fine tuning factors for CLAFT AT used to calculate the proportional band (Pb), integral time (Ti), and derivative time (Td).
  - When "1.0" is stored in [CLAFT AT Pb Fine Tuning], the proportional band (Pb) is calculated as standard tuning. When "1.5" is stored in [CLAFT AT Pb Fine Tuning], the proportional band (Pb) 1.5 times larger than the standard control is calculated.
  - When "1.0" is stored in [CLAFT AT Ti Fine Tuning], the integral time (Ti) is calculated as standard tuning. When "1.5" is stored in [CLAFT AT Ti Fine Tuning], the integral time (Ti), 1.5 times larger than the standard tuning is calculated.
  - When "1.0" is stored in [CLAFT AT Td Fine Tuning], the derivative time (Td) is calculated as standard tuning. When "1.5" is stored in [CLAFT AT Td Fine Tuning], the derivative time (Td) 1.5 times larger than the standard tuning is calculated.

  Note

  CLAFT AT
  This CLAFT AT is auto tuning method based on the concept of CLAFT, a self-tuning capability originally developed by Yamatake Corporation. The output operation of this CLAFT AT becomes the same as that of AT with normal limit cycle.

- **Item ID = 17 [CLAFT Self S Pb Fine Tuning]**
- **Item ID = 18 [CLAFT Self S Ti Fine Tuning]**
- **Item ID = 19 [CLAFT Self S Td Fine Tuning]**

Normally, this setting item is used at "factory setting".
Set the fine tuning factors for CLAFT Self S used to calculate the proportional band (Pb), integral time (Ti), and derivative time (Td).

- When "1.0" is stored in [CLAFT Self S Pb Fine Tuning], the proportional band (Pb) is calculated as standard tuning. When "1.5" is stored in [CLAFT Self S Pb Fine Tuning], the proportional band (Pb) 1.5 times larger than the standard tuning is calculated.
- When "1.0" is stored in [CLAFT Self S Ti Fine Tuning], the integral time (Ti) is calculated as standard tuning. When "1.5" is stored in [CLAFT Self S Ti Fine Tuning], the integral time (Ti), 1.5 times larger than the standard tuning is calculated.
- When "1.0" is stored in [CLAFT Self S Td Fine Tuning], the derivative time (Td) is calculated as standard tuning. When "1.5" is stored in [CLAFT Self S Td Fine Tuning], the derivative time (Td) 1.5 times larger than the standard tuning is calculated.

- **Item ID = 20 [Just-FiTTER Settling Bound]**

Normally, this setting item is used at "factory setting".
Specify a deviation range (%FS) to judge the settling of disturbance recovery by Just-FiTTER.
If the actual deviation is within this bound, a series of operations by Just-FiTTER is completed.
For example, if the deviation becomes ±4.0˚C or less when the PV range setting is 0 to 400˚C, and further if this setting is set at "0.1%FS", this is considered as being in the settled state.

Note

Just-FiTTER
This feature suppresses the overshoot when disturbance recovery or step response is performed.

- **Item ID = 21 [CLAFT Self S Step Size]**

Normally, this setting item is used at "factory setting".
Specify the minimum SP change size (%FS), with which the step responses triggering a series of CLAFT Self S operation are recognized. For example, if the SP value is changed 40˚C or more when the PV range setting is 0 to 400˚C and this setting is set at "10.0%FS", the controller considers this change is a step response to start the CLAFT Self S operations.

- **Item ID = 22 [CLAFT Self S Settling Bound]**

Normally, this setting item is used at "factory setting".
Specify a deviation range to judge the completion of step response settling by CLAFT Self S. If the actual deviation is within this bound, a series of operations by CLAFT Self S is completed. For example, if the deviation becomes ±4.0˚C or less when the PV range setting is 0 to 400˚C, and further if this setting is set at "1.0 %FS", this is considered as being in the settled state.

- **Item ID = 23 [CLAFT Self H Settling Bound]**

  Normally, this setting item is used at "factory setting".
  Specify a deviation range (%FS) to judge the hunching occurrence and the completion of settling by CLAFT Self H. If the actual deviation is within this bound, a series of operations by CLAFT Self H is completed. For example, if the deviation becomes ±4.0˚C or more when the PV range setting is 0 to 400˚C, and further if this setting is set at "1.0 %FS", this is considered as generation of hunching.

■ Ra_PID Constants

This setup is necessary to use the Ra_PID (RationaLOOP PID control) function block.
The constants necessary for the control calculation, such as PID constants and output limit values can be set and changed.

Parameter type ID: 235h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Proportional Band	0.1 to 1000.0%	5.0%	0		REAL
2	Integral Time	0.00 to 6000.00s	30.00s	0		
3	Derivative Time	0.00 to 6000.00s	30.00s	0		
4	DeltaD Factor	0.125 to 1.000	0.125	1		
5	SP Filtering Factor	0.0 to 10.0	0.0	1		
6	DeltaS Factor	1.0 to 5.0	1.0	1		
7	Output Low Limit	–1000.0 to +1000.0%	0.0%	0	The limits do not work in the MANUAL mode.	
8	Output High Limit	–1000.0 to +1000.0%	100.0%	0		
9	Manual Reset	0.0 to 100.0%	50.0%	0	The setting is effective only when the integral time is 0.00 sec.	
10	Just-FiTTER Ramp-down Factor	0.0 to 1.0	0.0	0		
11	Just-FiTTER Reset Factor	0.0 to 10.0	0.5	0		

- **Item ID = 1 [Proportional Band]**
- **Item ID = 2 [Integral Time]**
- **Item ID = 3 [Derivative Time]**

  These are the "Proportional band (P)", "Integral time (I)", and "Derivative time (D)" settings of the control constants. When using the PID constants self-tuning features (CLAFT AT, CLAFT Self S, CLAFT Self H, DeltaS, and DeltaD), these settings are automatically set corresponding to the characteristics of the controlled process. If optimal values are already known or if it is difficult that the self-tuning feature becomes effective, make the settings manually.

- **Item ID = 4 [DeltaD Factor]**

  Normally, this setting item is used at "factory setting".
  Specify a DeltaD factor for the derivative action. When "0.125" is specified, the general PID control is performed. When "1.0" is specified, the control equivalent to the PI control is performed.
  If a value between both settings is specified, an intermediate control characteristic is obtained. This setting is linked with the DeltaD feature. The PID constants are also automatically corrected to the most suitable values corresponding to the intermediate characteristic settings between the PI control and PID control.

  Note

  DeltaD feature

  This DeltaD feature is an automatic PID constants tuning function that achieves any intermediate characteristic between the PI control and PID control. The special control for the disturbance response characteristics can be performed easily using one variable.

- **Item ID = 5 [SP Filtering Factor]**

  Normally, this setting item is used at "factory setting".
  Specify an SP filter time constant factor used for "2 degrees of freedom PID control". When "0.0" is specified, the control becomes equivalent to "1 degree of freedom PID control". When a value close to "10.0" is specified, the control gets close to "2 degrees of freedom PID control".

- **Item ID = 6 [DeltaS Factor]**

  Specify the responsiveness vs. stability level of the control operation for the DeltaS feature.
  When "1.0" is specified, the control becomes the normal level PID control.
  When a value larger than "1.0" (about 1.1 to 2.0) is specified, the PID constants are automatically corrected to perform the PID control placing importance on the stability.

  Note

  DeltaS feature

  This DeltaS feature is an automatic PID constants tuning function that achieves a variable control characteristic corresponding to the level of importance on stability. The special control for the margin of the stability can be performed easily using one variable.

- **Item ID = 7 [Output Low Limit]**
- **Item ID = 8 [Output High Limit]**

  These settings are used to control the high and low limits of the MV value.
  The limits work even when the auto tuning is being operated. However, the limits do not work in the MANUAL mode.
  These settings also work as integral action limit. If the MV value reaches the output high limit or low limit, the integral action will not function.
  The settings are used to prevent reset wind-up that occurs if the PV does not rise for a long period of time.

- **Item ID = 9 [Manual Reset]**

  This setting is used when the integral time is set at "0". This setting is used to remove the offset that occurs during proportional action (without integral action). Specify an MV value suitable for deviation "0".

- **Item ID = 10 [Just-FiTTER Ramp-down Factor]**

  When "0.0" is specified, the Just-FiTTER feature becomes OFF. As a value close to "1.0" is specified, the overshoot suppression effect becomes more pronounced.
  To obtain the overshoot suppression effect without slowing of the disturbance recovery speed and step response speed, it is effective that a value close to "1.0" is specified.
  For reference, it is recommended that a value of "0.95" be the upper limit.

- **Item ID = 11 [Just-FiTTER Reset Factor]**

  As a larger value is specified, the overshoot suppression effect becomes more pronounced. If the overshoot cannot be suppressed even though a large value is stored in [Just-FiTTER Ramp-down Factor], make the disturbance recovery speed or step response speed gradually slower. At the same time, make this value gradually larger.

■ UP_PID Options

This setup is necessary to use the UP_PID (use-point PID control) function block. The settings necessary for the control calculation, such as control action, PV range, and auto-tuning method can be changed.

Parameter type ID: 241h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Control Action	0: Reverse action 1: Direct action	0	0		DINT
2	Control Method	0: Use-point auto 1: For extension 2: Use-point single	0	0		
3	PV Range Min	–99999.9 to +99999.9	0.0	0	The settings are used to determine the full scale range of the PID control.	REAL
4	PV Range Max	–99999.9 to +99999.9	1000.0	0		
5	Initialization on SP Changes	0: Automatic 1: Initialized on SP changes 2: Not initialized	0	0		DINT
6	Initial MV Value	–10.0 to +110.0%	0.0%	0		REAL
7	MV Rate-of-change Limit	0.0 to 100.0%	0.0%	0		
8	AUTO/MANUAL Transfer	0: Bump-less transfer 1: Preset manual value	0	0		DINT
9	Preset Manual Value	–10.0 to +110.0%	0.0%	0	This setting is effective only when item ID 8 [AUTO/MANUAL Transfer] is set at "setting 1 (Preset manual value)".	REAL
10	Auto Tuning Method	0: Standard AT 1: AT for overshoot suppression 2: AT for regulatory control	0	0		DINT

● Item ID = 2 [Control Method]

Select a control method.

Setting	Contents
0: Use-point auto	PID operation and FF operation are performed. The control constants of the FF operation are automatically calculated from the PID constants. This is a control method with the adjustment of control constants made simplified.
1: For extension	Do not use this setting.
2: Use-point single	Only the PID operation is performed and the FF operation is not performed. The operation equivalent to the PID_A control is performed.

[!] Handling Precaution

The "Use-point auto" control may not applicable depending on the characteristics of the controlled process. In this case, select "setting 2 (Use-point single)".

● Item ID = 10 [Auto Tuning Method]

Select a calculation method for the auto tuning control.

0	Standard AT Calculation method for general process.
1	AT for overshoot suppression Calculation method for process that overshoot occurs easily.
2	AT for regulatory control Calculation method for regulatory control process, where SP changes are not assumed .

For details about the auto tuning operation diagram, stop conditions, and cautions, ☞refer to Appendix 2, Auto Tuning.

● **Detailed description of the other items**

The descriptions of the other items are the same as those of the PID_A Constants. For details,

☞refer to PID_A Options on page 1-2.

■ UP_PID Constants

This setup is necessary to use the UP_PID (use-point PID control) function block. The constants necessary for the control calculation, such as PID constants and output limit values can be set and changed.

Parameter type ID: 242h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Proportional Band	0.1 to 1000.0%	100.0%	0		REAL
2	Integral Time	0.00 to 6000.00 s	0.00 s	0		
3	Derivative Time	0.00 to 6000.00 s	0.00 s	0		
4	Output Low limit for I-Action	–1000.0 to +1000.0%	0.0%	0		
5	Output High limit for I-Action	–1000.0 to +1000.0%	100.0%	0		
6	Output Low Limit	–1000.0 to +1000.0%	0.0%	0	The limits do not work in the MANUAL mode.	
7	Output High Limit	–1000.0 to +1000.0%	100.0%	0		
8	Manual Reset	0.0 to 100.0%	50.0%	0	The setting is effective only when the integral time is 0.00 sec.	

● **Detailed description of the items**

The descriptions of other items are the same as those of the PID_A Constants. For details,

☞refer to PID_A Constants on page 1-7.

■ TBL / TBR Setup

This setup is necessary to use the TBL (linearization table lookup) and TBR (linearization table reverse lookup) function blocks.

Parameter type ID: 301h Group ID: 001 to FFFh

Item ID	Item name	Setting range	Factory setting	User level	Remarks	Data type
1	Point X1	–99999.9 to +99999.9	0.0	0		REAL
2	Point Y1	–99999.9 to +99999.9	0.0	0		
3	Point X2	–99999.9 to +99999.9	0.0	0		
4	Point Y2	–99999.9 to +99999.9	0.0	0		
⋮	⋮	⋮	⋮	⋮		
39	Point X20	–99999.9 to +99999.9	0.0	0		
40	Point Y20	–99999.9 to +99999.9	0.0	0		

1 - 3 Calculation Monitor Parameters

Handling Precaution

The following Calculation Monitor Parameters cannot be set or changed. Therefore, the settings, such as SP cannot be set or changed.

■ PID_A Monitor

The status, such as PV, SP, MV, or mode can be checked when using the PID_A (standard PID control) function block.

Parameter type ID: 203h Group ID: 001 to FFFh

Item ID	Item name	Setting range	User level	Remarks	Data type
1	SP	The settings cannot be changed.	0		REAL
2	PV		0		
3	MV		0		
4	Mode		0	Bit 0: AUTO Bit 1: MANUAL Bit 2: AT executing Bit 3: ST learning Bit 31: ERR	DWORD
5	MODE Input parameter		0	0: AUTO 1: MANUAL	BOOL
6	PID Input parameter		0		DINT
7	PARA Input parameter		0		
8	MONI Input parameter		0		
9	AT Input parameter		0		
10	MV_IN Input parameter		0		REAL
11	E_OK Output parameter		0	0: ERR 1: OK	BOOL

■ PID_CAS Monitor

The status, such as PV, SP, MV, or mode can be checked when using the PID_CAS (cascade PID control) function block.

Parameter type ID: 214h Group ID: 001 to FFFh

Item ID	Item name	Setting range	User level	Remarks	Data type
1	M_SP	The settings cannot be changed.	0	SP on master	REAL
2	M_PV		0	PV on master	
3	S_SP		0	SP on slave	
4	S_PV		0	PV on slave	
5	MV		0	MV on slave	
6	Mode		0	Bit 0: AUTO Bit 1: MANUAL Bit 2: AT executing Bit 3: Reserved. Bit 4: READY Bit 5: RUN Bit 6: REMOTE Bit 7: LOCAL Bit 31: ERR	DWORD
7	MAN Input parameter		0	0: AUTO 1: MANUAL	BOOL
8	RUN Input parameter		0	0: READY 1: RUN	
9	LOCAL Input parameter		0	0: REMOTE 1: LOCAL	
10	S_LSP Input parameter		0		REAL
11	RSP_H Input parameter		0		
12	RSP_L Input parameter		0		
13	M_PID Input parameter		0		DINT
14	S_ PID Input parameter		0		
15	PARA Input parameter		0		
16	MONI Input parameter		0		
17	AT Input parameter		0		
18	MV_IN Input parameter		0		REAL
19	AT_MD Output parameter		0		BOOL
20	S_ RSP Output parameter		0		REAL
21	E_OK Output parameter		0	0: ERR 1: OK	BOOL

■ Ra_PID Monitor

The status, such as PV, SP, MV, or mode can be checked when using the Ra_PID (RationaLOOP PID control) function block.

Parameter type ID: 236h Group ID: 001 to FFFh

Item ID	Item name	Setting range	User level	Remarks	Data type
1	SP	The settings cannot be changed.	0		REAL
2	PV		0		
3	MV		0		
4	Mode		0	Bit 0: AUTO Bit 1: MANUAL Bit 2: AT executing Bit 3: Self S controlling Bit 31: ERR	DWORD
5	MODE Input parameter		0		BOOL
6	PID Input parameter		0		DINT
7	PARA Input parameter		0		
8	MONI Input parameter		0		
9	AT Input parameter		0		
10	MV_IN Input parameter		0		REAL
11	JF_IN Input parameter		0		
12	E_OK Output parameter		0		BOOL
13	JF Output parameter		0		REAL

■ UP_PID Monitor

The status, such as PV, SP, MV, or mode can be checked when using the UP_PID (use-point PID control) function block.

Parameter type ID: 243h Group ID: 001 to FFFh

Item ID	Item name	Setting range	User level	Remarks	Data type
1	U_SP(Use SP)	The settings cannot be changed.	0		REAL
2	U_PV(Use PV)		0		
3	S_PV(Source PV)		0		
4	MV		0		
5	Mode		0	Bit 0: AUTO Bit 1: MANUAL Bit 2: AT executing Bit 31: ERR	DWORD
6	MODE Input parameter		0	0: AUTO 1: MANUAL	BOOL
7	PID Input parameter		0		DINT
8	PARA Input parameter		0		
9	MONI Input parameter		0		
10	AT Input parameter		0		
11	MV_IN Input parameter		0		REAL
12	E_OK Output parameter		0	0: ERR 1: OK	BOOL

Chapter 2. STANDARD OPERATORS / FUNCTIONS / FUNCTION BLOCKS

This chapter describes the standard operators, functions and function blocks supported by the DMC50 CTRL module.
They are listed in alphabetical order.

If you want to find specific operators / functions / function blocks by their functionalities, see the following functional tables:

■ Standard operators

● Data operation

Name		Functional description	Page
1 gain	Assignment	Assigns data. Any types of data are available.	2-21
NEG	Sign change	Changes the sign of data. Available data types are DINT and REAL.	2-73

● Boolean operation

Name		Functional description	Page
&(AND)	AND	Outputs the boolean AND of two or more pieces of BOOL type data.	2-9
>=1(OR)	OR	Outputs the boolean OR of two or more pieces of BOOL type data.	2-20
=1(XOR)	Exclusive OR	Outputs the boolean exclusive-OR of 2 pieces of BOOL type data.	2-17

● Mathematical operation

Name		Functional description	Page
+	Addition	Adds two or more pieces of data together. Available data types are DINT and REAL.	2-12
–	Subtraction	Subtracts 1 piece of data from another. Available data types are DINT and REAL.	2-8
*	Multiplication	Multiplies two or more pieces of data. Available data types are DINT and REAL.	2-10
/	Division	Divides a piece of data by another. Available data types are DINT and REAL.	2-11

● Bitwise boolean operation

Name		Functional description	Page
AND_MASK	Bitwise AND	Outputs the bitwise AND of 2 pieces of DINT type data.	2-26
OR_MASK	Bitwise OR	Outputs the bitwise OR of 2 pieces of DINT type data.	2-76
XOR_MASK	Bitwise XOR	Outputs the bitwise exclusive OR of 2 pieces of DINT type data.	2-149
NOT_MASK	Bitwise inversion	Inverts and outputs DINT type data bitwise.	2-74

● Comparison test

	Name	Functional description	Page
<	Less than	Compares 2 pieces of data. (<)	2-13
<=	Less or equal	Compares 2 pieces of data. (≤)	2-14
>	Greater than	Compares 2 pieces of data. (>)	2-18
>=	Greater or equal	Compares 2 pieces of data. (≥)	2-19
=	Equal	Compares 2 pieces of data. (=)	2-16
<>	Not equal	Compares 2 pieces of data. (≠)	2-15

Available data types vary depending on each operator.

● Data conversion

Name		Functional description	Page
BOO	Convert to boolean	Converts a value to a BOOL type one. Any types of data can be converted.	2-34
ANA	Convert to integer	Converts a value to a DINT type one. Any types of data can be converted.	2-24
ANA_DP	Real to integer conversion	Multiplies REAL data by 10^{DP} and then rounds it off to an integer.	2-25
REAL	Convert to real	Converts a value to a REAL type one. Available data types are BOOL, DINT and TIME.	2-115
TMR	Convert to timer	Converts a value to a TIME type one. Available data types are DINT and REAL.	2-140
MSG	Convert to message string	Converts a value to a STRING type one. Any types of data can be converted.	2-69

● Others

Name		Functional description	Page
CAT	Message string concatenation	Concatenates two or more number of message strings into a single message string.	2-35
SYSTEM	Access to system	Reads out cycle time, etc.	2-132

■ Functions

● Mathematical functions

Name		Functional description	Page
ABS	Absolute value	Outputs the absolute value of REAL type data	2-22
EXPT	Exponential function *	Performs exponential operation on REAL type data and output the result.	2-45
LOG	Common logarithm	Outputs the common logarithm of REAL type data.	2-60
POW	Exponential function	Performs exponential operation on REAL type data and outputs the result.	2-91
SQRT	Square root	Outputs the square root of REAL type data.	2-130
TRUNC	Truncating the fractional portion	Drops the fractional portion of REAL type data.	2-144

* Different from POW, the exponent part is specified by DINT type data.

● Trigonometric functions

Name		Functional description	Page
ACOS	Arc cosine	Outputs the arc cosine of REAL type data. The output is in radians.	2-23
ASIN	Arc sine	Outputs the arc sine of REAL type data. The output is in radians.	2-28
ATAN	Arc tangent	Outputs the arc tangent of REAL type data. The output is in radians.	2-29
COS	Cosine	Outputs the cosine of REAL type data. The input is in radians.	2-38
SIN	Sine	Outputs the sine of REAL type data. The input is in radians.	2-129
TAN	Tangent	Outputs the tangent of REAL type data. The input is in radians.	2-133

● Register control

Name		Functional description	Page
ROL	Left rotation	Rotates the bits of DINT type data to the left.	2-118
ROR	Right rotation	Rotates the bits of DINT type data to the right.	2-119
SHL	Left shift	Shifts the bits of DINT type data leftward.	2-126
SHR	Right shift	Shifts the bits of DINT type data rightward.	2-127

● Data operation

Name		Functional description	Page
MIN	Minimum value	Compares 2 pieces of DINT type data and outputs the minimum value.	2-66
MAX	Maximum value	Compares 2 pieces of DINT type data and outputs the maximum value.	2-64
LIMIT	High/low limit	Limits DINT type data by the high and low limit values.	2-59
LIM_HI	REAL type high limit	Limits REAL type data by the high limit value.	2-56
LIM_LO	REAL type low limit	Limits REAL type data by the low limit value.	2-58
LIM_HILO	REAL type high/low limit	Limits REAL type data by the high and low limit values.	2-57
MOD	Modulus	Outputs the remainder of DINT type data.	2-68
MUX4	4-input multiplexer	Selects one out of 4 pieces of DINT type data.	2-70
MUX8	8-input multiplexer	Selects one out of 8 pieces of DINT type data.	2-71
MUX8REAL	REAL type 8-input multiplexer	Selects one out of 8 pieces of REAL type data.	2-72
ODD	Odd parity	Determines whether the DINT type data is odd or even.	2-75
RAND	Random value	Generates a random value of DINT type data. The range of random values can be specified.	2-114
SEL	Binary selection	Selects one out of 2 pieces of DINT type data.	2-122
SEL_BOOL	BOOL type binary selection	Selects one out of 2 pieces of BOOL type data.	2-123
SEL_REAL	REAL type binary selection	Selects one out of 2 pieces of REAL type data.	2-124
SEL_TMR	TIME type binary selection	Selects one out of 2 pieces of TIME type data.	2-125

● Data conversion

Name		Functional description	Page
ASCII	Character → ASCII code conversion	Outputs an ASCII code corresponding to the character specified in a message string.	2-27
CHAR	ASCII code → character conversion	Outputs an one character string corresponding to the specified ASCII code.	2-36
BIN3DEC	3-input boolean to integer conversion	Converts 3 pieces of BOOL type data to 0-to-7 DINT type data.	2-31
BIN8DEC	8-input boolean to integer conversion	Converts 8 pieces of BOOL type data to a 0-to-255 DINT type data.	2-32
SCAL_CNV	Scale conversion	Performs scaling operation of REAL type data. Can be used for conversion of industrial unit or 0-100% conversion.	2-121

● Message string operation

Name		Functional description	Page
DELETE	Message string deletion	Deletes a specified message string from a base message string. It is possible to specify the position and the length of the string to be deleted.	2-43
INSERT	Message string insertion	Inserts a specified message string into a base message string . It is possible to specify the position for insertion.	2-49
FIND	Message string finding	Searches for the specified message string and outputs its position.	2-47
MLEN	Message string length	Outputs the message string length.	2-67
LEFT	Left message string extraction	Extracts the specified message string from the left hand side of a base message string. It is possible to specify the length of the string to be extracted.	2-54
MID	Message string extraction	Extracts the specified message string from a base character string. It is possible to specify the position and the length of the string to be extracted.	2-65
REPLACE	Message string replacement	Replaces a message string with another one. Replaces the specified number of string from the specified position with another specified string.	2-116
RIGHT	Right message string extraction	Extracts the specified message string from the right hand side of a base message string. It is possible to specify the length of the string to be extracted.	2-117

■ Function blocks

● Boolean operation

Name		Functional description	Page
SR	Set dominant bistable	Sets a latch giving "set" a higher priority.	2-131
RS	Reset dominant bistable	Sets a latch giving "reset" a higher priority.	2-120
R_TRIG	Rising edge detection	Detects the rising edge of BOOL type data.	2-108
F_TRIG	Falling edge detection	Detects the falling edge of BOOL type data.	2-46

● Counter

Name		Functional description	Page
CTU	Up-counter	Outputs the countup value of DINT type data. Outputs the end signal when the count value has reached the target value.	2-40
CTD	Down-counter	Outputs the countdown value of DINT type data. Outputs the end signal when the count value has reached 0.	2-39
CTUD	Up/down counter	Integration of CTU (Up-counter) and CTD (Down-counter)	2-41

● Timer

Name		Functional description	Page
TON	ON delay timer	Turns on at a specified time after the rising edge has been detected. Also outputs the elapsed time.	2-142
TOF	OFF delay timer	Turns off at a specified time after the falling edge has been detected. Also outputs the elapsed time.	2-141
TP	Pulse timer	Turns on while a specified time after the rising edge has been detected. Also outputs the elapsed time.	2-143

● Integer analog

Name		Functional description	Page
CMP	Complete comparison	Compares 2 pieces of DINT type data and outputs either <, = or > as the result.	2-37

● Real analog

Name		Functional description	Page
AVERAGE	Moving average	Outputs the mean value of the specified sample numbers of REAL type data.	2-30
MAV	Moving average	This is an extension of the AVERAGE (Moving average).	2-62
HYSTER	Hysteresis	Determines with hysteresis whether the REAL type data has exceeded the high limit value.	2-48
LIM_ALRM	Limit alarm	Determines with hysteresis whether the REAL type data has exceeded the high limit value or whether it has come down below the low limit value respectively.	2-55
INTEGRAL	Integral	Outputs the integral of REAL type data	2-50
DERIVATE	Derivative	Outputs the derivative of REAL type data.	2-44
DED	Dead time	Outputs REAL type data after the dead time has elapsed.	2-42
LEAD_LAG	Lead/lag	Outputs the lead/lag calculated value of REAL type data. Can be used also as a first order digital filter.	2-52

● Signal generation

Name		Functional description	Page
BLINK	BOOL type signal blinking	Generates blinking signals at every specified cycle.	2-33
PLS_GEN	Pulse generator	Generates a pulse at every specified cycle.	2-90
RAMP_GEN	Ramp generator	Generates ramp values of REAL type up to the target value.	2-112
SIG_GEN	Signal generator	Generates 3 signals (i.e. pulse, up-counter and sine wave signals).	2-128

● Parameter access

Name		Functional description	Page
PAR_BOOL	Read BOOL type Parameter	Reads out a Parameter element value as BOOL type data.	2-77
PAR_INT	Read DINT type Parameter	Reads out a Parameter element value as DINT type data.	2-78
PAR_REAL	Read REAL type Parameter	Reads out a Parameter element value as REAL type data.	2-79
PAW_BOOL	Write BOOL type Parameter	Writes a BOOL type value into a Parameter data element.	2-80
PAW_INT	Write DINT type Parameter	Writes a DINT type value into a Parameter data element.	2-81
PAW_REAL	Write REAL type Parameter	Writes a REAL type value into a Parameter data element.	2-82

The word "Parameter" refers to one of the following: "System Parameters", "System Monitor Parameters", "Calculation Parameters", "Program Pattern Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

● Control operation

Name		Functional description	Page
PID_A	Standard PID operation	Performs a series form PID operation of which derivative action is applied on error.	2-84
PID_CAS	Cascade PID operation	Performs a PID operation for cascade control.	2-86
Ra_PID	RationaLOOP PID operation	Performs PID operation incorporated with superior disturbance recovery and over-shoot suppression feature for high precision control.	2-110
UP_PID	Use-point PID operation	Performs two-input one-output PID operation for disturbance suppression.	2-146

● Pattern generation

Name		Functional description	Page
PTN_MAIN	Program pattern main	Performs pattern operation, generating synchronized SPs of 2 channels.	2-98
PTN_SUB	Program pattern sub	Used when you want to generate synchronized SPs of three or more number of channels. Use this function block sublinked with the PTN_MAIN function block.	2-106
PTN_MODE	Program pattern mode	Easily carries out PTN_MAIN function block mode change, advance execution, PV start, etc.	2-102
PTN_TEV	Program pattern time event	Generates time events for the current segment.	2-107
PTN_EVR	Program pattern event	Reads out event thresholds for the current segment. Used for generating PV or error events.	2-96

● Other

Name		Functional description	Page
TBL	Linearization table lookup	Converts REAL type data using a linearization table.	2-134
TBR	Linearization table reverse lookup	Restores the original value from a converted value using the same linearization table.	2-138
ZONE7	Zone selector	Evaluates REAL type data using 7 zone selecting conditions and outputs a value between 0 and 7 (total of 8 zones).	2-150
PSVC	Power supply voltage compensation	Monitors the heater power voltage fluctuations to correct the manipulated variable output.	2-92

■ Functions and function blocks not supported by DMC50

Among the functions and function blocks supported by the ISaGRAF Standard, the following cannot be used by DMC50:

- ARCREATE (Create array of integer values)
- ARREAD (Read array element)
- ARWRITE (Write array element)
- DAY_TIME (Date / time)
- F_CLOSE (Close file)
- F_EOF (Detect EOF)
- F_ROPEN (Open file in Read mode)
- F_WOPEN (Open file in Write mode)
- FA_READ (Read integer from file)
- FA_WRITE (Write integer from file)
- FM_READ (Read message string from file)
- FM_WRITE (Write message string from file)
- SEMA (Semaphore)
- OPERATE (I / O channel operation)
- STACKINT (Stack of integer analogs)

■ Functions and function blocks for DMC50

In addition to the ISaGRAF standard functions/function blocks, the DMC50 supports its dedicated functions/function blocks.

● Functions for DMC50

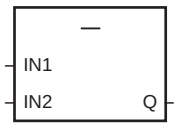
- ANA_DP (Real to integer conversion)
- BIN3DEC (3-input boolean to integer conversion)
- BIN8DEC (8-input boolean to integer conversion)
- LIM_HI (REAL type high limit)
- LIM_HILO (REAL type high/low limit)
- LIM_LO (REAL type low limit)
- MUX8REAL (REAL type 8-input multiplexer)
- SCAL_CNV (Scale conversion)
- SEL_BOOL (BOOL type binary selection)
- SEL_REAL (REAL type binary selection)
- SEL_TMR (TIME type binary selection)

● Functions blocks for DMC50

- DED (Dead time)
- LEAD_LAG (Lead/lag)
- MAV (Moving average)
- PAR_BOOL (Read BOOL type Parameter)
- PAR_INT (Read DINT type Parameter)
- PAR_REAL (Read REAL type Parameter)
- PAW_BOOL (Write BOOL type Parameter)
- PAW_INT (Write DINT type Parameter)
- PAW_REAL (Write REAL type Parameter)
- PID_A (Standard PID operation)
- PID_CAS (Cascade PID operation)
- PLS_GEN (Pulse generator)
- PSVC (Power supply voltage compensation)
- PTN_EVR (Program pattern event)
- PTN_MAIN (Program pattern main)
- PTN_MODE (Program pattern mode)
- PTN_SUB (Program pattern sub)
- PTN_TEV (Program pattern time event)
- Ra_PID (RationaLOOP PID operation)
- RAMP_GEN (Ramp generator)
- TBL (Linearization table lookup)
- TBR (Linearization table reverse lookup)
- UP_PID (Use-point PID operation)
- ZONE7 (Zone selector)

- (Subtraction) Standard operator

SYMBOL



◆ Functional description

Subtracts 1 piece of data from another.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL	
IN2	DINT, REAL	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	DINT, REAL	IN1 - IN2

◆ Description of action

Q = IN1 - IN2

IN1, IN2 and Q should be of the same data type.

& (AND) Standard operator

SYMBOL

◆ Functional description

Outputs the boolean(logical) AND of two or more pieces of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
INPUTn	BOOL	n is between 2 and 32.

◆ Output parameters

Parameter	Data type	Content
OUTPUT	BOOL	AND of inputs

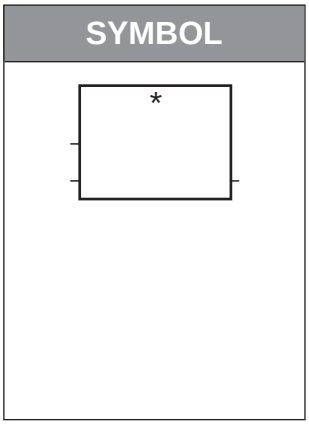
◆ Description of action

OUTPUT = INPUT1 & INPUT2 & .... & INPUTn

The number of inputs n can be any number between 2 and 32.

* (Multiplication) Standard operator

SYMBOL



◆ Functional description

Multiplies two or more pieces of data.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
INPUTn	DINT, REAL	n is between 2 and 32.

◆ Output parameters

Parameter	Data type	Content
OUTPUT	DINT, REAL	Multiplication of inputs

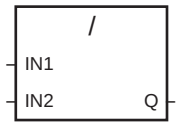
◆ Description of action

OUTPUT = INPUT1 X INPUT2 X ... X INPUTn

- The number of inputs n can be any number between 2 and 32.
- All the input parameters (INPUTn) and the output parameter (OUTPUT) should be of the same data type.

/ (Division) Standard operator

SYMBOL



◆ Functional description

Divides a piece of data by another.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL	
IN2	DINT, REAL	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	DINT, REAL	IN1 / IN2

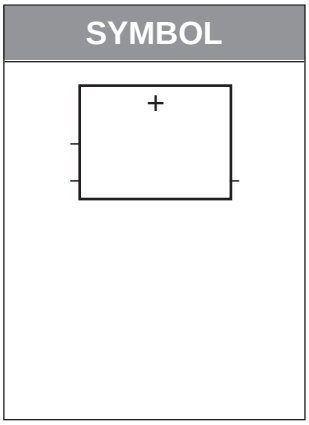
◆ Description of action

Q = IN1 / IN2

- IN1, IN2 and Q should be of the same data type.
- In case of DINT, the fractional portion of the calculation result is dropped.

+ (Addition) Standard operator

SYMBOL



◆ Functional description

Adds two or more pieces of data together.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
INPUTn	DINT, REAL	n is between 2 and 32. All the data should be of the same data type.

◆ Output parameters

Parameter	Data type	Content
OUTPUT	DINT, REAL	Addition of inputs

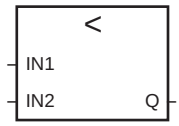
◆ Description of action

OUTPUT = INPUT1 + INPUT2 + ... + INPUTn

- The number of inputs n can be any number between 2 and 32.
- All the input parameters (INPUTn) and the output parameter (OUTPUT) should be of the same data type.

< (Less than) Standard operator

SYMBOL



◆ Functional description

Compares 2 pieces of data. (<)

Any types of data (DINT, REAL, TIME and STRING) are available.

◆ Input parameters

Parameter	Data type	Content
IN1	Any types of data	
IN2	Any types of data	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

◆ Description of action

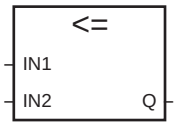
Q = TRUE when IN1 < IN2.

Q = FALSE when IN1 ≥ IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

<= (Less or equal) Standard operator

SYMBOL



◆ Functional description

Compares 2 pieces of data. (≤)
Available data types are DINT, REAL and STRING.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL, STRING	
IN2	DINT, REAL, STRING	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

◆ Description of action

Q = TRUE when IN1 ≤ IN2.
Q = FALSE when IN1 > IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

< > (Not equal) Standard operator

SYMBOL

◆ Functional description

Compares 2 pieces of data. (≠)
Available data types are DINT, REAL and STRING.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL, STRING	
IN2	DINT, REAL, STRING	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

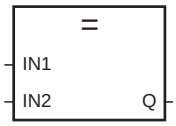
◆ Description of action

Q = TRUE when IN1 ≠ IN2.
Q = FALSE when IN1 = IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

= (Equal) Standard operator

SYMBOL



◆ Functional description

Compares 2 pieces of data. (=)
Available data types are DINT, REAL and STRING.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL, STRING	
IN2	DINT, REAL, STRING	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

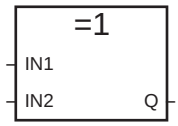
◆ Description of action

Q = TRUE when IN1 = IN2.
Q = FALSE when IN1 ≠ IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

=1 (XOR) (Exclusive OR) Standard operator

SYMBOL



◆ Functional description

Outputs the exclusive-OR of 2 pieces of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
IN1	BOOL	
IN2	BOOL	

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Exclusive-OR of inputs

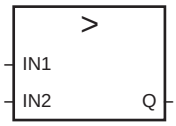
◆ Description of action

Q = XOR(IN1, IN2)

IN1	IN2	Q
TRUE	TRUE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	FALSE	FALSE

> (Greater than) Standard operator

SYMBOL



◆ Functional description

Compares 2 pieces of data. (>)
Any types of data (DINT, REAL, TIME and STRING) are available.

◆ Input parameters

Parameter	Data type	Content
IN1	Any types of data	
IN2	Any types of data	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

◆ Description of action

Q = TRUE when IN1 > IN2.
Q = FALSE when IN1 ≤ IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

>= (Greater or equal) Standard operator

SYMBOL

◆ Functional description

Compares 2 pieces of data. (≥)
Available data types are DINT, REAL and STRING.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT, REAL, STRING	
IN2	DINT, REAL, STRING	Should be of the same type as that of IN1.

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Result of comparison

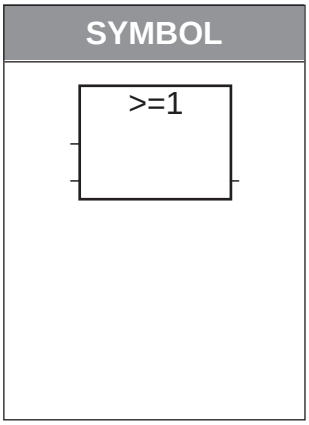
◆ Description of action

Q = TRUE when IN1 ≥ IN2.
Q = FALSE when IN1 < IN2.

- IN1 and IN2 should be of the same data type.
- In case of STRING, ASCII codes of the corresponding chracters of the strings are sequentially compared.

>=1 (OR) Standard operator

SYMBOL



◆ Functional description

Outputs the OR of two or more pieces of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
INPUTn	BOOL	n is between 2 and 32.

◆ Output parameters

Parameter	Data type	Content
OUTPUT	BOOL	OR of inputs

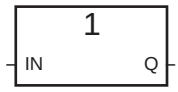
◆ Description of action

OUTPUT = INPUT1 OR INPUT2 OR .... OR INPUTn

The number of inputs n can be any number between 2 and 32.

1 gain (Assignment) Standard operator

SYMBOL



◆ Functional description

Assigns the value of data into a variable.
Any types of data (DINT, REAL, TIME and STRING) are available.

◆ Input parameters

Parameter	Data type	Content
IN	Any types of data	

◆ Output parameters

Parameter	Data type	Content
Q	Any types of data	

◆ Description of action

Q = IN

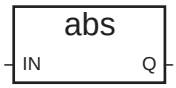
IN and Q should be of the same data type.

◆ Note

- This is a standard operator to assign a value into a variable. It is used in the LD language.
- In case of the FBD language, this standard operator is not necessary because a value is assigned to a variable by connecting them with a connection line.
- In case of the ST language, ":=" is used for assignment.

ABS (Absolute value) Function

SYMBOL



◆ Functional description

Outputs the absolute value of REAL type data.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	

◆ Output parameters

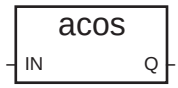
Parameter	Data type	Content
Q	REAL	Absolute value of IN

◆ Description of action

$Q = | IN |$

ACOS (Arc cosine) Function

SYMBOL



◆ Functional description

Outputs the arc cosine of REAL type data.
The output is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	-1.0 to +1.0

◆ Output parameters

Parameter	Data type	Content
Q	REAL	0.0 to π (radians)

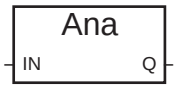
◆ Description of action

Q = acos(IN)

If an illegal value (i.e. a value not within the range between -1.0 and +1.0) is entered, Q = 0.0.

ANA (Convert to integer) Standard operator

SYMBOL



◆ Functional description

Converts a value to a DINT type one.
Any types of data can be converted.

◆ Input parameters

Parameter	Data type	Content
IN	Other than DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	

◆ Description of action

• When IN is of BOOL type:
Q = 0 if IN = FALSE or
Q = 1 if IN = TRUE.

• When IN is of REAL type:
Q = Integer part of IN (the fractional portion is dropped)
Example: If IN = 1234.56, Q = 1234.

• When IN is of TIME type:
Q = millisecond value
Example: If IN = T#1s46ms, Q = 1046.

• When IN is of STRING type:
Q = Message string in decimal representation
Example: If IN = '1234', Q = 1234.

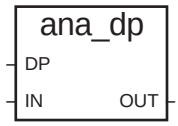
◆ Note

For the conversion from a REAL type value to a DINT type one, there is an extension, the ANA_DP (Real to integer conversion) function.

ANA_DP (Real to integer conversion)

Standard operator (for DMC50)

SYMBOL



◆ Functional description

Multiplies REAL data by 10^{DP} and then rounds it off to an integer.

◆ Input parameters

Parameter	Data type	Content
DP	DINT	Specification of the number of decimal places -30 to +30
IN	REAL	

◆ Output parameters

Parameter	Data type	Content
OUT	DINT	

◆ Description of action

OUT = IN X 10^{DP} (OUT is DINT type data with the fractional portion rounded off after the operation on the right-hand side was performed.)

If DP is not within the range between -30 and +30, operation is performed assuming that DP = 0.

[Sample of action]

DP	IN	OUT
0	123.456	123
1	123.456	1235
10	123.456e -10	123
-10	123.456e +10	123
-50	123.456	123
2	123.456e +10	Maximum value for DINT type
2	123.456e -10	0

◆ Note

This is an extension of ANA (Convert to integer).

AND_MASK (Bitwise AND) Standard operator

SYMBOL

◆ Functional description

Outputs the bitwise AND of 2 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	
MSK	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Bitwise AND of IN and MSK

◆ Description of action

Q = Bitwise AND of IN and MSK

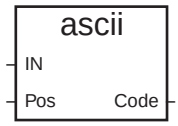
[Sample of action]

IN = 16#0000_0ABC= 2#0000_0000_0000_0000_0000_1010_1011_1100
MSK = 16#0000_0F0F = 2#0000_0000_0000_0000_0000_1111_0000_1111
Q = 16#0000_0A0C = 2#0000_0000_0000_0000_0000_1010_0000_1100

The '_' (underscore) used in the above "16#" or "2#" notations is only a delimiter to increase the readability of hexadecimal and binary numbers. It is neglected when evaluating the value of a number.

ASCII (Character → ASCII code conversion) Function

SYMBOL



◆ Functional description

Outputs the ASCII code corresponding to the character specified in a message string.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string other than empty
POS	DINT	Specifying the position of a character in a message string 1 to Len (Len is the IN message string length.)

◆ Output parameters

Parameter	Data type	Content
CODE	DINT	ASCII code of the specified character 0 to 255

◆ Description of action

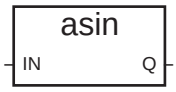
Outputs the ASCII code corresponding to the character at the position specified by POS in the message string IN.

• If POS exceeds the length of the message string, CODE = 0.
• If POS = 0, CODE = 0.

[Sample of action]
When IN = 'ABCDEFG' and POS = 3, CODE is 67 (43h).
When IN = 'ABCDEFG' and POS = 8, CODE is 0 (message string length exceeded).

ASIN (Arc sine) Function

SYMBOL



◆ Functional description

Outputs the arc sine of REAL type data.
The output is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	-1.0 to +1.0

◆ Output parameters

Parameter	Data type	Content
Q	REAL	-π/2 to +π/2 (radians)

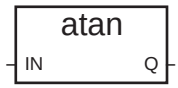
◆ Description of action

Q = asin(IN)

If an illegal value (i.e. a value not within the range between -1.0 and +1.0) is entered, Q = 0.0.

ATAN (Arc tangent) Function

SYMBOL



◆ Functional description

Outputs the arc tangent of REAL type data.
The output is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	Range of real numbers

◆ Output parameters

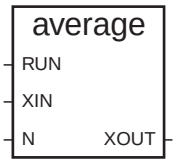
Parameter	Data type	Content
Q	REAL	-π/2 to +π/2 (radians)

◆ Description of action

Q = atan(IN)

AVERAGE (Moving average) Function block

SYMBOL



◆ Functional description

Outputs the mean value of the specified sample numbers of REAL type data.

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
XIN	REAL	Data
N	DINT	Number of samples for calculating the mean value 1 to 128

◆ Output parameters

Parameter	Data type	Content
XOUT	REAL	Mean value of N samples of XIN

◆ Description of action

If RUN = TRUE (Execution), IN is stored in the buffer at every cycle and the mean value of the latest N samples is output.
If RUN = FALSE (Reset), OUT = IN.

◆ Cautions

- The sampling time is defined in the cycle timing setting of the project as a fixed value.
- For N, make sure to specify a value between 1 and 128. If the value is out of this range, the action is unpredictable.
- If you change N (number of samples) while RUN = TRUE (Execution), the change is not reflected. It is necessary to make RUN equal to FALSE (Reset) once.

◆ Note

An extension, the MAV (Moving average) function block, is separately provided with the following features:
- Sampling time specification
- Maximum/minimum value dropping feature
- Saving the FB instance size with the sample count of 30 (compared with 128 samples for "AVERAGE")
- Efficient buffer utilization after initialization
- Allowing changes of N (number of samples) during execution

If the required sample count is 30 or less, it is recommended to use the MAV (Moving average) function block.

BIN3DEC (3-input boolean to integer conversion) Function (for DMC50)

SYMBOL

◆ Functional description

Converts 3 pieces of BOOL type data to 0-to-7 DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN1 to IN3	BOOL	

◆ Output parameters

Parameter	Data type	Content
OUT	DINT	0 to 7

◆ Description of action

$OUT = IN1 \times 2^0 + IN2 \times 2^1 + IN3 \times 2^2$

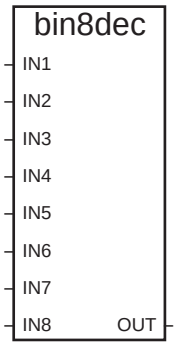
◆ Note

This function can be used to select an SP or PID group from binary data such as digital input.

BIN8DEC (8-input boolean to integer conversion)

Function (for DMC50)

SYMBOL



◆ Functional description

Converts 8 pieces of BOOL type data to a 0-to-255 DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN1 to IN8	BOOL	

◆ Output parameters

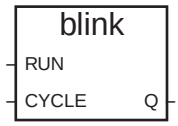
Parameter	Data type	Content
OUT	DINT	0 to 255

◆ Description of action

$$OUT = IN1 \times 2^0 + IN2 \times 2^1 + IN3 \times 2^2 + \ldots IN8 \times 2^7$$

BLINK (BOOL type signal blinking) Function block

SYMBOL



◆ Functional description

Generates blinking signals at every specified cycle.

◆ Input parameters

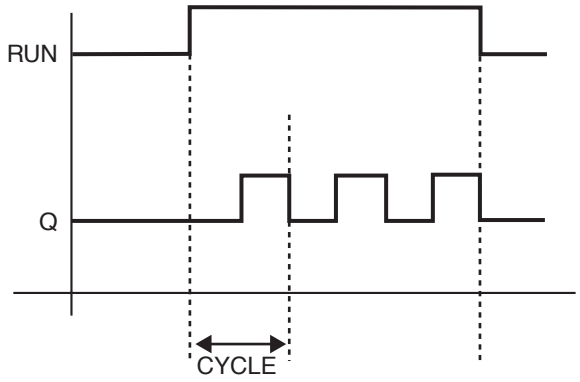
Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
CYCLE	TIME	Blinking cycle period, CYCLE ≥ the cycle timing setting

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Blinking output

◆ Description of action

If RUN = TRUE (Execution), Q repeatedly alternates between TRUE and FALSE at intervals of half the CYCLE period.
If RUN = FALSE (Reset), Q = FALSE.



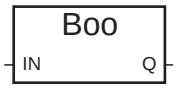
- Set CYCLE to an integral multiple of the cycle timing setting.
- If CYCLE = T#0s, Q repeatedly alternates between TRUE and FALSE at intervals of cycle timing setting.
- If CYCLE = The cycle timing setting, Q repeatedly alternates between TRUE and FALSE at intervals of the cycle timing setting.

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

BOO (Convert to boolean) Standard operator

SYMBOL



◆ Functional description

Converts a value data to a BOOL type one.
Any types of data can be converted.

◆ Input parameters

Parameter	Data type	Content
IN	Other than BOOL	

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	

◆ Description of action

• When IN is of DINT type:
Q = FALSE if IN = 0 or
Q = TRUE if IN ≠ 0.

• When IN is of REAL type:
Q = FALSE if IN = 0.0 or
Q = TRUE if IN ≠ 0.0.

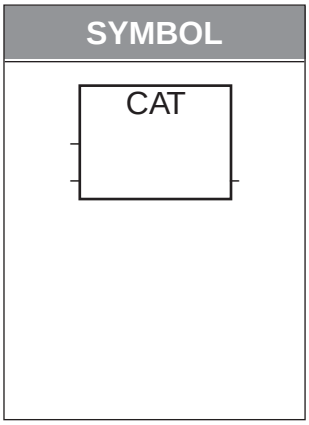
• When IN is of TIME type:
Q = FALSE if IN = T#0ms or
Q = TRUE if IN ≠ T#0ms.

• When IN is of STRING type:
Q = FALSE if IN = 'FALSE' or
Q = TRUE if IN = 'TRUE'. ('TRUE' should be in uppercase.)

Note: If IN is a message string other than 'TRUE', Q = FALSE.

CAT (Message string concatenation) Standard operator

SYMBOL



◆ Functional description

Concatenates two or more number of message strings into a single message string.

◆ Input parameters

Parameter	Data type	Content
INPUTn	STRING	Message string, n is a number between 2 and 32.

◆ Output parameters

Parameter	Data type	Content
OUTPUT	STRING	Concatenated message string

◆ Description of action

OUTPUT = Message string 1 + message string 2 + ... + message string n

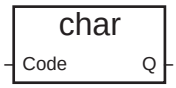
- The number of inputs n can be any number between 2 and 32.
- If the length of the message string concatenated by CAT exceeds the maximum length of the message string to be assigned, the characters as many as the maximum length are assigned starting from the first character (i.e. the character at the left end).

[Sample of action]

OUTPUT = 'ABC' + 'DEFG' + 'HIJK' = 'ABCDEFGHIJK'

CHAR (ASCII code → character conversion) Function

SYMBOL



◆ Functional description

Outputs the message string corresponding to the specified ASCII code.

◆ Input parameters

Parameter	Data type	Content
CODE	DINT	ASCII code 0 to 255

◆ Output parameters

Parameter	Data type	Content
Q	STING	Message string of one character

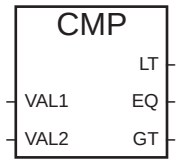
◆ Description of action

Outputs the one character string corresponding to the specified ASCII code.
If a value out of the range between 0 and 255 is passed to CODE, the value is divided by 256 with a remainder and the quotient is used.

[Sample of action]
If CODE = 65(16#41), Q = 'A'.
If CODE = 2625(16#A41), Q = 'A' (2625 / 256 = 10 with the remainder of 65).

CMP (Complete comparison) Function block

SYMBOL



◆ Functional description

Compares 2 pieces of DINT type data and outputs of either <, = or > as the result.

◆ Input parameters

Parameter	Data type	Content
VAL1	DINT	
VAL2	DINT	

◆ Output parameters

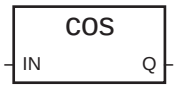
Parameter	Data type	Content
LT	BOOL	TRUE if VAL1 < VAL2.
EQ	BOOL	TRUE if VAL1 = VAL2.
GT	BOOL	TRUE if VAL1 > VAL2.

◆ Description of action

Condition	LT	EQ	GT
VAL1 < VAL2	TRUE	FALSE	FALSE
VAL1 = VAL2	FALSE	TRUE	FALSE
VAL1 > VAL2	FALSE	FALSE	TRUE

COS (Cosine) Function

SYMBOL



◆ Functional description

Outputs the cosine of REAL type data.
The input is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	Range of real numbers (in radians)

◆ Output parameters

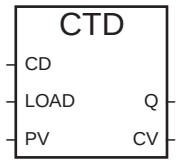
Parameter	Data type	Content
Q	REAL	-1.0 to +1.0

◆ Description of action

$Q = \cos(IN)$

CTD (Down-counter) Function block

SYMBOL



◆ Functional description

Outputs the countdown value of DINT type data.
Outputs the end signal when the count value has reached 0.

◆ Input parameters

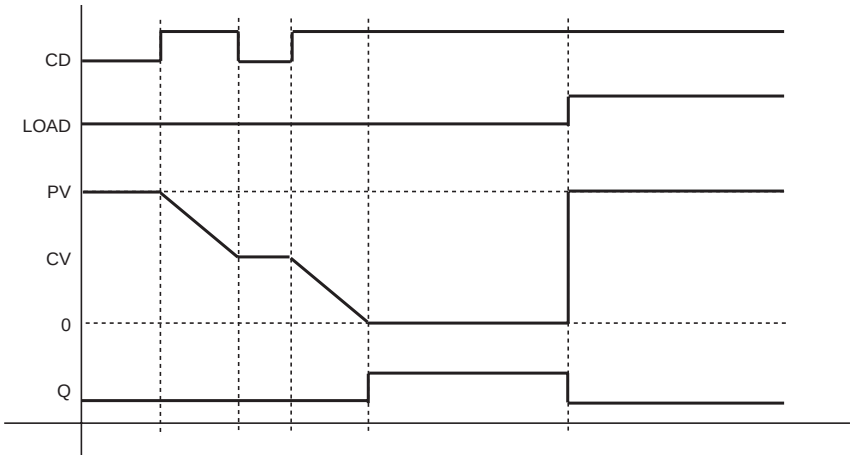
Parameter	Data type	Content
CD	BOOL	Countdown execution for TRUE, stop for FALSE
LOAD	BOOL	Load command (with higher priority) CV = PV for TRUE
PV	DINT	Initial count value, PV must > 0

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	End signal, TRUE if CV = 0
CV	DINT	Counter result

◆ Description of action

If LOAD = TRUE, CV is set to PV as the initial value of the counter (with the higher priority).
If CD = TRUE, CV decrements by one at every cycle.
When CV has reached 0, the counting-down action stops, making Q equal to TRUE.

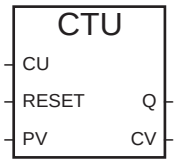


◆ Note

CTD can be used as a pulse counter by inputting pulse signals in CD.
(CD input cannot detect rising nor falling edge. When this is used as a pulse counter, it is necessary to input the pulses of one cycle time.)

CTU (Up-counter) Function block

SYMBOL



◆ Functional description

Outputs the countup value of DINT type data.
Outputs the end signal when the count value has reached the target value.

◆ Input parameters

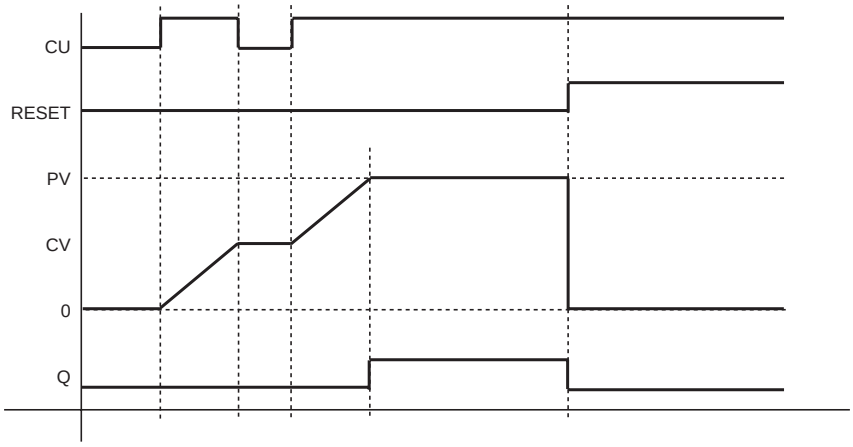
Parameter	Data type	Content
CU	BOOL	Countup execution for TRUE and stop for FALSE
RESET	BOOL	Reset command (with higher priority) CV = 0 for TRUE
PV	DINT	Counter target value, PV must > 0

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	End signal, TRUE if CV ≥ PV
CV	DINT	Counter result

◆ Description of action

If RESET = TRUE, CV is set to 0 as the initial value of the counter (with the higher priority).
If CU = TRUE, CV increments by one at every cycle.
When CV has reached PV (the target value), the counting-up action stops, making Q equal to TRUE.

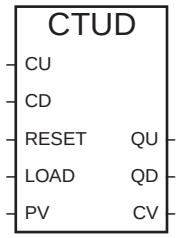


◆ Note

CTU can be used as a pulse counter by inputting pulse signals in CU.
(CU input cannot detect rising nor falling edge. When this is used as a pulse counter, it is necessary to input the pulses of one cycle time.)

CTUD (Up/down counter) Function block

SYMBOL



◆ Functional description

Integration of CTU (Up-counter) and CTD (Down-counter)

◆ Input parameters

Parameter	Data type	Content
CU	BOOL	Countup execution (higher priority than CD) for TRUE and stop for FALSE
CD	BOOL	Countdown execution for TRUE and stop for FALSE
RESET	BOOL	Reset command (higher priority than LOAD, CU and CD) CV = 0 for TRUE
LOAD	BOOL	Load command (higher priority than CU and CD) CV = PV for TRUE
PV	DINT	Counter target value, PV must > 0

◆ Output parameters

Parameter	Data type	Content
QU	BOOL	Up-counter end signal, TRUE if CV ≥ PV
QD	BOOL	Down-counter end signal, TRUE if CV = 0
CV	DINT	Counter result

◆ Description of action

For details,

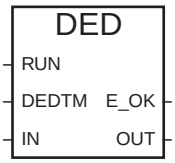
☞refer to CTU (Up-counter) on page 2-40 and CTD (Down-counter) on page 2-39.

- If both CU and CD are TRUE, operation is performed giving CU a higher priority. (Counting-up operation for CV)
- If both RESET and LOAD are TRUE, operation is performed giving RESET a higher priority (assuming CV = 0).

DED (Dead time)

Function Block (for DMC50)

SYMBOL



◆ Functional description

Outputs REAL type data after the dead time has elapsed.

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
DEDTM	TIME	Dead time
IN	REAL	

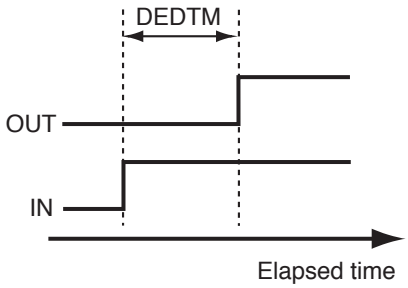
◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	REAL	IN after dead time

◆ Description of action

If RUN = TRUE (Execution), IN is output after DEDTM (dead time) has elapsed.
If RUN = FALSE (Reset), OUT = IN.

The number of buffers used for internal operation is 100. As the buffers are cyclically updated among these 100 buffers, if the dead time setting is long, the outputs are produced stepwise like staircases.
For example, when DEDTM = T#300s and the cycle timing setting = T#300ms, the output changes in increments of 3s.
If "DEDTM / cycle timing setting ≤ 100", buffers are updated at every application execution cycle.



◆ Note

Even though the dead time setting is long (i.e. DEDTM / cycle timing setting > 100), if you want to update buffers at every application execution cycle, connect two or more number of DED function blocks.

◆ Cautions

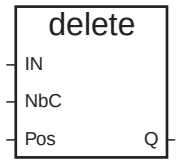
Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If the cycle timing setting of the project is 0, an execution error will occur.
The action is as follows when an execution error has occurred:
• OUT = IN
• E_OK = FALSE (Error)

DELETE (Message string deletion) Function

SYMBOL



◆ Functional description

Deletes a specified message string from a base message string.
It is possible to specify the position and the length of the string to be deleted.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Base message string
NBC	DINT	Number of characters to be deleted, NBC > 1
POS	DINT	First character position of the characters to be deleted, POS > 1

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Message string modified by deletion

◆ Description of action

Deletes NBC (number of characters) characters from the message string specified by IN starting from POS (a position in the message string).

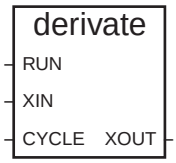
[Sample of action]
When IN = ‘ABCDEFGH’, NBC = 4, and POS = 3, Q is ‘ABGH’.
When IN = ‘ABCDEFGH’, NBC = 2, and POS = 1, Q is ‘CDEFGH’.

◆ Note

- If POS = 0, Q is a empty message string.
- If "POS > Message string length of IN" or "POS < 0", Q is IN.
  When IN = ‘ABCDEFGH’, NBC = 1, and POS = 9, Q is ‘ABCDEFGH’.
  When IN = ‘ABCDEFGH’, NBC = 2, and POS = -1, Q is ‘ABCDEFGH’.
- If "POS + NBC > Message string length of IN", only the valid part is deleted.
  When IN = ‘ABCDEFGH’, NBC = 3, and POS = 7, Q is ‘ABCDEF’.
  When IN = ‘ABCDEFGH’, NBC = 10, and POS = 1, Q is empty string.
- If "NBC <= 0", Q is an undefined message string.

DERIVATE (Derivative) Function block

SYMBOL



◆ Functional description

Outputs the derivative of REAL type data.

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution of derivative action, FALSE = Reset
XIN	REAL	Input data
CYCLE	TIME	Sampling time, CYCLE ≥ the cycle timing setting

◆ Output parameters

Parameter	Data type	Content
XOUT	REAL	Result of derivative action

◆ Description of action

Generates the output XOUT proportional to the changing rate of input XIN.

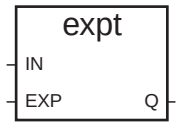
- CYCLE sets the execution interval.
- If RUN = TRUE, the derivative is calculated.
- If RUN = FALSE, a reset action is performed, setting XOUT to 0.0.

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

EXPT (Exponential function) Function

SYMBOL



◆ Functional description

Performs exponential operation on REAL type data and outputs the result.

The exponent number is specified as DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	Base number
EXP	DINT	Exponent number

◆ Output parameters

Parameter	Data type	Content
Q	REAL	$Q = IN^{EXP}$

◆ Description of action

$Q = IN^{EXP}$

◆ Cautions

- EXP (Exponent number) is specified as DINT type data. If you want to specify it as REAL type data, use the POW function.
- If you set a large number for EXP, use the POW function.

In the internal arithmetic, the operation is repeated as many times as EXP (like Q = IN X IN X...IN). Consequently, if EXP value is large, it takes a very long time to compute it. In some cases, it seems as if the system has entered an infinite loop.

F_TRIG (Falling edge detection) Function block

SYMBOL

◆ Functional description

Detects the falling edge of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
CLK	BOOL	

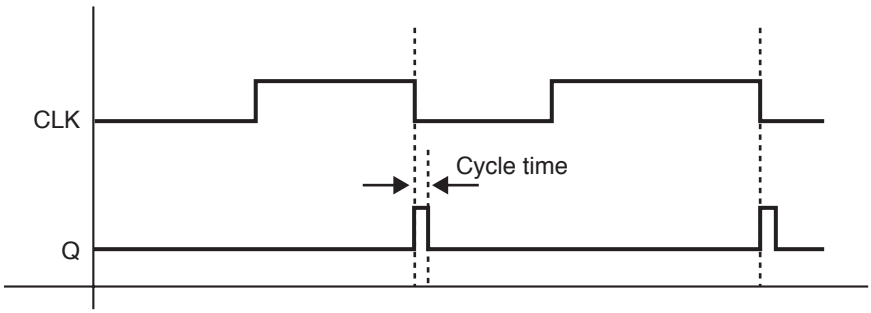
◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Q is TRUE when CLK is changed from TRUE to FALSE. FALSE in other cases.

◆ Description of action

When the falling edge of CLK is detected, Q is TRUE (for one cycle).
In other cases, Q is FALSE.

Falling edge is detected by comparing with the value of the previous cycle time.



FIND (Message string finding) Function

SYMBOL

◆ Functional description

Searches for the specified message string and outputs its position.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string
PAT	STRING	String pattern to be searched for

◆ Output parameters

Parameter	Data type	Content
POS	DINT	Position of the string pattern found

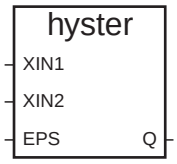
◆ Description of action

Searches the string pattern specified by PAT in the message string specified by IN.
The search is upper / lower case sensitive.
If a specified string pattern was found, the position where the pattern was found first is output to POS.
If there is no pattern found, POS is 0.

[Sample of action]
When IN = 'ABCDEFGH', and PAT = 'EFG', POS is 5.
When IN = 'ABCDABCD', and PAT = 'BC', POS is 2.
When IN = 'ABCDEFGH', and PAT = 'XYZ', POS is 0 (i.e. no pattern found).

HYSTER (Hysteresis) Function block

SYMBOL



◆ Functional description

Determines with hysteresis whether the REAL type data has exceeded the high limit value.

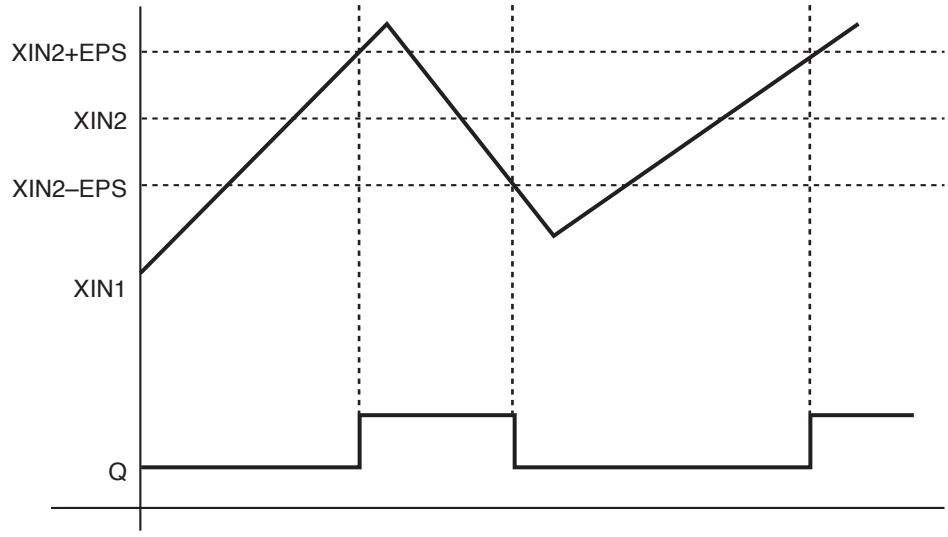
◆ Input parameters

Parameter	Data type	Content
XIN1	REAL	
XIN2	REAL	High limit value
EPS	REAL	Hysteresis value, EPS ≥ 0.0

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	

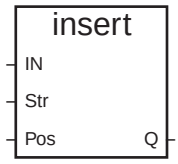
◆ Description of action



If EPS < 0.0, the action is unpredictable.

INSERT (Message string insertion) Function

SYMBOL



◆ Functional description

Inserts a specified message string into a base message string.
It is possible to specify the position for insertion.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Base message string
STR	STRING	Message string to be inserted
POS	DINT	Character position for insertion, POS > 1

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Message string modified by insertion

◆ Description of action

Inserts the message string STR into the message string specified by IN at POS (a position in the message string).

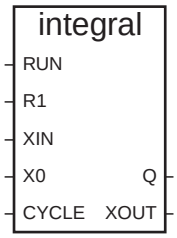
[Sample of action]
When IN = ‘ABCD’, STR = ‘##’, POS = 2, Q is ‘A##BCD’.

◆ Note

- If POS = 0, Q = empty message string.
- If POS < 0, Q is an undefined message string.
- If "POS > Message string length of IN", STR is concatenated with the end of IN.
  When IN = ‘ABCD’, STR = ‘##’, and POS = 9, Q is ‘ABCD##’.
- If "message string length after INSERT > allowable character string length of the variable connected to Q", the excess message string beyond the allowable message string length is deleted.

INTEGRAL (Integral) Function block

SYMBOL



◆ Functional description

Outputs the integral of REAL type data

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Integral execution, FALSE = Hold
R1	BOOL	Reset
XIN	REAL	Input data
X0	REAL	Initial value
CYCLE	TIME	Sampling time, CYCLE ≥ the cycle timing setting

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Inversion of R1
XOUT	REAL	Result of integration

◆ Description of action

Integrates the value of XIN over time.

- XOUT can be reset to X0 by making R1 equal to TRUE.
- CYCLE is set at the execution interval.
- XOUT is integrated while RUN = TRUE, and the integral is held at other times.
- Q is kept TRUE as long as the integral is not reset.

◆ Cautions

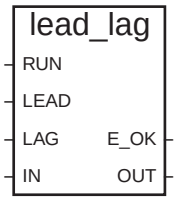
Must be executed at every application execution cycle to perform time management internally.

MEMO

LEAD_LAG (Lead / lag)

Function block (for DMC50)

SYMBOL



◆ Functional description

Outputs the lead / lag calculated value of REAL type data.
Can be used also as a first order digital filter.

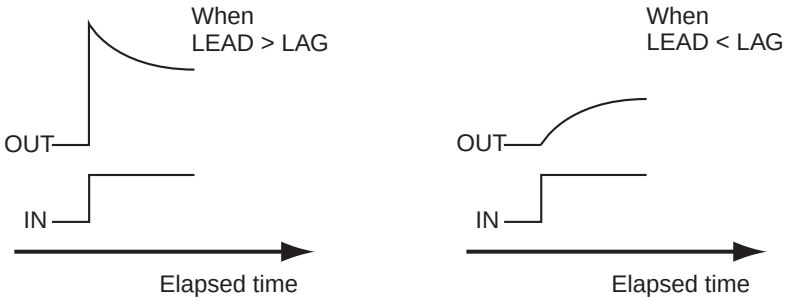
◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
LEAD	REAL	Lead time constant, LEAD ≥ 0.0 (unit: s)
LAG	REAL	Lag time constant, LAG ≥ 0.0 (unit: s)
IN	REAL	

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	REAL	Calculated value

◆ Description of action



If RUN = TRUE (Execution), the following operation is performed.
OUT = OUT_1 + (A X (IN - OUT_1) + B X (IN - IN_1))
A : Ts / (Ts + LAG)
B : LEAD / (Ts + LAG)
Ts : The cycle timing setting of the project (unit: s)
OUT_1: Previous OUT
IN_1 : Previous IN

If RUN = FALSE (Reset), OUT = IN.
When both LEAD and LAG are 0.0, OUT = IN.

◆ Note

If LEAD = 0.0, only the lag is left, and this function block can be used as a first order digital filter.

◆ Cautions

- Must be executed at every application execution cycle to perform time management internally.
- Though the unit of LEAD and LAG is time, they are input as REAL type data.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur.

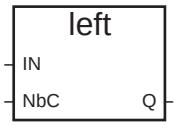
- The cycle timing setting of the project is 0.
- "LEAD < 0.0" or "LAG < 0.0".

The action is as follows when an execution error has occurred:

- OUT = IN
- E_OK = FALSE (Error)

LEFT (Left message string extraction) Function

SYMBOL



◆ Functional description

Extracts the specified message string from the left hand side of a base message string.
It is possible to specify the length of the string to be extracted.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string
NBC	DINT	Number of characters to be extracted, 0 < NBC ≤ Message string length of IN

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Extracted message string

◆ Description of action

Extracts characters of the length specified by NBC from the left hand side of the message string specified by IN.

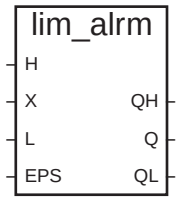
[Sample of action]
When IN = ‘ABCDEFGH’, and NBC = 2, Q is ‘AB’.
When IN = ‘ABCDEFGH’, and NBC = 4, Q is ‘ABCD’.

◆ Note

• If NBC = 0, Q is a empty message string.
• If NBC < 0, Q is IN.
• If "NBC >= Message string length of IN", Q is IN.

LIM_ALRM (Limit alarm) Function block

SYMBOL



◆ Functional description

Determines with hysteresis whether the REAL type data has exceeded the high limit value or whether it has come down below the low limit value respectively.

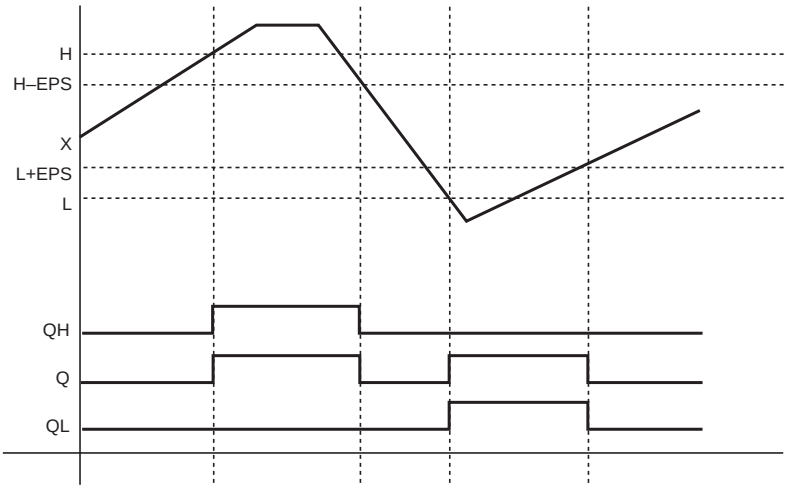
◆ Input parameters

Parameter	Data type	Content
H	REAL	High limit value
X	REAL	Input data
L	REAL	Low limit value
EPS	REAL	Hysteresis value, EPS ≥ 0.0

◆ Output parameters

Parameter	Data type	Content
QH	BOOL	High limit alarm
Q	BOOL	High limit/low limit alarms
QL	BOOL	Low limit alarm

◆ Description of action

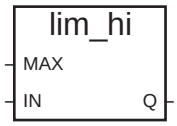


If EPS < 0.0, the action is unpredictable.

LIM_HI (REAL type high limit)

Function (for DMC50)

SYMBOL



◆ Functional description

Limits REAL type data by the high limit value.

◆ Input parameters

Parameter	Data type	Content
MAX	REAL	High limit value
IN	REAL	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	Value limited by the high limit value

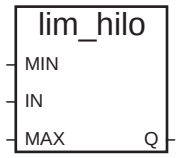
◆ Description of action

If IN ≥ MAX, Q is MAX.
If IN < MAX, Q is IN.

LIM_HILO (REAL type high/low limit)

Function (for DMC50)

SYMBOL



◆ Functional description

Limits REAL type data by the high and low limit values.

◆ Input parameters

Parameter	Data type	Content
MIN	REAL	Low limit value
IN	REAL	
MAX	REAL	High limit value

◆ Output parameters

Parameter	Data type	Content
Q	REAL	Value limited by the high and low limit values

◆ Description of action

If IN ≤ MIN, Q is MIN.
If MIN < IN < MAX, Q is IN.
If IN ≥ MAX, Q is MAX.

If MIN is set greater than MAX, the determination is made after interchanging MIN and MAX values.

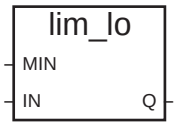
◆ Note

If you want to limit DINT type data by the high and low limit values, use the LIMIT (High / low limit) function.

LIM_LO (REAL type low limit)

Function (for DMC50)

SYMBOL



◆ Functional description

Limits REAL type data by the low limit value.

◆ Input parameters

Parameter	Data type	Content
MIN	REAL	Low limit value
IN	REAL	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	Value limited by the low limit value

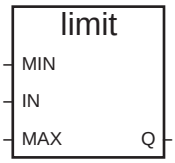
◆ Description of action

If IN ≤ MIN, Q is MIN.

If IN > MIN, Q is IN.

LIMIT (High / low limit) Function

SYMBOL



◆ Functional description

Limits DINT type data by the high and low limit values.

◆ Input parameters

Parameter	Data type	Content
MIN	DINT	Low limit value
IN	DINT	
MAX	DINT	High limit value

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Value limited by the high and low limit values

◆ Description of action

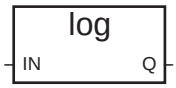
If IN ≤ MIN, Q is MIN.
If MIN < IN < MAX, Q is IN.
If IN ≥ MAX, Q is MAX.

◆ Note

If you want to limit REAL type data by the high and low limit values, use the LIM_HILO (Real type high/low limit) function.

LOG (Common logarithm) Function

SYMBOL



◆ Functional description

Outputs the common logarithm of REAL type data.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	IN > 0.0

◆ Output parameters

Parameter	Data type	Content
Q	REAL	$Q = LOG_{10}(IN)$

◆ Description of action

$$Q = LOG_{10}(IN)$$

◆ Note

The natural logarithm (log_e) is not supported. The natural logarithm is given by the following equation:

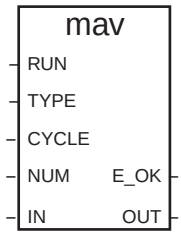
$$\log_e \chi = \log_{10} \chi \times \log_e 10 = \log_{10} \chi \times 2.30258$$

MEMO

MAV (Moving average)

Function block (for DMC50)

SYMBOL



◆ Functional description

Outputs the mean value of the specified sample numbers of REAL type data.
This is an extension of the AVERAGE (Moving average) function block.

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
TYPE	DINT	Averaging type 0 = Normal, 1 = Drops the maximum and minimum values
CYCLE	TIME	Sampling time (= Update time of the moving average)
NUM	DINT	Number of samples for calculating the mean value, 1 to 30
IN	REAL	Data

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	REAL	Mean value of NUM samples of IN

◆ Description of action

If RUN = TRUE (Execution), IN is stored in the buffer at every CYCLE (sampling time) and the mean value of the latest NUM samples is output. If TYPE = 1, the maximum and minimum values are removed from the buffered data and the mean value of the remaining data is output.

If RUN is changed from FALSE to TRUE, the mean value of the valid data is output until the buffer is filled with NUM samples of data.
If RUN = FALSE (Reset), OUT = IN.
The actual sampling time, "CYCLE'" is determined as follows:
CYCLE' = (Quotient of CYCLE / Ts) X Ts
where,
Ts : The cycle timing setting of the project (ms)
CYCLE : Sampling time (ms)

Example 1: When Ts = T#100ms and CYCLE = T#500ms
CYCLE / Ts = 500 / 100 = 5 Remainder: 0
CYCLE' = 5 X 100 = 500ms
Consequently, operation is performed with the sampling time of 500ms.
(A case where the CYCLE is divisible)

Example 2: When Ts = T#100ms and CYCLE = T#250ms
CYCLE / Ts = 250 / 100 = 2 Remainder: 50
CYCLE' = 2 X 100 = 200ms
Consequently, operation is performed with the sampling time of 200ms.

Example 3: When Ts = T#150ms and CYCLE = T#10000ms (i.e. 10s)
CYCLE / Ts = 10000 / 150 = 66 Remainder: 100
CYCLE' = 66 X 150 = 9900ms
Consequently, operation is performed with the sampling time of 9900ms (i.e. 9.9s).

◆ Note

Compared with the AVERAGE (Moving average) function block, this extension has the following additional features:
• Sampling time specification
• Maximum/minimum value dropping feature
• Saving the FB instance size with the sample count of 30 (compared with 128 samples for "AVERAGE")
• Efficient buffer utilization after initialization
• Allowing changes of N (number of samples) during execution.

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

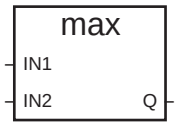
◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur.
• NUM ≤ 0 or NUM > 30.
• CYCLE ≤ 0 or CYCLE < the cycle timing setting.
• TYPE = neither 0 nor 1.
• The cycle timing setting of the project is 0s.

The action is as follows when an execution error has occurred:
• OUT = IN
• E_OK = FALSE (Error)

MAX (Maximum value) Function

SYMBOL



◆ Functional description

Compares 2 pieces of DINT type data and outputs the maximum value.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT	
IN2	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	

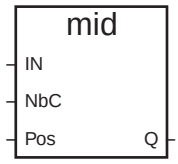
◆ Description of action

If IN1 > IN2, Q is IN1.

If IN1 ≤ IN2, Q is IN2.

MID (Message string extraction) Function

SYMBOL



◆ Functional description

Extracts the specified message string from a base message string.
It is possible to specify the position and the length of the string to be extracted.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string
NBC	DINT	Number of characters to be extracted, 0 < NBC ≤ Message string length of IN
POS	DINT	First character position of the characters to be extracted, POS > 1

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Extracted message string

◆ Description of action

Extracts characters of the length specified by NBC from the message string specified by IN starting from the position specified by POS.

[Sample of action]
When IN = ‘ABCDEFGH’, NBC = 2, and POS = 1, Q is ‘AB’.
When IN = ‘ABCDEFGH’, NBC = 4, and POS = 3, Q is ‘CDEF’.

◆ Note

- If POS ≤ 0, Q is a empty message string.
- If NBC = 0, Q is a empty message string.
- If NBC < 0, Q is an undefined message string.
- If "POS + NBC ≥ Message string length of IN", Q is an undefined message string.

MIN (Minimum value) Function

SYMBOL

◆ Functional description

Compares 2 pieces of DINT type data and outputs the minimum value.

◆ Input parameters

Parameter	Data type	Content
IN1	DINT	
IN2	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	

◆ Description of action

If IN1 < IN2, Q is IN1.

If IN1 ≥ IN2, Q is IN2.

MLEN (Message string length) Function

SYMBOL

◆ Functional description

Outputs the message string length.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string

◆ Output parameters

Parameter	Data type	Content
NbC	DINT	Message string length of IN

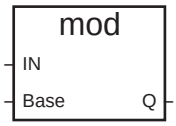
◆ Description of action

Q = Message string length of IN

[Sample of action]
When IN = ‘ABCDEFGH’, NbC = 8.
When IN = Empty message string, NbC = 0.

MOD (Modulus) Function

SYMBOL



◆ Functional description

Outputs the remainder of DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	
BASE	DINT	BASE > 0

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Remainder of "IN / BASE"

◆ Description of action

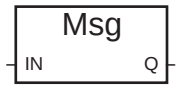
Q = Remainder of "IN / BASE"

[Sample of action]
When IN = 10 and BASE = 2, Q is 0.
When IN = 10 and BASE = 3, Q is 1.
When IN = -10 and BASE = 3, Q is -1.

If BASE ≤ 0, Q is -1.
When IN = 10 and BASE = 0, Q is -1.

MSG (Convert to message string) Standard operator

SYMBOL



◆ Functional description

Converts a value to a STRING type one.
Any types of data can be converted.

◆ Input parameters

Parameter	Data type	Content
IN	Other than STRING	

◆ Output parameters

Parameter	Data type	Content
Q	STRING	

◆ Description of action

• When IN is of BOOL type:
Q = 'FALSE' if IN = FALSE or
Q = 'TRUE' if IN = TRUE.

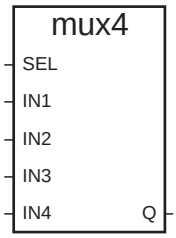
• When IN is of DINT type:
Q = integer message string
Example: If IN = 123, Q = '123'.

• When IN is of REAL type:
Q = Message string of the integer part of IN (the fractional portion is dropped)
Example: If IN = 123.45, Q = '123'.

• When IN is of TIME type:
Q = Message string of TIME type data
Example: If IN = T#7s200ms, Q = '7s200'.

MUX4 (4-input multiplexer) Function

SYMBOL



◆ Functional description

Selects one out of 4 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
SEL	DINT	Selector value 0 to 3
IN1 to IN4	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Selected IN

◆ Description of action

When SEL = 0, Q = IN1.
When SEL = 1, Q = IN2.
When SEL = 2, Q = IN3.
When SEL = 3, Q = IN4.
When SEL = Other than a value between 0 and 3, Q = 0.

◆ Note

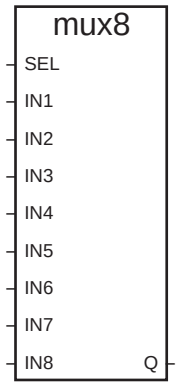
If you handle REAL type data, use the MUX8REAL (Real type 8-input multiplexer) function.

◆ Cautions

Note that the SEL value starts from 0 (0 to 3) and the IN number starts from 1 (IN1 to IN4).

MUX8 (8-input multiplexer) Function

SYMBOL



◆ Functional description

Selects one out of 8 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
SEL	DINT	Selector value 0 to 7
IN1 to IN8	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Selected IN

◆ Description of action

When SEL = 0, Q = IN1.
When SEL = 1, Q = IN2.
•
•
•
When SEL = 7, Q = IN8.
When SEL = Other than a value between 0 and 7, Q = 0.

◆ Note

If you handle REAL type data, use the MUX8REAL (Real type 8-input multiplexer) function.

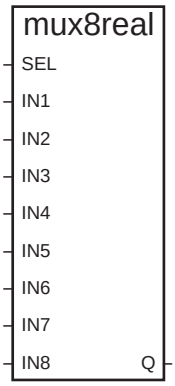
◆ Cautions

Note that the SEL value starts from 0 (0 to 7) and the IN number starts from 1 (IN1 to IN8).

MUX8REAL (REAL type 8-input multiplexer)

Function (for DMC50)

SYMBOL



◆ Functional description

Selects one out of 8 pieces of REAL type data.
This is an extension (supporting REAL type) of MUX8 (8-input multiplexer).

◆ Input parameters

Parameter	Data type	Content
SEL	DINT	Selector value 0 to 7
IN1 to IN8	REAL	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	Selected IN

◆ Description of action

When SEL = 0, Q = IN1.
When SEL = 1, Q = IN2.
•
•
•
When SEL = 7, Q = IN8.
When SEL = Other than a value between 0 and 7, Q = 0.0.

◆ Note

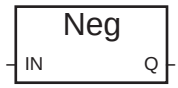
If you handle DINT type data, use the MUX4 or MUX8 function.

◆ Cautions

Note that the SEL value starts from 0 (0 to 7) and the IN number starts from 1 (IN1 to IN8).

NEG (Sign change) Standard operator

SYMBOL



◆ Functional description

Changes the sign of data.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
IN	DINT, REAL	

◆ Output parameters

Parameter	Data type	Content
Q	DINT, REAL	Q = -IN

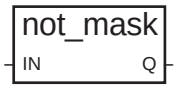
◆ Description of action

Q = -IN

IN and Q should be of the same data type.

NOT_MASK (Bitwise inversion) Standard operator

SYMBOL



◆ Functional description

Inverts and outputs DINT type data bitwise.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Bitwise inversion of IN

◆ Description of action

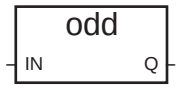
Q = Bitwise inversion of IN

[Sample of action]
IN = 16#0000_1234 = 2#0000_0000_0000_0000_0001_0010_0011_0100
Q = 16#FFFF_EDCB = 2#1111_1111_1111_1111_1110_1101_1100_1011

The '_' (underscore) used with "16#" or "2#" is only a delimiter to increase the readability of hexadecimal and binary numbers. It is neglected when evaluating the value of a number.

ODD (Odd parity) Function

SYMBOL



◆ Functional description

Determines whether the DINT type data is odd or even.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	

◆ Output parameters

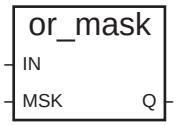
Parameter	Data type	Content
Q	BOOL	TRUE if IN is odd, or FALSE if IN is even.

◆ Description of action

If IN is odd, Q is TRUE.
If IN is even, Q is FALSE.

OR_MASK (Bitwise OR) Standard operator

SYMBOL



◆ Functional description

Outputs the bitwise OR of 2 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	
MSK	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Bitwise OR of IN and MASK

◆ Description of action

Q = Bitwise OR of IN and MASK

[Sample of action]

IN = 16#0000_3333 = 2#0000_0000_0000_0000_0011_0011_0011_0011

MSK = 16#0000_8421 = 2#0000_0000_0000_0000_1000_0100_0010_0001

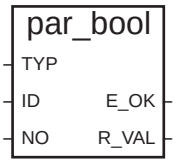
Q = 16#0000_B733 = 2#0000_0000_0000_0000_1011_0111_0011_0011

The '_' (underscore) used with "16#" or "2#" is only a delimiter to increase the readability of hexadecimal and binary numbers. It is neglected when evaluating the value of a number.

PAR_BOOL (Read BOOL type Parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Reads a Parameter element value as BOOL type data.
The word "Parameter" refers to data specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
R_VAL	BOOL	Read-out data

◆ Description of action

Reads the Parameter element value specified by TYP (Parameter type ID), ID (group ID) and NO (item ID) as BOOL type data.
When reading, type conversion is performed depending on the Parameter data type.

Parameter data type	Type conversion (Parameter value → R_VAL)
BOOL	No conversion
DINT, DWORD	0 → FALSE, Other than 0 → TRUE
REAL	0.0 → FALSE, Other than 0.0 → TRUE

◆ Action under abnormal conditions

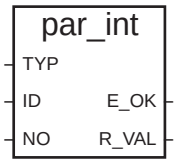
If the specified Parameter element element does not exist, an execution error will occur.

The action is as follows when an execution error has occurred:
• R_VAL = FALSE
• E_OK = FALSE (Error)

PAR_INT (Read DINT type Parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Reads a Parameter element value as DINT type data.
The word "Parameter" refers to data specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
R_VAL	DINT	Read-out data

◆ Description of action

Reads the Parameter element value specified by TYP (Parameter type ID), ID (group ID) and NO (item ID) as DINT type data.
When reading, type conversion is performed depending on the Parameter data type.

Parameter data type	Type conversion (Parameter value → R_VAL)
BOOL	FALSE → 0, TRUE → 1
DINT, DWORD	No conversion
REAL	N.XXX→ N Fractional portion dropped Limited by the DINT type maximum/minimum values

◆ Action under abnormal conditions

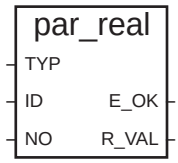
If the specified Parameter data element does not exist, an execution error will occur.

The action is as follows when an execution error has occurred:
• R_VAL = 0
• E_OK = FALSE (Error)

PAR_REAL (Read REAL type parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Reads a Parameter element value as REAL type data.
The word "Parameter" refers to data specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
R_VAL	REAL	Read-out data

◆ Description of action

Reads the Parameter element value specified by TYP (parameter type ID), ID (group ID) and NO (item ID) as REAL type data.
When reading, type conversion is performed depending on the Parameter data type.

◆ Action under abnormal conditions

Parameter data type	Type conversion (Parameter value → R_VAL)
BOOL	FALSE → 0.0, TRUE → 1.0
DINT	N → N.0
DWORD	Do not use. The action is unpredictable.
REAL	No conversion

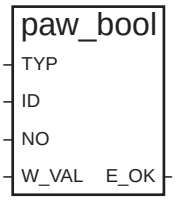
If the specified Parameter data element does not exist, an execution error will occur.

The action is as follows when an execution error has occurred:
• R_VAL = FALSE
• E_OK = FALSE (Error)

PAW_BOOL (Write BOOL type Parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Writes BOOL type data into a Parameter data element.
The word "Parameter" refers to that specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID
W_VAL	BOOL	Data to be written

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error

◆ Description of action

Writes into the Parameter data element specified by TYP (Parameter type ID), ID (group ID) and NO (item ID) as BOOL type data.
When writing, type conversion is performed depending on the Parameter data type.

Parameter data type	Type conversion (W_VAL → value to be written into the Parameter data element)
BOOL	No conversion
DINT, DWORD	FALSE → 0, TRUE → 1
REAL	FALSE → 0.0, TRUE → 1.0

◆ Action under abnormal conditions

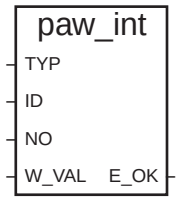
An execution error will occur if the specified Parameter data element does not exist, access is prohibited or the value is out of the allowable range.

The action is as follows when an execution error has occurred:
• No writing is performed.
• E_OK = FALSE (Error)

PAW_INT (Write DINT type Parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Writes DINT type data into a Parameter data element.
The word "Parameter" refers to data specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID
W_VAL	DINT	Data to be written

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error

◆ Description of action

Writes into the Parameter data element specified by TYP (Parameter type ID), ID (group ID) and NO (item ID) as DINT type data.
When writing, type conversion is performed depending on the Parameter data type.

Parameter data type	Type conversion (W_VAL → value to be written into the Parameter data element)
BOOL	0 → FALSE, Other than 0 → TRUE
DINT, DWORD	No conversion
REAL	N → N.0

◆ Action under abnormal conditions

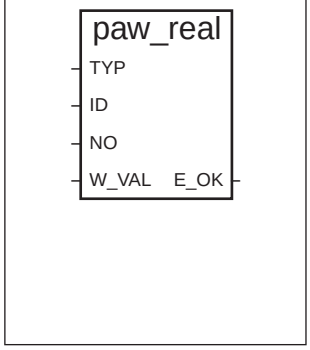
An execution error will occur if the specified Parameter data element does not exist, access is prohibited or the value is out of the allowable range.

The action is as follows when an execution error has occurred:
• No writing is performed.
• E_OK = FALSE (Error)

PAW_REAL (Write REAL type Parameter)

Function block (for DMC50)

SYMBOL



◆ Functional description

Writes REAL type data into a Parameter data element.
The word "Parameter" refers to data specifically provided for DMC50 such as "System Parameters", "Calculation Parameters", "Program Pattern Parameters", "System Monitor Parameters", "Calculation Monitor Parameters", and "User-defined Parameters".

◆ Input parameters

Parameter	Data type	Content
TYP	DINT	Parameter type ID
ID	DINT	Group ID
NO	DINT	Item ID
W_VAL	REAL	Data to be written

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error

◆ Description of action

Writes into the Parameter data element specified by TYP (Parameter type ID), ID (group ID) and NO (item ID) as REAL type data.
When writing, type conversion is performed depending on the Parameter data type

Parameter data type	Type conversion (W_VAL → value to be written into the Parameter data element)
BOOL	0 → FALSE, Other than 0 → TRUE
DINT	N.XXX → N (fractional portion dropped)
DWORD	Do not use. The action is unpredictable.
REAL	No conversion

◆ Action under abnormal conditions

An execution error will occur if the specified Parameter data element does not exist, access is prohibited or the value is out of the allowable range.

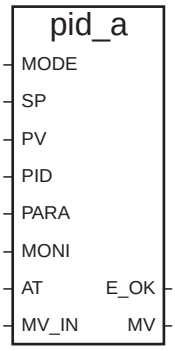
The action is as follows when an execution error has occurred:
• No writing is performed.
• E_OK = FALSE (Error)

MEMO

PID_A (Standard PID operation)

Function block (for DMC50)

SYMBOL



◆ Functional description

Performs a series form PID operation of which derivative action is applied on error.

◆ Input parameters

Parameter	Data type	Content
MODE	BOOL	TRUE = MANUAL mode, FALSE = AUTO mode
SP	REAL	SP (industrial unit)
PV	REAL	PV (industrial unit)
PID	DINT	Specifies the group ID of PID_A Constants.
PARA	DINT	Specifies the group ID of PID_A Options.
MONI	DINT	Specifies the group ID of PID_A Monitor.
AT	BOOL	Starts/stops auto tuning.
MV_IN	REAL	Inputs the value to be output in MANUAL mode.

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
MV	REAL	Manipulated variable

◆ Description of action

The PID_A function block is a series form PID operation of which derivative action is applied on error. It has the following additional features:

- AUTO / MANUAL transfer control
- Auto tuning (AT)
- Smart tuning (ST)
- 2 degrees of freedom PID control
- PID control with dead band

The PID operation is performed in AUTO mode (MODE = FALSE).

- The constants used in the PID operation (such as the PID constants, output high/low limit values, etc.) are grouped in the Calculation Parameter "PID_A Constants". (Specify the group ID of the "PID_A Constants" in PID.)
- The options used in the PID operation (such as the direct / reverse action, auto tuning method option, etc.) are grouped in the Calculation Parameter "PID_A Options". (Specify the group ID of the "PID_A Options" in PARA.)
- If you want to monitor the PID_A input/output data (such as PV, SP and MV) easily en bloc, use the Calculation Monitor Parameter "PID_A Monitor". (Specify the group ID of the "PID_A Monitor" in MONI. If you do not use this data, specify "0".)

In MANUAL mode (MODE = TRUE), MV = MV_IN.

- When the mode changes from AUTO to MANUAL, preset value output or bumpless transfer can be specified optionally. (This option is specified in the Calculation Parameter "PID_A Options".)
- When the mode changes from MANUAL to AUTO, the control output starts from the MV_IN input parameter value.

◆ Executing the auto tuning

In AUTO mode, the auto tuning starts when the AT rising edge is detected.

- At the completion of auto tuning, the PID values obtained by the selected AT method is written into the Calculation Parameter [PID_A Constants]. After completion, the ordinary PID calculation is performed.
- The auto tuning stops when the AT falling edge is detected.
- You can check whether the auto tuning is being executed or has been ended by the [Mode] item of the Calculation Monitor Parameter [PID_A Monitor].

For details about the auto tuning operation diagram, stop conditions, and cautions, ☞refer to Appendix 2, Auto Tuning.

◆ Cautions

- To use the PID_A function block, it is required that the corresponding Calculation Parameters and Calculation Monitor Parameter (PID_A Options, PID_A Constants and PID_A Monitor) have been downloaded in the controller. Download the Calculation Parameters and Calculation Monitor Parameter with each group ID specified in the input parameters PID, PARA and MONI, respectively.
- If you specify a value changing at every application execution cycle (such as ramp SP and RSP) in SP, set the [Initialization on SP changes] item of the Calculation Parameter [PID_A Options] to "Option 2 (Not initialized)" to prevent unnecessary PID calculation initialization.

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur.

- The Calculation Parameters and Calculation Monitor Parameter specified in the input parameters PID, PARA and MONI have not been downloaded in the controller.
- The cycle timing setting of the project is 0s.

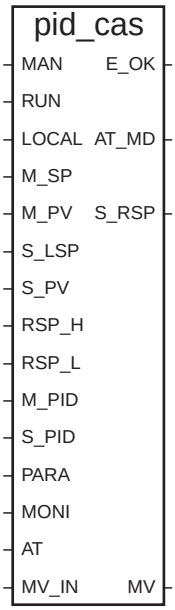
The action is as follows when an execution error has occurred:

- MV = MV_IN
- E_OK = FALSE (Error)

PID_CAS (Cascade PID operation)

Function block (for DMC50)

SYMBOL



◆ Functional description

Performs a PID operation for cascade control.

◆ Input parameters

Parameter	Data type	Content
MAN	BOOL	TRUE = MANUAL mode, FALSE = AUTO mode
RUN	BOOL	TRUE = RUN mode, FALSE = READY mode
LOCAL	BOOL	TRUE = LOCAL mode, FALSE = REMOTE mode
M_SP	REAL	Master SP (industrial unit)
M_PV	REAL	Master PV (industrial unit)
S_LSP	REAL	Slave LSP (industrial unit), For LOCAL mode
S_PV	REAL	Slave PV (industrial unit)
RSP_H	REAL	RSP scale high limit (industrial unit)
RSP_L	REAL	RSP scale low limit (industrial unit)
M_PID	DINT	Specifies the group ID of PID_CAS Constants (for master).
S_PID	DINT	Specifies the group ID of PID_CAS Constants (for slave).
PARA	DINT	Specifies the group ID of PID_CAS Options.
MONI	DINT	Specifies the group ID of PID_CAS Monitor.
AT	BOOL	Starts/stops auto tuning.
MV_IN	REAL	Inputs the value to be output in MANUAL mode.

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
AT_MD	BOOL	TRUE = During AT, FALSE = AT stopped, For read only.
S_RSP	REAL	Remote SP (industrial unit) for slave control, For read only.
MV	REAL	Manipulated variable

◆ Description of action

The PID_CAS function block is a PID operation for cascade control and has the following additional features:

- AUTO/MANUAL, READY/RUN, and LOCAL / REMOTE transfer control
- Auto tuning (AT)
- 2 degrees of freedom PID control
- PID control with dead band
- The constants used in the PID operation (such as the PID constants, output high/low limit values, etc.) are grouped in the Calculation Parameter [PID_CAS Constants].
  (Specify the group IDs of the [PID_CAS Constants] in M_PID and S_PID so as to specify for master and slave, respectively.)
- The options used in the PID operation (such as the direct /reverse action, auto tuning method option, etc.) are grouped in the Calculation Parameter [PID_CAS Options].(Specify the group ID of the [PID_CAS Options] in PARA.)
- If you want to monitor the PID_CAS input/output data (such as PV, SP and MV) easily en bloc, use the Calculation Monitor Parameter [PID_CAS Monitor]. (Specify the group ID of the [PID_CAS Monitor] in MONI. If you do not use this data, specify "0".)

◆ Modes for PID_CAS

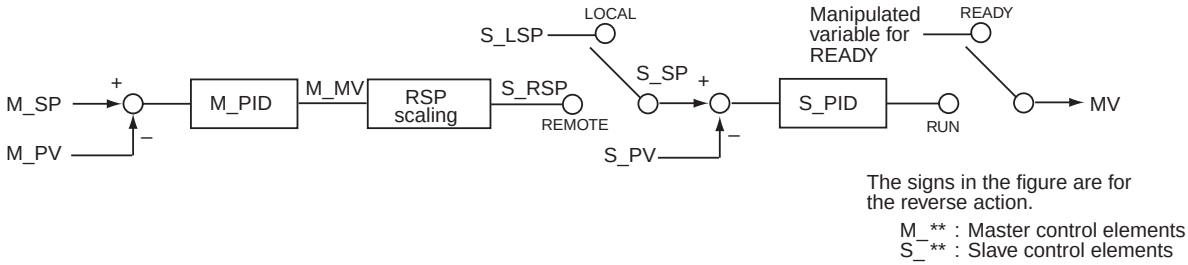
PID_CAS has the following modes:

Mode	Description
AUTO	Mode when the MAN input parameter is FALSE. Performs the PID operation for cascade control.
MANUAL	Mode when the MAN input parameter is TRUE. Outputs the value entered in MV_IN to MV without any processing.
READY	Mode when the RUN input parameter is FALSE. Outputs the manipulated variable for READY to MV.
RUN	Mode when the RUN input parameter is TRUE. Performs the PID operation for cascade control and outputs to MV.
REMOTE	Mode when the LOCAL input parameter is FALSE. Performs cascade control with the master and slave.
LOCAL	Mode when the LOCAL input parameter is TRUE. Performs the slave PID operation only.
AT	Mode when AT is activated while the system is in AUTO and RUN modes at the same time. Mode while auto tuning is executing.

Operation for each mode combination is described in the following table:

AUT/MAN	RDY/RUN	REM/LOC	AT	Operation
AUTO	READY	REMOTE	Non-AT	MV = Manipulated value for READY
		LOCAL		
	RUN	REMOTE	AT	Executing AT for master control
			Non-AT	Cascade control with the master and slave
		LOCAL	AT	Executing AT for slave control
			Non-AT	Slave PID control only
MANUAL	READY	REMOTE	Non-AT	MV = MV_IN Output the value in MV_IN without any processing
		LOCAL		
	RUN	REMOTE		
		LOCAL		

◆ Actions in AUTO mode



✻ M_PID and S_PID perform the same operation as that of PID_A (Standard PID operation).

The processing in AUTO mode (MODE = FALSE) is as follows:

- In READY mode, MV = manipulated variable for READY.
  The setting for the manipulated variable for READY is made at the calculation Parameter "PID_CAS options".
- In RUN and REMOTE modes at the same time, cascade control is executed.
  (1) Performs the master PID operation and calculates the master MV (M_MV).
  (2) Performs the RSP scaling processing from the M_MV and calculates the slave SP (S_SP).
  In the RSP scaling processing, the values of RSP_H and RSP_L are used to convert the master MV (0-100%) to the industrial unit of the slave SP. The converted value of the slave SP can be checked at the output parameter S_RSP.
  (3) Performs the slave PID operation using the S_SP and calculates the MV.
- In RUN and LOCAL modes at the same time, only the slave PID control is executed.
  Performs the slave PID operation using the S_LSP and calculates the MV.

◆ Actions in MANUAL mode

In MANUAL mode (MODE = TRUE), MV = MV_IN.

- When the mode changes from AUTO to MANUAL, preset value output or bumpless transfer can be specified optionally. (This option is specified in the Calculation Parameter "PID_CAS Options".)
- When the mode changes from MANUAL to AUTO, the control output starts from the MV_IN input parameter value.

◆ Executing the auto tuning

In RUN and AUTO modes at the same time, the auto tuning starts when the AT rising edge is detected. Depending on the SP mode (REMOTE/LOCAL), either of the master control or the slave control is selected.

Mode	Description of AT execution
REMOTE	AT execution for master control
LOCAL	AT execution for slave control

- Execute the AT for slave control in LOCAL mode first. Then execute the AT for master control in REMOTE mode.
- At the completion of auto tuning, the PID values obtained by the selected AT method is written into the Calculation Parameter [PID_CAS constants]. After completion, the mode changes to AUTO and the ordinal PID calculation is performed.
- The auto tuning stops when the AT falling edge is detected.
- You can check whether the auto tuning is being executed or has been ended by the value of AT_MD or the [Mode] item of the Calculation Monitor Parameter [PID_CAS Monitor].

For details about the auto tuning operation diagram, stop conditions, and cautions, ☞refer to Appendix 2, Auto Tuning.

◆ Cautions

- To use the PID_CAS function block, it is required that the corresponding Calculation Parameters and Calculation Monitor Parameter (PID_CAS Options, PID_CAS Constants and PID_CAS Monitor) have been downloaded in the controller. Download the Calculation Parameters and Calculation Monitor Parameter with each group ID specified in the input parameters M_PID, S_PID, PARA and MONI, respectively.
- If you specify a value changing at every application execution cycle (such as ramp SP and RSP) in M_SP in REMOTE mode, set the [Initialization on SP changes] item of the Calculation Parameter [PID_CAS Options] to "Option 2 (Not initialized: method 1)" to prevent unnecessary PID calculation initialization.
- If you specify a value changing at every application execution cycle (such as ramp SP and RSP) in LSP (on the slave) in LOCAL mode, set the [Initialization on SP Changes (slave)] item of the Calculation Parameter [PID_CAS Options] to "Option 2 (Not initialized)" to prevent unnecessary PID calculation initialization.
- Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

- The Calculation Parameters and Calculation Monitor Parameter specified in the input parameters M_PID, S_PID, PARA and MONI have not been downloaded in the module.
- The cycle timing setting of the project is 0s.

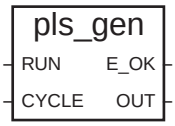
The action is as follows when an execution error has occurred:

- MV = MV_IN
- E_OK = FALSE (Error)

PLS_GEN (Pulse generator)

Function block (for DMC50)

SYMBOL



◆ Functional description

Generates a pulse at every specified cycle.

◆ Input parameters

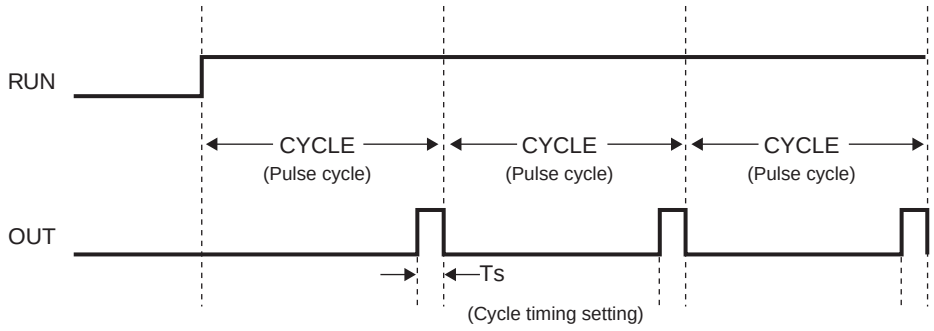
Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
CYCLE	TIME	Pulse cycle period, CYCLE ≥ the cycle timing setting

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	BOOL	Pulse output

◆ Description of action

If RUN = TRUE (Execution), the action is as follows.



If RUN = FALSE (Reset), OUT = FALSE.

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

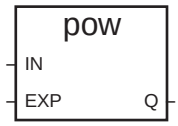
- CYCLE < the cycle timing setting.
- CYCLE = T#0.
- The cycle timing setting of the project is 0s.

The action is as follows when an execution error has occurred:

- OUT = FALSE
- E_OK = FALSE (Error)

POW (Exponential function) Function

SYMBOL



◆ Functional description

Performs exponential operation on REAL type data and outputs the result.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	Base number
EXP	REAL	Exponent number

◆ Output parameters

Parameter	Data type	Content
Q	REAL	$Q = IN^{EXP}$

◆ Description of action

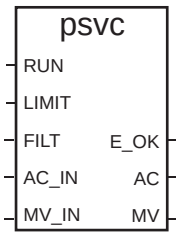
$Q = IN^{EXP}$

The operation is available even if the IN is a negative value.

PSVC (Power supply voltage compensation)

Function block (for DMC50)

SYMBOL



◆ Functional description

Monitors the heater power voltage fluctuations to correct the manipulated variable output.
PSVC (Power Supply Voltage Compensation)

◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
LIMIT	REAL	Compensation range: 0.0 to 50.0% (Normally 20.0% is sufficient) Executes the compensation when AC_IN = 100% ± LIMIT
FILT	REAL	Filter constant for AC_IN: 0.0 to 2000.0s This is for the operation during heater break or power failure.
AC_IN	REAL	AC voltage input. Used to input the AUX_IN (AC voltage).
MV_IN	REAL	Manipulated variable before compensation

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = During compensation, FALSE = Compensation stopped
AC	REAL	AC value after filter operation. This is for reference.
MV	REAL	Manipulated variable after compensation

◆ Description of action

If the voltage of the power supply used for actuators such as heaters fluctuates, the manipulated variable corresponding to the fluctuations is calculated so that the fluctuation does not affect the controllability.
For example, if the voltage of the heater power decreases, the manipulated variable is increased for compensation. On the contrary, if the voltage of the heater power increases, the manipulated variable is decreased for compensation.

- LIMIT is used to set a compensation range (0.0 to 50.0%). Normally, this parameter is set at about 20.0%.
- Set FILT to perform a heavy filter operation for the AC_IN.
- Connect the AUX_IN1 or AUX_IN2 of the I / O variables (input variables) to the AC_IN.
- Connect the MV (Manipulated value) of the PID operation to the MV_IN.

When RUN = TRUE (Execution), the following compensation operation is performed:
AC_IN' = First order digital filter operation (AC_IN)
$MV = MV\_IN \times (100.0 / AC\_IN')^2$

Then, the output parameters are as follows:
• E_OK = TRUE (During compensation)
• AC = AC_IN' (AC value after filter calculation)
• MV = Manipulated variable after compensation

In the following conditions, no compensation operation is performed:
• RUN = FALSE (Reset).
• The AC_IN is not within the range of 100% ± LIMIT.
• Three seconds has not elapsed yet after the AC_IN entered the range of 100% ± LIMIT.
• The LIMIT has been set out of the range between 0.0 and 50.0% (Error).
• The FILT has been set out of the range between 0.0 and 2000.0s (Error).

If any of them is the case, the output parameters are as follows:
• E_OK = FALSE (Compensation stopped)
• AC = AC_IN
• MV = MV_IN

◆ FILT input parameters

[Purpose]

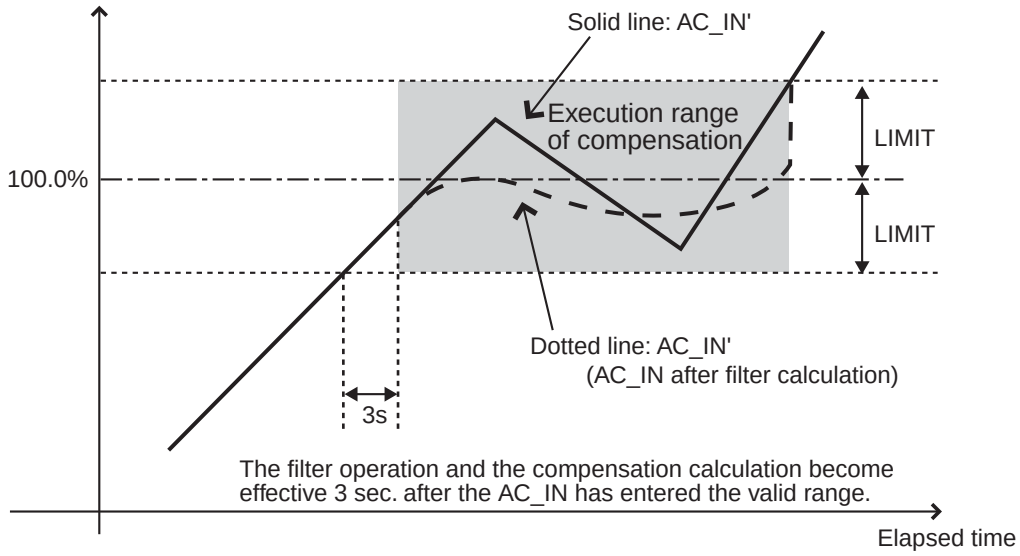
A heavy filter operation is performed for the AC_IN not to compensate for rapid fluctuation of the power voltage but to compensate solely for the amount gradually shifted as time goes on.
For the I/O variable AUX_IN connected to the AC_IN, the input filter is activated using the AUX-IN Options in the System Parameters.
If this filter constant is too large, a problem caused by the filter operation may arise when the heater power is turned ON.
For example, assuming that the power to the DMC50 is kept turned ON and only the heater power is turned ON or OFF in a system, the AUX-IN rises with a heavy filter operation when the heater power is changed from OFF to ON. Therefore, it takes a long time to recognize the AC voltage input as a normal value which can be compensated.
If the filter constant is small, no problems may arise.

[Operation specifications with FILT input parameters]

When using a heavy filter for the AC voltage input, set the filter constant to the FILT input parameters instead of the AUX-IN Options in the System Parameters.
The following shows the filter operation specifications inside the PSVC function block:

- The filter process by the FILT input is operated, independent of the input filter of the AUX-IN Options.
- The filter operation works only when the AC_IN is within its valid range (100% ± LIMIT).
- The filter operation is stopped immediately when the AC_IN is out of its valid range (100% ± LIMIT).
- The filter operation is not performed for 3 sec. when the AC_IN enters the valid range from out of the valid range.
- If RUN = FALSE (Reset), the filter operation is not performed.

The filter operation and the valid compensation range are as follows:



◆ Cautions

- Must be executed at every application execution cycle to perform time management internally.
- If a rather heavy filter operation is required to perform for AC voltage input, make the settings for the filter constant not in the AUX-IN Options in the System Parameters but in the FILT input parameter.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

- The LIMIT has been set out of the range between 0.0 and 50.0% (Error).
- The FILT has been set out of the range between 0.0 and 2000.0s (Error).
- The cycle timing setting of the project is 0s.

The action is as follows when an execution error has occurred:

- E_OK* = FALSE (Compensation stopped)
- AC = AC_IN
- MV = MV_IN

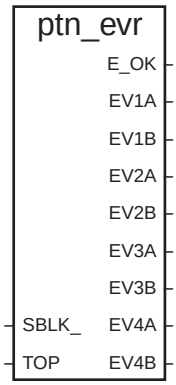
*: Different from the specifications of the other function blocks, This E_OK indicates not only the execution error but also the execution status of compensation (i.e. whether the compensation is being performed or stopped).

MEMO

PTN_EVR (Read pattern event data)

Function block (for DMC50)

SYMBOL



◆ Functional description

Reads out the event thresholds for the current segment.
Used for generating PV or error events.

◆ Input parameters

Parameter	Data type	Content
SBLK_	DINT	Makes a connection to the SB_LK output of PTN_MAIN/PTN_SUB.
TOP	DINT	top event number (1 to 17)

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
EV1A	REAL	Event 1 threshold A
EV1B	REAL	Event 1 threshold B
EV2A	REAL	Event 2 threshold A
EV2B	REAL	Event 2 threshold B
EV3A	REAL	Event 3 threshold A
EV3B	REAL	Event 3 threshold B
EV4A	REAL	Event 4 threshold A
EV4B	REAL	Event 4 threshold B

◆ Description of action

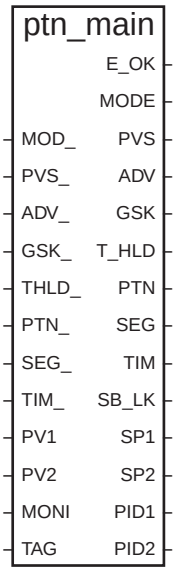
For details,
☞refer to Module Type Controller DMC50 User's Manual "Application Developer's Guide", No. CP-SP-1134E.

MEMO

PTN_MAIN (Pattern main)

Function block (for DMC50)

SYMBOL



◆ Functional description

Performs program pattern operation, generating synchronized SPs of 2 channels.

◆ Input parameters

Parameter	Data type	Content
MOD_	DINT	Mode change request (0 to 6)
PVS_	DINT	PV start request (0 to 6)
ADV_	DINT	Request to advance (0 to 8)
GSK_	DINT	Request to resume from G.SOAK standby state (0 to 3)
THLD_	BOOL	TRUE = Request of temporary HOLD, FALSE = No request
PTN_	DINT	Pattern number change request (1 to 99) Specifies a group ID of Pattern Setup.
SEG_	DINT	Segment number change request (1 to 99) Specifies a group ID of Segment Setup.
TIM_	DINT	Segment elapsed time change request (0.0 to 30000.0)
PV1	REAL	Inputs the PV for PV start and G.SOAK processing on SP1.
PV2	REAL	Inputs the PV for PV start and G.SOAK processing on SP2.
MONI	DINT	Specifies a group ID of the Pattern FB Monitor.
TAG	STRING	Pattern FB tag

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
MODE	DINT	Mode (0 to 6)
PVS	DINT	PV start status (0 to 4)
ADV	DINT	Advance status (0 to 8)
GSK	DINT	G.SOAK standby status (0 to 3)
T_HLD	BOOL	TRUE = Temporary HOLD, FALSE = Other than temporary HOLD
PTN	DINT	Current pattern number
SEG	DINT	Current segment number
TIM	DINT	Segment elapsed time
SB_LK	DINT	Sub-link connection Makes a connection to the SBLK_ input of PTN_SUB / PTN_EVR / PTN_TEV.
SP1	REAL	Current SP1
SP2	REAL	Current SP2
PID1	DINT	PID group for the control loop with SP1
PID2	DINT	PID group for the control loop with SP2

◆ Description of action

This manual contains only the auxiliary explanation for input/output parameters. For details,

☞refer to Module Type Controller DMC50 User's Manual "Application Developer's Guide", No. CP-SP-1134E.

[MOD_ input]

The values for MOD_ input (mode change request) are as follows:

0: No request
1: READY
2: RUN
3: HOLD
4: FAST
5: END
6: SPHOLD

[PVS_ input]

The values for PVS_ input (PV start request) are as follows:
(Operation is triggered by the rising edge of the input)

0: No request
1: SP1 ramp-up segments
2: SP1 ramp-down segments
3: SP1 ramp-up/down segments
4: SP2 ramp-up segments
5: SP2 ramp-down segments
6: SP2 ramp-up/down segments

[ADV_ input]

The values for ADV_ input (advance request) are as follows:
(Operation is triggered by the rising edge of the input)

0: No request
1: Advance of 1 segment
2: Advance to the segment specified by SEG_
3: Advance to the pattern and segment specified by PTN_ and SEG_
4: Advance to the pattern, segment and the segment elapsed time specified by PTN_, SEG_ and TIM_
5: Relative advance by TIM_ in time from the current segment elapsed time
6: Absolute advance by TIM_ in time from the pattern starting point
7: SP1 search
8: SP2 search

[GSK_ input]

The values for GSK_ input (request to resume from the G.SOAK standby state) are as follows:

0: No request
1: Resume on SP1
2: Resume on SP2
3: Resume on both SP1 and SP2

[MODE output]

The values for MODE output (mode) are as follows:

0: ERR (error status)
1: READY
2: RUN
3: HOLD
4: FAST
5: END
6: SPHOLD

[PVS output]

The values of PVS output (PV start status) are as follows:
(Value is retained only for one execution cycle)

0: Status other than start
1: Start other than the PV start from READY mode
2: PV start from READY mode
3: No PV start execution at the pattern link destination
4: PV start execution at the pattern link destination

[ADV output]

The values for ADV output (advance status) are as follows:
(Value is retained only for one execution cycle)

0: No request
1: Advance of 1 segment
2: Advance to the segment specified by SEG_
3: Advance to the pattern and segment specified by PTN_ and SEG_
4: Advance to the pattern, segment and the segment elapsed time specified by PTN_, SEG_ and TIM_
5: Relative advance by TIM_ in time from the current segment elapsed time
6: Absolute advance by TIM_ in time from the pattern starting point
7: SP1 search
8: SP2 search

[GSK output]

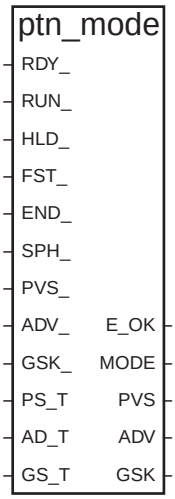
The values for GSK output (whether or not the operations are in G.SOAK standby state) are as follows:

0: Resuming normal operations on both SP1 and SP2
1: In standby state on SP1
2: In standby state on SP2
3: In standby state on both SP1 and SP2

PTN_MODE (Pattern mode)

Function block (for DMC50)

SYMBOL



◆ Functional description

Easily carries out PTN_MAIN function block mode change, advance execution, PV start, etc.

◆ Input parameters

Parameter	Data type	Content
RDY_	BOOL	TRUE = READY request, FALSE = No request
RUN_	BOOL	TRUE = RUN request, FALSE = No request
HLD_	BOOL	TRUE = HOLD request, FALSE = No request
FST_	BOOL	TRUE = FAST request, FALSE = No request
END_	BOOL	TRUE = END request, FALSE = No request
SPH_	BOOL	TRUE = SPHOLD request, FALSE = No request
PVS_	BOOL	TRUE = PV start request, FALSE = No request
ADV_	BOOL	TRUE = Request to advance, FALSE = No request
GSK_	BOOL	TRUE = Request to resume from G.SOAK standby state, FALSE = No request
PS_T	DINT	PV start type (0 to 6)
AD_T	DINT	Advance type (0 to 8)
GS_T	DINT	Type of resuming from G.SOAK standby state (0 to 3)

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
MODE	DINT	Mode change request (0 to 6)
PVS	DINT	PV start request (0 to 6)
ADV	DINT	Advance request (0 to 8)
GSK	DINT	Request to resume from G.SOAK standby state (0 to 3)

◆ Description of action

This manual contains only the auxiliary explanation for input / output parameters. For details,

☞refer to Module Type Controller DMC50 User's Manual "Application Developer's Guide", No. CP-SP-1134E.

[PS_T input]

The values for PS_T input (PV start type) are as follows:

0: No request
1: SP1 ramp-up segments
2: SP1 ramp-down segments
3: SP1 ramp-up /down segments
4: SP2 ramp-up segments
5: SP2 ramp-down segments
6: SP2 ramp-up /down segments

[AD_T input]

The values for AD_T input (advance type) are as follows:

0: No request
1: Advance of 1 segment
2: Advance to the segment specified by SEG_
3: Advance to the pattern and segment specified by PTN_ and SEG_
4: Advance to the pattern, segment and segment elapsed time specified by PTN_, SEG_ and TIM_
5: Relative advance by TIM_ in time from the current segment elapsed time
6: Absolute advance by TIM_ in time from the pattern starting point
7: SP1 search
8: SP2 search

[GS_T input]

The values for GS_T input (request to resume from G.SOAK standby state) are as follows:

0: No request
1: Resume on SP1
2: Resume on SP2
3: Resume on both SP1 and SP2

[MODE output]

The values for MODE output (mode change request) are as follows:
(Value is retained only for one execution cycle)

0: ERR (error status)
1: READY
2: RUN
3: HOLD
4: FAST
5: END
6: SPHOLD

[PVS output]

The values for PVS output (PV start request) are as follows:
(Value is retained only for one execution cycle)

0: No request
1: SP1 ramp-up segments
2: SP1 ramp-down segments
3: SP1 ramp-up / down segments
4: SP2 ramp-up segments
5: SP2 ramp-down segments
6: SP2 ramp-up / down segments

[ADV output]

The values for ADV output (advance request) are as follows:
(Value is retained only for one execution cycle)

0: No request
1: Advance of 1 segment
2: Advance to the segment specified by SEG_
3: Advance to the pattern and segment specified by PTN_ and SEG_
4: Advance to the pattern, segment and segment elapsed time by PTN_, SEG_ and TIM_
5: Relative advance by TIM_ in time from the segment elapsed time
6: Absolute advance by TIM_ in time from the pattern starting point
7: SP1 search
8: SP2 search

[GSK output]

The values for GSK output (request to resume from G.SOAK standby state) are as follows:
(Value is retained only for one execution cycle)

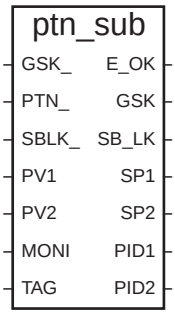
0: No request
1: Resume on SP1
2: Resume on SP2
3: Resume on both SP1 and SP2

MEMO

PTN_SUB (Pattern sub)

Function block (for DMC50)

SYMBOL



◆ Functional description

Used when you want to generate synchronized SPs of three or more number of channels.
Use this function block sublinked with the PTN_MAIN function block.

◆ Input parameters

Parameter	Data type	Content
GSK_	DINT	Request to resume from G.SOAK standby state (0 to 3)
PTN_	DINT	Pattern number change request (1 to 99) Specifies a group ID of Pattern Setup.
SBLK_	DINT	Makes a connection to the SB_LK output of PTN_MAIN.
PV1	REAL	Inputs the PV for G.SOAK processing on SP1.
PV2	REAL	Inputs the PV for G.SOAK processing on SP2.
MONI	DINT	Specifies a group ID of the Pattern FB Monitor.
TAG	STRING	Pattern FB tag

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
GSK	DINT	G.SOAK standby status (0 to 3)
SB_LK	DINT	Sub-link connection Makes a connection to the SBLK_ input of PTN_EVR/PTN_TEV.
SP1	REAL	Current SP1
SP2	REAL	Current SP2
PID1	DINT	PID group for the control loop with SP1
PID2	DINT	PID group for the control loop with SP2

◆ Description of action

This manual contains only the auxiliary explanation for input / output parameters. For details,
☞refer to Module Type Controller DMC50 User's Manual "Application Developer's Guide", No. CP-SP-1134E.

[GSK_ input]
The values for GSK_ input (request to resume from G.SOAK standby state) are as follows:

0: No request
1: Resume on SP1
2: Resume on SP2
3: Resume on both SP1 and SP2

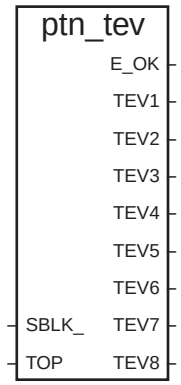
[GSK output]
The values for GSK output (G.SOAK standby status) are as follows:

0: Resuming operations on both SP1 and SP2
1: In standby state on SP1
2: In standby state on SP2
3: In standby state on both SP1 and SP2

PTN_TEV (Pattern time event)

Function block (for DMC50)

SYMBOL



◆ Functional description

Generates time events for the current segment.

◆ Input parameters

Parameter	Data type	Content
SBLK_	DINT	Makes a connection to the SB_LK output of PTN_MAIN/PTN_SUB.
TOP	DINT	Top event number (1 to 13)

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
TEV1	BOOL	Time event 1 status (TRUE = ON, FALSE = OFF)
TEV2	BOOL	Time event 2 status (TRUE = ON, FALSE = OFF)
TEV3	BOOL	Time event 3 status (TRUE = ON, FALSE = OFF)
TEV4	BOOL	Time event 4 status (TRUE = ON, FALSE = OFF)
TEV5	BOOL	Time event 5 status (TRUE = ON, FALSE = OFF)
TEV6	BOOL	Time event 6 status (TRUE = ON, FALSE = OFF)
TEV7	BOOL	Time event 7 status (TRUE = ON, FALSE = OFF)
TEV8	BOOL	Time event 8 status (TRUE = ON, FALSE = OFF)

◆ Description of action

For details,

☞refer to Module Type Controller DMC50 User's Manual "Application Developer's Guide", No. CP-SP-1134E.

R_TRIG (Rising edge detection) Function block

SYMBOL

◆ Functional description

Detects the rising edge of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
CLK	BOOL	

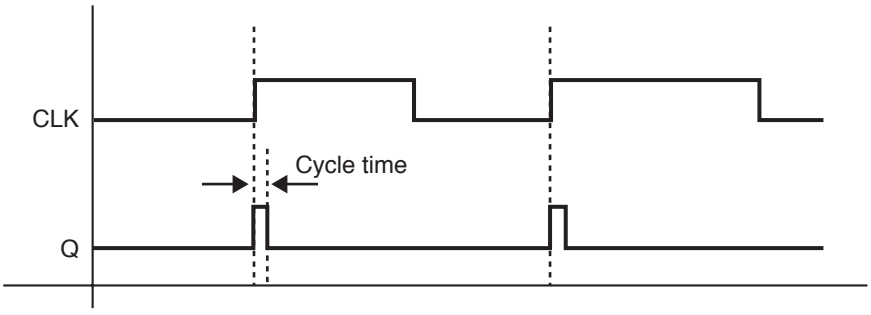
◆ Output parameters

Parameter	Data type	Content
Q	BOOL	Q is TRUE when CLK is changed from FALSE to TRUE. FALSE in other cases.

◆ Description of action

When the rising edge of CLK is detected, Q is TRUE (for one cycle time).
In other cases, Q is FALSE.

Rising edge is detected by comparing with the value of the previous cycle.



◆ Cautions

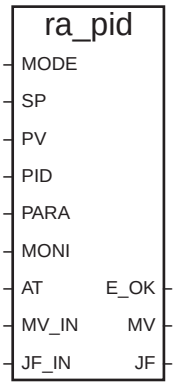
When this function block is executed for the first time (including the occasion of power-on), the operation is performed assuming that the previous value of CLK is FALSE. Note that the rising edge is detected if CLK = TRUE at the time of first execution.

MEMO

Ra_PID (RationaLOOP PID operation)

Function block (for DMC50)

SYMBOL



◆ Functional description

Performs PID operation incorporated with superior disturbance recovery and over-shoot suppression feature for high precision control.

◆ Input parameters

Parameter	Data type	Content
MODE	BOOL	TRUE = MANUAL mode, FALSE = AUTO mode
SP	REAL	SP (industrial unit)
PV	REAL	PV (industrial unit)
PID	DINT	Specifies a group ID of Ra_PID Constants
PARA	DINT	Specifies a group ID of Ra_PID Options
MONI	DINT	Specifies a group ID of Ra_PID Monitor
AT	BOOL	Starts / stops auto tuning.
MV_IN	REAL	Inputs the value to be output in MANUAL mode.
JF_IN	REAL	Just-FiTTER input Normally input 0.0.

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
MV	REAL	Manipulated variable
JF	REAL	Just-FiTTER output

◆ Description of action

The Ra_PID function block (RationaLOOP PID operation) has the following features:

- AUTO / MANUAL transfer control
- Controller feature (Ra_PID, Just-FiTTER)
- Response and stability tuning feature (Delta D, Delta S)
- PID constants self-tuning feature (CLAFT AT, CLAFT Self S, CLAFT Self H)

The RationaLOOP PID operation is performed in AUTO mode (MODE = FALSE).

- The constants used in the RationaLOOP PID operation (such as the PID constants, output high / low limit values, etc.) are grouped in the Calculation Parameter [Ra_PID Constants]. (Specify a group ID of the [Ra_PID Constants] in PID.)
- The options used in the RationaLOOP PID operation (such as the direct / reverse action, auto tuning method option, etc.) are grouped in the Calculation Parameter [Ra_PID Options]. (Specify a group ID of the [Ra_PID Options] in PARA.)

- If you want to monitor the Ra_PID input / output data (such as PV, SP and MV) easily en bloc, use the calculation monitor data [Ra_PID Monitor]. (Specify the group ID of the [Ra_PID Monitor] in MONI. If you do not use this data, specify "0".)

In MANUAL mode (MODE = TRUE), MV = MV_IN.

- When the control mode changes from AUTO to MANUAL, preset value output or bumpless transfer can be specified optionally. (This option is specified in the Calculation Parameter [Ra_PID Options].)
- When the control mode changes from MANUAL to AUTO, the control output starts from the MV_IN input parameter value.

◆ Executing the auto tuning

In AUTO mode, the auto tuning starts when the rising edge of AT is detected.

- At the completion of auto tuning, the PID values obtained by the selected AT method is written into the Calculation Parameter [Ra_PID Constants]. After completion, the ordinal PID calculation is performed.
- The auto tuning stops when the AT falling edge is detected.
- You can check whether the auto tuning is being executed or has been ended by the [Mode] item of a Calculation Monitor Parameter [Ra_PID Monitor].

For details about the auto tuning operation diagram, stop conditions, and cautions, ☞refer to Appendix 2, Auto Tuning.

◆ Cautions

To use the Ra_PID function block, it is required that the corresponding Calculation Parameters and Calculation Monitor Parameter (Ra_PID Options, Ra_PID Constants and Ra_PID Monitor) have been downloaded in the controller. Download the Calculation Parameters and Calculation Monitor Parameter with each group ID specified in the input parameters PID, PARA and MONI, respectively.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

- The Calculation Parameters and Calculation Monitor Parameter specified in the input parameters PID, PARA and MONI have not been downloaded in the controller.
- The cycle timing setting of the project is 0s.

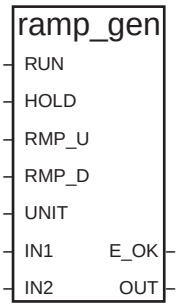
The action is as follows when an execution error has occurred:

- MV = MV_IN
- E_OK = FALSE (Error)

RAMP_GEN (Ramp generator)

Function block (for DMC50)

SYMBOL



◆ Functional description

Generates ramp values of REAL type up to the target value.

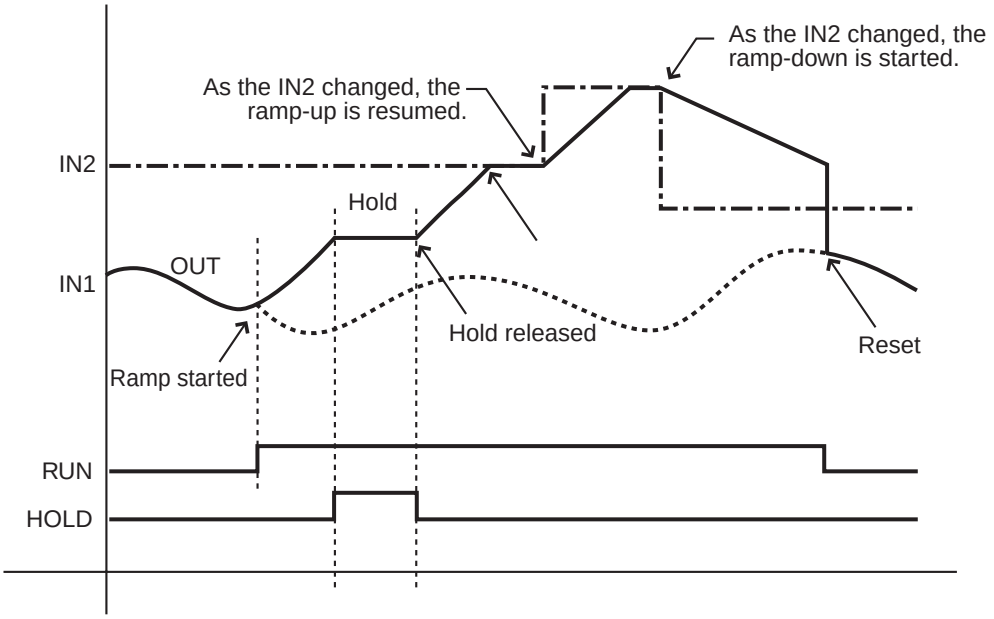
◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALSE = Reset
HOLD	BOOL	TRUE = Suspend the ramp action (Hold the current value). FALSE = Continue the ramp action (release the hold).
RMP_U	REAL	Ramp-up rate
RMP_D	REAL	Ramp-down rate
UNIT	DINT	Ramp time unit 0: Ramp rate/s 1: Ramp rate/min 2: Ramp rate/h
IN1	REAL	Starting value of ramp action
IN2	REAL	Target value of ramp action

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	REAL	Ramp value

◆ Description of action



The following shows the operation for each RUN parameter value:

FALSE : OUT = IN1

FALSE → TRUE : Ramp operation toward IN2 is started after OUT = IN1 is assigned to the start value.

TRUE : Ramp operation is performed until OUT = IN2. However, after the value has reached IN2, the ramp operation is resumed corresponding to changes in IN2.

The following shows the operation for each HOLD parameter value when the ramp operation is being performed with RUN = TRUE:

HOLD = FALSE : Ramp operation continues.

HOLD = TRUE : Ramp operation is suspended.

In the ramp operation, the amount added per each execution cycle (Delta) is calculated from the following expression:

Delta = (ramp X Ts) / t_unit

ramp : Ramp-up = RMP_U, Ramp-down = -RMP_D

Ts : The cycle timing setting (ms)

t_unit : 1000 ms for UNIT = 0
60000 ms for UNIT = 1
3600000 ms for UNIT = 2

At each execution cycle, OUT is updated with the value as calculated below:

OUT = Start + (Delta X Count)

Start : Value at ramp start

Count : Count value incremented at each execution cycle after the ramp start

- If Delta is small, OUT is not changed at every execution cycle, but changed at once per a certain number of cycles to reach IN2 (target value). However, if Delta is too small, the value may not reach the target value.
- When RUN is TRUE, and when RMP_U and RMP_D are set at "0.0", OUT becomes IN2. (IN2 is output to OUT directly.)

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

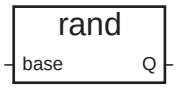
- RMP_U < 0.0 or RMP_D < 0.0.
- UINT is neither 0,1, nor 2.
- The cycle timing setting of the project is 0s.

The action is as follows when an execution error has occurred:

- OUT = IN1
- E_OK = FALSE (Error)

RAND (Random value) Function

SYMBOL



◆ Functional description

Generates a random value of DINT type data.
The range of random values can be specified.

◆ Input parameters

Parameter	Data type	Content
BASE	DINT	Maximum random value, BASE > 1

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Random value, 0 to BASE-1

◆ Description of action

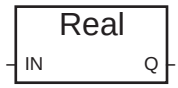
Q = Random value (between 0 and BASE-1)

If BASE ≤ 1, Q is 0.

[Sample of action]
When BASE = 10, Q is a random value between 0 and 9.
When BASE = 1, Q is fixed to 0.

REAL (Convert to real) Standard operator

SYMBOL



◆ Functional description

Converts a value to a REAL type one.
Available data types are BOOL, DINT and TIME.

◆ Input parameters

Parameter	Data type	Content
IN	BOOL, DINT TIME	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	

◆ Description of action

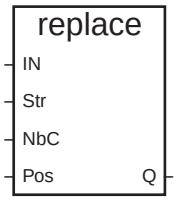
• When IN is of BOOL type:
Q = 0.0 if IN = FALSE or
Q = 1.0 if IN = TRUE.

• When IN is of DINT type:
Q = IN
Example: If IN = 1234, Q = 1234.0.

• When IN is of TIME type:
Q = milliseconds value
Example: If IN = T#1s46ms, Q = 1046.0.

REPLACE (Message string replacement) Function

SYMBOL



◆ Functional description

Replaces a message string with another one.
Replaces the specified number of message string from the specified position with another specified message string .

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Base message string
STR	STRING	Message string to be inserted
NBC	DINT	Number of characters to be deleted
POS	DINT	First character position of the message string to be inserted, POS > 1

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Message string modified by replacement

◆ Description of action

Performs replacement by deleting NBC characters from the message string specified by IN starting from POS (a position in the message string) first, then inserting the message string STR there.

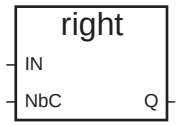
[Sample of action]
When IN = ‘ABCDEF’, STR = ‘123’, NBC = 2, and POS = 2, Q is ‘A123DEF’.

◆ Note

- If POS = 0, Q is a empty message string.
- If POS < 0 or NBC ≤ 0, Q is an undefined message string .
- If POS > "Message string length of IN", STR is concatenated with the end of IN.
  When IN = ‘ABC’, STR = ‘123’, NBC = 2, and POS = 4, Q is ‘ABC123’.
- If the length of the message string has exceeded 255 after replacement, Q is a empty message string.

RIGHT (Right message string extraction) Function

SYMBOL



◆ Functional description

Extracts the specified message string from the right hand side of a base message string.
It is possible to specify the length of the string to be extracted.

◆ Input parameters

Parameter	Data type	Content
IN	STRING	Message string
NBC	DINT	Number of characters to be extracted, 0 < NBC ≤ Message string length of IN

◆ Output parameters

Parameter	Data type	Content
Q	STRING	Extracted message string

◆ Description of action

Extracts characters of the length specified by NBC from the right hand side of the message string specified by IN.

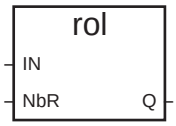
[Sample of action]
When IN = ‘ABCDEFGH’, and NBC = 2, Q is ‘GH’.
When IN = ‘ABCDEFGH’, and NBC = 4, Q is ‘EFGH’.

◆ Note

• If NBC ≤ 0, Q is a empty message string.
• If NBC ≥ "Message string length of IN", Q is IN.

ROL (Left rotation) Function

SYMBOL



◆ Functional description

Rotates the bits of DINT type data to the left.

◆ Input parameters

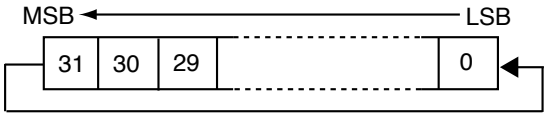
Parameter	Data type	Content
IN	DINT	
NBR	DINT	Number of bits to be rotated to the left, 1 to 31

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Result of left rotation

◆ Description of action

Rotates all the 32 bits of IN to the left by NBR positions. The MSB is copied into the LSB.



[Sample of action]
When IN = 16#12345678 and NBR = 4, Q is 16#23456781.

- If NBR ≤ 0, Q = IN. (No rotation conducted)
- If NBR ≥ 32, Q is an undefined value.

ROR (Right rotation) Function

SYMBOL

◆ Functional description

Rotates the bits of DINT type data to the right.

◆ Input parameters

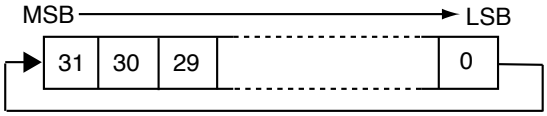
Parameter	Data type	Content
IN	DINT	
NBR	DINT	Number of bits to be rotated to the right, 1 to 31

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Result of right rotation

◆ Description of action

Rotates all the 32 NBR bits of IN to the right by NBR positions. The LSB is copied into the MSB.



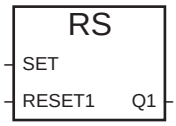
[Sample of action]

When IN = 16#12345678 and NBR = 4, Q is 16#81234567.

- If NBR ≤ 0, Q = IN. (No rotation conducted)
- If NBR ≥ 32, Q is an undefined value.

RS (Reset dominant bistable) Function block

SYMBOL



◆ Functional description

Sets a reset dominant latch.

◆ Input parameters

Parameter	Data type	Content
SET	BOOL	
RESET1	BOOL	

◆ Output parameters

Parameter	Data type	Content
Q1	BOOL	If SET = TRUE, Q1 is TRUE. If RESET1 = TRUE, Q1 is FALSE (reset dominant).

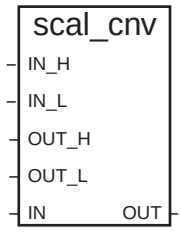
◆ Description of action

SET	RESET1	Q1	Results of Q1
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	FALSE (reset dominant)
FALSE	TRUE	TRUE	FALSE (reset dominant)
TRUE	FALSE	FALSE	TRUE
TRUE	FALSE	TRUE	TRUE
TRUE	TRUE	FALSE	FALSE (reset dominant)
TRUE	TRUE	TRUE	FALSE (reset dominant)

SCAL_CNV (Scale conversion)

Function (for DMC50)

SYMBOL



◆ Functional description

Performs scaling operation of REAL type data.

Can be used for conversion of industrial unit or 0-100% conversion.

◆ Input parameters

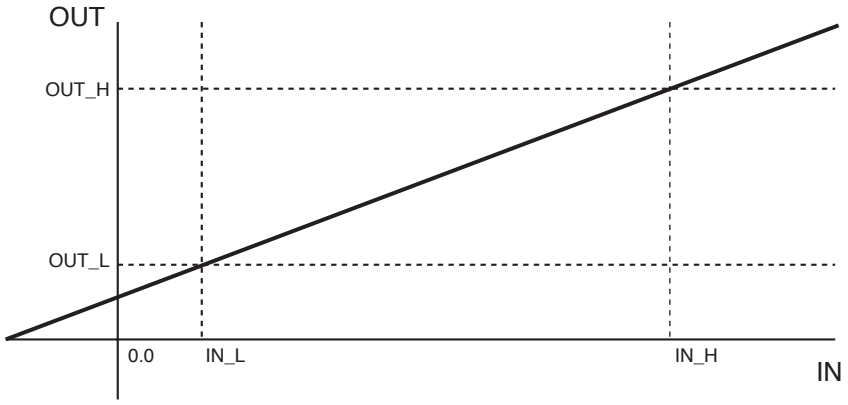
Parameter	Data type	Content
IN_H, IN_L	REAL	Input scale high limit/low limit
OUT_H, OUT_L	REAL	Output scale high limit/low limit
IN	REAL	Input data

◆ Output parameters

Parameter	Data type	Content
OUT	REAL	Output data

◆ Description of action

OUT : = (IN - IN_L) / (IN_H - IN_L) X (OUT_H - OUT_L) + OUT_L

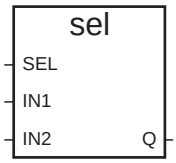


• When IN_H = IN_L, OUT = 0.0.

• If you perform 0-100% conversion, set OUT_L to 0.0 and OUT_H to 100.0.

SEL (Binary selection) Function

SYMBOL



◆ Functional description

Selects one out of 2 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
SEL	BOOL	Selector value
IN1	DINT	
IN2	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	

◆ Description of action

If SEL = FALSE, Q = IN1.
If SEL = TRUE, Q = IN2.

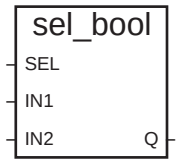
◆ Note

For handling data of other types, the following functions are provided:
SEL_BOOL (BOOL type binary selection), SEL_REAL (REAL type binary selection), SEL_TMR (TIME type binary selection).

SEL_BOOL (BOOL type binary selection)

Function (for DMC50)

SYMBOL



◆ Functional description

Selects one out of 2 pieces of BOOL type data.

◆ Input parameters

Parameter	Data type	Content
SEL	BOOL	Selector value
IN1	BOOL	
IN2	BOOL	

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	

◆ Description of action

If SEL = FALSE, Q = IN1.
If SEL = TRUE, Q = IN2.

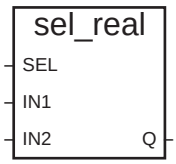
◆ Note

For handling data of other types, the following functions are provided:
SEL (DINT type binary selection), SEL_REAL (REAL type binary selection),
SEL_TMR (TIME type binary selection).

SEL_REAL (REAL type binary selection)

Function (for DMC50)

SYMBOL



◆ Functional description

Selects one out of 2 pieces of REAL type data.

◆ Input parameters

Parameter	Data type	Content
SEL	BOOL	Selector value
IN1	REAL	
IN2	REAL	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	

◆ Description of action

If SEL = FALSE, Q = IN1.
If SEL = TRUE, Q = IN2.

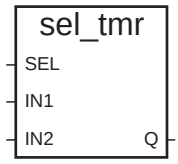
◆ Note

For handling data of other types, the following functions are provided:
SEL_BOOL (BOOL type binary selection), SEL (DINT type binary selection),
SEL_TMR (TIME type binary selection).

SEL_TMR (TIME type binary selection)

Function (for DMC50)

SYMBOL



◆ Functional description

Selects one out of 2 pieces of TIME type data.

◆ Input parameters

Parameter	Data type	Content
SEL	BOOL	Selector value
IN1	TIME	
IN2	TIME	

◆ Output parameters

Parameter	Data type	Content
Q	TIME	

◆ Description of action

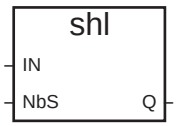
If SEL = FALSE, Q = IN1.
If SEL = TRUE, Q = IN2.

◆ Note

For handling data of other types, the following functions are provided:
SEL_BOOL (BOOL type binary selection), SEL (DINT type binary selection),
SEL_REAL (REAL type binary selection).

SHL (Left shift) Function

SYMBOL



◆ Functional description

Shifts the bits of DINT type data leftward.

◆ Input parameters

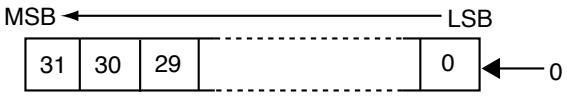
Parameter	Data type	Content
IN	DINT	
NBS	DINT	Number of bits to be shifted leftward, 1 to 31

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Result of left shift

◆ Description of action

Shifts all the 32 bits of IN leftward by NBS positions. (0s are inserted into the LSB.)



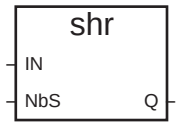
[Sample of action]

When IN = 16#12345678 and NBS = 4, Q is 16#23456780.

- If NBS ≤ 0, Q = IN. (No shift conducted)
- If NBS ≥ 32, Q is an undefined value.

SHR (Right shift) Function

SYMBOL



◆ Functional description

Shifts the bits of DINT type data rightward.

◆ Input parameters

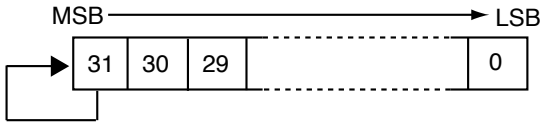
Parameter	Data type	Content
IN	DINT	
NBS	DINT	Number of bits to be shifted rightward, 1 to 31

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Result of right shift

◆ Description of action

Shifts all the 32 bits of IN rightward by NBS positions. (The same bit value as the previous MSB is copied to the MSB in a bit shift.)

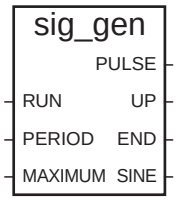


[Sample of action]
When IN = 16#12345678 and NBS = 4, Q is 16#01234567.
When IN = 16#AAAAAAAA and NBS = 1, Q is 16#D5555555.

- If NBS ≤ 0, Q = IN. (No shift conducted)
- If NBS ≥ 32, Q is an undefined value.

SIG_GEN (Signal generator) Function block

SYMBOL



◆ Functional description

Generates 3 signals (i.e. pulse, up-counter and sine wave signals).

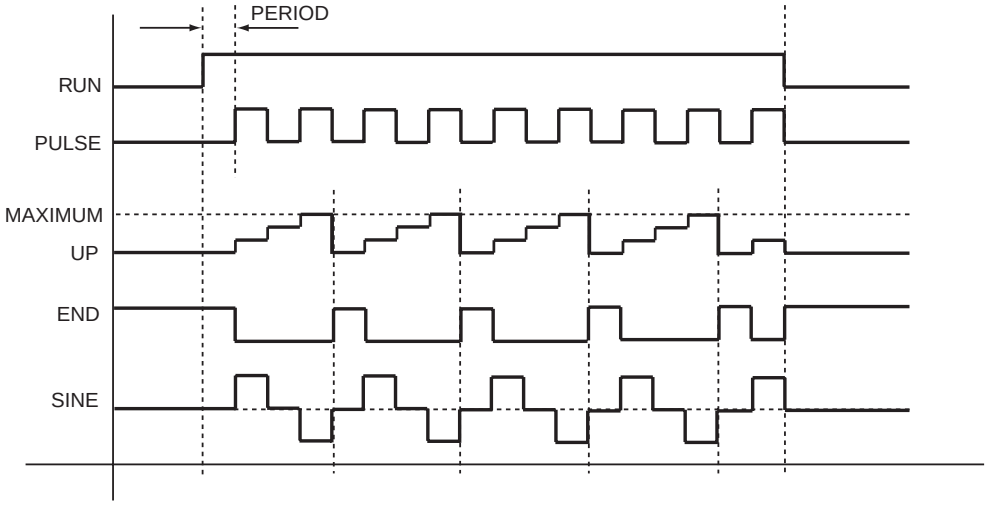
◆ Input parameters

Parameter	Data type	Content
RUN	BOOL	TRUE = Execution, FALE = Reset
PERIOD	TIME	Update time (pulse width) PERIOD ≥ the cycle timing setting
MAXIMUM	DINT	Maximum count value, MAXIMUM ≥ 0

◆ Output parameters

Parameter	Data type	Content
PULSE	BOOL	Inverted at every update time
UP	DINT	Incremented at every update time
END	BOOL	End signal of up-counter
SINE	REAL	Sine wave signal (cycle period is a update time X 4.)

◆ Description of action



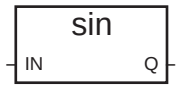
If RUN = TRUE, SINE = sin (2π X UP / (MAXIMUM + 1)).
(The cycle period is PERIOD X (MAXIMUM + 1).)

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

SIN (Sine) Function

SYMBOL



◆ Functional description

Outputs the sine of REAL type data.
The input is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	Range of real numbers (in radians)

◆ Output parameters

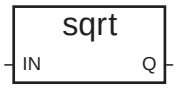
Parameter	Data type	Content
Q	REAL	-1.0 to +1.0

◆ Description of action

Q = sin(IN)

SQRT (Square root) Function

SYMBOL



◆ Functional description

Outputs the square root of REAL type data.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	IN ≥ 0.0

◆ Output parameters

Parameter	Data type	Content
Q	REAL	

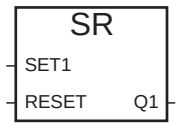
◆ Description of action

$Q = \sqrt{IN}$

If IN < 0.0, Q = 0.0.

SR (Set dominant bistable) Function block

SYMBOL



◆ Functional description

Sets a set dominant latch.

◆ Input parameters

Parameter	Data type	Content
SET1	BOOL	
RESET	BOOL	

◆ Output parameters

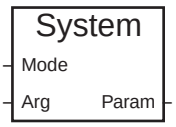
Parameter	Data type	Content
Q1	BOOL	If SET1 = TRUE, Q1 is TRUE (set dominant). If RESET = TRUE, Q1 is FALSE.

◆ Description of action

SET1	RESET	Q1	Results of Q1
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE (set dominant)
TRUE	FALSE	TRUE	TRUE (set dominant)
TRUE	TRUE	FALSE	TRUE (set dominant)
TRUE	TRUE	TRUE	TRUE (set dominant)

SYSTEM (Access to system) Standard operator

SYMBOL



◆ Functional description

Reads out cycle time, etc.

◆ Input parameters

Parameter	Data type	Content
MODE	DINT	Command
ARG	DINT	0 if MODE ≠ SYS_TWRITE. New cycle time (ms) if MODE = SYS_TWRITE.

◆ Output parameters

Parameter	Data type	Content
PARAM	DINT	Result of access

◆ Description of action

Though MODE is data of DINT type, it is input as a predefined literal.

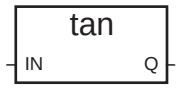
MODE	ARG	Description
SYS_TALLOWED	0	Reads out the current setting value of the cycle time.
SYS_TCURRENT	0	Reads out the execution time of the previous cycle.
SYS_TMAXIMUM	0	Reads out the maximum value of the execution time.
SYS_TOVERFLOW	0	Reads out the number of overflow times of the execution time.
SYS_TRESET	0	Resets the maximum value of execution time and the number of overflow times.
SYS_TWRITE	Any value	Changes the cycle time setting.
SYS_ERR_TEST	0	Checks the run time error. No error when PARAM = 0.

◆ Cautions

The cycle time and the execution time are data of DINT type, and are expressed in milliseconds.

TAN (Tangent) Function

SYMBOL



◆ Functional description

Outputs the tangent of REAL type data.
The input is in radians.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	any value other than ±π/2 X n (n = 1, 3, 5.....) in radians

◆ Output parameters

Parameter	Data type	Content
Q	REAL	

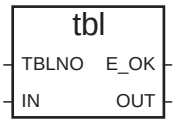
◆ Description of action

Q = tan(IN)

TBL (Linearization table lookup)

Function block (for DMC50)

SYMBOL



◆ Functional description

Converts REAL type data using a linearization table.

◆ Input parameters

Parameter	Data type	Content
TBLNO	DINT	Specifies the group ID of TBL/TBR Setup Parameters.
IN	REAL	Input data

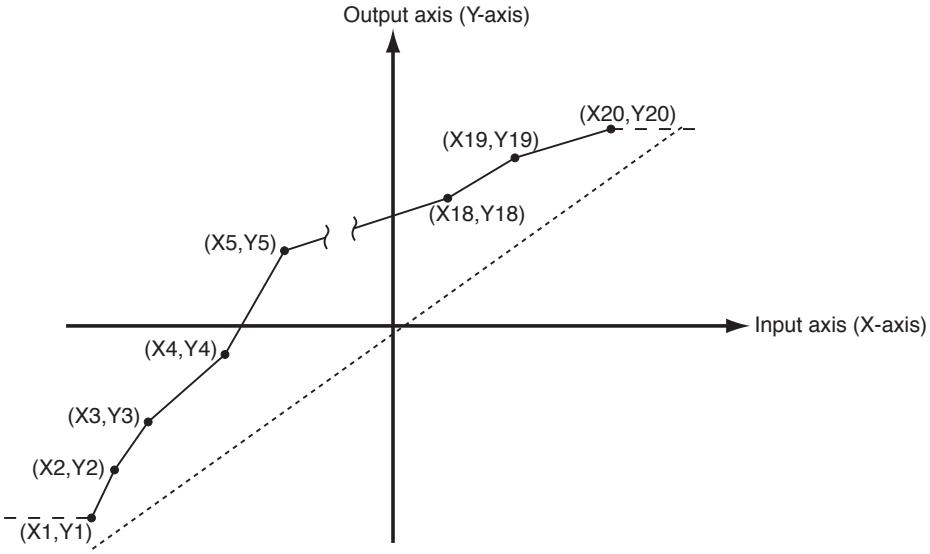
◆ Output parameters

Parameter	Data type	Content
OUT	REAL	Output data

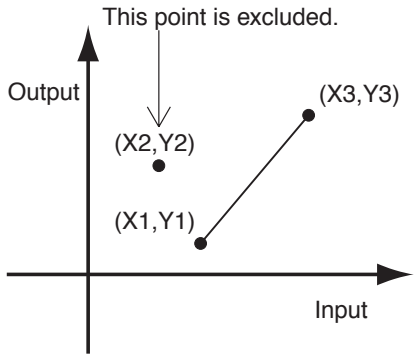
◆ Description of action

The linearization table specified by the TBLNO is used to perform the conversion process. The linearization table is a Calculation Parameter [TBL / TBR Setup].

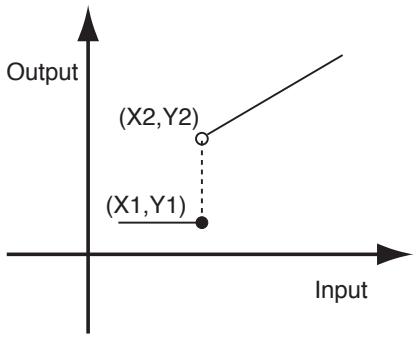
- Item Xn represents the input axis (X-axis) while Yn represents the output (Y-axis).
- If IN ≤ X1, OUT becomes Y1.
- If IN > X (last point), OUT is Y (last point).



- Make the settings so that X value increases monotonously along the input axis as the index number increases (X1 ≤ X2 ≤ ... ≤ X19 ≤ X20).
- Points whose X values do not increase as their index numbers increase are excluded.

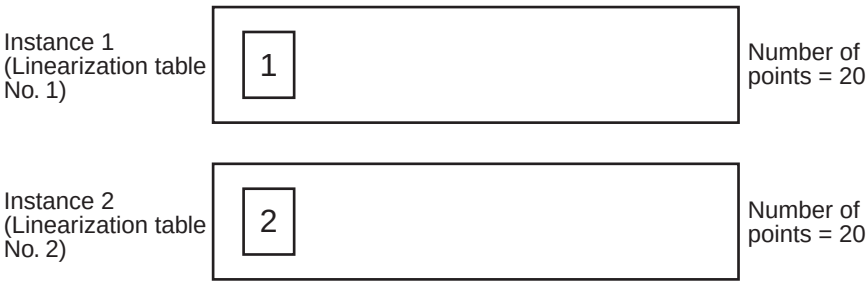


- If the setting is made so that X1 = X2 on the input axis, Y1 becomes the output.



◆ Linearization table number

To create a linearization table, add an instance of the Calculation Parameter [TBL / TBR Setup] in SLP-D50. Here, one instance corresponds to one linearization table . When using multiple linearization tables, create a separate instance for each table. For example, when making two linearization tables, two instances need to be created.



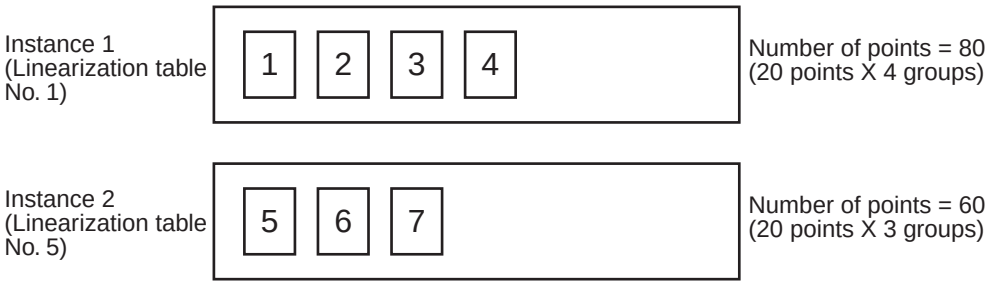
◆ How to use linearization table with 20 points or more

The following describes how to create a linearization table with 20 points or more :
(The linearization table has 20 points per group.)
A linearization table is created by adding an instance in SLP-D50 with:

- Top group ID = Linearization table No.
- Number of groups = Number of desired points / 20

(The number of desired points must be a multiple of 20.)
For example, to create a linearization table with linearization table No. 1 and 80 points, add an instance with the top group ID = 1 and the number of groups = 4. As a result, all the points in the groups 1 to 4 of the instance makes up a linearization table.

The linearization table number is input to the TBLNO argument of the TBL function block. In a linearization table with 20 points or more, however, only the top group ID of the instance is valid as the linearization table number. In the following example, only "1" and "5" are valid as linearization table numbers, and "2 to 4" and "6 to 7" are invalid number designations:



◆ Cautions

To use this function block, it is necessary that an instances of the Calculation Parameter [TBL / TBR Setup] has been downloaded in the controller.

◆ Note

SLP-D50 provides the linearization table support facility. Use of this facility makes it possible to verify the contents of an edited linearization table through the graph display (linearization table lookup/linearization table reverse lookup).

◆ Action under abnormal conditions

If a specified group ID is not present, or if the top group ID is not specified for a table of 20 or more points, an execution error will occur.

The action is as follows when an execution error has occurred:

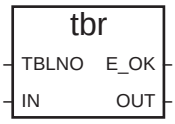
- OUT = IN (The value of input IN is output without any processing.)
- E_OK = FALSE (Error)

MEMO

TBR (Linearization table reverse lookup)

Function block (for DMC50)

SYMBOL



◆ Functional description

Restores the original value from a converted value using the same linearization table.

◆ Input parameters

Parameter	Data type	Content
TBLNO	DINT	Specifies the group ID of TBL / TBR Setup.
IN	REAL	Input data

◆ Output parameters

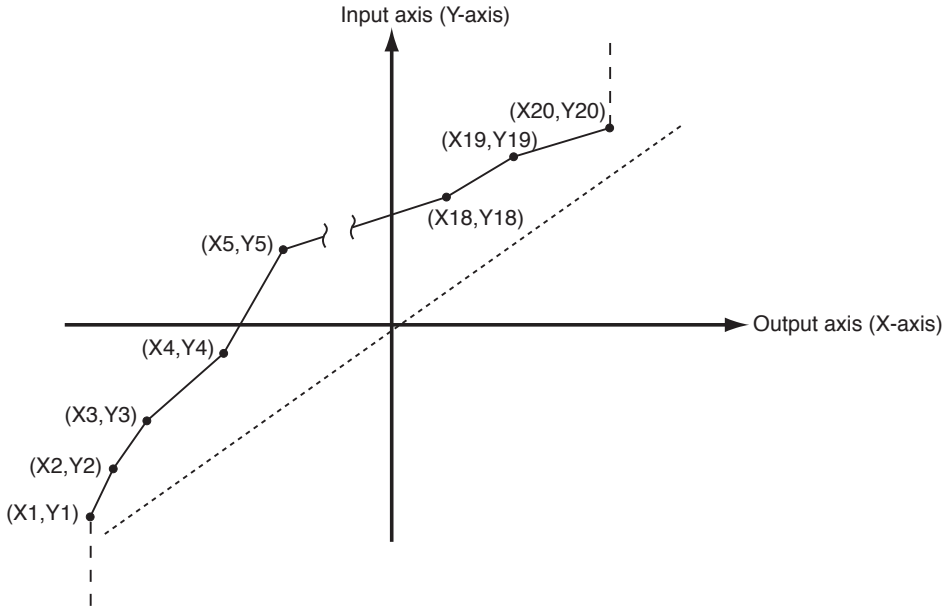
Parameter	Data type	Content
OUT	REAL	Output data

◆ Description of action

Restores the original value from the value converted with the TBL (Linearization table lookup) function block.

A linearization table specified by the TBLNO is used to perform the reverse conversion process. The linearization table is a Calculation Parameter [TBL / TBR Setup].

- Item Yn represents the input (Y-axis) value, while Xn represents the output (X-axis) value.
- If the IN ≤ Y1, OUT becomes X1.
- If the IN > Y (last point), OUT is X (last point).



◆ Cautions

- To use this function block, it is necessary that an instance of the Calculation Parameter [TBL / TBR Setup] has been downloaded in the controller.
- The TBL (Linearization table lookup) and TBR (Linearization table reverse lookup) function blocks use the same linearization table. The linearization table reverse lookup function block can perform reverse conversion correctly only when it is assured that Yn is monotone increasing (decreasing) with respect to Xn on the linearization table. If increase and decrease are mixed in the linearization table, data cannot be converted back into the original one correctly.

◆ Note

SLP-D50 provides the linearization table support facility. Use of this facility makes it possible to verify the contents of an edited linearization table through the graph display (linearization table lookup/linearization table reverse lookup).

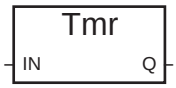
◆ Action under abnormal conditions

If a specified group ID is not present, or if the top group ID is not specified for a table of 20 or more points, an execution error will occur.

The action is as follows when an execution error has occurred:
- OUT = IN (The value of input IN is outputted without any processing.)
- E_OK = FALSE (Error)

TMR (Convert to timer) Standard operator

SYMBOL



◆ Functional description

Converts a value to a TIME type one.
Available data types are DINT and REAL.

◆ Input parameters

Parameter	Data type	Content
IN	DINT, REAL	

◆ Output parameters

Parameter	Data type	Content
Q	TIME	

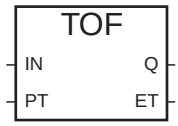
◆ Description of action

• When IN is of DINT type:
Q = millisecond value given by IN
Example: When IN = 1234, Q is T#1s234ms.
If IN < 0, Q is undefined.

• When IN is of REAL type:
Q = millisecond value given by the integer part of IN (the fractional portion is dropped)
Example: When IN = 1234.56, Q is T#1s234ms.
If IN < 0.0, Q is undefined.

TOF (OFF delay timer) Function block

SYMBOL



◆ Functional description

Turns off at a specified time after the falling edge has been detected.
Also outputs the elapsed time.

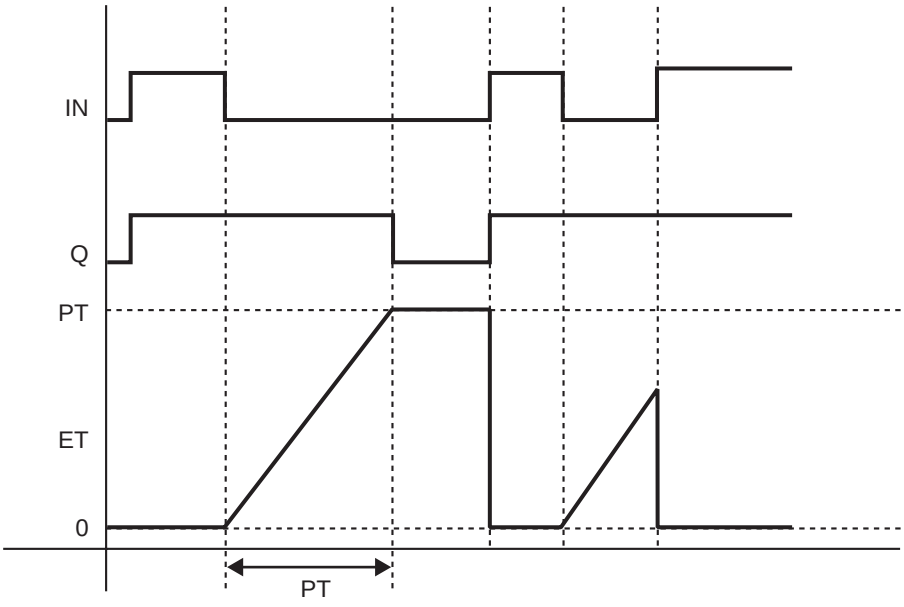
◆ Input parameters

Parameter	Data type	Content
IN	BOOL	Timer starts when the falling edge is detected Timer stops and is reset when the rising edge is detected
PT	TIME	Preset time

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	FALSE if ET = PT.
ET	TIME	Elapsed time

◆ Description of action

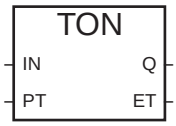


◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

TON (ON delay timer) Function block

SYMBOL



◆ Functional description

Turns on at a specified time after the rising edge has been detected.
Also outputs the elapsed time.

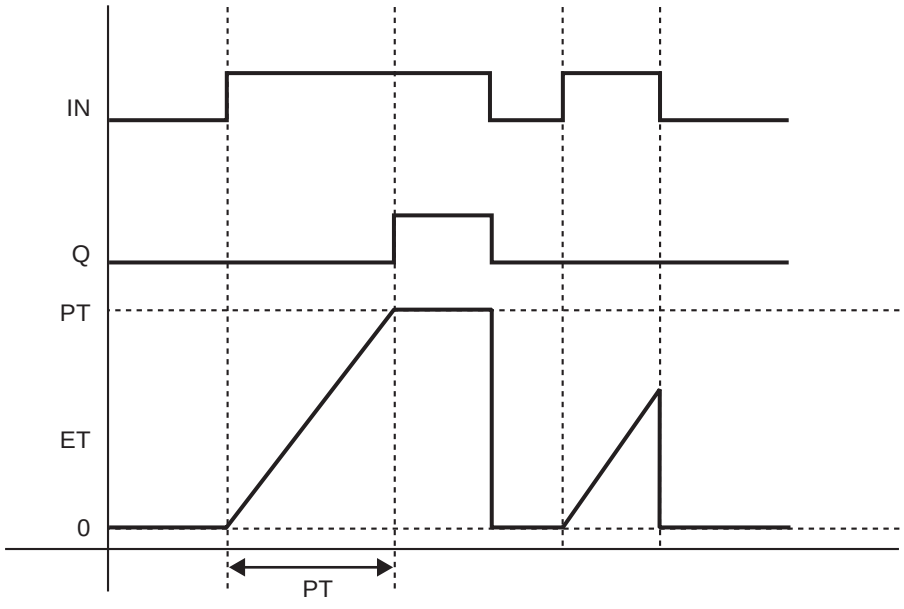
◆ Input parameters

Parameter	Data type	Content
IN	BOOL	Timer starts when the rising edge is detected Timer stops and is reset when the falling edge is detected
PT	TIME	Preset time

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	TRUE if ET = PT.
ET	TIME	Elapsed time

◆ Description of action

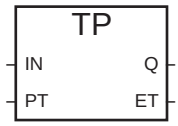


◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

TP (Pulse timer) Function block

SYMBOL



◆ Functional description

Turns on during a specified amount of time after the rising edge has been detected.
Also outputs the elapsed time.

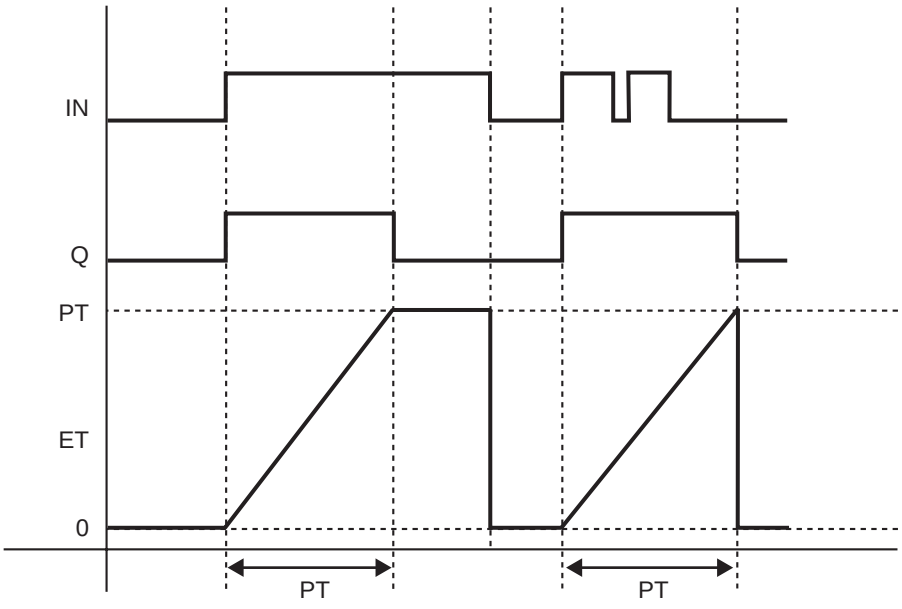
◆ Input parameters

Parameter	Data type	Content
IN	BOOL	Timer starts when the rising edge is detected
PT	TIME	Preset time

◆ Output parameters

Parameter	Data type	Content
Q	BOOL	TRUE if ET < PT while the timer is working.
ET	TIME	Elapsed time

◆ Description of action



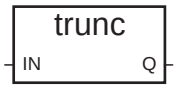
- Changes of IN are ignored while the timer is on.
- Reset occurs if ET has reached PT (counted by the timer) and IN = FALSE. (ET becomes 0; Q becomes FALSE)

◆ Cautions

Must be executed at every application execution cycle to perform time management internally.

TRUNC (Truncate the fractional portion) Function

SYMBOL



◆ Functional description

Drops the fractional portion of REAL type data.

◆ Input parameters

Parameter	Data type	Content
IN	REAL	

◆ Output parameters

Parameter	Data type	Content
Q	REAL	Integer part of IN

◆ Description of action

Q = Integer part of IN

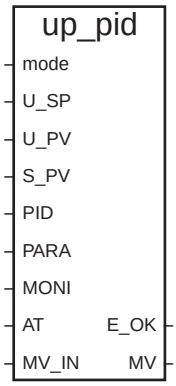
[Sample of action]
When IN = 10.567, Q = 10.0.
When IN = -10.567, Q = -10.0.

MEMO

UP_PID (Use-point PID operation)

Function block (for DMC50)

SYMBOL



◆ Functional description

Performs two-input one-output PID operation for disturbance suppression.

! Handling precautions

The UP_PID (Use-point PID operation) is a control method assuming a narrow range of processes to be controlled. Consequently, this function block may not be suitable for certain processes depending on their characteristics. It is recommended to use any of PID_A (standard PID operation), PID_CAS (cascade PID operation) and Ra_PID (RationaLOOP PID operation) function blocks first.

This function block is not suitable for applications where:

- Set point is changed during operation
- Rise time or overshoot amount is important
- You want to control with the PID constants just obtained by the auto tuning (The values obtained by the auto tuning are the constants to be used as guidelines. To get the best control performance, it is necessary to determine the best PID constants manually.)

◆ Input parameters

Parameter	Data type	Content
MODE	BOOL	TRUE = MANUAL mode, FALSE = AUTO mode
U_SP	REAL	SP at the use point (industrial unit)
U_PV	REAL	PV at the use point (industrial unit)
S_PV	REAL	PV at the disturbance source (industrial unit)
PID	DINT	Specifies a group ID of UP_PID Constants.
PARA	DINT	Specifies a group ID of UP_PID Options.
MONI	DINT	Specifies a group ID of UP_PID Monitor.
AT	BOOL	Starts/stops auto tuning.
MV_IN	REAL	Inputs the value to be output in MANUAL mode.

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
MV	REAL	Manipulated variable

◆ Description of action

The UP_PID function block (use-point PID operation) has the following additional features:

- AUTO/MANUAL tranfer control
- Auto tuning (AT)

Input the PV you actually want to control in the U_PV (use PV).
Input the PV for disturbance detection in the S_PV (source PV). This is not used when the control method is "use single". (You can specify the control method in the Calculation Parameter [UP_PID Options].)

The use-point PID operation is performed in AUTO mode (MODE = FALSE).

- The constants used in the use-point PID operation (such as the PID constants, output high/low limit values, etc.) are grouped in the Calculation Parameter [UP_PID Constants]. (Specify a group ID of the "UP_PID Constants" in PID.)
- The options used in the use-point PID operation (such as the direct/reverse action, auto tuning method, etc.) are grouped in the Calculation Parameter [UP_PID Options]. (Specify a group ID of the [UP_PID Options] in PARA.)
- If you want to monitor the UP_PID input/output data (such as PV, SP and MV) easily en bloc, use the Calculation Monitor Parameter [UP_PID Monitor]. (Specify a group ID of the [UP_PID Monitor] in MONI. If you do not use this Parameter, specify "0".)

In MANUAL mode (MODE = TRUE), MV = MV_IN.

- When the mode changes from AUTO to MANUAL, preset value output or bumpless transfer can be specified optionally. (This option is specified in the Calculation Parameter [UP_PID Options].)
- When the mode changes from MANUAL to AUTO, the control output starts from the MV_IN input parameter value.

◆ Executing the auto tuning

In AUTO mode, the auto tuning starts when the AT rising edge is detected.

- At the completion of auto tuning, the PID values obtained by the selected AT method is written into the Calculation Parameter [UP_PID Constants]. After completion, the ordinal use-point PID calculation is performed.
- The auto tuning stops when the AT falling edge is detected.
- You can check whether the auto tuning is being executed or has been ended by the [Mode] item of the Calculation Monitor Parameter [UP_PID Monitor].

For details about the auto tuning operation diagram, stop conditions, and cautions,
☞refer to Appendix 2, Auto Tuning.

◆ Cautions

- To use the UP_PID function block, it is required that the corresponding Calculation Parameters and Calculation Monitor Parameter (UP_PID Options, UP_PID Constants and UP_PID Monitor) have been downloaded in the controller. Download the Calculation Parameters and Calculation Monitor Parameter with each group ID specified in the input parameters PID, PARA and MONI, respectively.
- If you specify a value changing at every application execution cycle (such as ramp SP and RSP) in U_SP, set the [Initialization on SP Changes] item of the Calculation Parameter [UP_PID Options] to "Option 2 (Not initialized)" to prevent unnecessary PID calculation initialization.
- Must be executed at every application execution cycle to perform time management internally.

◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

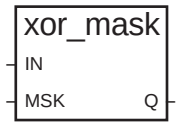
- The Calculation Parameters and Calculation Monitor Parameter specified in the input parameters PID, PARA and MONI have not been downloaded in the controller.
- The cycle timing setting of the project is 0s.

The action is as follows when an execution error has occurred:

- MV = MV_IN
- E_OK = FALSE (Error)

XOR_MASK (Bitwise XOR) Standard operator

SYMBOL



◆ Functional description

Outputs the bitwise exclusive OR of 2 pieces of DINT type data.

◆ Input parameters

Parameter	Data type	Content
IN	DINT	
MSK	DINT	

◆ Output parameters

Parameter	Data type	Content
Q	DINT	Bitwise exclusive OR of IN and MSK

◆ Description of action

Q = Bitwise exclusive OR of IN and MSK

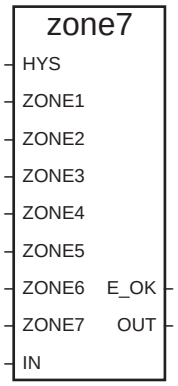
[Sample of action]
IN = 16#3333_CCCC = 2#0011_0011_0011_0011_1100_1100_1100_1100
MSK = 16#F0F0_F0F0 = 2#1111_0000_1111_0000_1111_0000_1111_0000
Q = 16#C3C3_3C3C = 2#1100_0011_1100_0011_0011_1100_0011_1100

The '_' (underscore) used with "16#" or "2#" is only a delimiter to increase the readability of hexadecimal and binary numbers. It is neglected when evaluating the value of a number.

ZONE7 (Zone selector)

Function block (for DMC50)

SYMBOL



◆ Functional description

Evaluates REAL type data using seven zone selecting conditions and outputs a value between 0 and 7 (total of 8 zones).
Used to switch the PID group according to the SP or PV values.

◆ Input parameters

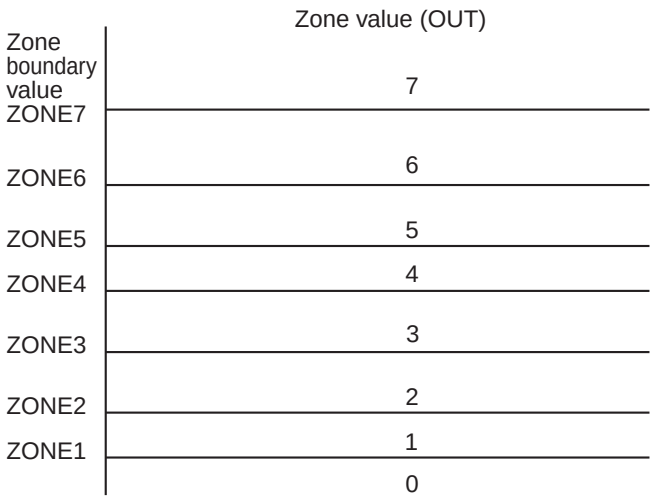
Parameter	Data type	Content
HYS	REAL	Zone-to-zone hysteresis, HYS ≥ 0.0
ZONE1 to 7	REAL	Zone boundary values, ZONE1 < ZONE2 < .... < ZONE7
IN	REAL	Input data

◆ Output parameters

Parameter	Data type	Content
E_OK	BOOL	TRUE = Normal, FALSE = Error
OUT	REAL	Zone value (0 to 7)

◆ Description of action

If ZONE (n) ≤ IN < ZONE (n+1), OUT = n. (n = 0 to 7)



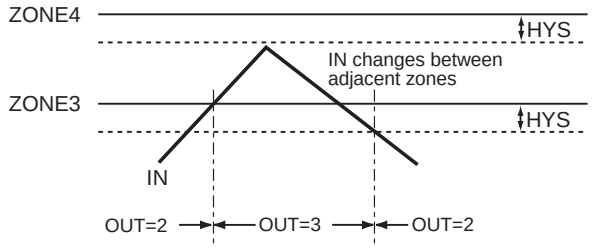
◆ Hysteresis action

If setting HYS to a value equal to or greater than 0.0, and hysteresis is applied when zones are switched.
The hysteresis action operates when IN changes between adjacent zones in a decreasing direction. However, when IN changes between nonadjacent zones such as from ZONE4 < IN < ZONE5 to ZONE2 < IN < ZONE3, zone switching is made with no hysteresis.

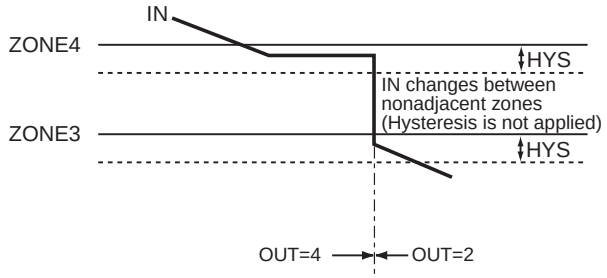
Output is defined as follows:

- While IN is increasing (no hysteresis while increasing):
  If ZONE (n) ≤ IN < ZONE (n+1), OUT = n.
- While IN is decreasing:
  [if ZONE(n) < IN < ZONE(n+1) - HYS]
  OUT = n.

[if ZONE(n) - HYS < IN < ZONE(n) and IN is changing between adjacent zones]
OUT = n.



[if ZONE(n) - HYS < IN < ZONE(n) and IN is changing between nonadjacent zones]
OUT = n - 1.



◆ Action under abnormal conditions

If any of the following conditions is met, an execution error will occur:

- Zone values (ZONE1 to 7) are not monotonously increasing.
  (Any setting other than "ZONE 1 < .... < ZONE 7" is an error.)
- Each zone width is not greater than HYS.
- HYS < 0.0

The action is as follows when an execution error has occurred:

- OUT = 0
- E_OK = FALSE (Error)

MEMO

Appendix 1. ISaGRAF VARIABLES AND PARAMETER DATA TYPES

■ Data types of ISaGRAF variables

Data types of ISaGRAF variables are as follows:

Data type	Description	Range
BOOL	Boolean value	0 to 1 (0: FALSE, 1: TRUE)
DINT	Double-precision integer	-2147483647 to +2147483647
REAL	Real number	Approx. 10^{-38} to 10^{38} (Maximum precision of 7 digits in decimal)
TIME	Time type data	0 to T#23h59m59s999ms
STRING *	Character string	Variable-length, maximum of 255 single byte characters

* Not accessible by the CPL communications.

Note

- You must assign a data type for each variable to be used in your programs with the dictionary editor of the ISaGRAF workbench while editing your project.
- BOOL type is also referred to as boolean type in ISaGRAF.
- DINT type is 32-bit signed integer data which is also referred to as integer type in ISaGRAF. It starts from "-2147483647".
- REAL type is 32-bit floating-point (single-precision) data defined in IEEE754.
- TIME type is 32-bit unsigned integer data (in units of millisecond) which is also referred to as timer type in ISaGRAF.
- STRING type is also referred to as variable-length message string type, message string type, or MESSAGE type on the ISaGRAF. This type is not accessible by the CPL communications.

■ Parameter data types

The data types for each element of "Parameter", data for DMC50 only, are as follows:

Data type	Description	Range
BOOL	Boolean value	0 to 1 (0: FALSE, 1: TRUE)
DINT	Double-precision integer	-2147483647 to +2147483647
DWORD	32-bit binary	0 to 16#FFFFFFFF
REAL	Real number	Approx. 10^{-38} to 10^{38} (Maximum precision of 7 digits in decimal)

Note

- For the data type of a specific item, refer to the corresponding Parameter table.
- DINT type is 32-bit signed integer data. It starts from "-2147483647".
- DWORD type is 32-bit data. This type is mainly used for handling bit data, etc. as a binary image.
- REAL type is 32-bit floating-point (single-precision) data defined in IEEE754.

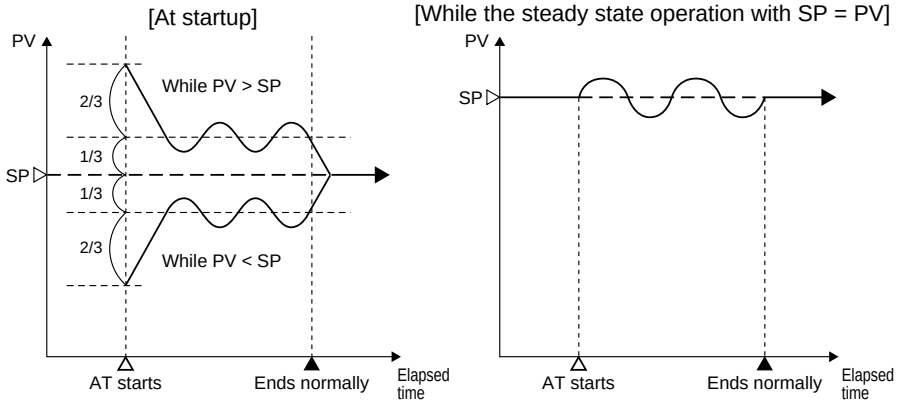
Appendix 2. AUTO TUNING

■ Action during auto tuning

● PID_A, PID_CAS, and UP_PID function blocks

In auto tuning, PID constants are determined by the limit cycle operation regardless of the [Auto Tuning Method] setting.

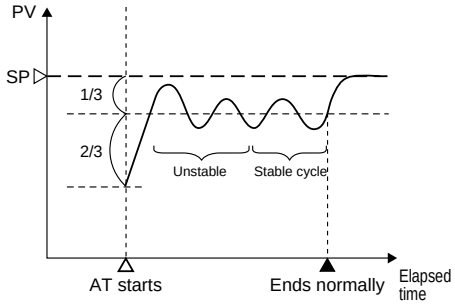
- The on/off outputs in a limit cycle are , respectively, the values of [Output High Limit] and [Output Low Limit] items in the PID_A Constants(or PID_CAS Constants, or UP_PID Constants) selected at the start of auto tuning.
- Output is alternated between on and off at the point where the difference between the SP and PV at the start of auto tuning are divided at the ratio of "2:1". The on/off switching point will not change until the end of auto tuning, even if the SP is changed during auto tuning execution.
- A two-cycle on/off output is performed during auto tuning.
  For example, if a reverse action is conducted while "SP ≥ PV", auto tuning ends after the output turns "on → off → on → off → on".



● Ra_PID function block

In auto tuning, PID constants are determined by the limit cycle operation.

- The on/off outputs in a limit cycle are , respectively, the values of [Output High Limit] and [Output Low Limit] items in the Ra_PID Constants selected at the start of auto tuning.
- Output is alternated between on and off at the point where the difference between the SP and PV at the start of auto tuning are divided at the ratio of "2:1". The on/off switching point will not change until the end of auto tuning even if the SP is changed during auto tuning execution.
- For making adjustments of PID constants with high precision, it checks to make sure that the PV is sufficiently attenuated and amplified in each limit cycle. It ends the limit cycle operation if stable cycles are observed, and calculate the PID parameters.
- The number of cycles to establish stable cycles depends on the process to be controlled.



■ Conditions to stop auto tuning

● PID_A function block

If any of the following conditions is met during auto tuning, auto tuning is stopped without writing the PID constants:

- The falling edge of the AT input parameter has been detected.
- The mode has been changed from AUTO to MANUAL. (Auto tuning not available in MANUAL mode)
- The PARA input parameter (group ID of PID_A Options) has changed.
- The PID input parameter (group ID of PID_A Constants) has changed.
- The control action (direct / reverse action) has changed.
- ISaGRAF has entered a mode other than the real time mode (RT).
- The cycle time has changed.
- The E_OK output parameter has become FALSE (ERR).
- A power failure has occurred.

● PID_CAS function block

If any of the following conditions is met during auto tuning, auto tuning is stopped without writing the PID constants:

- The falling edge of the AT input parameter has been detected.
- The mode has been changed from RUN to READY. (Auto tuning not available in READY mode)
- The mode has been changed from AUTO to MANUAL. (Auto tuning not available in MANUAL mode)
- The mode has been changed from LOCAL to REMOTE.
- The mode has been changed from REMOTE to LOCAL.
- The PARA input parameter (group ID of PID_CAS Options) has changed.
- The M_PID/S_PID input parameter (group ID of PID_CAS Constants) has changed.
- The control action (direct / reverse action) has changed.
- ISaGRAF has entered a mode other than the real time mode (RT).
- The cycle time has changed.
- The E_OK output parameter has become FALSE (ERR).
- A power failure has occurred.

● **Ra_PID function block**

If any of the following conditions is met during auto tuning, auto tuning is stopped without writing the PID constants:

- The falling edge of the AT input parameter has been detected.
- The mode has been changed from AUTO to MANUAL. (Auto tuning not available in MANUAL mode)
- The PARA input parameter (group ID of Ra_PID Options) has changed.
- The PID input parameter (group ID of Ra_PID Constants) has changed.
- The control action (direct /reverse action) has changed.
- ISaGRAF has entered a mode other than the real time mode (RT).
- The cycle time has changed.
- The E_OK output parameter has become FALSE (ERR).
- A power failure has occurred.

● **UP_PID function block**

If any of the following conditions is met during auto tuning, auto tuning is stopped without writing the PID constants:

- The falling edge of the AT input parameter has been detected.
- The mode has been changed from AUTO to MANUAL. (Auto tuning not available in MANUAL mode)
- The PARA input parameter (group ID of UP_PID Options) has changed.
- The PID input parameter (group ID of UP_PID Constants) has changed.
- The control action (direct / reverse action) has changed.
- The control method (use-point/single control) has changed.
- ISaGRAF has entered a mode other than the real time mode (RT).
- The cycle time has changed.
- The E_OK output parameter has become FALSE (ERR).
- A power failure has occurred.

■ Cautions for auto tuning

If PV vibrates a lot due to noises or other causes, auto tuning may not be executed correctly. (To be more specific, auto tuning immediately stops and the PID constants have been updated to very small values.) If this is the case, remove the vibration by passing the PV through a light filter, then execute the auto tuning.

Revision History

Printed Date	Manual Number	Edition	Revised pages	Description
02-12	**CP-SP-1130E**	**1st Edition**		

Specifications are subject to change without notice.

YAMATAKE

Yamatake Corporation
Control Products Division
Head office : Totate International Building
2-12-19 Shibuya Shibuya-ku Tokyo 150-8316 Japan
Inquiries to : International Business Division
Phone : 81-3-3486-2331, Fax : 81-3-3486-2300 (Sales)
Phone : 81-466-20-2307, Fax : 81-466-27-9264 (Customer Service)
http://www.yamatake.com

Printed in Japan.
1st Edition: Issued in Dec., 2002(W)

This has been printed on recycled paper.