

Module Type Controller

Control Module / Communication Module

DMC50

The DMC50 is a module type multi-loop controller for the precise control of analog process variables such as temperature, flow rate, PH and liquid levels.

- Each product in the line up is provided with a control module to control analog process variables and with a communication module for digital communications.
- The control module features multiple advanced PID controller functions and supplementary operational functions that provide optimal accuracy, high stability and excellent response. It has the ability to allow the user to freely organize process logics and is effective for high level designing of equipment.
 - The communication module not only fulfills RS-485 communication but also supports Ethernet and facilitates the freedom to handle all systemization needs.
- The control module is equipped with multiple analog inputs/outputs and multiple digital inputs/outputs as well as communications capabilities all in a single module so that it can operate as a stand-alone unit and can be, therefore, installed separately in the field.
- The DMC50 furthermore comes equipped with ISaGRAF* to allow an optional control method for the user's various requirements.
- A development environment has been incorporated to allow the user to freely design custom control processes.
 - Operation types: More than 100 types such as four rules, statistics, logic, time and control.
 - Computing capacity: More than 8-loop calculations are possible in terms of PID computing capability.
 - Language: Optimum Language selection is possible for responding to an application in writing complicated process programs.
 - Highly efficient program development is possible due to the application of the structured programming concept.
 - Its development took into consideration easy utilization which lead to the design of a process controller having the functions of parameter setting, trend monitoring, etc.
- The controller has a computing speed that is compatible for the control of pressure, flow rate, and fast temperature ramp-up.
 - The controller computes a 50ms update cycle for the inputs and outputs, and is capable of executing applications very quickly.
- Reinforcements for stability and overshoot control algorithms have been incorporated to the design.
 - Controllability that cannot be accomplished by conventional methods becomes possible with the utilization of the "Rationa Loop" high accuracy control algorithm,



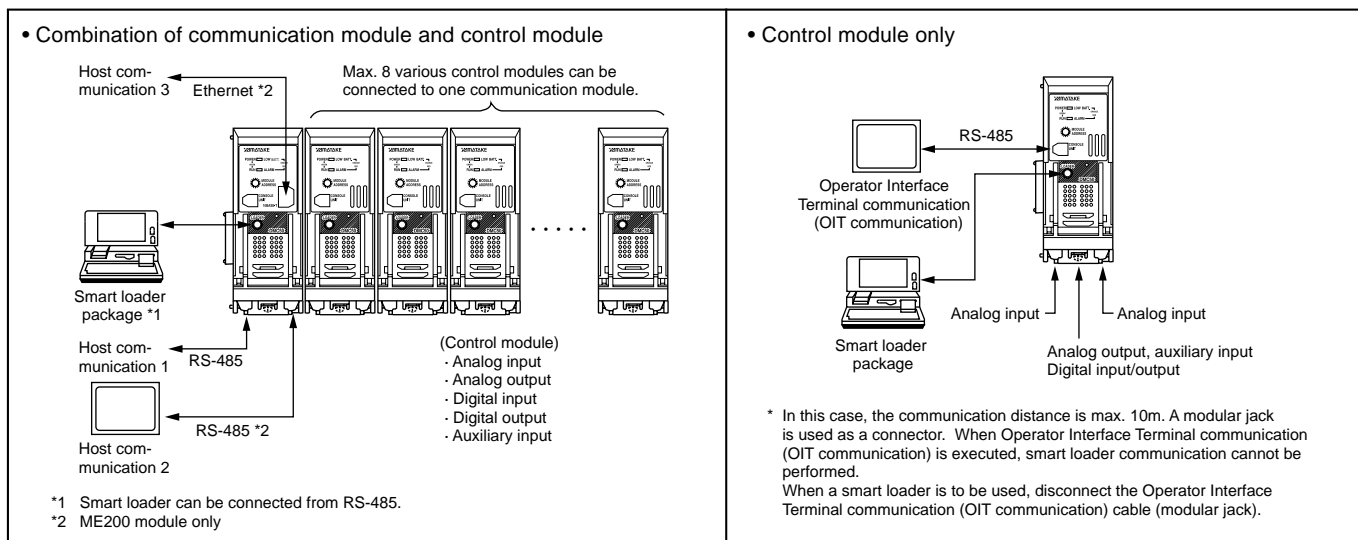
which has become required by system upgrading demands for direct heating systems.

*ISaGRAF is a registered trademark of AlterSys.

■ Features

- **Module commonality**
 - High reliability: Designed to be able to be used for 5 years (45,000 hours) continuous use at an ambient temperature of 50°C and a load of 80%.
 - Maintenance: The base unit, main body, and terminal unit separate therefore greatly facilitating inspection and maintenance.
 - Less wiring required: Capable of supplying electrical power to each individual module via the connected base units and can exchange data between the modules, thus resulting in savings in wiring work and in also the wire required.
- **Control module**
- **High resolution. processing**
 - Thermocouple input: 1/100°C of a specific range.
 - RTD input: 1/5000°C (Special model)
 - 4 to 20 mA input: 1/32000 resolution
- **Many analog process variables can be processed.**
 - Analog input: 4 universal inputs (Max. 8 inputs with voltage or thermocouple. 2 heater voltage compensation inputs are equipped.)
 - Analog output: 4 points
- **Digital input and output signals are provided to be used for relay sequence inputs/outputs such as interlock signals and operation signals.**
 - Digital input: 12 photo-coupler inputs
 - Digital output: 16 transistor outputs
- **Communication module**
 - Network accommodation:**
 - Each control module connected to a communication module can be communicated with a CPL (Controller Peripheral Link: Upper level communications protocol of Yamatake) client host and/or a loader via Ethernet (10BASE-T) and/or RS-485.

■ Configuration Examples



■ Common General Specifications

	Module type	Communication module		Control module	
	Type	Ethernet type	RS-485 type	High resolution (2/4 loops) type	Special (2/4 100ps) type
	Basic model number	DMC50ME20□	DMC50MR20□	DMC50CH20□/40□	DMC50CS20□/40□
Memory backup		Clock data Parameter: RAM with battery backup Battery backup time: 10 years without power in room temperature Firmware: Flash ROM		Clock data User program, parameter: RAM with battery backup Battery backup time: 10 years without power in room temperature Adjustment data at factory shipment: EE-PROM Firmware: Flash ROM	
Clock (real time clock) accuracy		-100 to +20ppm (standard condition) / -200 to +20ppm (operating condition)			
Power supply	Rated voltage	21.6 to 26.4Vdc			
	Max. current consumption	0.2A		0.6A	
	Power ON inrush current	30A max. per module (50μs max.)			
	Controller startup time	10s max.			
	Power failure dead time	400μs max.			
Insulation resistance	20MΩ min. between power (24V) and isolated input/output				
Dielectric strength	Between power (24V) and isolated input/output: 500Vdc 50/60Hz 1min. Between isolated input and output: 500Vdc 50/60Hz 1min.			Between power (24V) and other isolated input/output: 500Vdc 50/60Hz 1min. Between isolated input and output: 500Vdc 50/60Hz 1min.	
Standard conditions	Ambient temperature	23±2°C			
	Ambient humidity	60±5%RH (no condensation allowed)			
	Power supply	24Vdc±2%			
	Vibration resistance	0m/s ²			
	Shock resistance	0m/s ²			
	Mounting angle	Reference plane (vertical) ±3°C			
Operating conditions	Ambient temperature	0 to 50°C			
	Ambient temperature for guaranteed accuracy	—		0 to 50°C	10 to 40°C
	Ambient humidity	20 to 90%RH (no condensation allowed)			
	Vibration resistance	0.00 to 1.96 m/s ²			
	Shock resistance	0.00 to 9.81 m/s ²			
	Mounting angle	Reference plane ±10°C			
Transport/storage conditions	Ambient temperature	-20 to +70°C			0 to 50°C
	Ambient humidity	10 to 95%RH (no condensation allowed)			
	Vibration resistance	0.00 to 4.90m/s ² (10 to 60Hz for 2 hours each in X,Y and Z directions)			
	Shock resistance	0 to 490 m/s ² (3 times each vertically)			
	Package drop test	Drop height: 60cm (1 angle, 3 edges and 6 planes; free fall)			
Others	Max. number of connectable modules	Max. 8 control modules connectable to one communication module			
	Mounting	DIN rail mounting or direct panel mounting			
	Case material	Modified polyphenylene ether resin			
	Case color	DIC547 (No.15 revision)			
	Weight	600g max.			
	Approvals	EN61326-1-1997, AI-1998			

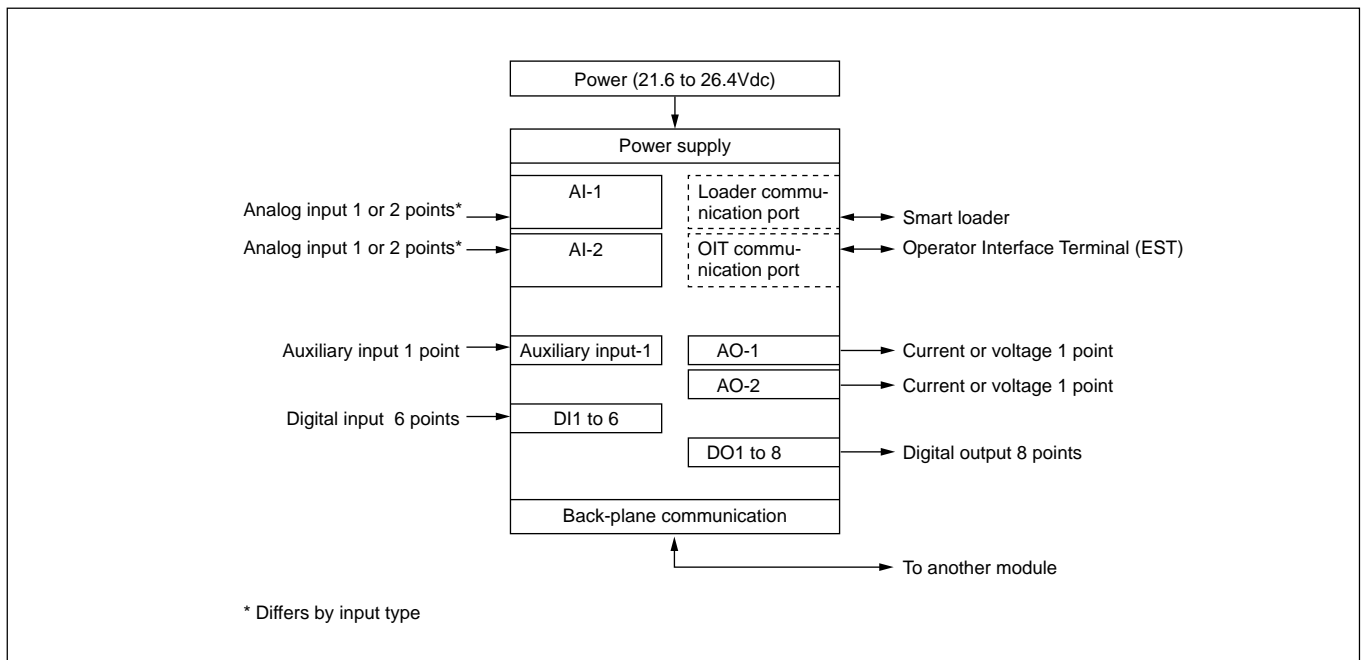
Control Module Individual Specifications

The optimum control module can be selected based on the desired input type, accuracy and number of input/output points. A single control module can handle multiple analog controls and logic operations. The main body is incorporated with advanced PID control and other complex functions. In addition, it is capable of processing text language, thus facilitating flexible management of sophisticated process control.

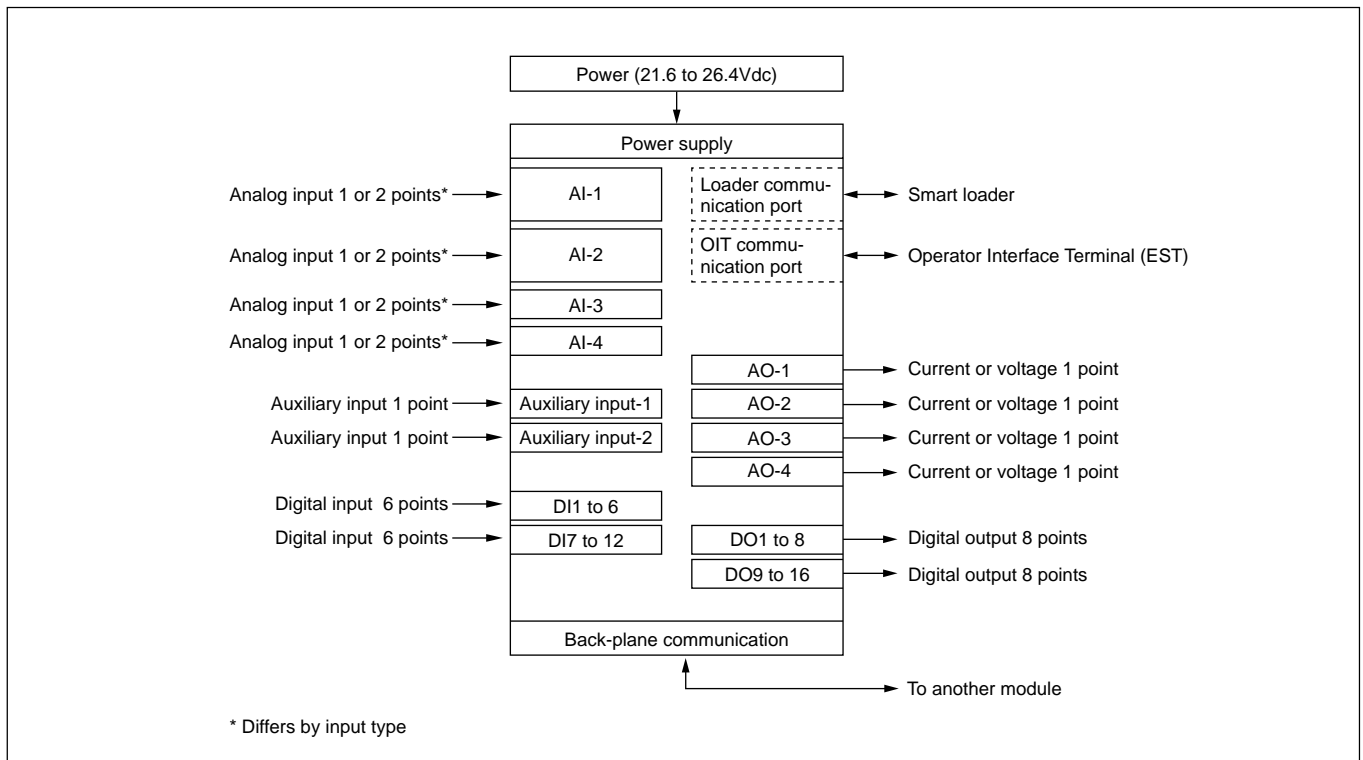


Input/output configuration

• 2 loop type



• 4 loop type



Individual Specifications

Model No.		DMC50CH20□	DMC50CH40□	DMC50CS20□	DMC50CS40□
Model		High resolution model		Special model	
Type		2 loop type	4 loop type	2 loop type	4 loop type
Analog input	Number of inputs	2 to 4	4 to 8	2	4
	Input type	Multi-range of T/C, RTD, linear voltage and linear current		Multi-range of RTD (for 4-wire model) and linear voltage	
	Input accuracy	See Table 1.		See Table 2.	
	Input sampling time	50ms min.			
	Input bias current	T/C input: Range No.2,7,8,9,10,11 ±0.20μA max. Other ranges ±0.70μA max. Linear voltage input: Range No.40,41 ±0.20μA max. Range No.38,39 ±2.0μA max. Other ranges ±8.0μA max.		Linear voltage input: Range No.38,39 ±2.0μA max. Other ranges ±8.0μA max.	
	Input impedance	Linear current input: 10Ω±50% (under operating conditions)		—	
	RTD measuring current	1.04mA±0.2mA (flow-out from a-terminal for 3 wire type, and d-terminal for 4 wire type)		2.08mA±0.2mA (flow-out from d-terminal)	
	Allowable parallel resistance	T/C input and linear voltage (excluding L04, L08, L05, L07,V01) 1MΩ Min. Linear voltage L04,L08 3MΩ Min. Linear voltage L05, L07, V01 20MΩ Min.		Linear voltage (excluding L04, L08, L05, L07, V01) 1MΩ Min. Linear voltage L04, L08 3MΩ Min. Linear voltage L05, L07, V01 20MΩ Min.	
	Input voltage range	-2V to +12V (linear voltage, T/C)			
	Burnout	T/C and linear voltage: Upscale, downscale and none selection RTD: Yes or no selection except 3 wire type Linear current: 4-20mAdc input Downscale only 0-20mAdc input Near to 0%FS		Linear voltage: Upscale, downscale and none selection RTD: Yes or no selection	
	Overtime	110%FS or more: upscaled -10%FS or less: downscaled			
	Cold junction compensation accuracy	±0.7°C (under standard conditions)		—	
	Cold junction compensation method	Internal or external compensation (at 0°C) selectable		—	
	Scaling	-999999 to +999999 (These settings are available for linear inputs only. Reverse scaling can be performed.)			
	Ambient temperature effect on cold junction compensation	Max ±0.5°C over 0 to 50°C range except for the standard condition (23±2°C)		—	
Effect of wiring resistance	RTD input: 0.01%FS/Ω max. (within 0 to 10Ω wiring resistance range) Allowable wiring resistance: 10Ω max.		RTD input: 0.01°C/Ω max. (within 0 to 1Ω wiring resistance range) Allowable wiring resistance: 1Ω max.		
Long-term stability (reference value)	<ul style="list-style-type: none"> In the first half year, less than 3 times ambient temperature effect (1°C) After that, for every half a year, less than ambient temperature effect (1°C) Note: The above shows reference values under standard conditions.				
Analog output	Number of outputs	2	4	2	4
	Output type	Linear output (0-20mAdc, 4-20mAdc, 0-10Vdc) and time proportional current output selectable			
	Output accuracy	0 to 20mAdc: ±0.08%FS (accuracy ±0.5%FS for output 1.0% or less) 4 to 20mAdc: ±0.10%FS 0 to 10Vdc: ±0.10%FS (accuracy ±0.5%FS for output 1.0% or less) at load resistance more than 100kΩ.			
	Output resolution	30,000 min. for 0 to 20mAdc, 30,000min. for 0 to 10Vdc			
	Output update time	Linear output <ul style="list-style-type: none"> While in operating mode: Execution cycle time While in application inactive mode: 100ms Time proportional current output: <ul style="list-style-type: none"> Time proportioning cycle (1 to 120s, 1s unit). When MV value changes in this cycle, output change can be performed without waiting for it to update. Minimum ON-OFF time is 1s. 			
	Maximum load resistance	Current output, time proportional current output: 500Ω			
	Max. output current	Linear current output: 25mAdc; time proportional current, linear voltage output: 40mAdc			
	Output response time	Linear output: 50ms (90% response time)			
	Open terminal voltage	20V max.			
	OFF leakage current	Time proportional current output: 100μA max. (at load shorted, under standard conditions)			
Auxiliary input	Number of inputs	1	2	1	2
	Input type	0-5Vac / 0-6Vac / 0-10Vac / 0-12Vac / 1-5Vdc selectable by setting			
	Input accuracy	0-5Vac: ±3.0%FS 0-12Vac: ±1.25%FS 0-6Vac: ±2.5%FS 1-5Vdc: ±0.2%FS 0-10Vac: ±1.5%FS			

Model No.		DMC50CH20□	DMC50CH40□	DMC50CS20□	DMC50CS40□
Auxiliary input	Input sampling time	100ms			
	Input impedance	AC: 100kΩ min. (under operating conditions) DC: 1MΩ min. (under operating conditions)			
	Input voltage range	AC: 14.4Vac max. DC: -1 to +6Vdc max.			
	Burnout	AC: Equivalent to 0Vac input DC: Upscale			
Digital input	Number of inputs	6	12	6	12
	Isolation	Photo-coupler (bidirectional)			
	Input voltage	24Vdc±10%			
	Connectable output types	No voltage contact (relay contact) and open collector			
	Min. ON voltage	15.0V/0.9mA min.			
	Max. OFF voltage	7.0V max.			
	Max. OFF current	0.4mA max.			
	Current limit resistance	15kΩ ±5%			
	Sampling time	Variable by operation of ISaGRAF During operating mode: Execution cycle time During application inactive mode: 100ms			
	Min. detectable pulse width	Variable by operation of ISaGRAF During operating mode: More than execution cycle time During application inactive mode: 100ms			
Terminal	6 points common (possible for both source/sink side common)				
Digital output	Number of outputs	8	16	8	16
	Output type	Open drain type FET output			
	External supply voltage	10.8 to 26.4Vdc			
	Max. output current	70mA/point, 400mA max. per 8 points			
	OFF leakage current	100μA max.			
	ON residual voltage	1.5V max.			
	Over-current protection	None			

Model No.		DMC50CH□00	DMC50CS□00	DMC50CH□01	DMC50CS□01
Back plane communication	Connection	Multilink connector		Multilink connector + Extension terminal	
	Transmission line type	Bus type (Max. 8 control modules for one communication module) RS-485 conformed 3 wire type			
	Mode	Half-duplex			
	Max. cable length	20m			

Table 1 High resolution model (DMC50CH*)
input type • accuracy**

No.	Range symbol	Range	Indication accuracy $\pm^{\circ}\text{C}$ ($\pm\%$ FS)
1	K: CA	-200.00 to +1200.00 $^{\circ}\text{C}$	0.7 (0.05) *1
2	K: CA	-200.00 to +400.00 $^{\circ}\text{C}$	0.3 (0.05) *1
3	E: CRC	0.00 to 800.00 $^{\circ}\text{C}$	0.4 (0.05)
4	J: IC	0.00 to 800.00 $^{\circ}\text{C}$	0.4 (0.05)
5	N: NiCr-Ni	0.00 to 1300.00 $^{\circ}\text{C}$	0.65 (0.05)
6	PLII	0.00 to 1300.00 $^{\circ}\text{C}$	0.65 (0.05)
7	T: CC	-200.00 to +300.00 $^{\circ}\text{C}$	0.25 (0.05)
8	B: PR30-6	0.00 to 1800.00 $^{\circ}\text{C}$	1.8 (0.1) *2
9	R: PR13	0.00 to 1600.00 $^{\circ}\text{C}$	1.2 (0.075) *3
10	S: RP10	0.00 to 1600.00 $^{\circ}\text{C}$	1.2 (0.075)
11	PR40-20	0.00 to 1900.00 $^{\circ}\text{C}$	4.75 (0.25) *4
12	WRe5-26	0.00 to 2300.00 $^{\circ}\text{C}$	1.25 (0.05)
13	WRe5-26	0.00 to 1400.00 $^{\circ}\text{C}$	1.25 (0.09)
14	DIN L	-200.00 to +800.00 $^{\circ}\text{C}$	0.5 (0.05) *5
15	DIN U	-200.00 to +400.00 $^{\circ}\text{C}$	0.3 (0.05) *6
16	Ni-NiMo	0.00 to +1300.00 $^{\circ}\text{C}$	1.3 (0.1)
21	Pt100	-200.00 to +500.00 $^{\circ}\text{C}$	0.35 (0.05)
22	Pt100	-60.00 to +100.00 $^{\circ}\text{C}$	0.15 (0.063)
31		0 to 20mA	(0.04)
32		4 to 20mA	(0.05)
33		0 to 10V	(0.04)
34		0 to 5V	(0.08)
35		1 to 5V	(0.10)
36		-1 to +1V	(0.05)
37		0 to 1V	(0.10)
38		-100 to +100mV	(0.05)
39		0 to 100mV	(0.10)
40		-10 to +10mV	(0.07)
41		0 to 10mV	(0.14)

*1: K and T T/C for less than -100°C : $\pm 1.5^{\circ}\text{C}$ for range K29, $\pm 0.6^{\circ}\text{C}$ for K24, $\pm 0.5^{\circ}\text{C}$ for range T44

*2: B T/C: $\pm 72.0^{\circ}\text{C}$ for less than 260°C , $\pm 3.6^{\circ}\text{C}$ for 260 to 800°C

*3: R and S T/C: $\pm 1.6^{\circ}\text{C}$ for less than 100°C

*4: PR40-20 T/C: $\pm 23.7^{\circ}\text{C}$ for less than 300°C , $\pm 14.2^{\circ}\text{C}$ for 300 to 800°C

*5: DIN standard L T/C: $\pm 0.75^{\circ}\text{C}$ for less than -100°C

*6: DIN standard U T/C: $\pm 1.0^{\circ}\text{C}$ for less than -100°C , $\pm 0.5^{\circ}\text{C}$ for -100 to 0°C

Table 2 Special model (DMC50CS*)
input type • accuracy**

No.	Range symbol	Range	Indication accuracy $\pm^{\circ}\text{C}$ ($\pm\%$ FS)
23	Pt100	16.000 to 37.000 $^{\circ}\text{C}$	0.10
24	Pt100	-50.000 to +150.000 $^{\circ}\text{C}$	0.15
33		0 to 10V	(0.04)
34		0 to 5V	(0.08)
35		1 to 5V	(0.10)
36		-1 to +1V	(0.05)
37		0 to 1V	(0.10)
38		-100 to +100mV	(0.05)
39		0 to 100mV	(0.10)

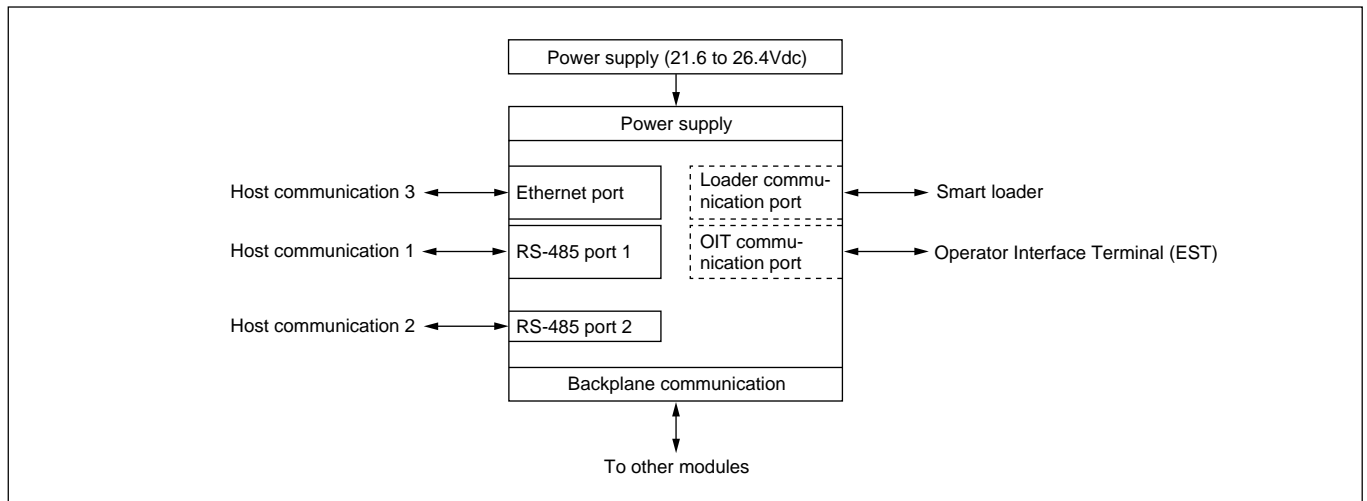
Communication Module Individual Specifications

There are two types of communication modules for the ME model (Ethernet/ RS-485) and the MR model (RS-485). A single communication module can be connected to a maximum of 8 control modules and electricity is supplied to each control module by the backplane.

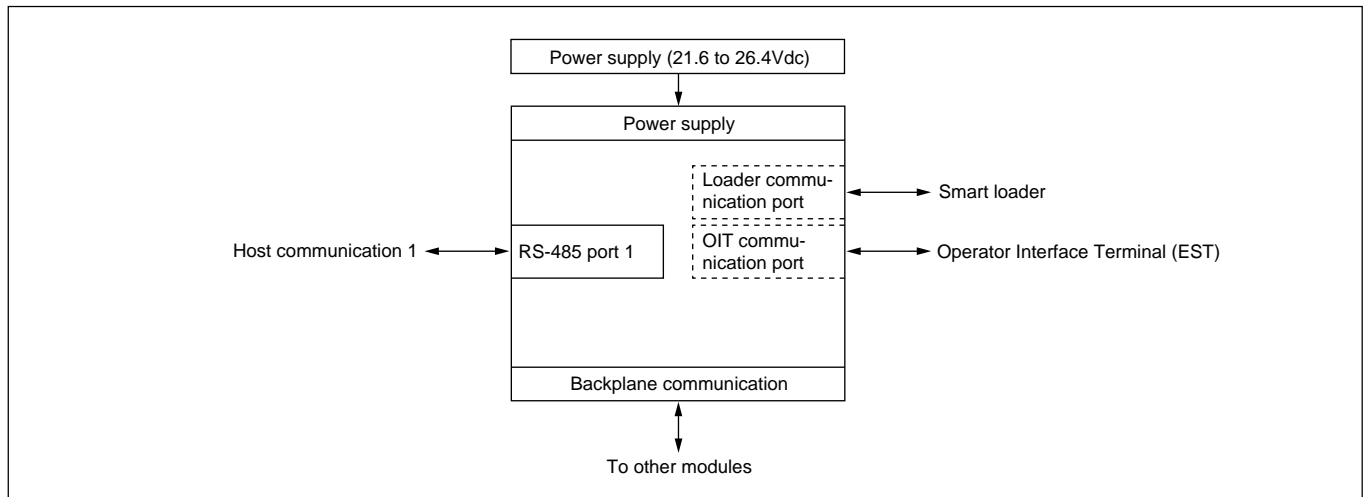


Input/output configuration

• DMC50ME20 □ model



• DMC50MR20 □ model



Individual specifications

Model No.	DMC50ME200	DMC50ME201	DMC50MR200	DMC50MR201
Number of host communication connections	3		1	
Host communication connection system	Host communication 1: RS-485 Host communication 2: RS-485 Host communication 3: Ethernet		Host communication 1: RS-485	
Host communication priority system	Basically, priority is given to the newest input.		—	
Number of digital output	4 points		2 points	
Backplane communication	Connection by a multilink connector × 2	Connection by a multilink connector + extension terminal	Connection by a multilink connector × 2	Connection by a multilink connector + extension terminal

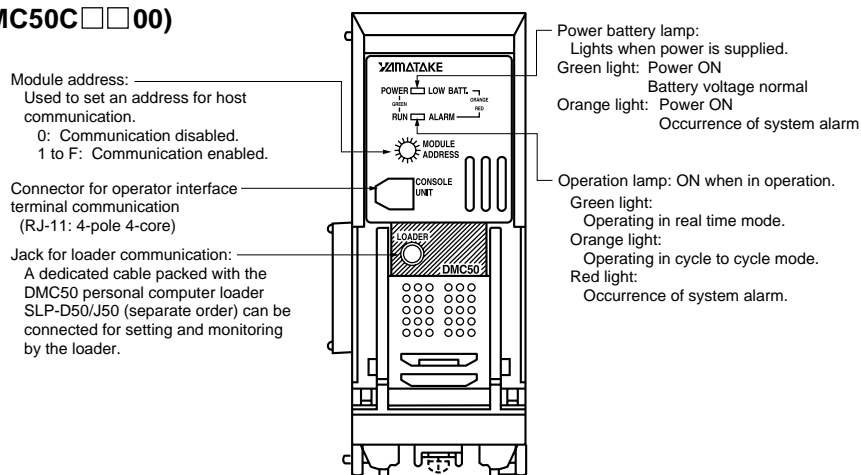
● Communication specifications

Model No.	Host communications 1,2	Host communication 3	Backplane communication
Transmission line type	Conforms to RS-485, 5-wire/3-wire, bus type	Ethernet, 10BASE-T	Bus type (max. 8 control modules for one communication module)
Mode	Half-duplex	CSMA/CD	Half-duplex
Max. transmission distance	500m	100m/segment	20m
Terminating resistor	An 150Ω terminating resistor can be driven for both ends of each line.	—	—
Transmission speed	9600bps 19200bps 38400bps	10Mbps	—
Communication protocol	CPL	TCP/IP (carrier protocol) CPL (application protocol) Protocol dedicated to loader (application protocol)	—
Connection	Terminal	10BASE-T modular jack	Terminal
Others	Synchronization: Asynchronous communication Transmission speed deviation: Within 2.0% Bit length: 8 bits Stop bit length: 1 bit Parity bit: Even parity	Number of port: 8 Number of connection: Max. 8 lines Connector: RJ-45 connector	—

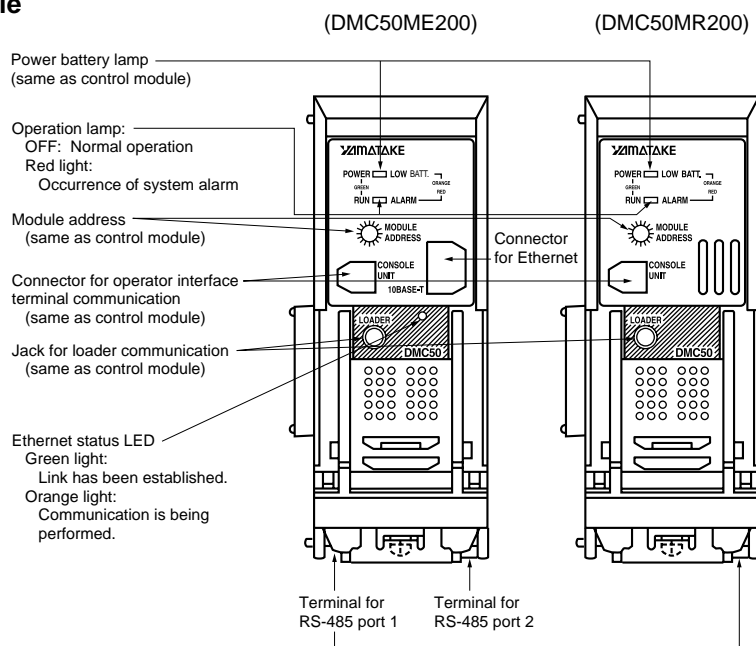
Name and function of each part

■ Body

● Control module (DMC50C□□00)



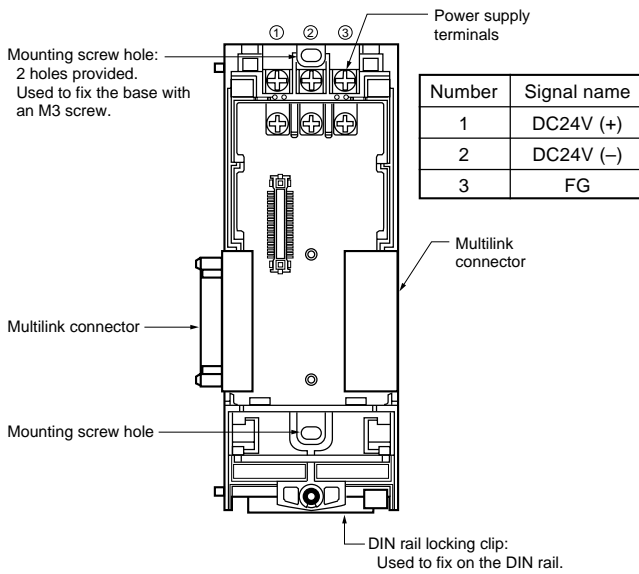
● Communication module



■ Base

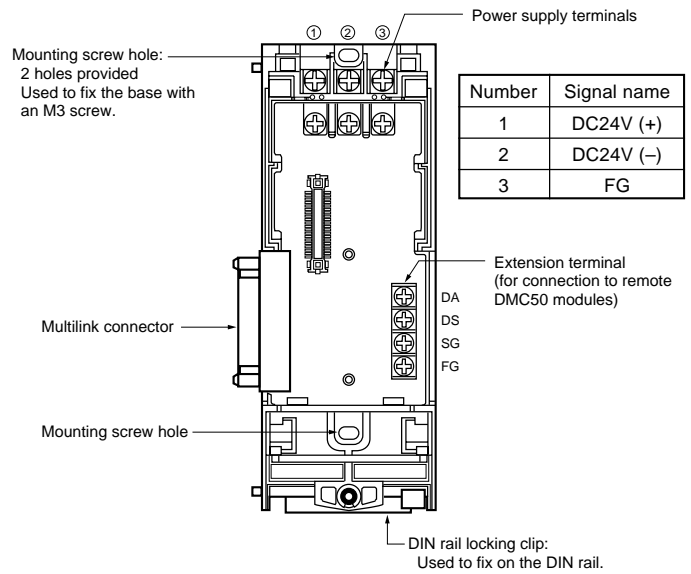
• Standard base

(Module model numbers: CH200, CH400, CS200, CS400, MR200, ME200)

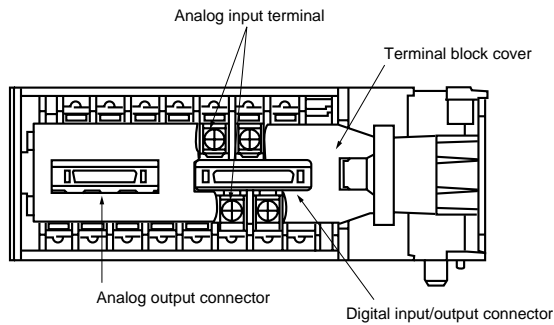


• Extension base

(Module model numbers: CH201, CH401, CS201, CS401, MR201, ME201)



■ Terminal Block



■ Module Connection

- In the case of the standard base, the DMC50 can be connected to other modules by using the multilink connectors on the right and the left sides of the base.
- In the case of the extension base, each linked module can be independently wired from the extension terminals on the right side of base unit. (The left base side can be connected with a standard base.)
- Connect the DMC50 modules together before mounting them onto the DIN rail or screw mounting.
- By linking the modules in this way, wiring can be minimized for the electrical and CPL communication connections between each module.

■ Model Selection Guide

● Control module

I Basic	II		III		IV				Description
	Module model No.		Additional treatment		Special treatment				
DMC50									Module type controller
		CH20							High resolution 2-loop type
		CH40							High resolution 4-loop type
		CS20							Special 2-loop type
		CS40							Special 4-loop type
		0							Standard base
		1							Extension base
			0	0					No additional treatment
			T	0					Tropicalization
			K	0					Antisulfide treatment
			D	0					With inspection certificate
			B	0					Tropicalization + inspection certificate
			L	0					Antisulfide treatment + inspection certificate
			0	Y					Traceability certificate available
			T	Y					Tropicalization + traceability certificate available
			K	Y					Antisulfide treatment + traceability certificate available
					0	0	0	0	(None)

● Communication module

I Basic	II		III		IV				Description
	Module model No.		Additional treatment		Special treatment				
DMC50									Module type controller
		MR20							RS-485 1 channel
		ME40							Ethernet + RS-485 2 channels
		0							Standard base
		1							Extension base
			0	0					No additional treatment
			T	0					Tropicalization
			K	0					Antisulfide treatment
			D	0					With inspection certificate
			B	0					Tropicalization + inspection certificate
			L	0					Antisulfide treatment + inspection certificate
			0	Y					Traceability certificate available
			T	Y					Tropicalization + traceability certificate available
			K	Y					Antisulfide treatment + traceability certificate available
					0	0	0	0	(None)

● Personal computer software

I Basic model No.	II		Description
	Others		
SLP-D50			Personal computer loader for DMC50 (CD-ROM)
		E50	English version, with a dedicated cable

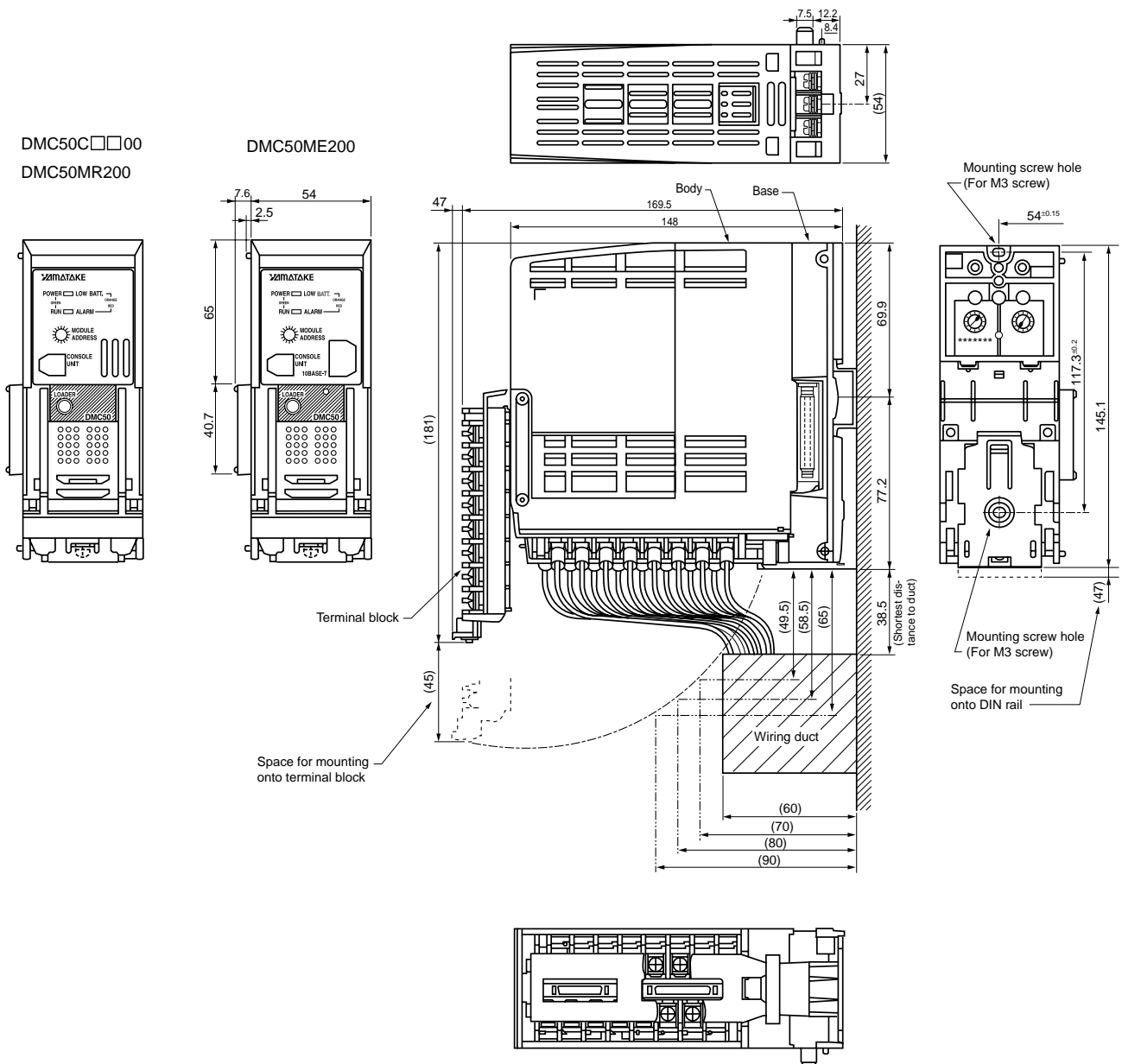
■ Applicable Connector Type

Analog output connector (CN1)	TX20A-26PH1-D2P1-D1 (made by Japan Aviation Electronics Industry, Ltd.)
Digital input/output connector (CN2)	TX20A-36PH1-D2P1-D1 (made by Japan Aviation Electronics Industry, Ltd.)

■ Dimensions

● DMC50C□□00, DMC50M□□200

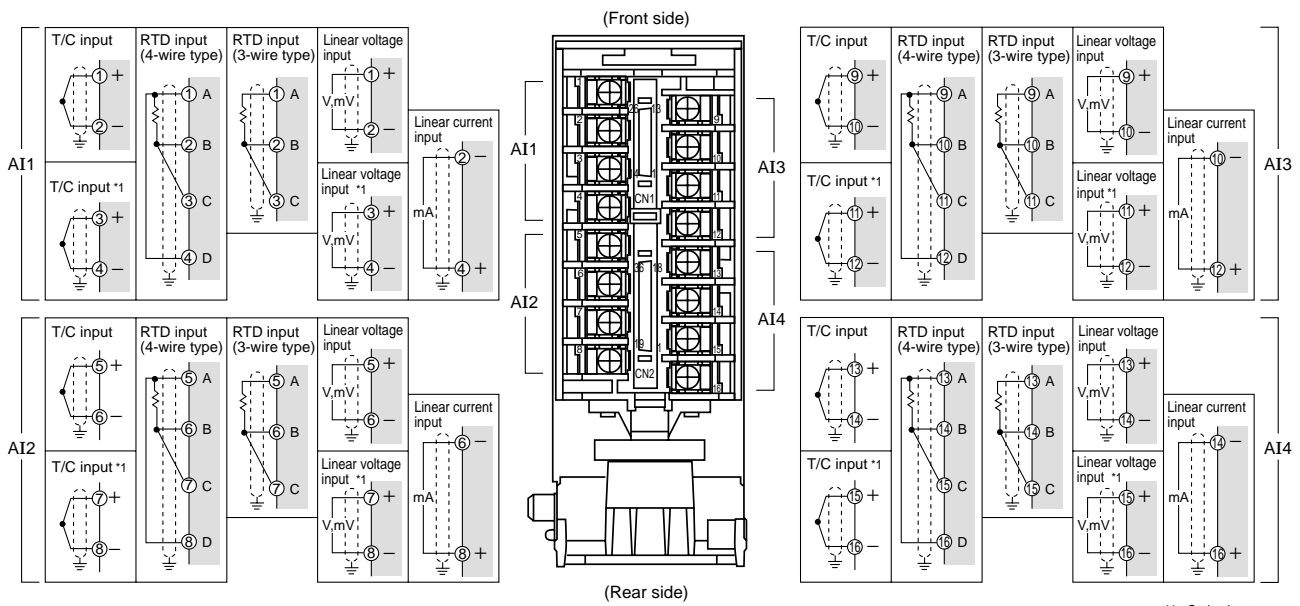
(Unit: mm)



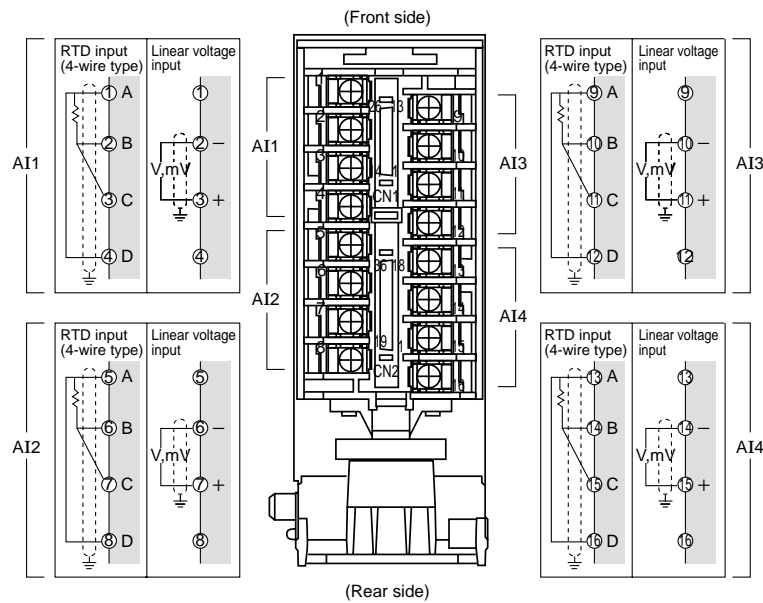
■ Control module terminal layout

● Analog input terminal

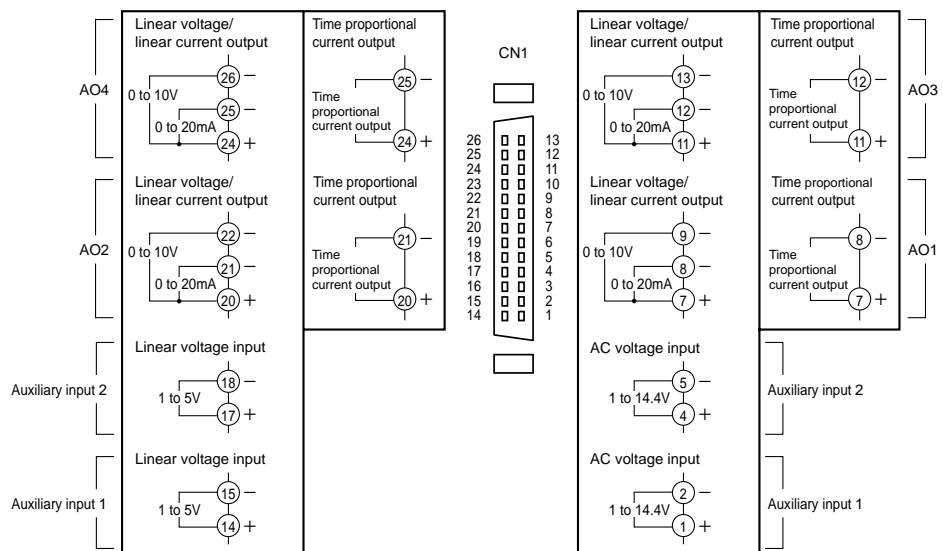
DMC50CH□00 (high resolution model)



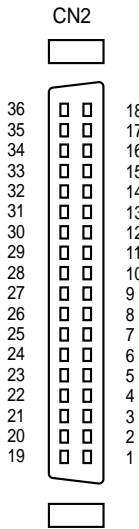
DMC50CS□00 (Special model)



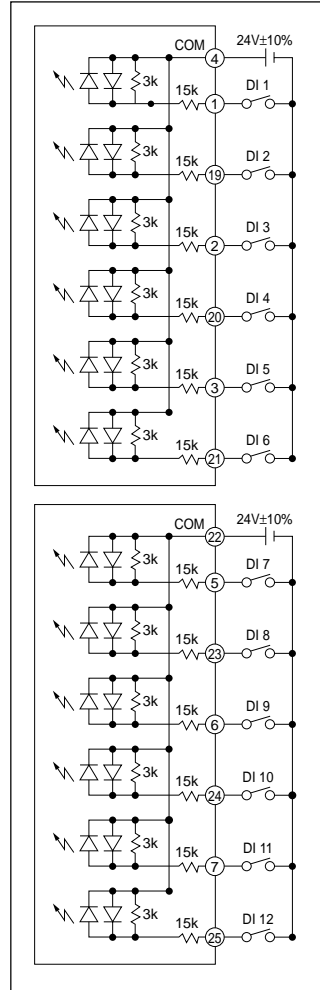
● Connector 1 (CN1)



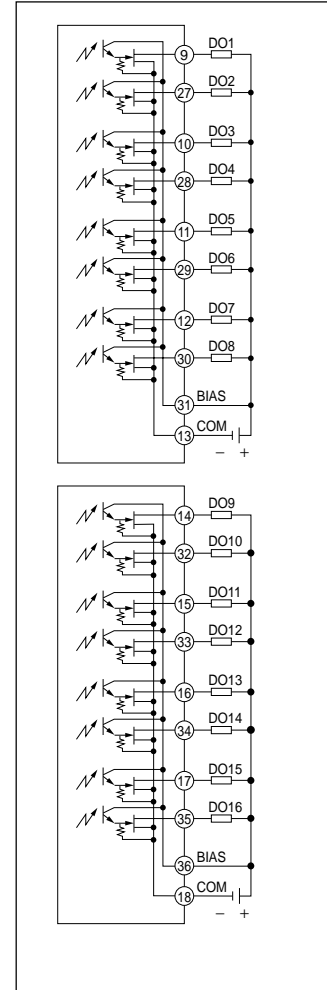
Connector 2 (CN2)



Digital input terminal wiring diagram

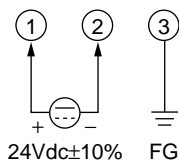


Digital output terminal wiring diagram



Power supply connection

Connect the power supply as follows:



Handling Precautions

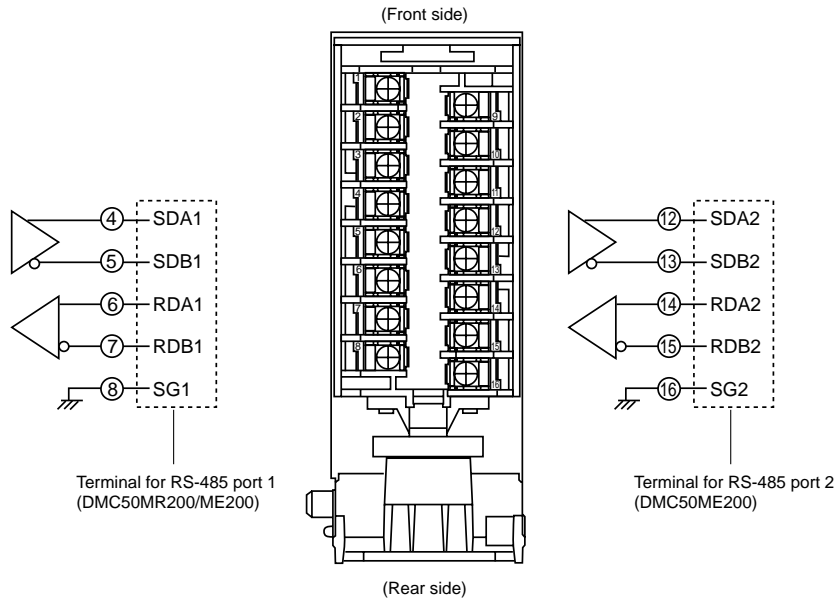
- When selecting a switching power supply unit, ensure that it generates minimal switching noise.
- The power supply lines and grounding wires to the controller should not be bundled with the power cable or should not be run in the same wiring conduit or duct.
- The power supply is mutually connected between the coupled modules. Supply the power to only one of the coupled modules.
- There should be sufficient power supply to provide for the total power consumption of the coupled modules.

Wiring precautions

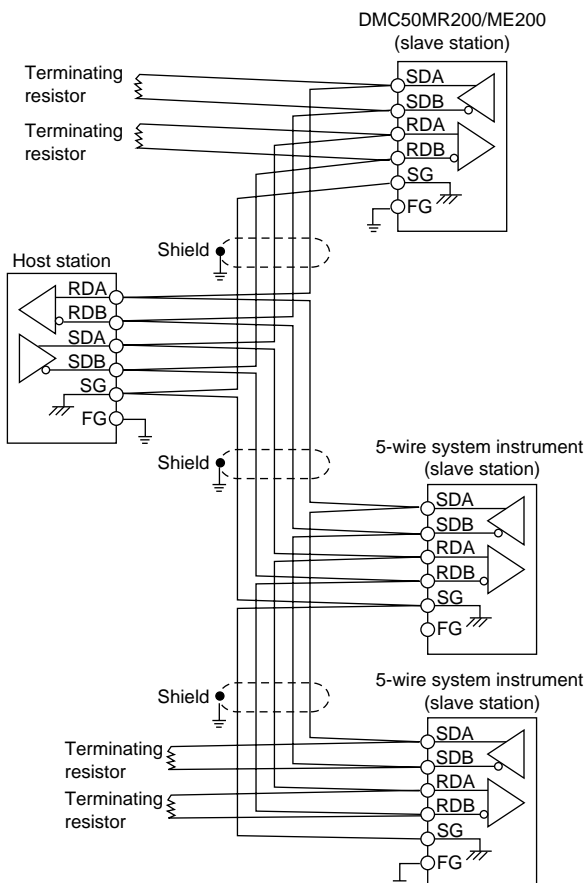
- Perform wiring only after confirming the controller model number and terminal number found on the label on the side of the main body.
- Make sure to use crimp terminals that conform to 3.5 mm screws for terminal connection.
- Do not allow the sides of the crimp terminals to come into contact with each other.
- Make sure to separate the input/output signal wires by at least 50 cm from the power cable and the power supply line. Also, do not run the signal wires in the same conduit or duct as the other cables.
- If the controller is connected in parallel with other instruments, make sure to check those instruments' specifications and conditions of use before instrumentation.
- It takes about 10 seconds before the controller stabilizes and begins to function after turning on the power. After stabilizing, the controller will be ready to perform its functions. However, it requires to warm up in order to attain its specified accuracy. Please allow a warm-up time of at least 2 hours.
- Before turning the power on after completing wiring, check and verify that the wiring as been properly done.

Communication module terminal layout

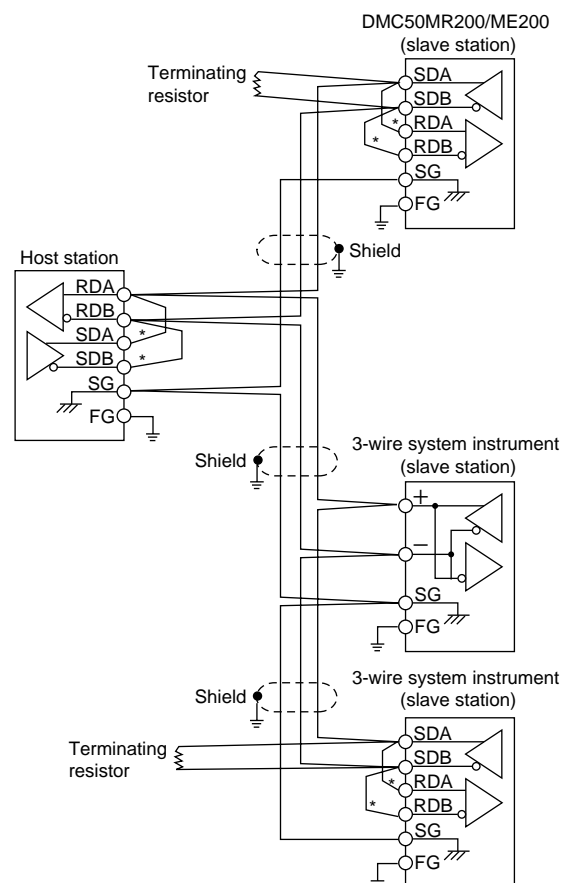
● RS-485 terminal layout



● Connection with 5-wire system instrument



● Connection used with 3-wire system instrument



- Connect two terminating resistors of $150\Omega \pm 5\%$ 1/2W min. to the instrument on each end of the transmission line.
- Ground the shield FGs at only one end to a single location, not at both ends.
- Use a shielded twisted-pair cable as the RS-485 communication line.

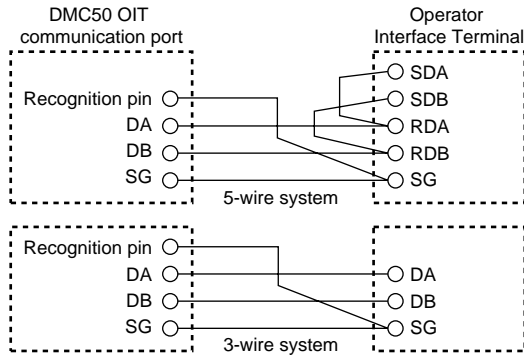
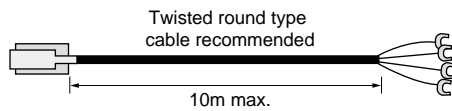
- Connect two terminating resistors of $150\Omega \pm 5\%$ 1/2W min. to the instrument on each end of the transmission line.
- Ground the shield FGs at only one end to a single location, not at both ends.
- The areas marked with an asterisk (*) are to be connected externally.
- Use a shielded twisted-pair cable as the RS-485 communication line.

● Communication connections for Operator Interface Terminal

Communication connector for the DMC50 Operator Interface Terminal

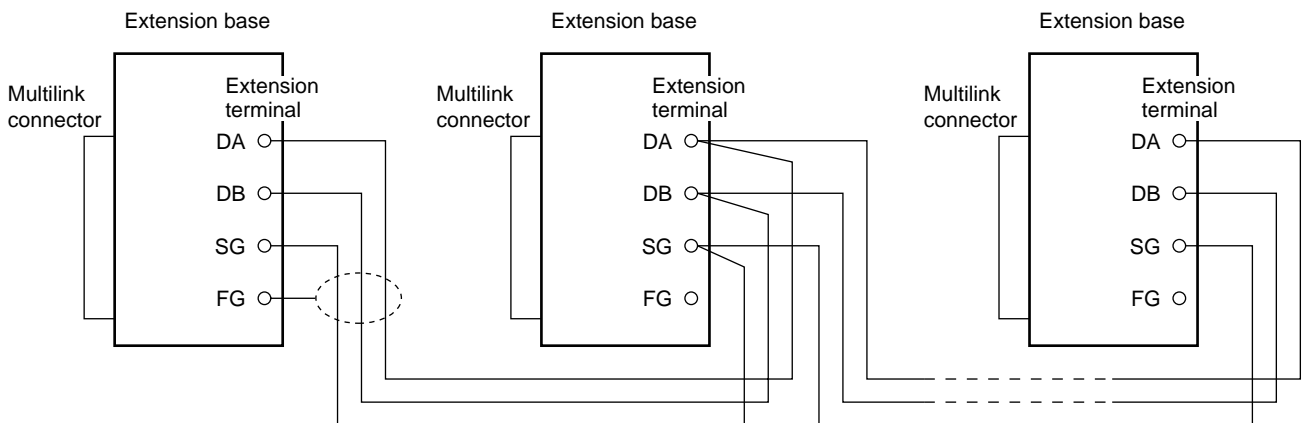


Plug for 4-pin modular-jack (RJ-11: 4 poles 4 cores)



- A communication connector for an Operator Interface Terminal and a communication jack for a loader cannot be connected at the same time.
 - Disconnect the communication cable for the loader when communication for the Operator Interface Terminal is to be used.
 - Disconnect the communication cable for the Operator Interface Terminal when loader communication is to be used

● Connection of backplane communication



- Ground the shield FGs at only one end to a single location, not at both ends.
- Use a shielded twisted-pair cable as the RS-485 communication line.

Program Language/Instructional Terms

■ Quick LD (ladder circuit)

● Contact

Symbol	Name and function
	Direct contact
	Negated contact
	Contact with positive (rising) edge detection
	Contact with negative (falling) edge detection

● Language construct

Symbol	Name and function
	Jump Jumps to a designated label name when conditions are met.
	Return Does not execute the programs lower than the return symbol when conditions are met.

■ FBD (Function block diagram)

Standard instruction

• Data operation

1 gain	Assignment
NEG	Sign change

• Boolean operation

& (AND)	Boolean AND
> = 1 (OR)	Boolean OR
= 1 (XOR)	Boolean exclusive OR

• Mathematical operation

+	Addition
-	Subtraction
*	Multiplication
/	Division

• Bitwise boolean operation

AND_MASK	Bitwise AND
OR_MASK	Bitwise OR
XOR_MASK	Bitwise XOR
NOT_MASK	Bitwise NOT

● Coil

Symbol	Name and function
	Direct coil
	Negated coil
	Set action coil
	Reset action coil
	Coil with positive (rising) edge detection
	Coil with negative (falling) edge detection

• Comparison test

<	Less than
< =	Less or equal
>	Greater than
> =	Greater or equal
=	Equal
< >	Not equal

• Data conversion

BOO	Convert to boolean
ANA	Convert to integer
ANA_DP	Real to integer conversion with decimal point position designation
REAL	Convert to real
TMR	Convert to timer
MSG	Convert to message string

• Others

CAT	Message string concatenation
SYSTEM	Access to system

Function

• Mathematical functions

ABS	Absolute value
EXPT	Exponential function
LOG	Common logarithm
POW	Exponential function
SQRT	Square root
TRUNC	Truncate the decimal part

• Trigonometric function

ACOS	Arc cosine
ASIN	Arc sine
ATAN	Arc tangent
COS	Cosine
SIN	Sine
TAN	Tangent

• Register control

ROL	Left rotation
ROR	Right rotation
SHL	Left shift
SHR	Right shift

• Data operation

MIN	Minimum value
MAX	Maximum value
LIMIT	High/low limit
LIMIT_HI	Real type high limit
LIMIT_LO	Real type low limit
LIMIT_HILO	Real type high/low limit
MOD	Modulus (division remainder)
MUX4	4-input multiplexer
MUX8	8-input multiplexer
MUX8REAL	Real type 8-input multiplexer
ODD	Odd parity
RAND	Random value
SEL	Binary selection
SEL_BOOL	Boolean type binary selection
SEL_REAL	Real type binary selection
SEL_TMR	Timer type binary selection

• Data conversion

ASCII	Character → ASCII code conversion
CHAR	ASCII code → character conversion
BIN3DEC	3-input boolean to integer conversion
BIN8DEC	8-input boolean to integer conversion
SCAL_CNV	Scale conversion

• Message string operation

DELETE	Message string deletion
INSERT	Message string insertion
FIND	Message string finding
MLEN	Message string length
LEFT	Left message string extraction
MID	Message string extraction
REPLACE	Message string replacement
RIGHT	Right message string extraction

Function block

• Boolean operation

SR	Set dominant bistable
RS	Reset dominant bistable
R_TRIG	Rising edge detection
F_TRIG	Falling edge detection

• Counter

CTU	Up counter
CTD	Down counter
CTUD	Up/down counter

• Timer

TON	ON delay timer
TOF	OFF delay timer
TP	Pulse timer

• Integer analog

CMP	Complete comparison
-----	---------------------

• Real analog

AVERAGE	Moving average
MAV	Moving average
HYSTER	Hysteresis
LIM_ALARM	Limit alarm
INTEGRAL	Integral
DERIVATE	Derivative
DED	Dead time
LEAD_LAG	Lead/lag

• Signal generation

BLINK	Boolean type signal blinking
PLS_GEN	Pulse generator
PAMP_GEN	Ramp generator
SIG_GEN	Signal generator

Special function block

• Parameter access

PAR_BOOL	Read boolean type parameter
PAR_INT	Read integer type parameter
PAR_REAL	Read real type parameter
PAW_BOOL	Write boolean type parameter
PAW_INT	Write integer type parameter
PAW_REAL	Write real type parameter

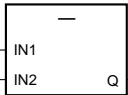


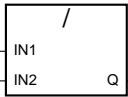

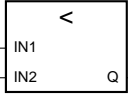
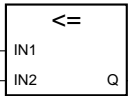
• Control operation

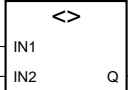
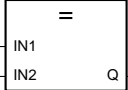
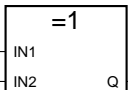
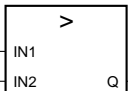
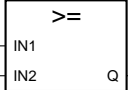
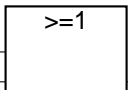
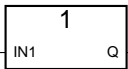
PID_A	Standard PID operation
PID_CAS	Cascade PID operation
Ra_PID	Rationa Loop PID operation
UP_PID	Use-point PID operation

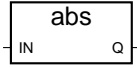
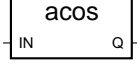
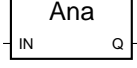
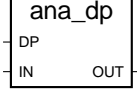
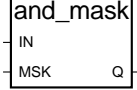

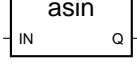
• Others

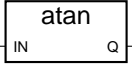


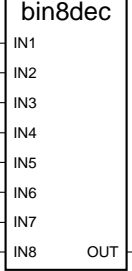
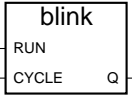
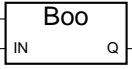
TBL	Linearization table lookup
TBR	Linearization table reverse lookup
ZONE7	Zone selector
PSVC	Power supply voltage compensation

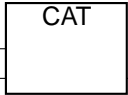
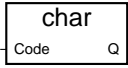
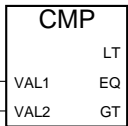
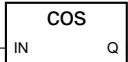
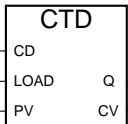
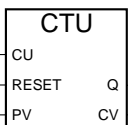
● Standard instruction, function, function block and special function block

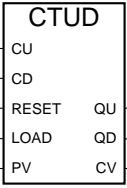
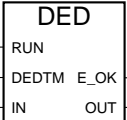



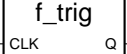
<p align="center">- (Subtraction)</p> 	<p>Input parameter IN1: Integer type real type IN2: Integer type real type, Same data type as IN1</p> <p>Output parameter Q: Integer type real type, $IN1 - IN2$</p> <p>Description 1 piece of data is subtracted from another.</p>
<p align="center">& (AND)</p> 	<p>Input parameter INPUTn: Boolean type, n is 2 to 32.</p> <p>Output parameter OUTPUT: Boolean type, AND of input</p> <p>Description AND of 2 or more pieces of Boolean type data</p>
<p align="center">* (Multiplication)</p> 	<p>Input parameter INPUTn: Integer type real type, n is 2 to 32, All data is of the same data type.</p> <p>Output parameter OUTPUT: Integer type real type, Multiplication of inputs</p> <p>Description 2 pieces of data are multiplied by each other.</p>
<p align="center">/ (Division)</p> 	<p>Input parameter IN1: Integer type real type IN2: Integer type real type, Same data type as IN1</p> <p>Output parameter Q: Integer type real type, $IN1 / IN2$</p> <p>Description 1 piece of data is divided by another.</p>
<p align="center">+ (Addition)</p> 	<p>Input parameter INPUTn: Integer type real type, n is 2 to 32, All data is of the same data type.</p> <p>Output parameter OUTPUT: Integer type real type, Addition of inputs</p> <p>Description 2 pieces of data are added together.</p>
<p align="center">< (Less than)</p> 	<p>Input parameter IN1: Integer type real type timer type message type IN2: Integer type real type timer type message type, Sama data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at $IN1 < IN2$ FALSE at $IN1 \geq IN2$</p> <p>Description 2 pieces of data are compared.</p>
<p align="center"><= (Less or equal)</p> 	<p>Input parameter IN1: Integer type real type message type IN2: Integer type real type message type, Same data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at $IN1 \leq IN2$ FALSE at $IN1 > IN2$</p> <p>Description 2 pieces of data are compared.</p>


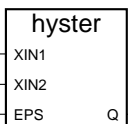
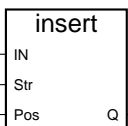
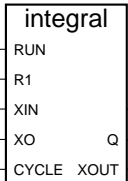
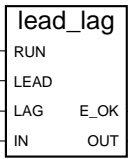
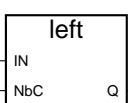
<p style="text-align: center;"><> (Not equal)</p> 	<p>Input parameter IN1: Integer type real type message type IN2: Integer type real type message type, Same data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at IN1 ≠ IN2 FALSE at IN1 = IN2</p> <p>Description 2 pieces of data are compared.</p>
<p style="text-align: center;">= (Equal)</p> 	<p>Input parameter IN1: Integer type real type message type IN2: Integer type real type message type, Same data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at IN1 = IN2 FALSE at IN1 ≠ IN2</p> <p>Description 2 pieces of data are compared.</p>
<p style="text-align: center;">=1 (XOR) (Exclusive OR)</p> 	<p>Input parameter IN1: Boolean type IN2: Boolean type</p> <p>Output parameter Q: Boolean type, Exclusive-OR of inputs</p> <p>Description Exclusive-OR of 2 pieces of Boolean type data</p>
<p style="text-align: center;">> (Greater than)</p> 	<p>Input parameter IN1: Integer type real type timer type message type IN2: Integer type real type timer type message type, Sama data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at IN1 > IN2 FALSE at IN1 ≤ IN2</p> <p>Description 2 pieces of data are compared.</p>
<p style="text-align: center;">>= (Greater or equal)</p> 	<p>Input parameter IN1: Integer type real type message type IN2: Integer type real type message type, Same data type as IN1</p> <p>Output parameter Q: Boolean type, TRUE at IN1 ≥ IN2 FALSE at IN1 < IN2</p> <p>Description 2 pieces of data are compared.</p>
<p style="text-align: center;">>=1 (OR)</p> 	<p>Input parameter INPUTn: Boolean type, n is 2 to 32.</p> <p>Output parameter Q: Boolean type, OR of inputs</p> <p>Description OR of 2 or more pieces of Boolean type data</p>
<p style="text-align: center;">1 gain (Assignment)</p> 	<p>Input parameter IN: Any types of data</p> <p>Output parameter Q: Any types of data</p> <p>Description Data assignment</p>

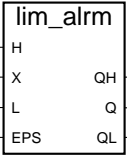
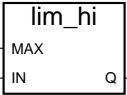

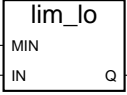

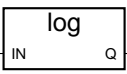
<p>ABS (Absolute value)</p> 	<p>Input parameter IN: Real type</p> <p>Output parameter Q: Real type, Absolute value of IN</p> <p>Description Absolute value of real type data</p>
<p>ACOS (Arc cosine)</p> 	<p>Input parameter IN: Real type, -1.0 to $+1.0$</p> <p>Output parameter Q: Real type, 0.0 to π (radian unit), $\text{acos}(\text{IN})$</p> <p>Description Arc cosine of real type data</p>
<p>ANA (Convert to integer)</p> 	<p>Input parameter IN: Those excluding integer type data</p> <p>Output parameter Q: Integer type, For a boolean type IN, 0 at IN=FALSE and 1 at IN=TRUE. For a real type IN, it is the integer type portion of IN. (Digits below the decimal point are ignored) For a timer type IN, it is a ms numerical value. For a message type IN, its message string is represented by a decimal number.</p> <p>Description Data is converted to integer type data.</p>
<p>ANA_DP (Real to integer conversion with decimal point position designation)</p> 	<p>Input parameter DP: Integer type, Designation of number of digits below the decimal point, -30 to $+30$ IN: Real type</p> <p>Output parameter Q: Integer type, $\text{IN} \times 10^{\text{DP}}$</p> <p>Description Real data is multiplied by 10^{DP} and then rounded off to an integer.</p>
<p>AND_MASK (Bitwise AND)</p> 	<p>Input parameter IN: Integer type MASK: Integer type</p> <p>Output parameter Q: Integer type, bitwise AND of IN and MASK</p> <p>Description Bitwise AND of 2 pieces of integer type data</p>
<p>ASCII (Character→ASCII code conversion)</p> 	<p>Input parameter IN: Message type, Message string other than NULL POS: Integer type, Designated location of a character in a message string, 1 to Len (Len is the IN character string length.)</p> <p>Output parameter CODE: Integer type, ASCII code of a designated character, 0 to 255</p> <p>Description The designated character in a message string is converted to the corresponding ASCII code.</p>
<p>ASIN (Arc sine)</p> 	<p>Input parameter IN: Real type, -1.0 to $+1.0$</p> <p>Output parameter Q: Real type, $-\pi/2$ to $+\pi/2$ (radian unit) $\text{asin}(\text{IN})$</p> <p>Description Arc sine of real type data</p>

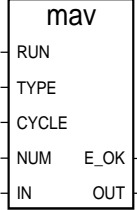
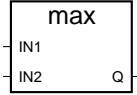
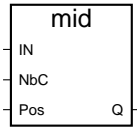

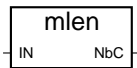
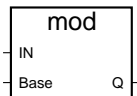
<p>ATAN (Arc tangent)</p> 	<p>Input parameter IN: Real type, Range of real numbers</p> <p>Output parameter Q: Real type, $-\pi/2$ to $+\pi/2$ (radian unit) atan(IN)</p> <p>Description Arc tangent of real type data</p>
<p>AVERAGE (Moving average)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset XIN: Real type, Input data N: Integer type, Number of samples for calculating the mean value, 1 to 128</p> <p>Output parameter XOUT: Real type, Mean value of N units (number of samples) of XIN</p> <p>Description Mean value of real type data in the designated sample numbers</p>
<p>BIN3DEC (3-input boolean to integer conversion)</p> 	<p>Input parameter IN1 to IN3: Boolean type</p> <p>Output parameter OUT: Integer type, $IN1 \times 2^0 + IN2 \times 2^1 + IN3 \times 2^2$, 0 to 7</p> <p>Description Converts 3 pieces of Boolean type data to a 0 to 7 integer type data.</p>
<p>BIN8DEC (8-input boolean to integer conversion)</p> 	<p>Input parameter IN1 to IN8: Boolean type</p> <p>Output parameter OUT: Integer type, $IN1 \times 2^0 + IN2 \times 2^1 + IN3 \times 2^2 \dots + IN8 \times 2^7$, 0 to 255</p> <p>Description Converts 8 pieces of Boolean type data to a 0 to 255 integer type data.</p>
<p>BLINK (Boolean type signal blinking)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset CYCLE: Timer type, Blinking cycle, $CYCLE \geq$ Cycle time setting</p> <p>Output parameter Q: Boolean type, Blinking output, If RUN=TRUE, TRUE/FALSE is repeated at half the CYCLE time. If RUN=FALSE, FALSE.</p> <p>Description A blinking signal is generated at each designated cycle.</p>
<p>BOO (Convert to boolean)</p> 	<p>Input parameter IN: Those excluding Boolean type</p> <p>Output parameter Q: Boolean type, For an integer type IN, FALSE at IN=0 and TRUE at IN≠0 For a real type IN, FALSE at IN=0.0 and TRUE at IN≠0.0 For a timer type IN, FALSE at IN=T#0ms and TRUE at IN≠#0ms For a character string type IN, FALSE at IN≠'TRUE' and TRUE at IN='TRUE'</p> <p>Description Data is converted to Boolean type</p>

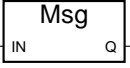
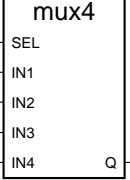
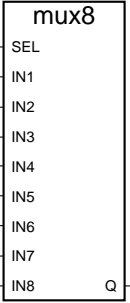
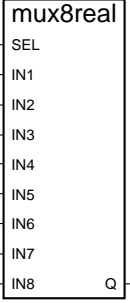
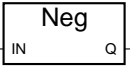
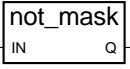
<p>CAT (Message string concatenation)</p>	<p>Input parameter INPn: Message type, Message string, n is 2 to 32.</p> <p>Output parameter OUTPUT: Message type, Concatenated message string, Message string 1 + message string 2 + ... + message string n</p> <p>Description Concatenation of multiple message strings</p>
	
<p>CHAR (ASCII code → character conversion)</p>	<p>Input parameter CODE: Integer type, ASCII code, 0 to 255</p> <p>Output parameter Q: Message type, Message string of 1 character</p> <p>Description The designated ASCII code is converted to the corresponding character.</p>
	
<p>CMP (Complete comparison)</p>	<p>Input parameter VAL1: Integer type VAL2: Integer type</p> <p>Output parameter LT: Boolean type TRUE at VAL1<VAL2 EQ: Boolean type TRUE at VAL1=VAL2 GT: Boolean type TRUE at VAL1>VAL2</p> <p>Description 2 pieces of integer type data are compared and the <, = and > results are output.</p>
	
<p>COS (Cosine)</p>	<p>Input parameter IN: Real type, Range of real numbers (radian unit)</p> <p>Output parameter Q: Real type, -1.0 to +1.0, cos(IN)</p> <p>Description Cosine of real type data</p>
	
<p>CTD (Down-counter)</p>	<p>Input parameter CD: Boolean type, Countdown execution at TRUE and stop at FALSE LOAD: Boolean type, Load command (dominant), CV=PV at TRUE PV: Integer type, Initial count value, Must be PV>0</p> <p>Output parameter Q: Boolean type, End signal, TRUE at CV=0 CV: Integer type, Counter result</p> <p>Description Countdown value of integer type data</p>
	
<p>CTU (Up-counter)</p>	<p>Input parameter CU: Boolean type, CountUP execution at TRUE and stop at FALSE RESET: Boolean type, Reset command (dominant), CV=0 at TRUE PV: Integer type, Counter target value, Must be PV>0</p> <p>Output parameter Q: Boolean type, End signal, TRUE at CV≥PV CV: Integer type, Counter result</p> <p>Description UPcount value of integer type data</p>
	

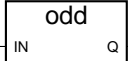

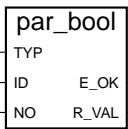
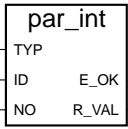
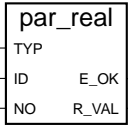
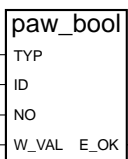
<p style="text-align: center;">CTUD (Up/down counter)</p> 	<p>Input parameter</p> <p>CU: Boolean type, CountUP execution (higher priority than CD) at TRUE and stop at FALSE CD: Boolean type, Countdown execution at TRUE and stop at FALSE RESET: Boolean type, Reset command (higher priority than LOAD, CU and CD), CV=0 at TRUE LOAD: Boolean type, Load command (higher priority than CU and CD), CV=PV at TRUE PV: Integer type, Counter target value, Must be PV>0</p> <p>Output parameter</p> <p>QU: Boolean type, UP-counter end signal, TRUE at CV≥PV QD: Boolean type, Down-counter end signal, TRUE at CV=0 CV: Integer type, Counter result</p> <p>Description</p> <p>Integration of CTU (up-counter) and CTD (down-counter)</p>
<p style="text-align: center;">DED (Dead time)</p> 	<p>Input parameter</p> <p>RUN: Boolean type, TRUE=Execution, FALSE=Reset DEDTM: Timer type, Dead time IN: Real type, Input data</p> <p>Output parameter</p> <p>E_OK: Boolean type, TRUE=Normal, FALSE=Error OUT: Real type, IN after dead time</p> <p>Description</p> <p>Outputting of real type data after dead time</p>
<p style="text-align: center;">DELETE (Message string deletion)</p> 	<p>Input parameter</p> <p>IN: Message type, Message string as base NbC: Integer type, Number of characters to delete, NBC>1 POS: Integer type, location of the delete begin position, POS>1</p> <p>Output parameter</p> <p>Q: Message type, Message string modified by deletion</p> <p>Description</p> <p>The designated characters are deleted from the message string at the designated location.</p>
<p style="text-align: center;">DERIVATE (Derivative)</p> 	<p>Input parameter</p> <p>RUN: Boolean type, TRUE=Execution, FALSE=Reset XIN: Real type, Input data CYCLE: Timer type, Sampling cycle, CYCLE≥cycle time setting</p> <p>Output parameter</p> <p>XOUT: Real type, Derivative result, The generation of output XOUT is proportional to the changing rate of input XIN.</p> <p>Description</p> <p>Derivative of real type data</p>
<p style="text-align: center;">EXPT (Exponential function)</p> 	<p>Input parameter</p> <p>IN: Real type, Base number EXP: Integer type, Exponent number</p> <p>Output parameter</p> <p>Q: Real type, IN^{EXP}</p> <p>Description</p> <p>Exponential operation</p>
<p style="text-align: center;">F_TRIG (Falling edge detection)</p> 	<p>Input parameter</p> <p>CLK: Boolean type</p> <p>Output parameter</p> <p>Q: Boolean type, TRUE when CLK is TRUE→FALSE, and FALSE when CLK is other than this.</p> <p>Description</p> <p>Falling edge detection of Boolean type data</p>

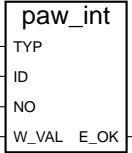
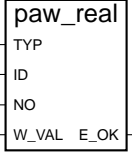
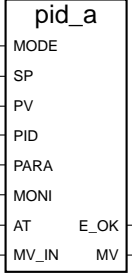
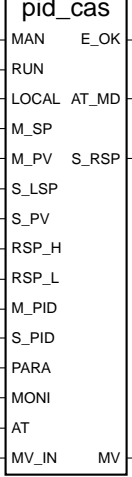
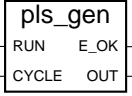
<p>FIND (Message string finding)</p> 	<p>Input parameter IN: Message type, Message string PAT: Message type, String pattern to find</p> <p>Output parameter POS: Integer type, Location of found string pattern within message string</p> <p>Description The designated message string is found and its location is output.</p>
<p>HYSTER (Hysteresis)</p> 	<p>Input parameter XIN1: Real type, Input data XIN2: Real type, High limit set point EPS: Real type, Hysteresis value, $EPS \geq 0.0$</p> <p>Output parameter Q: Boolean type</p> <p>Description Judgement with hysteresis on whether the real type data has exceeded the high limit value</p>
<p>INSERT (Message string insertion)</p> 	<p>Input parameter IN: Message type, Base message string STR: Message type, Message string to insert POS: Integer type, begin location for insertion of message string, $POS > 1$</p> <p>Output parameter Q: Message type, message string modified by insertion</p> <p>Description Insertion of the designated message string at the designated location of the base message string</p>
<p>INTEGRAL (Integral)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Integral execution, FALSE=Hold R1: Boolean type, Reset XIN: Real type, Input data X0: Real type, Initial value CYCLE: Timer type, Sampling cycle, $CYCLE \geq \text{Cycle time setting}$</p> <p>Output parameter Q: Boolean type, Inverting of R1 XOUT: Real type, Integral result</p> <p>Description Integral of real type data</p>
<p>LEAD_LAG (Lead/lag)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset LEAD: Real type, Lead time constant, $LEAD \geq 0.0$ (unit: s) LAG: Real type, Lag time constant, $LAG \geq 0.0$ (unit: s) IN: Real type, Input data</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error OUT: Real type, Calculated value, $OUT = OUT_{-1} + (A \times (IN - OUT_{-1}) + B \times (IN - IN_{-1}))$ A: $Ts / (Ts + LAG)$ B: $LEAD / (Ts + LAG)$ Ts: Cycle time setting of project (unit: s) OUT₋₁: Previous OUT ON₋₁: Previous IN</p> <p>Description Lead/lag calculation of real type data</p>
<p>LEFT (Left message string extraction)</p> 	<p>Input parameter IN: Message type, Message string NbC: Integer type, Number of characters to extract, $0 < NbC \leq \text{IN message string length}$</p> <p>Output parameter Q: Message type, Extracted message string</p> <p>Description The designated characters are extracted from the left hand side of the designated message string.</p>

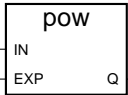
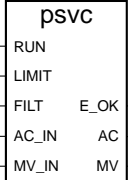
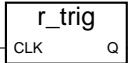
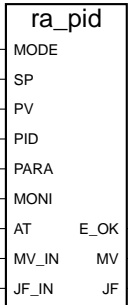
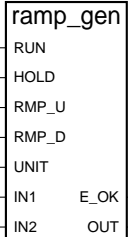
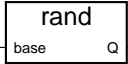
<p>LIM_ALARM (Limit alarm)</p> 	<p>Input parameter H: Real type, High limit set point X: Real type, Input data L: Real type, Low limit set point EPS: Real type, Hysteresis value, $EPS \geq 0.0$</p> <p>Output parameter QH: Boolean type, High limit alarm Q: Boolean type, High limit and low limit alarms QL: Boolean type, Low limit alarm</p> <p>Description Determines using hysteresis whether the real type data has exceeded the high limit value or the low limit value.</p>
<p>LIM_HI (Real type high limit)</p> 	<p>Input parameter MAX: Real type, High limit value IN: Real type, Input data</p> <p>Output parameter Q: Real type, Value limited by the high limit value, MAX at $IN \geq MAX$ IN at $IN < MAX$</p> <p>Description The real type data is limited by the high limit value.</p>
<p>LIM_HILO (Real type high/low limit)</p> 	<p>Input parameter MIN: Real type, Low limit value IN: Real type, Input data MAX: Real type, High limit value</p> <p>Output parameter Q: Real type, Value limited by the high and low limit values, MIN at $IN \leq MIN$ IN at $MIN < IN < MAX$ MAX at $IN \geq MAX$</p> <p>Description The real type data is limited by the high and low limit values.</p>
<p>LIM_LO (Real type low limit)</p> 	<p>Input parameter MIN: Real type, Low limit value IN: Real type, Input data</p> <p>Output parameter Q: Real type, Value limited by the low limit value, MIN at $IN \leq MIN$ IN at $IN > MIN$</p> <p>Description The real type data is limited by the low limit value.</p>
<p>LIMIT (High/low limit)</p> 	<p>Input parameter MIN: Integer type, Low limit value IN: Integer type, Input data MAX: Integer type, High limit value</p> <p>Output parameter Q: Integer type, Value limited by the high and low limit values, MIN at $IN \leq MIN$ IN at $MIN < IN < MAX$ MAX at $IN \geq MAX$</p> <p>Description The integer type data is limited by the high an low limit values.</p>
<p>LOG (Common logarithm)</p> 	<p>Input parameter IN: Real type, $IN > 0.0$</p> <p>Output parameter Q: Real type, $LOG_{10}(IN)$</p> <p>Description Common logarithm of real type data</p>

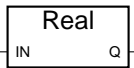
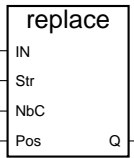
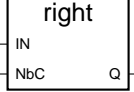
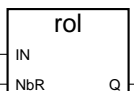

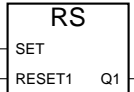
<p>MAV (Moving average)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset TYPE: Integer type, Averaging type, 0=Normal, 1=Ignores the maximum value/minimum value CYCLE: Timer type, Sampling cycle (updates the cycle of a moving average) NUM: Integer type, Number of samples to calculate the mean value, 1 to 30 IN: Real type, Input data</p> <p>Output parameter E_K: Boolean type, TRUE=Normal, FALSE=Error OUT: Real type, Mean value of NUM samples of IN</p> <p>Description Mean value of the real type data in the number of designated samples</p>
<p>MAX (Maximum value)</p> 	<p>Input parameter IN1: Integer type IN2: Integer type</p> <p>Output parameter Q: Integer type, IN1 at IN1>IN2, IN2 at IN1≤IN2</p> <p>Description The maximum value found in 2 pieces of integer type data</p>
<p>MID (Message string extraction)</p> 	<p>Input parameter IN: Message type, Message string NBC: Integer type, Number of characters to extract, Message string length of 0<NBC≤IN POS: Integer type, Location of character to extract, POS>1</p> <p>Output parameter Q: Message type, Extracted message string</p> <p>Description The designated characters are extracted from the message string at the designated location.</p>
<p>MIN (Minimum value)</p> 	<p>Input parameter IN1: Integer type IN2: Integer type</p> <p>Output parameter Q: Integer type, IN1 at IN1<IN2, IN2 at IN1≥IN2</p> <p>Description The minimum value found in 2 pieces of integer type data.</p>
<p>MLEN (Message string length)</p> 	<p>Input parameter IN: Message type, Message string</p> <p>Output parameter NBC: Integer type, Message string length of IN</p> <p>Description Outputs the message string length.</p>
<p>MOD (Modulus)</p> 	<p>Input parameter IN: Integer type BASE: Integer type, BASE>0</p> <p>Output parameter Q: Integer type, Residual of IN / BASE</p> <p>Description Residual of integer type data</p>

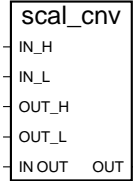
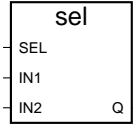
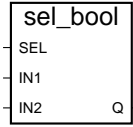
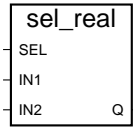
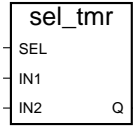
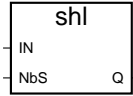
<p>MSG (Convert to message string)</p> 	<p>Input parameter IN: Those excluding message type</p> <p>Output parameter Q: Message type, For a boolean type IN, 'FALSE' at IN=FALSE, 'TRUE' at IN=TRUE For an integer type IN, message string of integer For a real type IN, message string in integer part of IN (the decimal part is truncated) For a timer type IN, message string of timer type data</p> <p>Description Data is converted to message type.</p>
<p>MUX4 (4-input multiplexer)</p> 	<p>Input parameter SEL: Integer type, Select value, 0 to 3 IN1 to IN4: Integer type</p> <p>Output parameter Q: Integer type, Selected IN, IN1 at SEL=0 IN2 at SEL=1 IN3 at SEL=2 0 at other than 1 to 3</p> <p>Description Selects one out of 4 pieces of integer type data.</p>
<p>MUX8 (8-input multiplexer)</p> 	<p>Input parameter SEL: Integer type, Select value, 0 to 7 IN1 to IN8: Integer type</p> <p>Output parameter Q: Integer type, Selected IN, IN1 at SEL=0 IN2 at SEL=1 . . . IN8 at SEL=7 0 at other than 0 to 7</p> <p>Description Selects one out of 8 pieces of integer type data.</p>
<p>MUX8REAL (Real type 8-input multiplexer)</p> 	<p>Input parameter SEL: Integer type, Select value, 0 to 7 IN1 to IN8: Real type</p> <p>Output parameter Q: Real type, Selected IN, IN1 at SEL=0 IN2 at SEL=1 . . . IN8 at SEL=7 0.0 at other than 0 to 7</p> <p>Description Selects one out of 8 real type data.</p>
<p>NEG (Sign change)</p> 	<p>Input parameter IN: Integer type real type</p> <p>Output parameter Q: Integer type real type, -IN</p> <p>Description Sign change of data</p>
<p>NOT_MASK (Bitwise inversion)</p> 	<p>Input parameter IN: Integer type</p> <p>Output parameter Q: Integer type, bitwise inversion</p> <p>Description Bitwise inversion of integer type data</p>


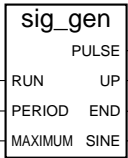
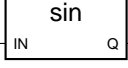
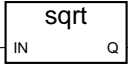
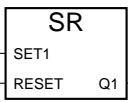
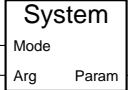
<p>ODD (Odd parity)</p> 	<p>Input parameter IN: Integer type</p> <p>Output parameter Q: Boolean type, TRUE when IN is odd, FALSE when IN is even.</p> <p>Description Determines whether the integer type data is odd or even.</p>
<p>OR_MASK (bitwise OR)</p> 	<p>Input parameter IN: Integer type MASK: Integer type</p> <p>Output parameter Q: Integer type, bitwise OR of IN and MASK</p> <p>Description Bitwise OR of 2 integer type data</p>
<p>PAR_BOOL (Read Boolean type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error R_VAL: Boolean type, Read-out data</p> <p>Description A DMC50's dedicated parameter is read out as Boolean type data.</p>
<p>PAR_INT (Read Integer type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID</p> <p>Output parameter E_OK: Integer type, TRUE=Normal, FALSE=Error R_VAL: Integer type, Read-out data</p> <p>Description A DMC50's dedicated parameter is read out as integer type data.</p>
<p>PAR_REAL (Read Real type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error R_VAL: Real type, Read-out data</p> <p>Description A DMC50's dedicated parameter is read out as real type data.</p>
<p>PAW_BOOL (Write Boolean type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID W_VAL: Boolean type, data to be written</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error</p> <p>Description Boolean type data is written into a DMC50's dedicated parameter.</p>

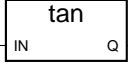
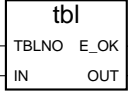
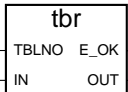
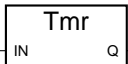

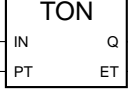
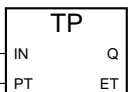
<p>PAW_INT (Write Integer type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID W_VAL: Integer type, data to be written</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error</p> <p>Description Integer type data is written into a DMC50's dedicated parameter.</p>
<p>PAW_REAL (Write Real type parameter)</p> 	<p>Input parameter TYPE: Integer type, Parameter type ID ID: Integer type, Group ID NO: Integer type, Item ID W_VAL: Real type, data to be written</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error</p> <p>Description The real type data is written into a DMC50's dedicated parameter.</p>
<p>PID_A (Standard PID operation)</p> 	<p>Input parameter MODE: Boolean type, TRUE=MANUAL mode, FALSE=AUTO mode SP: Real type, SP (industrial unit) PV: Real type, PV (industrial unit) PID: Integer type, Designates the group ID of PID_A constants PARA: Integer type, Designates the group ID of PID_A options MONI: Integer type, Designates the group ID of PID_A monitor AT: Boolean type, Start/stop of auto-tuning MV_IN: Real type, Outputting value at MANUAL mode</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error MV: Real type, Manipulated variable</p> <p>Description A series form PID operation of which derivative action applied on error</p>
<p>PID_CAS (Cascade PID operation)</p> 	<p>Input parameter MAN: Boolean type, TRUE=MANUAL mode, FALSE=AUTO mode RUN: Boolean type, TRUE=RUN mode, FALSE=READY mode LOCAL: Boolean type, TRUE=LOCAL mode, FALSE=REMOTE mode M_SP: Real type, Master side SP (industrial unit) M_PV: Real type, Master side PV (industrial unit) S_LSP: Real type, Slave side LSP (industrial unit), For LOCAL mode S_PV: Real type, Slave side PV (industrial unit) RSP_H: Real type, RSP scale upper limit (industrial unit) RSP_L: Real type, RSP scale lower limit (industrial unit) M_PID: Integer type, Designation of group ID for PID_CAS constants (master side) S_PID: Integer type, Designation of group ID for PID_CAS constants (slave side) PARA: Integer type, Designation of group ID for PID_CAS options MONI: Integer type, Designation of group ID for PID_CAS monitor AT: Boolean type, Start/stop of auto-tuning MV_IN: Real type, Outputting value at MANUAL mode</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error AT_MD: Boolean type, TRUE=During AT, FALSE=AT stop S_RSP: Real type, Slave side remote SP (industrial unit) MV: Real type, Manipulated variable</p> <p>Description PID operation for cascade control</p>
<p>PLS_GEN (Pulse generator)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset CYCLE: Timer type, CYCLE≥Cycle time setting</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error OUT: Boolean type, Pulse output</p> <p>Description A one cycle time pulse is generated at each designated cycle.</p>

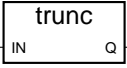
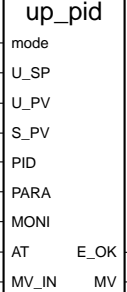
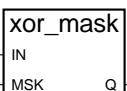
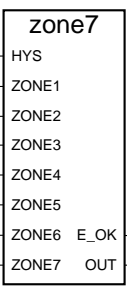
<p>POW (Exponential function)</p> 	<p>Input parameter IN: Real type, Base number EXP: Real type, Exponent number</p> <p>Output parameter Q: Real type, IN^{EXP}</p> <p>Description Exponential operation of real type data</p>
<p>PSVC (Power supply voltage compensation)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset LIMIT: Real type, Compensation range, 0.00 to 50.0% (Normally 20.0% is sufficient) Executing the compensation at $AC_IN=100\% \pm LIMIT$ FILT: Real type, Filter constant for AC_IN, 0.0 to 2000.0s AC_IN: Real type, AC voltage input, an AUX_IN (AC voltage) input MV_IN: Real type, Manipulated variable before compensation</p> <p>Output parameter E_OK: Boolean type, TRUE=During compensation, FALSE=Compensation stop AC: Real type, AC value after filter operation MV: Real type, Manipulated variable after compensation</p> <p>Description Heater power voltage (AUX_IN) fluctuations are monitored and the manipulated variable output is compensated for.</p>
<p>R_TRIG (Rising edge detection)</p> 	<p>Input parameter CLK: Boolean type</p> <p>Output parameter Q: Boolean type, TRUE when CLK is FALSE→TRUE, and FALSE otherwise.</p> <p>Description Rising edge detection of Boolean type data</p>
<p>Ra_PID (Rationa Loop PID operation)</p> 	<p>Input parameter MODE: Boolean type, TRUE=MANUAL mode, FALSE=AUTO mode SP: Real type, SP (industrial unit) PV: Real type, PV (industrial unit) PID: Integer type, Designates the group ID of Ra_PID constants PARA: Integer type, Designates the group ID of Ra_PID options MONI: Integer type, Designates the group ID of Ra_PID monitor AT: Boolean type, Start/stop of auto-tuning MV_IN: Real type, Output value at MANUAL mode JF_IN: Real type, Just-FITTER input, Inputting 0.0 normally</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error MV: Real type, Manipulated variable JF: Real type, Just-FITTER output</p> <p>Description PID operation incorporated with superior disturbance recovery and over-shoot suppression feature for high precision control</p>
<p>RAMP_GEN (Ramp generator)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset HOLD: Boolean type, TRUE=Holding the ramp action, FALSE=Continuing the ramp action (canceling the hold) RMP_U: Real type, Ramp-Up variable RMP_D: Real type, Ramp-Down variable UNIT: Integer type, Ramp time unit, 0: Ramp variable/s, 1: Ramp variable/min, 2: Ramp variable/h IN1: Real type, Starting point of ramp action IN2: Real type, Target point of ramp action</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error OUT: Real type, Ramp data</p> <p>Description A ramp of real data generated from IN1 up to the target value IN2.</p>
<p>RAND (Random value)</p> 	<p>Input parameter BASE: Integer type, Maximum value of random value $BASE > 1$</p> <p>Output parameter Q: Integer type, Random value 0 to $BASE - 1$</p> <p>Description Random value generation of integer type data</p>

<p>REAL (Convert to real)</p> 	<p>Input parameter IN: Boolean type Integer type Timer type</p> <p>Output parameter Q: Real type, For a boolean type IN, 0.0 at IN=FALSE and 1.0 at IN=TRUE. For an integer type IN, IN. For a timer type IN, mSec numerical value.</p> <p>Description Data is converted to real type data.</p>
<p>REPLACE (Message string replacement)</p> 	<p>Input parameter IN: Message type, Base message string STR: Message type, Message string to insert NBC: Integer type, Number of characters to delete prior to insert POS: Integer type, Begin location for insertion of message string, POS>1</p> <p>Output parameter Q: Message type, Message string modified by replacement</p> <p>Description Message for the number of designated characters is deleted at the designated character location, and then the designated message string is inserted into the base string for replacement.</p>
<p>RIGHT (Right message string extraction)</p> 	<p>Input parameter IN: Message type, Message string NBC: Integer type, Number of characters to extract, Message string length of 0<NBC≤IN</p> <p>Output parameter Q: Message type, Message string extracted</p> <p>Description The designated characters are extracted from the right hand side of the designated message string.</p>
<p>ROL (Left rotation)</p> 	<p>Input parameter IN: Integer type NBR: Integer type, Number of bits for left rotation, 1 to 31</p> <p>Output parameter Q: Integer type, Result of left rotation for IN</p> <p>Description Integer type data bits are rotated left.</p>
<p>ROR (Right rotation)</p> 	<p>Input parameter IN: Integer type NBR: Integer type, Number of bits for right rotation, 1 to 31</p> <p>Output parameter Q: Integer type, Result of right rotation for IN</p> <p>Description Integer type data bits are rotated right.</p>
<p>RS (Reset dominant bistable)</p> 	<p>Input parameter SET: Boolean type RESET1: Boolean type</p> <p>Output parameter Q: Boolean type, TRUE at SET=TRUE, In case of RESET1=TRUE, FALSE (reset dominant)</p> <p>Description Reset dominant bistable latch</p>

<p>SCAL_CNV (Scale conversion)</p> 	<p>Input parameter IN_H: Real type, Input scale high limit IN_L: Real type, Input scale low limit OUT_H: Real type, Output scale high limit OUT_L: Real type, Output scale low limit IN: Real type, Input data</p> <p>Output parameter OUT: Real type, Output data, $OUT = (IN-IN_L) / (IN_H-IN_L) \times (OUT_H-OUT_L) + OUT_L$</p> <p>Description Scaling operation of real type data</p>
<p>SEL (Binary selection)</p> 	<p>Input parameter SEL: Boolean type, For selection IN1: Integer type IN2: Integer type</p> <p>Output parameter Q: Integer type, IN1 at SEL=FALSE, IN2 at SEL=TRUE</p> <p>Description Selects one out of 2 pieces of integer type data.</p>
<p>SEL_BOOL (Boolean type binary selection)</p> 	<p>Input parameter SEL: Boolean type, For selection IN1: Boolean type IN2: Boolean type</p> <p>Output parameter Q: Boolean type, IN1 at SEL=FALSE IN2 at SEL=TRUE</p> <p>Description Selects one out of 2 pieces of Boolean type data.</p>
<p>SEL_REAL (Real type binary selection)</p> 	<p>Input parameter SEL: Boolean type, For selection IN1: Real type IN2: Real type</p> <p>Output parameter Q: Real type, IN1 at SEL=FALSE, IN2 at SEL=TRUE</p> <p>Description Selects one out of 2 pieces of real type data</p>
<p>SEL_TMR (Timer type binary selection)</p> 	<p>Input parameter SEL: Boolean type, For selection IN1: Timer type IN2: Timer type</p> <p>Output parameter Q: Timer type, IN1 at SEL=FALSE, IN2 at SEL=TRUE</p> <p>Description Selects one out of 2 pieces of timer data.</p>
<p>SHL (Left shift)</p> 	<p>Input parameter IN: Integer type NbS: Integer type, Number of bits for left shift 1 to 31</p> <p>Output parameter Q: Integer type, Result of left shift for IN (0 inserted into LSB)</p> <p>Description Left bit shift of integer type data</p>

<p>SHR (Right shift)</p> 	<p>Input parameter IN: Integer type NBS: Integer type, Number of bits for right shift, 1 to 31</p> <p>Output parameter Q: Integer type, Result of right shift for IN (save value copied to MSB)</p> <p>Description Right bit shift of integer type data</p>
<p>SIG_GEN (Signal generator)</p> 	<p>Input parameter RUN: Boolean type, TRUE=Execution, FALSE=Reset PERIOD: Timer type, Sampling time (pulse width), PERIOD≥cycle time setting MAXIMUM: Integer type, Maximum count value, MAXIMUM≥0</p> <p>Output parameter PULSE: Boolean type, Inverse at every sampling time, UP: Integer type, Up-counter of every sampling time END: Boolean type, End signal of up-counter SINE: Real type, Sine waveform signal (One cycle of waveform is a counting duration.)</p> <p>Description Generation of 3 signals (pulse, up-counter and sine waveform signal)</p>
<p>SIN (Sine)</p> 	<p>Input parameter IN: Real type, Range of real values (radian unit)</p> <p>Output parameter Q: Real value, -1.0 to +1.0 Sin(IN)</p> <p>Description Sine of real type data</p>
<p>SQRT (Square root)</p> 	<p>Input parameter IN: Real type, IN≥0.0</p> <p>Output parameter Q: Real type, \sqrt{IN}</p> <p>Description Square root of real type data</p>
<p>SR (Set dominant bistable)</p> 	<p>Input parameter SET1: Boolean type RESET: Boolean type</p> <p>Output parameter Q: Boolean type, TRUE (set dominant) at SET1=TRUE FALSE at RESET=TRUE</p> <p>Description Set dominant bistable latch</p>
<p>SYSTEM (Access to system)</p> 	<p>Input parameter MODE: Integer type, Command ARG: Integer type, 0 at MODE≠SYS_TWRITE Timer type, New cycle time at MODE=SYS_TWRITE</p> <p>Output parameter Q: Integer type, Access result, Reads out the setting value of cycle time in case of MODE=SYS_TALLOWED. Reads out the execution time of previous cycle in case of MODE=SYS_TCURRENT. Reads out the maximum value of execution time in case of MODE=SYS_TMAXIMUM. Reads out the number of overflow of execution time in case of MODE=SYS_TOVERFLOW. Resets the maximum value of execution time and the number of overflow in case of MODE=SYS_TRESET. Changes the setting value of cycle time in case of MODE=SYS_TWRITE. Checks the run time error in case of MODE=SYS_ERR_TEST.</p> <p>Description Read out cycle time, etc</p>

<p>TAN (Tangent)</p> 	<p>Input parameter IN: Real type, $\pm \pi/2 \times a$ ($n = 1,3,5, \dots$), Radian unit</p> <p>Output parameter Q: Real type, $\tan(\text{IN})$</p> <p>Description Tangent of real type data</p>
<p>TBL (Linearization table lookup)</p> 	<p>Input parameter TBLND: Integer type, Sets the group ID of TBL/TBR setup parameters IN: Real type, Input data</p> <p>Output parameter OUT: Real type, Output data</p> <p>Description Pieewise linearization table lookup for input data</p>
<p>TBR (Linearization table reverse lookup)</p> 	<p>Input parameter TBLND: Integer type, Sets the group ID of TBL/TBR setup parameters IN: Real type, Input data</p> <p>Output parameter OUT: Real type, Output data</p> <p>Description Pieewise linearizatin table reverse lookup for input data.</p>
<p>TMR (Convert to timer)</p> 	<p>Input parameter IN: Integer type, real type</p> <p>Output parameter Q: Timer type, In case of integer type IN, mSec value explained by IN In case of real type IN, mSec value explained by the integer portion of IN (The decimal part is truncated)</p> <p>Description Data is converted into timer type data.</p>
<p>TOF (OFF delay timer)</p> 	<p>Input parameter IN: Boolean type, Timer start at falling edge detection, timer stop and reset at rising edge detection PT: Timer type, Preset timer value</p> <p>Output parameter Q: Boolean, FALSE at $\text{ET}=\text{PT}$ ET: Timer type, Elapsed timer value</p> <p>Description Stops the timer at a specified time after falling edge detection.</p>
<p>TON (ON delay timer)</p> 	<p>Input parameter IN: Boolean type, Timer start at rising edge detection, timer stop and reset at falling edge detection PT: Timer type, Preset timer value</p> <p>Output parameter Q: Boolean, TRUE at $\text{ET}=\text{PT}$ ET: Timer type, Elapsed timer value</p> <p>Description Starts the timer at a specified time after rising edge detection.</p>
<p>TP (Pulse timer)</p> 	<p>Input parameter IN: Boolean type, Timer start at rising edge detection PT: Timer type, Preset timer value</p> <p>Output parameter Q: Boolean type, TRUE at $\text{ET}<\text{PT}$ during timer movement ET: Timer type, Elapsed timer value</p> <p>Description Starts the timer upon rising edge detection for a specified period of time.</p>

<p>TRUNC (Truncate the decimal part)</p> 	<p>Input parameter IN: Real type</p> <p>Output parameter Q: Real type, Integer portion of IN</p> <p>Description The number of real type data is rounded to nearest integer value toward zero.</p>
<p>UP_PID (Use-point PID operation)</p> 	<p>Input parameter MODE: Boolean type, TRUE=MANUAL mode, FALSE=AUTO mode U_SP: Real type, Use SP (industrial unit) U_PV: Real type, Use PV (industrial unit) S_PV: Real type, Source PV (industrial type) PID: Integer type, Designates the group ID of UP_PID constants PARA: Integer type, Designates the group ID of UP_PID options MONI: Integer type, Designates the group ID of UP_PID monitor AT: Boolean type, Start/stop of auto-tuning MV_IN: Real type, Outputting value at MANUAL mode</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error MV: Real type, Manipulated variable</p> <p>Description 2-input and one-output PID operation for disturbance rejection</p>
<p>XOR_MASK (Bitwise XOR)</p> 	<p>Input parameter IN: Integer type MASK: Integer type</p> <p>Output parameter Q: Integer type, Bitwise exclusive OR of IN and MASK</p> <p>Description Bitwise exclusive OR of 2 pieces of integer type data</p>
<p>ZONE7 (Zone selector)</p> 	<p>Input parameter NYS: Real type, Zone-to-zone hysteresis $HYS \geq 0.0$ ZONE1 to 7: Zone boundary values $ZONE1 < ZONE2 < \dots < ZONE7$ IN: Real type, Input data</p> <p>Output parameter E_OK: Boolean type, TRUE=Normal, FALSE=Error OUT: Real type, Zone value (0 to 7) $OUT = n$ at $ZONE(n) \leq IN < ZONE(n+1)$</p> <p>Description Determines the zone number by using 7 zone conditions on the input and then outputs a value between 0 to 7 (total 8 zones).</p>

■ ST Configuration Text

● Basic instructions

Name	Function
Assignment	<p>Expression: =</p> <p>Meaning: The value of an expression is assigned to a variable.</p> <p>Syntax: <variable> := <any_expression></p> <p>Operands: Variable should be an internal or output variable. Variable and expression should be of the same type.</p>
RETURN	<p>Expression: RETURN</p> <p>Meaning: Ends the program currently being executed.</p> <p>Syntax: RETURN</p> <p>Operands: (none)</p>
IF-THEN-ELSE	<p>Expression: IF...THEN...ELSIF...THEN...ELSE...END_IF</p> <p>Meaning: Executes one of the lists of ST statements according to the evaluation of each Boolean type expression.</p> <p>Syntax: IF <conditional expression> THEN <statement>; <statement>; ... ELSIF <conditional expression> THEN <statement>; <statement>; ... ELSE <statement>; <statement>; END_IF:</p>
CASE	<p>Expression: CASE...OF...ELSE...END_CASE</p> <p>Meaning: Executes one of the lists of ST statements according to the evaluation of each integer expression.</p> <p>Syntax: CASE <integer type variable> OF <integer type data> : <statement>; <integer type data>, <integer type data> : <statement>; ... ELSE <statement>; END_CASE;</p>
WHILE	<p>Expression: WHILE...DO...END_WHILE</p> <p>Meaning: Executes a group of ST statements in repetition. Evaluates the condition before any repetition.</p> <p>Syntax: WHILE <conditional expression> DO <statement>; <statement>; ... END_WHILE</p>
REPEAT	<p>Expression: REPEAT...UNTIL... END_REPEAT</p> <p>Meaning: Executes the group of ST statements in repetition. Evaluates the condition after any repetition.</p> <p>Syntax: REPEAT <statement>; <statement>; ... UNTIL <conditional equation>; END_REPEAT</p>
FOR	<p>Expression: FOR...TO...BY...DO...END_FOR</p> <p>Meaning: Executes a repetitive operation to a finite number of cycles according to the index of the integer type variable.</p> <p>Syntax: FOR<index> :=<min>TO<max>BY<step>DO <statement>; <statement>; END_FOR;</p> <p>Operand: index: Internal integer type variable increased at each iteration min: Initial value of index max: Maximum value of index step: The size of increment for index at each iteration</p>
EXIT	<p>Expression: EXIT</p> <p>Meaning: Executes EXIT from a FOR, WHILE, or REPEAT repetitive statement.</p> <p>Syntax: EXIT;</p> <p>Operand: (none)</p>

● Special Boolean instructions

Name	Function
REDGE (Rising edge detection)	Expression: REDGE Meaning: Performs the rising edge detection of a Boolean type expression. Syntax: <return value>: =REDGE (<boo_expression>, <memo_variable>); Operands: <boo_expression> Boolean type variable or expression of which rising edge is to be detected. <memo_variable> Internal variable to store the last state of the expression. Return value: TRUE When changed from FALSE to TRUE. FALSE Other than the above.
FEDGE (Falling edge detection)	Expression: FEDGE Meaning: Performs the falling edge detection of a Boolean type expression. Syntax: <return value>: =FEDGE (<boo_expression>, <memo_variable>) Operands: <boo_expression> Boolean type variable or expression of which falling edge is to be detected. <memo_variable> Internal variable to store the last state of the expression. Return value: TRUE When changed from TRUE to FALSE. FALSE Other than the above.

* ISaGRAF is a registered trademark of Alter Sys.

* Windows is a registered trademark of Microsoft Corporation of the USA.

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>

This has been printed on recycled paper.

Printed in Japan. (H)
1st Edition: Issued in May, 2002