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 “RationaLOOP” 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.

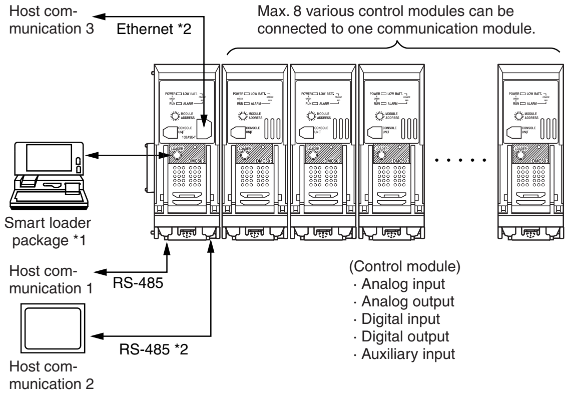
- ● Control module
- High resolution. processing
  · Thermocouple input: 1/100°C of a specific range.
  · RTD input: 1/5000°C (Special model)
  · 4 to 20 mA input: 1/32000 resolution
- Many analog process variables can be processed.
  · Analog input: 4 universal inputs (Max. 8 inputs with voltage or 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.

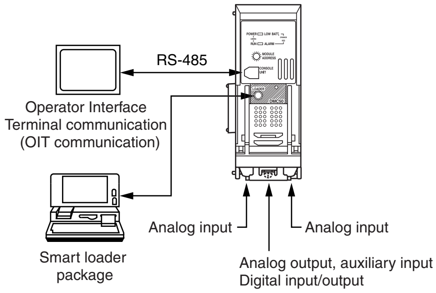
Configuration Examples

• Combination of communication module and control module



*1 Smart loader can be connected from RS-485.
*2 ME200 module only

• Control module only


RS-485
Operator Interface Terminal communication (OIT communication)
Analog input
Analog input
Smart loader package
Analog output, auxiliary input
Digital input/output

* In this case, the communication distance is max. 10m. A modular jack is used as a connector. When Operator Interface Terminal communication (OIT communication) is executed, smart loader communication cannot be performed.
When a smart loader is to be used, disconnect the Operator Interface Terminal communication (OIT communication) cable (modular jack).

Common General Specifications

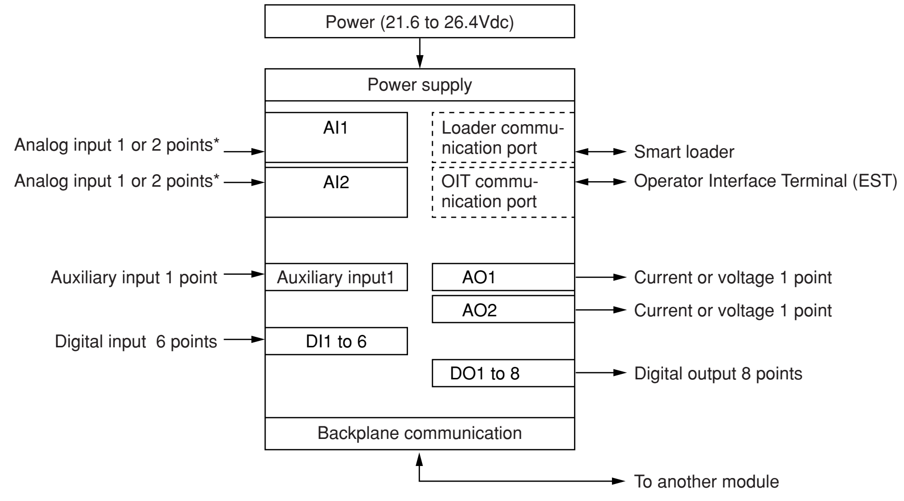
	Module type	Communication module		Control module	
	Type	Ethernet type	RS-485 type	High resolution (2/4 loops) type	Special (2/4 100ps) type
	Basic model number	**DMC50ME20□**	**DMC50MR20□**	**DMC50CH20□/40□**	**DMC50CS20□/40□**
Memory backup		Clock data Parameter: RAM with battery backup Battery backup 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.

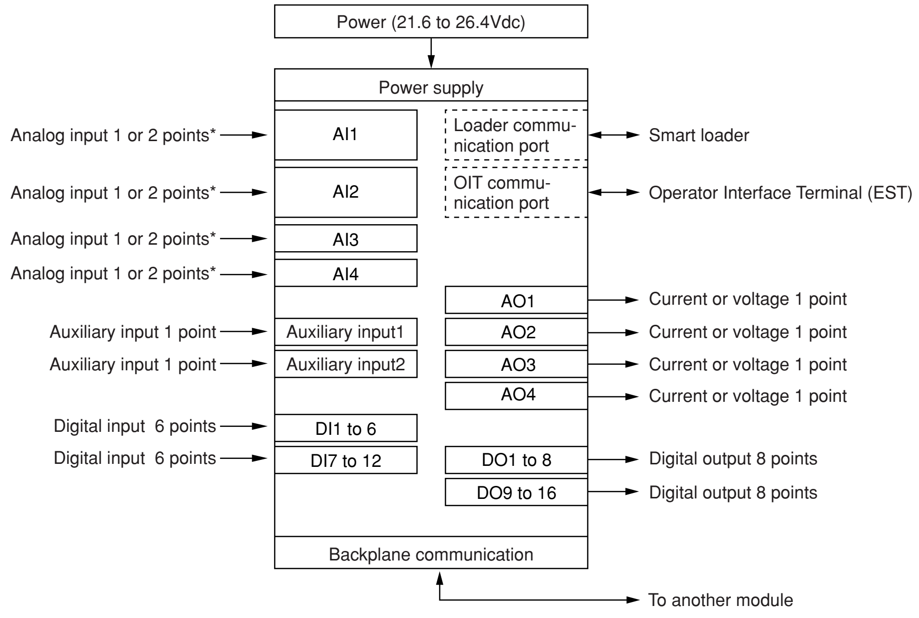
Input/output configuration

- **2 loop type**



* Differs by input type

- **4 loop type**



* Differs by input 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	
	Overrange	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)	• 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 • While in operating mode: Execution cycle time • While in application inactive mode: 100ms Time proportional current output: 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
Backplane 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 ±°C (±%FS)
1	K: CA	–200.00 to +1200.00 °C	0.7 (0.05) *1
2	K: CA	–200.00 to +400.00 °C	0.3 (0.05) *1
3	E: CRC	0.00 to 800.00 °C	0.4 (0.05)
4	J: IC	0.00 to 800.00 °C	0.4 (0.05)
5	N: NIcr-Ni	0.00 to 1300.00 °C	0.65 (0.05)
6	PLII	0.00 to 1300.00 °C	0.65 (0.05)
7	T: CC	–200.00 to +300.00 °C	0.25 (0.05)
8	B: PR30-6	0.00 to 1800.00 °C	1.8 (0.1) *2
9	R: PR13	0.00 to 1600.00 °C	1.2 (0.075) *3
10	S: RP10	0.00 to 1600.00 °C	1.2 (0.075)
11	PR40-20	0.00 to 1900.00 °C	4.75 (0.25) *4
12	WRe5-26	0.00 to 2300.00 °C	1.25 (0.05)
13	WRe5-26	0.00 to 1400.00 °C	1.25 (0.09)
14	DIN L	–200.00 to +800.00 °C	0.5 (0.05) *5
15	DIN U	–200.00 to +400.00 °C	0.3 (0.05) *6
16	Ni-NiMo	0.00 to +1300.00 °C	1.3 (0.1)
21	Pt100	–200.00 to +500.00 °C	0.35 (0.05)
22	Pt100	–60.00 to +100.00 °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: ±1.5°C for range K29, ±0.6°C for K24, ±0.5°C for range T44

*2: B T/C: ±72.0°C for less than 260°C, ±3.6°C for 260 to 800°C

*3: R and S T/C: ±1.6°C for less than 100°C

*4: PR40-20 T/C: ±23.7°C for less than 300°C, ±14.2°C for 300 to 800°C

*5: DIN standard L T/C: ±0.75°C for less than –100°C

*6: DIN standard U T/C: ±1.0°C for less than –100°C, ±0.5°C for –100 to 0°C

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

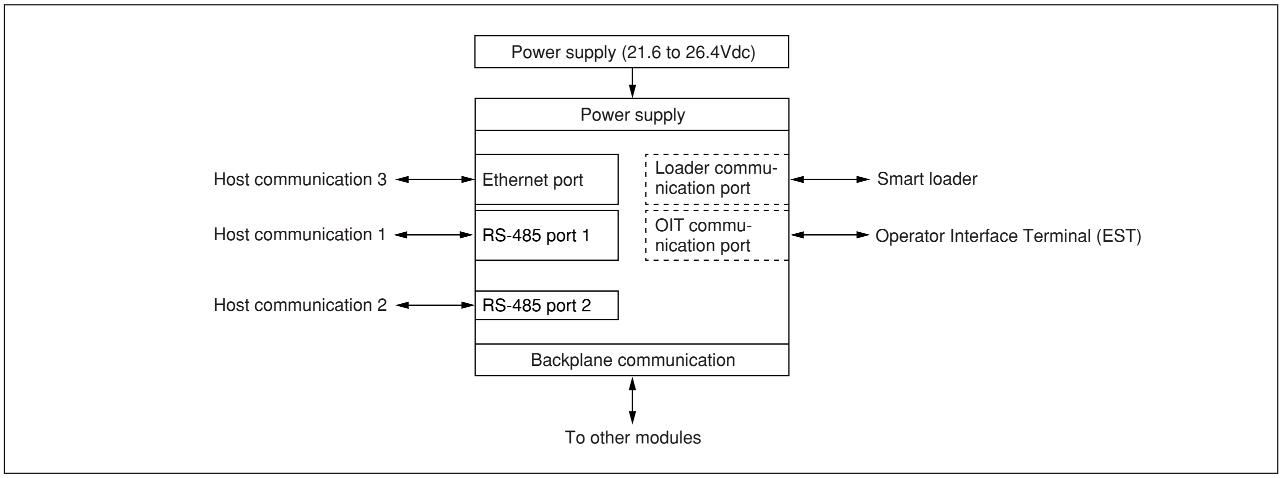
No.	Range symbol	Range	Indication accuracy ±°C (±%FS)
23	Pt100	16.000 to 37.000 °C	0.10
24	Pt100	–50.000 to +150.000 °C	0.15
33		0 to 10V	(0.04)
34		0 to 5V	(0.08)
35		1 to 5V	(0.10)
36		–1 to +1V	(0.05)
37		0 to 1V	(0.10)
38		–100 to +100mV	(0.05)
39		0 to 100mV	(0.10)

Communication Module Individual Specifications

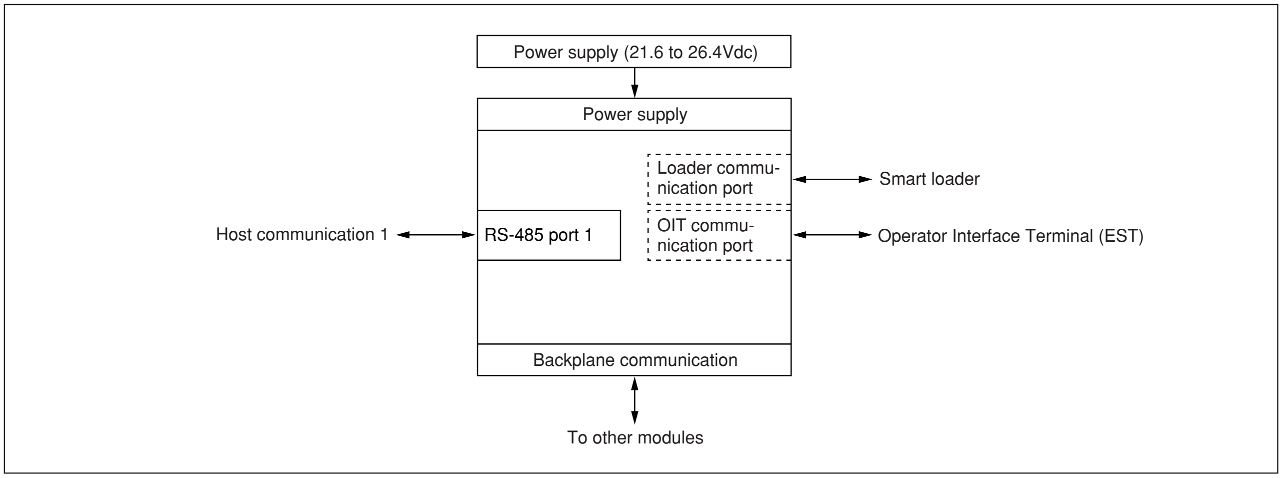
There are two types of communication modules for the ME model (Ethernet/ RS-485) and the MR model (RS-485). A single communication module can be connected to a maximum of 8 control modules and electricity is supplied to each control module by the backplane.

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

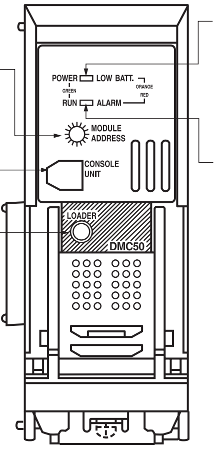
● Communication specifications

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)



Module address:
Used to set an address for host communication.
0: Communication disabled.
1 to F: Communication enabled.

Connector for Operator Interface Terminal communication
(RJ-11: 4-pole 4-core)

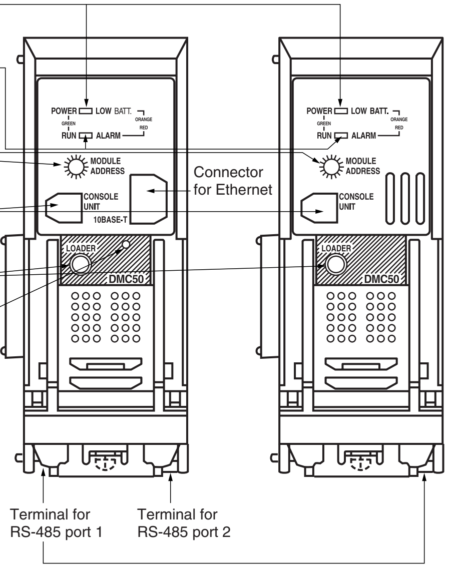
Jack for loader communication:
A dedicated cable packed with the DMC50 personal computer loader SLP-D50/J50 (separate order) can be connected for setting and monitoring by the loader.

Power battery lamp:
Lights when power is supplied.
Green light: Power ON
Battery voltage normal
Orange light: Power ON
Occurrence of system alarm

Operation lamp: ON when in operation.
Green light:
Operating in real time mode.
Orange light:
Operating in cycle to cycle mode.
Red light:
Occurrence of system alarm.

● Communication module

(DMC50ME200) (DMC50MR200)



Power battery lamp
(same as control module)

Operation lamp:
OFF: Normal operation
Red light:
Occurrence of system alarm

Module address
(same as control module)

Connector for operator interface terminal communication
(same as control module)

Jack for loader communication
(same as control module)

Ethernet status LED
Green light:
Link has been established.
Orange light:
Communication is being performed.

Connector for Ethernet

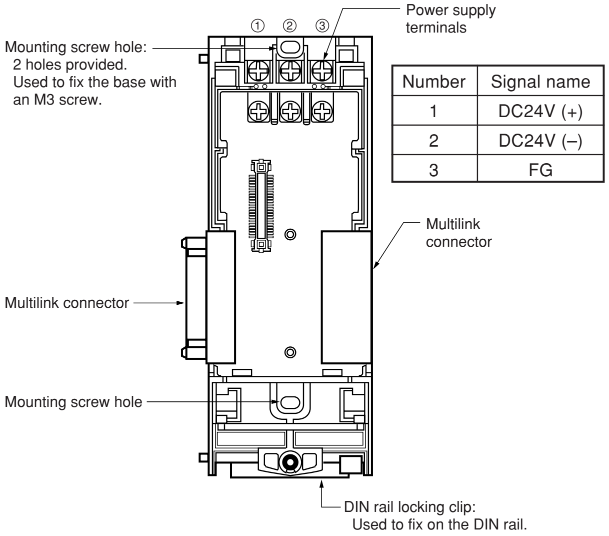
Terminal for RS-485 port 1

Terminal for RS-485 port 2

■ Base

• Standard base

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

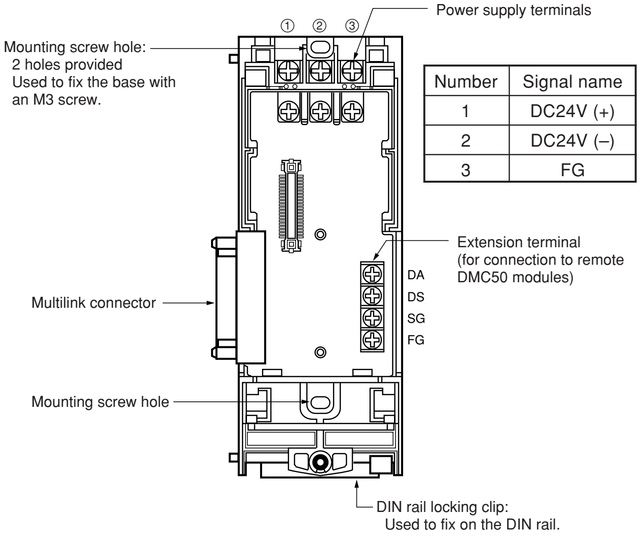


①
②
③
Power supply terminals
Mounting screw hole: 2 holes provided. Used to fix the base with an M3 screw.
Multilink connector
Multilink connector
Mounting screw hole
DIN rail locking clip: Used to fix on the DIN rail.

Number	Signal name
1	DC24V (+)
2	DC24V (–)
3	FG

• Extension base

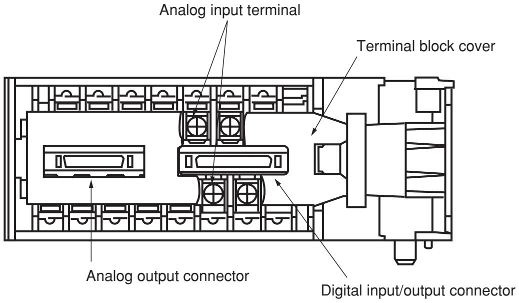
(Module model numbers: CH201, CH401, CS201, CS401, MR201, ME201)



①
②
③
Power supply terminals
Mounting screw hole: 2 holes provided Used to fix the base with an M3 screw.
Extension terminal (for connection to remote DMC50 modules)
DA
DS
SG
FG
Multilink connector
Mounting screw hole
DIN rail locking clip: Used to fix on the DIN rail.

Number	Signal name
1	DC24V (+)
2	DC24V (–)
3	FG

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

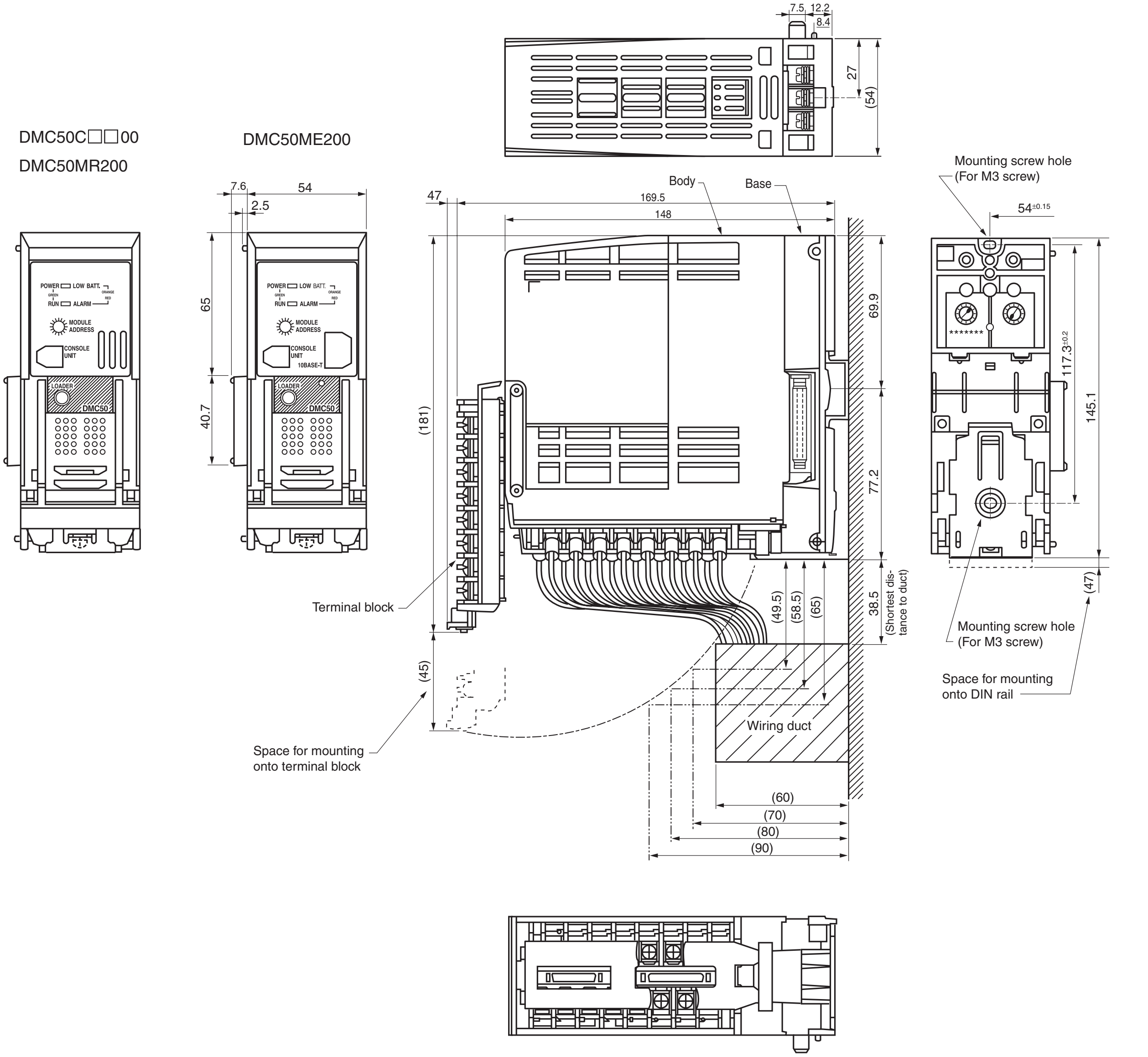
Applicable Connector Type

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

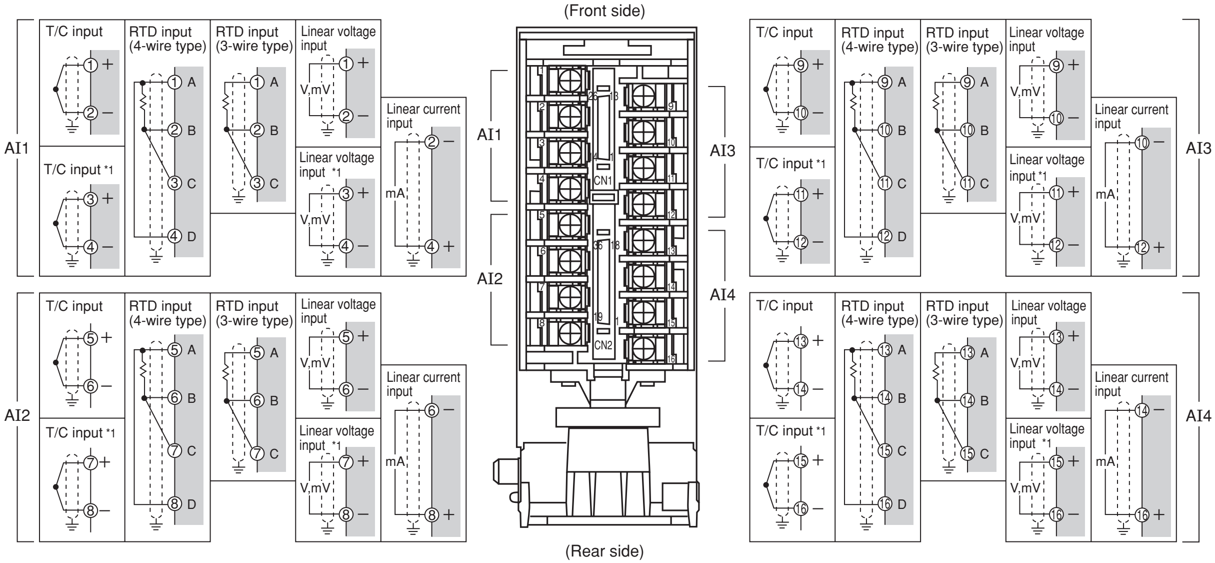
(Unit: mm)



■ Control module terminal layout

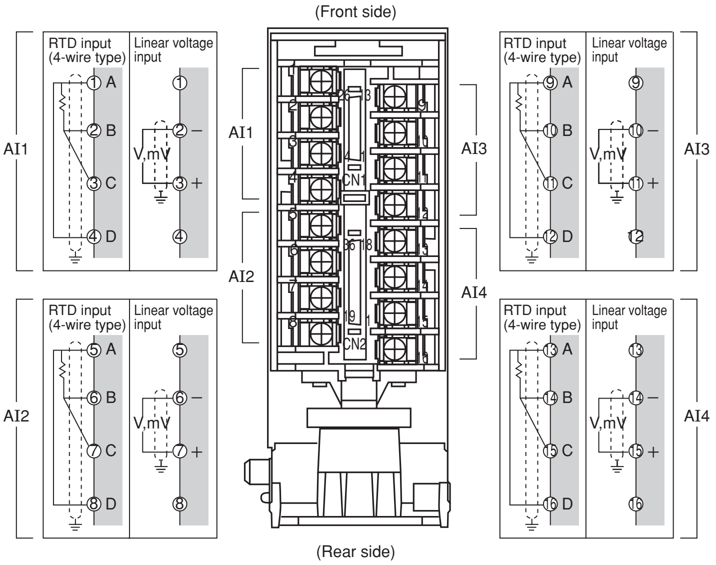
● Analog input terminal

DMC50CH□00 (high resolution model)

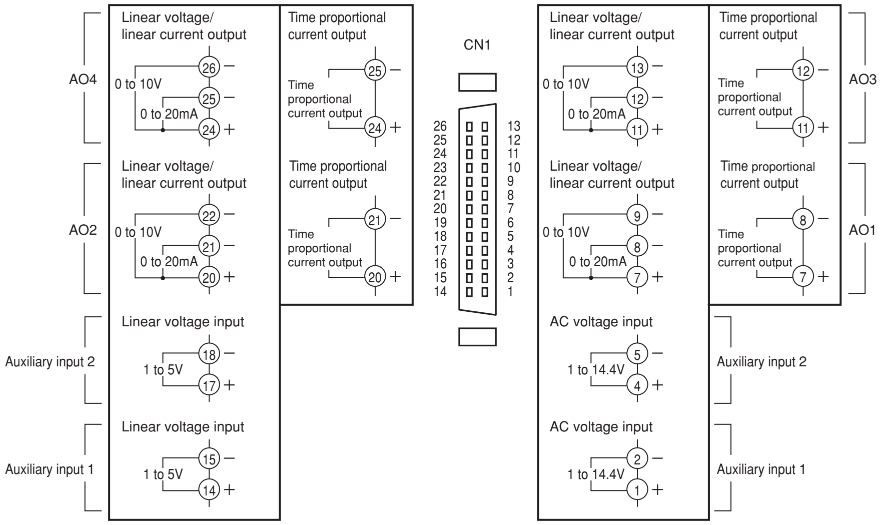


*1: Option input

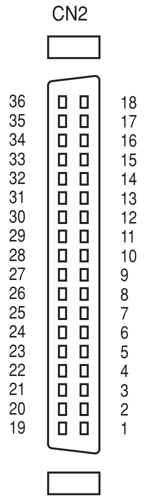
DMC50CS□00 (Special model)



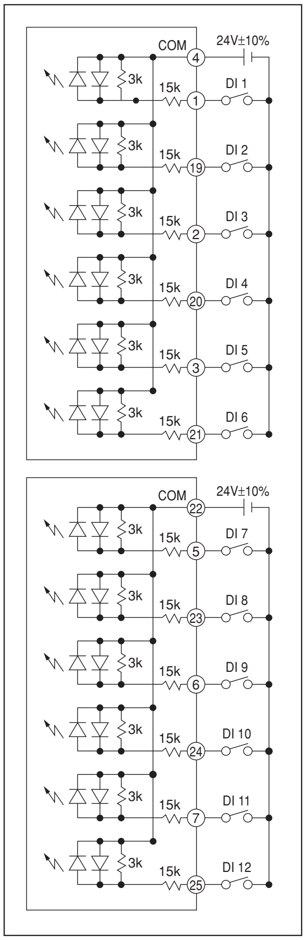
● Connector 1 (CN1)



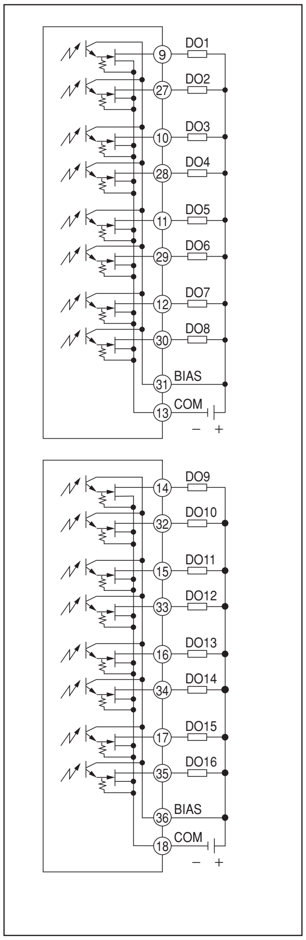
● Connector 2 (CN2)



Digital input terminal wiring diagram

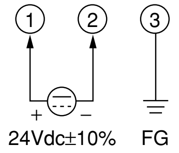


Digital output terminal wiring diagram



■ Power supply connection

Connect the power supply as follows:



! Handling Precautions

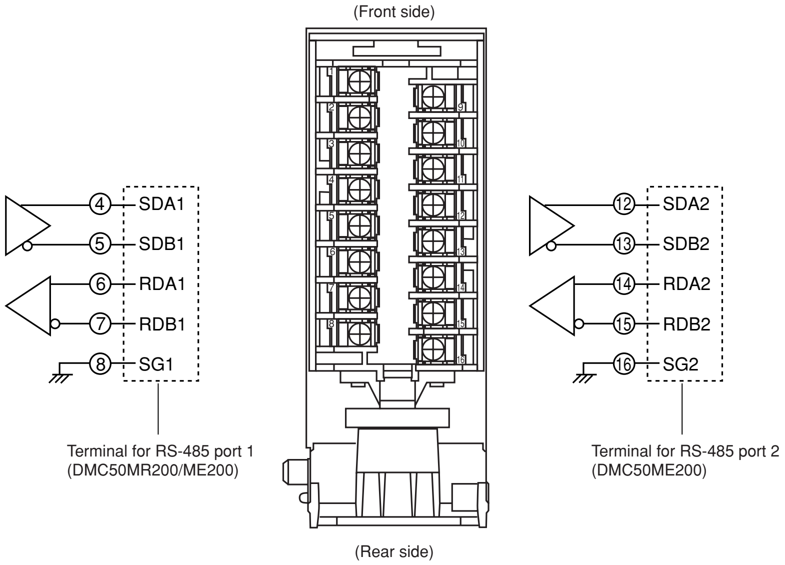
- When selecting a switching power supply unit, ensure that it generates minimal switching noise.
- The power supply lines and grounding wires to the controller should not be bundled with the power cable or should not be run in the same wiring conduit or duct.
- The power supply is mutually connected 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

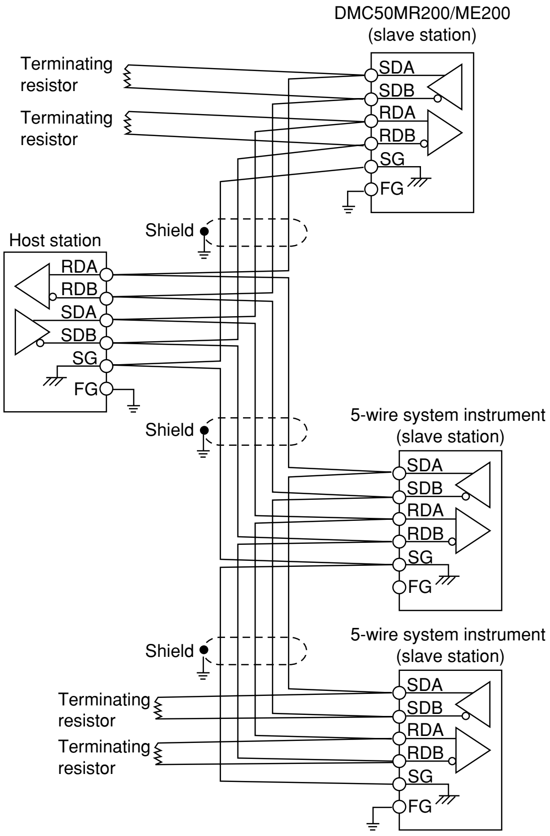
- Perform wiring only after confirming the controller model number and terminal number found on the label on the side of the main body.
- Make sure to use crimp terminals that conform to 3.5 mm screws for terminal connection.
- Do not allow the sides of the crimp terminals to come into contact with each 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

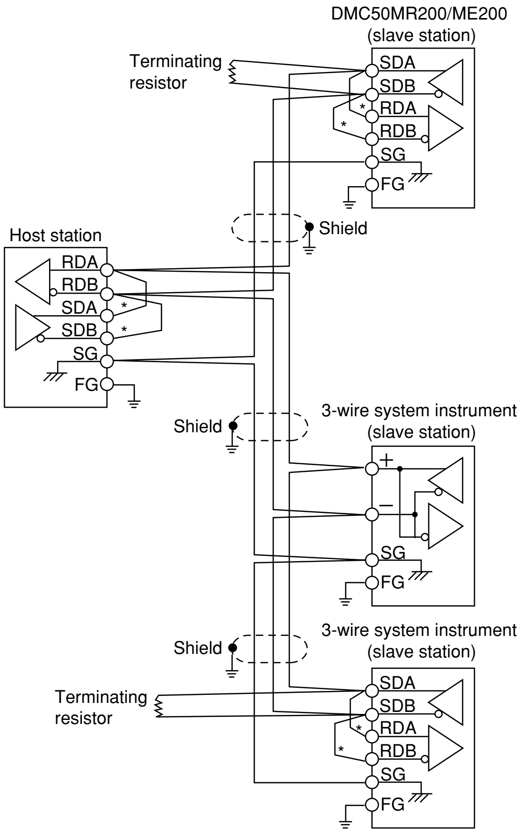


● Connection with 5-wire system instrument



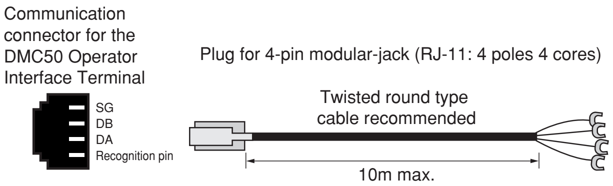
- Connect two terminating resistors of 150Ω±5% 1/2W min. to the instrument on each end of the transmission line.
- Ground the shield FGs at only one end to a single location, not at both ends.
- Use a shielded twisted-pare cable as the RS-485 communication line.

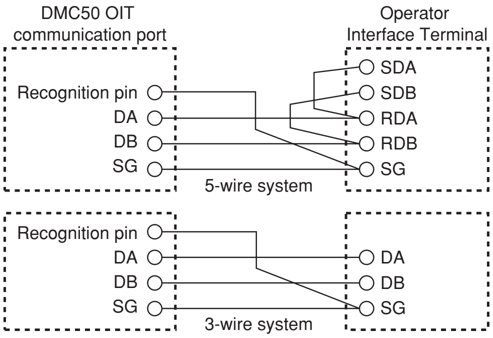
● Connection used with 3-wire system instrument



- Connect two terminating resistors of 150Ω±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-pare cable as the RS-485 communication line.

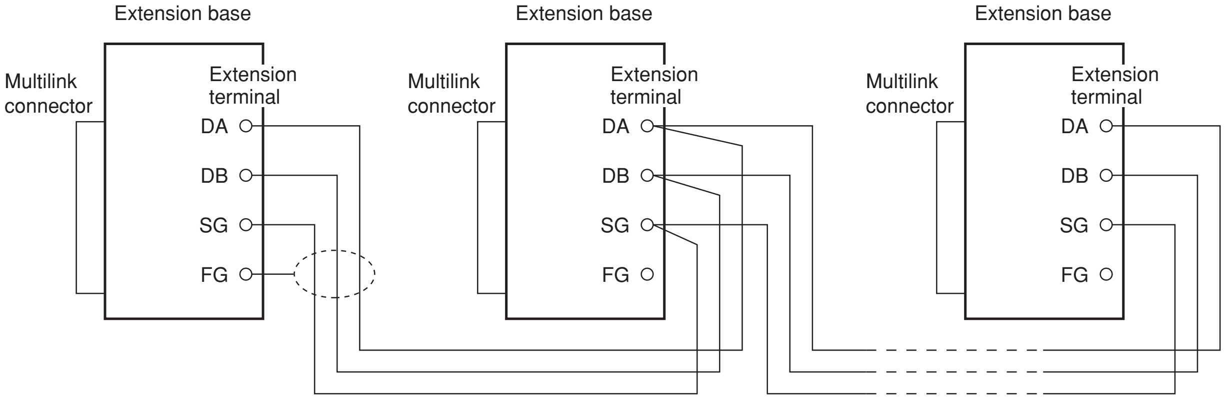
● Communication connections for Operator Interface Terminal


Communication connector for the DMC50 Operator Interface Terminal
SG
DB
DA
Recognition pin
Plug for 4-pin modular-jack (RJ-11: 4 poles 4 cores)
Twisted round type cable recommended
10m max.



- 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-pare cable as the RS-485 communication line.

Program Language/Instructional Terms

■ Quick LD (ladder circuit)

● Contact

Symbol	Name and function
─┤ ├─	Direct contact
─┤\├─	Negated contact
─┤P├─	Contact with positive (rising) edge detection
─┤N├─	Contact with negative (falling) edge detection

● Language construct

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

● Coil

Symbol	Name and function
─( )─	Direct coil
─(\)─	Negated coil
─(S)─	Set action coil
─(R)─	Reset action coil
─(P)─	Coil with positive (rising) edge detection
─(N)─	Coil with negative (falling) edge detection

■ 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

• 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
SYTEM	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_ALRM	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

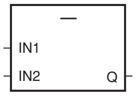
PID_A	Standard PID operation
PID_CAS	Cascade PID operation
Ra_PID	RationaLOOP PID operation
UP_PID	Use-point PID operation

• 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

– (Subtraction)

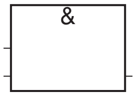


Input parameter
IN1: Integer type | real type
IN2: Integer type | real type, Same data type as IN1

Output parameter
Q: Integer type | real type, IN1 – IN2

Description
1 piece of data is subtracted from another.

& (AND)

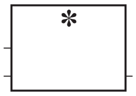


Input parameter
INPUTn: Boolean type, n is 2 to 32.

Output parameter
OUTPUT: Boolean type, AND of input

Description
AND of 2 or more pieces of Boolean type data

* (Multiplication)

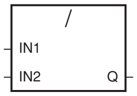


Input parameter
INPUTn: Integer type | real type, n is 2 to 32, All data is of the same data type.

Output parameter
OUTPUT: Integer type | real type, Multiplication of inputs

Description
2 pieces of data are multiplied by each other.

/ (Division)

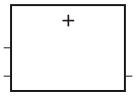


Input parameter
IN1: Integer type | real type
IN2: Integer type | real type, Same data type as IN1

Output parameter
Q: Integer type | real type, IN1 / IN2

Description
1 piece of data is divided by another.

+ (Addition)

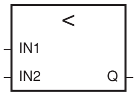


Input parameter
INPUTn: Integer type | real type, n is 2 to 32, All data is of the same data type.

Output parameter
OUTPUT: Integer type | real type, Addition of inputs

Description
2 pieces of data are added together.

< (Less than)

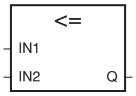


Input parameter
IN1: Integer type | real type |timer type | message type
IN2: Integer type | real type |timer type | message type, Sama data type as IN1

Output parameter
Q: Boolean type, TRUE at IN1 < IN2
FALSE at IN1 ≥ IN2

Description
2 pieces of data are compared.

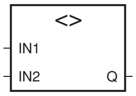
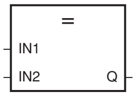
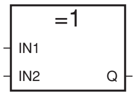
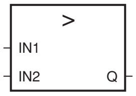
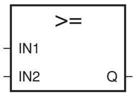
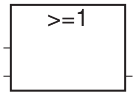
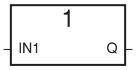
<= (Less or equal)



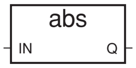
Input parameter
IN1: Integer type | real type | message type
IN2: Integer type | real type | message type, Same data type as IN1

Output parameter
Q: Boolean type, TRUE at IN1 ≤ IN2
FALSE at IN1 > IN2

Description
2 pieces of data are compared.

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

ABS (Absolute value)

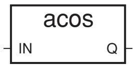


Input parameter
IN: Real type

Output parameter
Q: Real type, Absolute value of IN

Description
Absolute value of real type data

ACOS (Arc cosine)

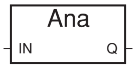


Input parameter
IN: Real type, –1.0 to +1.0

Output parameter
Q: Real type, 0.0 to π (radian unit), acos(IN)

Description
Arc cosine of real type data

ANA
(Convert to integer)

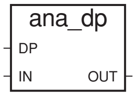


Input parameter
IN: Those excluding integer type data

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.

Description
Data is converted to integer type data.

ANA_DP
(Real to integer conversion with decimal point position designation)

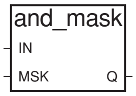


Input parameter
DP: Integer type, Designation of number of digits below the decimal point, –30 to +30
IN: Real type

Output parameter
Q: Integer type, IN x 10^{DP}

Description
Real data is multiplied by 10^{DP} and then rounded off to an integer.

AND_MASK
(Bitwise AND)

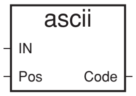


Input parameter
IN: Integer type
MASK: Integer type

Output parameter
Q: Integer type, bitwise AND of IN and MASK

Description
Bitwise AND of 2 pieces of integer type data

ASCII
(Character→ ASCII code conversion)

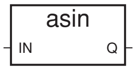


Input parameter
IN: Message type, Message string other than NULL
POS: Integer type, Designated location of a character in a message string, 1 to Len (Len is the IN character string length.)

Output parameter
CODE: Integer type, ASCII code of a designated character, 0 to 255

Description
The designated character in a message string is converted to the corresponding ASCII code.

ASIN (Arc sine)

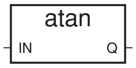


Input parameter
IN: Real type, –1.0 to +1.0

Output parameter
Q: Real type, –π/2 to +π/2 (radian unit) asin(IN)

Description
Arc sine of real type data

ATAN (Arc tangent)

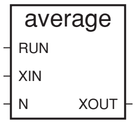


Input parameter
IN: Real type, Range of real numbers

Output parameter
Q: Real type, $-\pi/2$ to $+\pi/2$ (radian unit) atan(IN)

Description
Arc tangent of real type data

AVERAGE (Moving average)

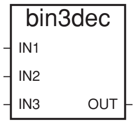


Input parameter
RUN: Boolean type, TRUE=Execution, FALSE=Reset
XIN: Real type, Input data
N: Integer type, Number of samples for calculating the mean value, 1 to 128

Output parameter
XOUT: Real type, Mean value of N units (number of samples) of XIN

Description
Mean value of real type data in the designated sample numbers

BIN3DEC (3-input boolean to integer conversion)

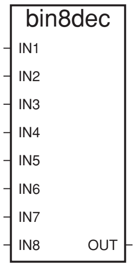


Input parameter
IN1 to IN3: Boolean type

Output parameter
OUT: Integer type, $IN1\times2^0+IN2\times2^1+IN3\times2^2$, 0 to 7

Description
Converts 3 pieces of Boolean type data to a 0 to 7 integer type data.

BIN8DEC (8-input boolean to integer conversion)

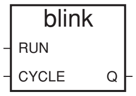


Input parameter
IN1 to IN8: Boolean type

Output parameter
OUT: Integer type, $IN1\times2^0+IN2\times2^1+IN3\times2^2...+IN8\times2^7$, 0 to 255

Description
Converts 8 pieces of Boolean type data to a 0 to 255 integer type data.

BLINK (Boolean type signal blinking)

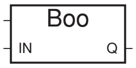


Input parameter
RUN: Boolean type, TRUE=Execution, FALSE=Reset
CYCLE: Timer type, Blinking cycle, CYCLE≥Cycle time setting

Output parameter
Q: Boolean type, Blinking output, If RUN=TRUE, TRUE/FALSE is repeated at half the CYCLE time.
If RUN=FALSE, FALSE.

Description
A blinking signal is generated at each designated cycle.

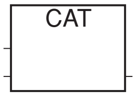
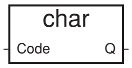
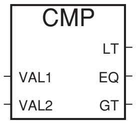
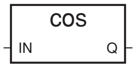
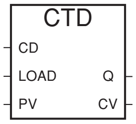
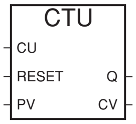
BOO (Convert to boolean)

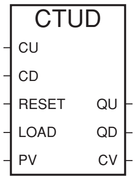
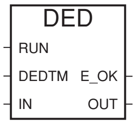
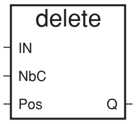
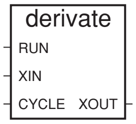
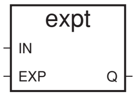
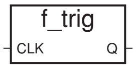


Input parameter
IN: Those excluding Boolean type

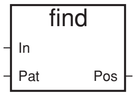
Output parameter
Q: Boolean type, For an integer type IN, FALSE at IN=0 and TRUE at IN≠0
For a real type IN, FALSE at IN=0.0 and TRUE at IN≠0.0
For a timer type IN, FALSE at IN=T#0ms and TRUE at IN≠#0ms
For a character string type IN, FALSE at IN≠'TRUE' and TRUE at IN='TRUE'

Description
Data is converted to Boolean type

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

CTUD **(Up/down counter)**  	**Input parameter** **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 **Output parameter** **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 **Description** Integration of CTU (up-counter) and CTD (down-counter)
DED (Dead time)  	**Input parameter** **RUN**: Boolean type, TRUE=Execution, FALSE=Reset **DEDTM**: Timer type, Dead time **IN**: Real type, Input data **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **OUT**: Real type, IN after dead time **Description** Outputting of real type data after dead time
DELETE **(Message string deletion)**  	**Input parameter** **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 **Output parameter** **Q**: Message type, Message string modified by deletion **Description** The designated characters are deleted from the message string at the designated location.
DERIVATE (Derivative)  	**Input parameter** **RUN**: Boolean type, TRUE=Execution, FALSE=Reset **XIN**: Real type, Input data **CYCLE**: Timer type, Sampling cycle, CYCLE≥cycle time setting **Output parameter** **XOUT**: Real type, Derivative result, The generation of output XOUT is proportional to the changing rate of input XIN. **Description** Derivative of real type data
EXPT **(Exponential function)**  	**Input parameter** **IN**: Real type, Base number **EXP**: Integer type, Exponent number **Output parameter** **Q**: Real type, IN^{EXP} **Description** Exponential operation
F_TRIG **(Falling edge detection)**  	**Input parameter** **CLK**: Boolean type **Output parameter** **Q**: Boolean type, TRUE when CLK is TRUE→FALSE, and FALSE when CLK is other than this. **Description** Falling edge detection of Boolean type data

FIND
(Message string finding)



Input parameter

IN: Message type, Message string
PAT: Message type, String pattern to find

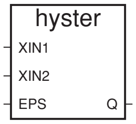
Output parameter

POS: Integer type, Location of found string pattern within message string

Description

The designated message string is found and its location is output.

HYSTER (Hysteresis)



Input parameter

XIN1: Real type, Input data
XIN2: Real type, High limit set point
EPS: Real type, Hysteresis value, EPS≥0.0

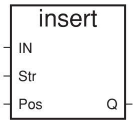
Output parameter

Q: Boolean type

Description

Judgement with hysteresis on whether the real type data has exceeded the high limit value

INSERT
(Message string insertion)



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

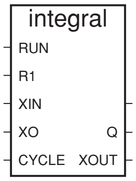
Output parameter

Q: Message type, message string modified by insertion

Description

Insertion of the designated message string at the designated location of the base message string

INTEGRAL (Integral)



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≥Cycle time setting

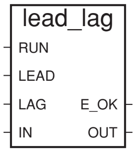
Output parameter

Q: Boolean type, Inverting of R1
XOUT: Real type, Integral result

Description

Integral of real type data

LEAD_LAG (Lead/lag)



Input parameter

RUN: Boolean type, TRUE=Execution, FALSE=Reset
LEAD: Real type, Lead time constatnt, LEAD≥0.0 (unit: s)
LAG: Real type, Lag time constant, LAG≥0.0 (unit: s)
IN: Real type, Input data

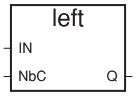
Output parameter

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

Description

Lead/lag calculation of real type data

LEFT
(Left message string extraction)



Input parameter

IN: Message type, Message string
NBC: Integer type, Number of characters to extract, 0<NBC≤IN message string length

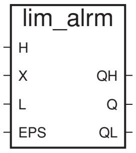
Output parameter

Q: Message type, Extracted message string

Description

The designated characters are extracted from the left hand side of the designated message string.

LIM_ALARM
(Limit alarm)

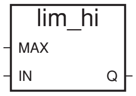


Input parameter
- **H**: Real type, High limit set point
- **X**: Real type, Input data
- **L**: Real type, Low limit set point
- **EPS**: Real type, Hysteresis value, EPS≥0.0

Output parameter
- **QH**: Boolean type, High limit alarm
- **Q**: Boolean type, High limit and low limit alarms
- **QL**: Boolean type, Low limit alarm

Description
Determines using hysteresis whether the real type data has exceeded the high limit value or the low limit value.

LIM_HI (Real type high limit)

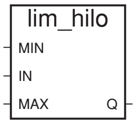


Input parameter
- **MAX**: Real type, High limit value
- **IN**: Real type, Input data

Output parameter
- **Q**: Real type, Value limited by the high limit value, MAX at IN≥MAX
  IN at IN<MAX

Description
The real type data is limited by the high limit value.

LIM_HILO
(Real type high/low limit)

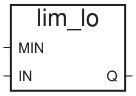


Input parameter
- **MIN**: Real type, Low limit value
- **IN**: Real type, Input data
- **MAX**: Real type, High limit value

Output parameter
- **Q**: Real type, Value limited by the high and low limit values, MIN at IN≤MIN
  IN at MIN<IN<MAX
  MAX at IN≥MAX

Description
The real type data is limited by the high and low limit values.

LIM_LO
(Real type low limit)

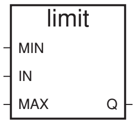


Input parameter
- **MIN**: Real type, Low limit value
- **IN**: Real type, Input data

Output parameter
- **Q**: Real type, Value limited by the low limit value, MIN at IN≤MIN
  IN at IN>MIN

Description
The real type data is limited by the low limit value.

LIMIT (High/low limit)

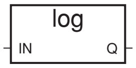


Input parameter
- **MIN**: Integer type, Low limit value
- **IN**: Integer type, Input data
- **MAX**: Integer type, High limit value

Output parameter
- **Q**: Integer type, Value limited by the high and low limit values, MIN at IN≤MIN
  IN at MIN<IN<MAX
  MAX at IN≥MAX

Description
The integer type data is limited by the high an low limit values.

LOG (Common logarithm)

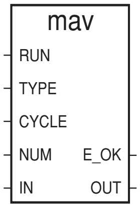


Input parameter
- **IN**: Real type, IN>0.0

Output parameter
- **Q**: Real type, LOG_{10} (IN)

Description
Common logarithm of real type data

MAV (Moving average)

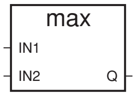


Input parameter
RUN: Boolean type, TRUE=Execution, FALSE=Reset
TYPE: Integer type, Averaging type, 0=Normal, 1=Ignores the maximum value/minimum value
CYCLE: Timer type, Sampling cycle (updates the cycle of a moving average)
NUM: Integer type, Number of samples to calculate the mean value, 1 to 30
IN: Real type, Input data

Output parameter
E_K: Boolean type, TRUE=Normal, FALSE=Error
OUT: Real type, Mean value of NUM samples of IN

Description
Mean value of the real type data in the number of designated samples

MAX (Maximum value)

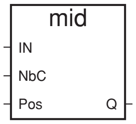


Input parameter
IN1: Integer type
IN2: Integer type

Output parameter
Q: Integer type, IN1 at IN1>IN2, IN2 at IN1≤IN2

Description
The maximum value found in 2 pieces of integer type data

MID (Message string extraction)

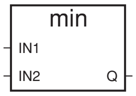


Input parameter
IN: Message type, Message string
NBC: Integer type, Number of characters to extract, Message string length of 0<NBC≤IN
POS: Integer type, Location of character to extract, POS>1

Output parameter
Q: Message type, Extracted message string

Description
The designated characters are extracted from the message string at the designated location.

MIN (Minimum value)

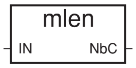


Input parameter
IN1: Integer type
IN2: Integer type

Output parameter
Q: Integer type, IN1 at IN1<IN2, IN2 at IN1≥IN2

Description
The minimum value found in 2 pieces of integer type data.

MLEN (Message string length)

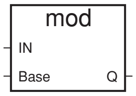


Input parameter
IN: Message type, Message string

Output parameter
NBC: Integer type, Message string length of IN

Description
Outputs the message string length.

MOD (Modulus)

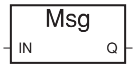


Input parameter
IN: Integer type
BASE: Integer type, BASE>0

Output parameter
Q: Integer type, Residual of IN / BASE

Description
Residual of integer type data

MSG
(Convert to message string)



Input parameter

IN: Those excluding message type

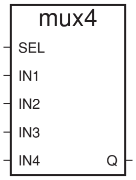
Output parameter

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

Description

Data is converted to message type.

MUX4
(4-input multiplexer)



Input parameter

SEL: Integer type, Select value, 0 to 3
IN1 to IN4: Integer type

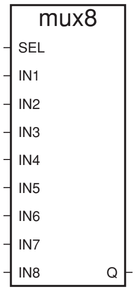
Output parameter

Q: Integer type, Selected IN, IN1 at SEL=0
IN2 at SEL=1
IN3 at SEL=2
0 at other than 1 to 3

Description

Selects one out of 4 pieces of integer type data.

MUX8
(8-input multiplexer)



Input parameter

SEL: Integer type, Select value, 0 to 7
IN1 to IN8: Integer type

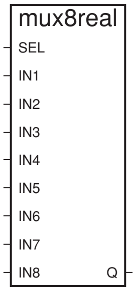
Output parameter

Q: Integer type, Selected IN, IN1 at SEL=0
IN2 at SEL=1
•
•
•
IN8 at SEL=7
0 at other than 0 to 7

Description

Selects one out of 8 pieces of integer type data.

MUX8REAL
(Real type 8-input multiplexer)



Input parameter

SEL: Integer type, Select value, 0 to 7
IN1 to IN8: Real type

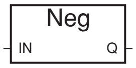
Output parameter

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

Description

Selects one out of 8 real type data.

NEG (Sign change)



Input parameter

IN: Integer type | real type

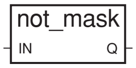
Output parameter

Q: Integer type | real type, –IN

Description

Sign change of data

NOT_MASK
(Bitwise inversion)



Input parameter

IN: Integer type

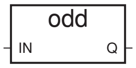
Output parameter

Q: Integer type, bitwise inversion

Description

Bitwise inversion of integer type data

ODD (Odd parity)

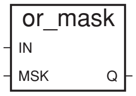


Input parameter
IN: Integer type

Output parameter
Q: Boolean type, TRUE when IN is odd,
FALSE when IN is even.

Description
Determines whether the integer type data is odd or even.

OR_MASK (bitwise OR)

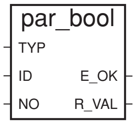


Input parameter
IN: Integer type
MASK: Integer type

Output parameter
Q: Integer type, bitwise OR of IN and MASK

Description
Bitwise OR of 2 integer type data

PAR_BOOL (Read Boolean type parameter)

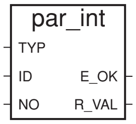


Input parameter
TYPE: Integer type, Parameter type ID
ID: Integer type, Group ID
NO: Integer type, Item ID

Output parameter
E_OK: Boolean type, TRUE=Normal, FALSE=Error
R_VAL: Boolean type, Read-out data

Description
A DMC50's dedicated parameter is read out as Boolean type data.

PAR_INT (Read Integer type parameter)

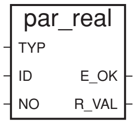


Input parameter
TYPE: Integer type, Parameter type ID
ID: Integer type, Group ID
NO: Integer type, Item ID

Output parameter
E_OK: Integer type, TRUE=Normal, FALSE=Error
R_VAL: Integer type, Read-out data

Description
A DMC50's dedicated parameter is read out as integer type data.

PAR_REAL (Read Real type parameter)

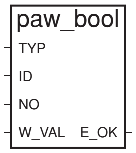


Input parameter
TYPE: Integer type, Parameter type ID
ID: Integer type, Group ID
NO: Integer type, Item ID

Output parameter
E_OK: Boolean type, TRUE=Normal, FALSE=Error
R_VAL: Real type, Read-out data

Description
A DMC50's dedicated parameter is read out as real type data.

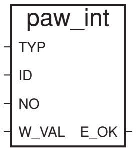
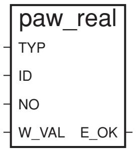
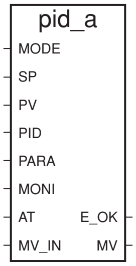
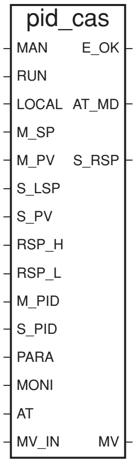
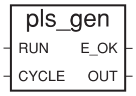
PAW_BOOL (Write Boolean type parameter)



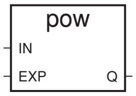
Input parameter
TYPE: Integer type, Parameter type ID
ID: Integer type, Group ID
NO: Integer type, Item ID
W_VAL: Boolean type, data to be written

Output parameter
E_OK: Boolean type, TRUE=Normal, FALSE=Error

Description
Boolean type data is written into a DMC50's dedicated parameter.

PAW_INT **(Write Integer type parameter)**  	**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 **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **Description** Integer type data is written into a DMC50's dedicated parameter.
PAW_REAL **(Write Real type parameter)**  	**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 **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **Description** The real type data is written into a DMC50's dedicated parameter.
PID_A **(Standard PID operation)**  pid_a MODE SP PV PID PARA MONI AT E_OK MV_IN MV	**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 **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **MV**: Real type, Manipulated variable **Description** A series form PID operation of which derivative action applied on error
PID_CAS **(Cascade PID operation)**  pid_cas MAN E_OK RUN LOCAL AT_MD M_SP M_PV S_RSP S_LSP S_PV RSP_H RSP_L M_PID S_PID PARA MONI AT MV_IN MV	**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 **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 **Description** PID operation for cascade control
PLS_GEN (Pulse generator)  	**Input parameter** **RUN**: Boolean type, TRUE=Execution, FALSE=Reset **CYCLE**: Timer type, CYCLE≥Cycle time setting **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **OUT**: Boolean type, Pulse output **Description** A one cycle time pulse is generated at each designated cycle.

POW (Exponential function)

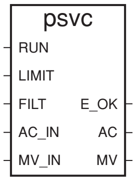


Input parameter
IN: Real type, Base number
EXP: Real type, Exponent number

Output parameter
Q: Real type, IN^{EXP}

Description
Exponential operation of real type data

PSVC (Power supply voltage compensation)

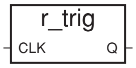


Input parameter
RUN: Boolean type, TRUE=Execution, FALSE=Reset
LIMIT: Real type, Compensation range, 0.00 to 50.0% (Normally 20.0% is sufficient)
Executing the compensation at AC_IN=100%±LIMIT
FILT: Real type, Filter constant for AC_IN, 0.0 to 2000.0s
AC_IN: Real type, AC voltage input, an AUX_IN (AC voltage) input
MV_IN: Real type, Manipulated variable before compensation

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

Description
Heater power voltage (AUX_IN) fluctuations are monitored and the manipulated variable output is compensated for.

R_TRIG (Rising edge detection)

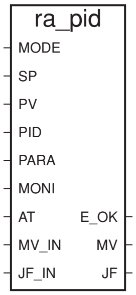


Input parameter
CLK: Boolean type

Output parameter
Q: Boolean type, TRUE when CLK is FALSE→TRUE, and FALSE otherwise.

Description
Rising edge detection of Boolean type data

Ra_PID (RationaLOOP PID operation)

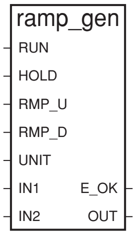


Input parameter
MODE: Boolean type, TRUE=MANUAL mode, FALSE=AUTO mode
SP: Real type, SP (industrial unit)
PV: Real type, PV (industrial unit)
PID: Integer type, Designates the group ID of Ra_PID constants
PARA: Integer type, Designates the group ID of Ra_PID options
MONI: Integer type, Designates the 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

Output parameter
E_OK: Boolean type, TRUE=Normal, FALSE=Error
MV: Real type, Manipulated variable
JF: Real type, Just-FiTTER output

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

RAMP_GEN (Ramp generator)

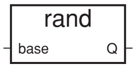


Input parameter
RUN: Boolean type, TRUE=Execution, FALSE=Reset
HOLD: Boolean type, TRUE=Holding the ramp action, FALSE=Continuing the ramp action (canceling the hold)
RMP_U: Real type, Ramp-Up variable
RMP_D: Real type, Ramp-Down variable
UNIT: Integer type, Ramp time unit, 0: Ramp variable/s, 1: Ramp variable/min, 2: Ramp variable/h
IN1: Real type, Starting point of ramp action
IN2: Real type, Target point of ramp action

Output parameter
E_OK: Boolean type, TRUE=Normal, FALSE=Error
OUT: Real type, Ramp data

Description
A ramp of real data generated from IN1 up to the target value IN2.

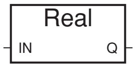
RAND (Random value)



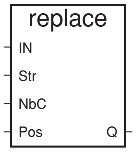
Input parameter
BASE: Integer type, Maximum value of random value BASE>1

Output parameter
Q: Integer type, Random value 0 to BASE _ 1

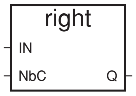
Description
Random value generation of integer type data

<table>
<tr><td><b>REAL</b>
<b>(Convert to real)</b>
 </td><td><b>Input parameter</b>
<b>IN</b>: Boolean type | Integer type | Timer type

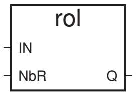
<b>Output parameter</b>
<b>Q</b>: Real type, For a boolean type IN, 0.0 at IN=FALSE and 1.0 at IN=TRUE.
For an integer type IN, IN.
For a timer type IN, mSec numerical value.

<b>Description</b>
Data is converted to real type data.</td></tr>
<tr><td><b>REPLACE</b>
<b>(Message string replacement)</b>
 </td><td><b>Input parameter</b>
<b>IN</b>: Message type, Base message string
<b>STR</b>: Message type, Message string to insert
<b>NBC</b>: Integer type, Number of characters to delete prior to insert
<b>POS</b>: Integer type, Begin location for insertion of message string, POS>1

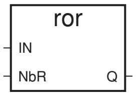
<b>Output parameter</b>
<b>Q</b>: Message type, Message string modified by replacement

<b>Description</b>
Message for the number of designated characters is deleted at the designated character location, and then the designated mesage string is inserted into the base string for replacement.</td></tr>
<tr><td><b>RIGHT</b>
<b>(Right message string extraction)</b>
 </td><td><b>Input parameter</b>
<b>IN</b>: Message type, Message string
<b>NBC</b>: Integer type, Number of characters to extract, Message string length of 0<NBC≤IN

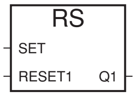
<b>Output parameter</b>
<b>Q</b>: Message type, Message string extracted

<b>Description</b>
The designated characters are extracted from the right hand side of the designated message string.</td></tr>
<tr><td><b>ROL (Left rotation)</b>
 </td><td><b>Input parameter</b>
<b>IN</b>: Integer type
<b>NBR</b>: Integer type, Number of bits for left rotation, 1 to 31

<b>Output parameter</b>
<b>Q</b>: Integer type, Result of left rotation for IN

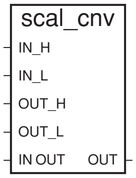
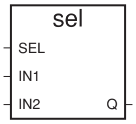
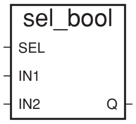
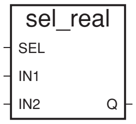
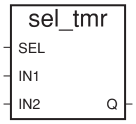
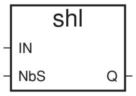
<b>Description</b>
Integer type data bits are rotated left.</td></tr>
<tr><td><b>ROR (Right rotation)</b>
 </td><td><b>Input parameter</b>
<b>IN</b>: Integer type
<b>NbR</b>: Integer type, Number of bits for right rotation, 1 to 31

<b>Output parameter</b>
<b>Q</b>: Integer type, Result of right rotation for IN

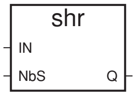
<b>Description</b>
Integer type data bits are rotated right.</td></tr>
<tr><td><b>RS</b>
<b>(Reset dominant bistable)</b>
 </td><td><b>Input parameter</b>
<b>SET</b>: Boolean type
<b>RESET1</b>: Boolean type

<b>Output parameter</b>
<b>Q</b>: Boolean type, TRUE at SET=TRUE,
In case of RESET1=TRUE, FALSE (reset dominant)

<b>Description</b>
Reset dominant bistable latch</td></tr>
</table>

SCAL CNV **(Scale conversion)** 	**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 **Output parameter** **OUT**: Real type, Output data, OUT = (IN–IN_L) / (IN_H–IN_L) × (OUT_H–OUT_L) + OUT_L **Description** Scaling operation of real type data
SEL (Binary selection) 	**Input parameter** **SEL**: Boolean type, For selection **IN1**: Integer type **IN2**: Integer type **Output parameter** **Q**: Integer type, IN1 at SEL=FALSE, IN2 at SEL=TRUE **Description** Selects one out of 2 pieces of integer type data.
SEL_BOOL **(Boolean type binary selection)** 	**Input parameter** **SEL**: Boolean type, For selection **IN1**: Boolean type **IN2**: Boolean type **Output parameter** **Q**: Boolean type, IN1 at SEL=FALSE IN2 at SEL=TRUE **Description** Selects one out of 2 pieces of Boolean type data.
SEL_REAL **(Real type binary selection)** 	**Input parameter** **SEL**: Boolean type, For selection **IN1**: Real type **IN2**: Real type **Output parameter** **Q**: Real type, IN1 at SEL=FALSE, IN2 at SEL=TRUE **Description** Selects one out of 2 pieces of real type data
SEL_TMR **(Timer type binary selection)** 	**Input parameter** **SEL**: Boolean type, For selection **IN1**: Timer type **IN2**: Timer type **Output parameter** **Q**: Timer type, IN1 at SEL=FALSE, IN2 at SEL=TRUE **Description** Selects one out of 2 pieces of timer data.
SHL (Left shift) 	**Input parameter** **IN**: Integer type **NBS**: Integer type, Number of bits for left shift 1 to 31 **Output parameter** **Q**: Integer type, Result of left shift for IN (0 inserted into LSB) **Description** Left bit shift of integer type data

SHR (Right shift)

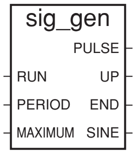


Input parameter
IN: Integer type
NBS: Integer type, Number of bits for right shift, 1 to 31

Output parameter
Q: Integer type, Result of right shift for IN (save value copied to MSB)

Description
Right bit shift of integer type data

SIG_GEN (Signal generator)

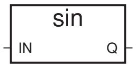


Input parameter
RUN: Boolean type, TRUE=Execution, FALE=Reset
PERIOD: Timer type, Sampling time (pulse width), PERIOD≥cycle time setting
MAXIMUM: Integer type, Maximum count value, MAXIMUM≥0

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.)

Description
Generation of 3 signals (pulse, up-counter and sine waveform signal)

SIN (Sine)

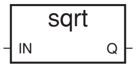


Input parameter
IN: Real type, Range of real values (radian unit)

Output parameter
Q: Real value, –1.0 to +1.0 Sin(IN)

Description
Sine of real type data

SQRT (Square root)

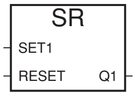


Input parameter
IN: Real type, IN≥0.0

Output parameter
Q: Real type, $\sqrt{\text{IN}}$

Description
Square root of real type data

SR (Set dominant bistable)

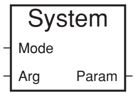


Input parameter
SET1: Boolean type
RESET: Boolean type

Output parameter
Q: Boolean type, TRUE (set dominant) at SET1=TRUE
FALSE at RESET=TRUE

Description
Set dominant bistable latch

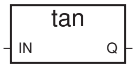
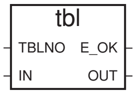
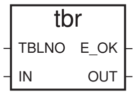
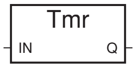
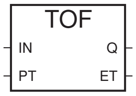
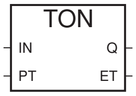
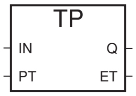
SYSTEM (Access to system)

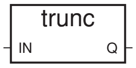
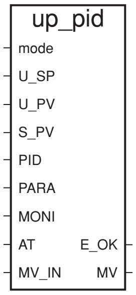
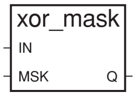
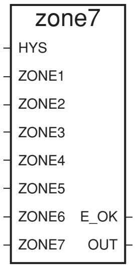


Input parameter
MODE: Integer type, Command
ARG: Integer type, 0 at MODE≠SYS_TWRITE
Timer type, New cycle time at MODE=SYS_TWRITE

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.

Description
Read out cycle time, etc

TAN (Tangent) 	**Input parameter** **IN**: Real type, ± π/2 × a (n = 1,3,5, ...), Radian unit **Output parameter** **Q**: Real type, tan(IN) **Description** Tangent of real type data
TBL (Linearization table lookup) 	**Input parameter** **TBLND**: Integer type, Sets the group ID of TBL/TBR setup parameters **IN**: Real type, Input data **Output parameter** **OUT**: Real type, Output data **Description** Piecewise linearization table lookup for input data
TBR (Linearization table reverse lookup) 	**Input parameter** **TBLND**: Integer type, Sets the group ID of TBL/TBR setup parameters **IN**: Real type, Input data **Output parameter** **OUT**: Real type, Output data **Description** Piecewise linearization table reverse lookup for input data.
TMR (Convert to timer) 	**Input parameter** **IN**: Integer type, real type **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) **Description** Data is converted into timer type data.
TOF (OFF delay timer) 	**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 **Output parameter** **Q**: Boolean, FALSE at ET=PT **ET**: Timer type, Elapsed timer value **Description** Stops the timer at a specified time after falling edge detection.
TON (ON delay timer) 	**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 **Output parameter** **Q**: Boolean, TRUE at ET=PT **ET**: Timer type, Elapsed timer value **Description** Starts the timer at a specified time after rising edge detection.
TP (Pulse timer) 	**Input parameter** **IN**: Boolean type, Timer start at rising edge detection **PT**: Timer type, Preset timer value **Output parameter** **Q**: Boolean type, TRUE at ET<PT during timer movement **ET**: Timer type, Elapsed timer value **Description** Starts the timer upon rising edge detection for a specified period of time.

TRUNC **(Truncate the decimal part)**  	**Input parameter** **IN**: Real type **Output parameter** **Q**: Real type, Integer portion of IN **Description** The number of real type data is rounded to nearest integer value toward zero.
UP_PID **(Use-point PID operation)**  up_pid mode U_SP U_PV S_PV PID PARA MONI AT E_OK MV_IN MV	**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 **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **MV**: Real type, Manipulated variable **Description** 2-input and one-output PID operation for disturbance rejection
XOR_MASK **(Bitwise XOR)**  	**Input parameter** **IN**: Integer type **MASK**: Integer type **Output parameter** **Q**: Integer type, Bitwise exclusive OR of IN and MASK **Description** Bitwise exclusive OR of 2 pieces of integer type data
ZONE7 (Zone selector)  zone7 HYS ZONE1 ZONE2 ZONE3 ZONE4 ZONE5 ZONE6 E_OK ZONE7 OUT	**Input parameter** **NYS**: Real type, Zone-to-zone hysteresis HYS≥0.0 **ZONE1 to 7**: Zone boundary values ZONE1< ZONE2< .... <ZONE7 **IN**: Real type, Input data **Output parameter** **E_OK**: Boolean type, TRUE=Normal, FALSE=Error **OUT**: Real type, Zone value (0 to 7) OUT=n at ZONE(n)≤IN<ZONE(n+1) **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).

■ ST Configuration Text

● Basic instructions

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

● 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.

⚠ RESTRICTIONS ON USE

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

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

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

Specifications are subject to change without notice.

azbil

Yamatake Corporation
Advanced Automation Company

1-12-2 Kawana, Fujisawa
Kanagawa 251-8522 Japan
URL: http://www.azbil.com

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

Printed on recycled paper.

(07)