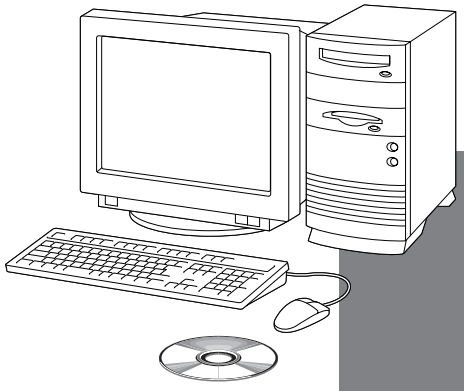
YAMATAKE

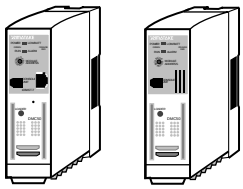
No. CP-SP-1093E

Module Type Controller DMC50

User's Manual

Communications Connection





Thank you for purchasing the Module Type Controller DMC50.
This manual contains information for ensuring correct use of DMC50's communication facilities.
This manual should be read by those who design and maintain equipment that use DMC50 and its communication facilities.
Be sure to keep this manual nearby for handy reference.

Yamatake Corporation

RESTRICTIONS ON USE

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

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

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

REQUEST

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

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

Considerable effort has been made to ensure that this User's Manual is free from inaccuracies and omissions.
If you should find any inaccuracies or omissions, please contact Yamatake Corporation.

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

SAFETY PRECAUTIONS

■ About Icons

Safety precautions are for ensuring safe and correct use of this product, and for preventing injury to the operator and other people or damage to property. You must observe these safety precautions. The safety precautions described in this manual are indicated by various icons.
As the following describes the icons and their meanings, be sure to read and understand the descriptions before reading this manual:

⚠WARNING	Warnings are indicated when mishandling this product might result in death or serious injury to the user.
⚠CAUTION	Cautions are indicated when mishandling this product might result in minor injury to the user, or only physical damage to this product.

■ Examples

	Triangles warn the user of a possible danger that may be caused by wrongful operation or misuse of this product. These icons graphically represent the actual danger. (The example on the left warns the user of the danger of electrical shock.)
	White circles with a diagonal bar notify the user that specific actions are prohibited to prevent possible danger. These icons graphically represent the actual prohibited action. (The example on the left notifies the user that disassembly is prohibited.)
	Black filled-in circles instruct the user to carry out a specific obligatory action to prevent possible danger. These icons graphically represent the actual action to be carried out. (The example on the left instructs the user to remove the plug from the outlet.)

⚠WARNING

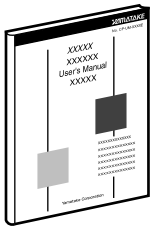
	Before removing / mounting or wiring the DMC50, be sure to turn the source power OFF. Doing so might cause electric shock.
	Do not touch any terminals or metal parts connected to the DMC50 when mounting or dismounting this instrument. Doing so might cause electric shock.
	Do not disassemble the DMC50. Doing so might cause electric shock or faulty operation.
	Earth the FG terminal with a ground resistance of a maximum of 100 Ω before connecting it to the measurement target or external control circuits. Failure to do so might cause electric shock or fire.
	Do not touch electrically charged parts such as the power terminals. Doing so might cause electric shock.
	Provide safety measures externally for this instrument so that the entire system can operate safely even in the event of this instrument malfunctioning or if any abnormality as a result of some external cause occurs. Faulty operation of this instrument could result in serious injury or accident.

⚠CAUTION

	Use the DMC50 within the operating ranges recommended in the specifications (temperature, humidity, voltage, vibration, shock, mounting direction, atmosphere, etc.). Failure to do so might cause fire or faulty operation.
	Do not block ventilation holes. Doing so might cause fire or faulty operation.
	Do not allow lead clippings, chips or water to enter the DMC50 case. Doing so might cause fire or faulty operation.
	Wire the DMC50 properly according to predetermined standards. Also wire the DMC50 using designed power leads according to recognized installation methods. Failure to do so might cause electric shock, fire or faulty operation.
	Inputs to the current input terminals (2•4, 6•8, 10•12, 14•16) on this instrument should comply within the electrical current limit stated in the specifications irrespective of the operating conditions of this instrument. Failure to do so might cause fire or faulty operation.
	Firmly tighten the terminal screws and mounting screw at the torque listed in the specifications. Insufficient tightening of terminal screws might cause electric shock or fire.
	Do not use unused terminals on the DMC50 as relay terminals. Doing so might cause electric shock, fire or faulty operation.
	We recommend attaching the terminal cover after wiring the DMC50. Failure to do so might cause electric shock.

The Role of This Manual

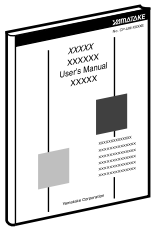
Seven different manuals in total are available for DMC50. Read each manual according to your specific requirements. Following is a brief outline of each of the manuals.
To obtain manuals, contact Yamatake Corporation or your Yamatake dealer.



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

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

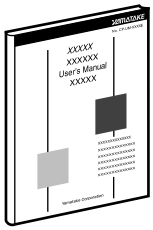
This manual consists of two parts, one for the control module and the other for the communication module. It describes the installation, wiring, specifications, and hardware troubleshooting of those controller.



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

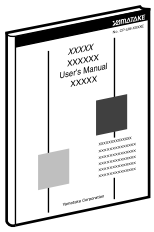
The user who operates DMC50 for the first time must read this manual thoroughly. This manual is intended to help the user understand the overview of the controller operation and basic operating procedures.

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



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

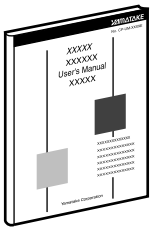
This Manual.
The user who uses the communication facilities of DMC50 must read this manual thoroughly. This manual describes the communication facilities of this controller, such as CPL communication and Ethernet communication.



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

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

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

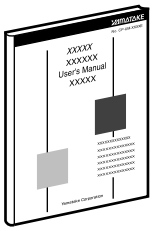


Module Type Controller DMC50 User's Manual " Application Developer's Guide" Manual No. CP-SP-1134E

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

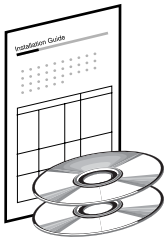
This manual describes the pattern function blocks, utilization of ISaGRAF, and application examples.

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



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

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



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

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

Organization of This User's Manual

This manual is organized as follows:

Chapter 1. OVERVIEW

This chapter summarizes the communication features of this controller and communication system configurations using the communication ports of this controller.

Chapter 2. CONNECTIONS AND COMMUNICATIONS

This chapter describes how to make connections and settings for communications through this controller. Additionally, this chapter also describes actual communications connections and operations with smart loader package SLP-D50.

Chapter 3. ETHERNET COMMUNICATION

This chapter describes the specifications and sample programs for communications with this controller over Ethernet.

Chapter 4. CPL COMMAND REFERENCE

This chapter describes the communication format, communication troubleshooting, and details of each command of the Yamatake’s CPL protocol communicating with this controller.

Chapter 5. CONNECTIONS WITH EST-Z Series

This chapter describes how to connect Yamatake's smart terminal EST-Z Series to this controller.

Contents

Conventions Used in This Manual

The following conventions are used in this manual:

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

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

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

COM module : This indicates the communication module (ME20X, MR20X).

CTRL module : This indicates the control module (CS40X, CS20X, CH40X, CH20X).

Loader : This indicates the smart loader package SLP-D50.

[OK] button : This indicates the selection button shown on the personal computer screen.

[Communication path]
: This indicates the message or menu shown on the personal computer screen.

>> : This indicates the contents shown on the personal computer or unit as a result of operation or unit status after completion of operation.

[F4] key : This indicates the key on the keyboard.

TCP/IP Setup window
: This indicates the name of window shown on the personal computer screen.

Chapter 1. OVERVIEW

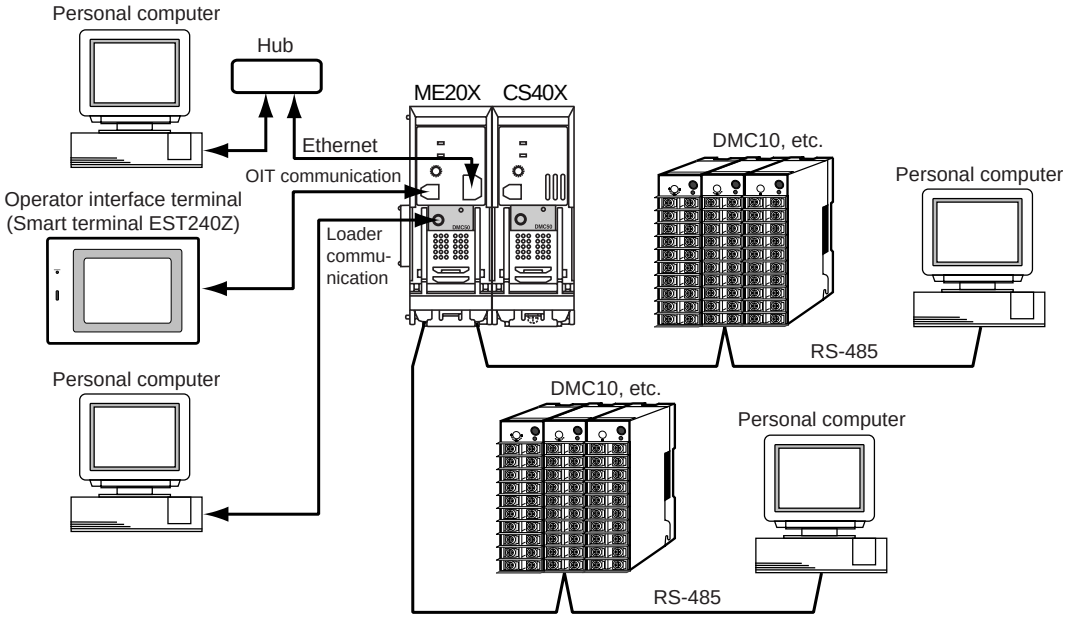
With a communication module (hereafter referred to as COM module) connected, remote monitoring of DMC50 over RS-485 or Ethernet is made possible.

■ Features

The communication facilities of DMC50 provide the following features:

- **Allowing various connection methods**

  RS-485 port(s) (two ports for ME20X and one port for MR20X), an OIT communication port, a loader port, and an Ethernet port (only for ME20X) are provided.
  Therefore, DMC50 can be connected from multiple host units by means of various methods and/or through various routes.



- **Enabling Aaccess from Ethernet, a standard network**

  Use of ME200 makes it possible to construct a system including commercially available Ethernet units.
  Additionally, DMC50 can be accessed from an existing Ethernet network, allowing efficient utilization of existing resources.

- **Supporting TCP/IP, a standard communication protocol**

  A general purpose personal computer or workstation can easily communicate with DMC50 by an application program created with commercially available program development software.

- **Providing an OIT port on every module**

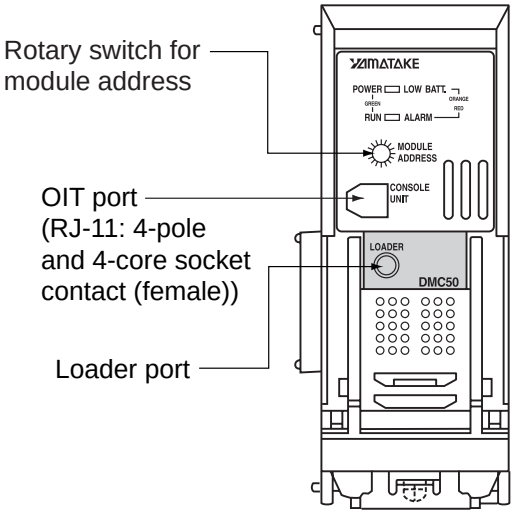
  All control modules (hereafter referred to as CTRL module) are equipped with an OIT port. This enables one-to-one connection from an operator interface terminal without use of COM module.

- **Complying with CPL protocol**

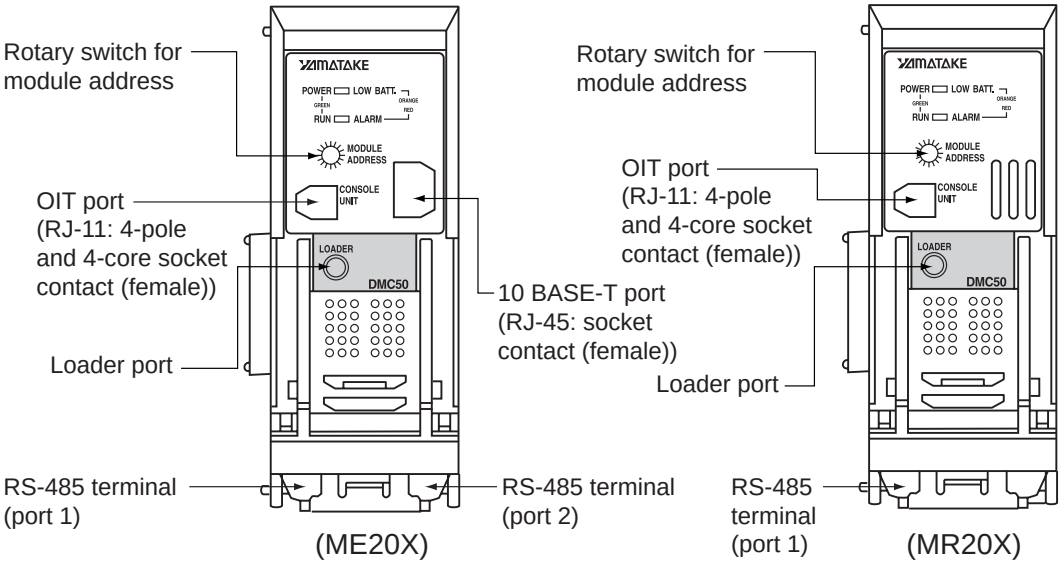
  DMC50 complies with the CPL (Controller Peripheral Link), Yamatake's host communication protocol. Therefore, DMC50 can be easily accessed from an operator interface terminal.

■ Communication ports

The following describes the names and positions of the communication ports provided on DMC50:



CTRL module



COM module

! Handling Precautions

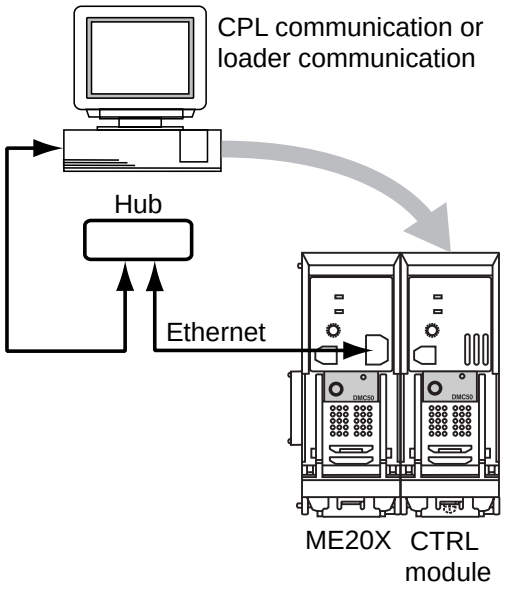
The loader port and OIT port cannot be used at the same time. If the plug for the loader communication and the connector for the OIT communication are connected at the same time, the communication will not work correctly.

■ Connection ports and network configurations

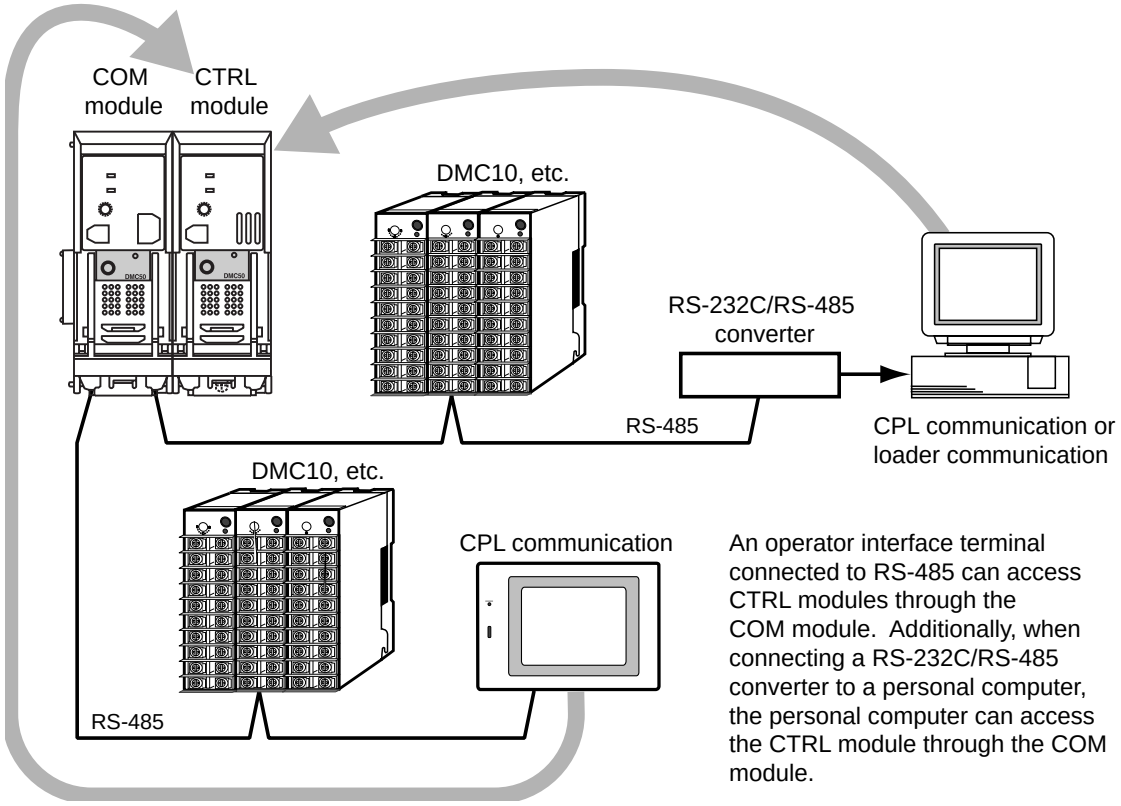
The following shows examples of network system configurations with DMC50:

● Ethernet connections

Workstations and personal computers connected to Ethernet can access the CTRL module through ME20X.



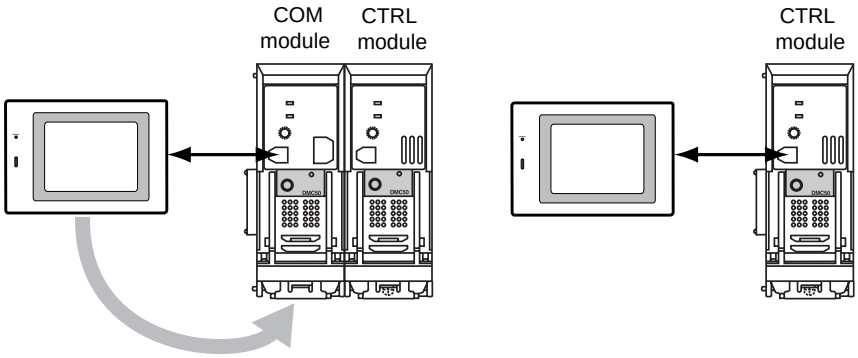
● RS-485 connections



An operator interface terminal connected to RS-485 can access CTRL modules through the COM module. Additionally, when connecting a RS-232C/RS-485 converter to a personal computer, the personal computer can access the CTRL module through the COM module.

- **OIT port connections**

An operator interface terminal connected to the OIT port can access the CTRL module through the COM module. Additionally, an operator interface terminal port can directly access the CTRL module through the own operator interface terminal port on the CTRL module.

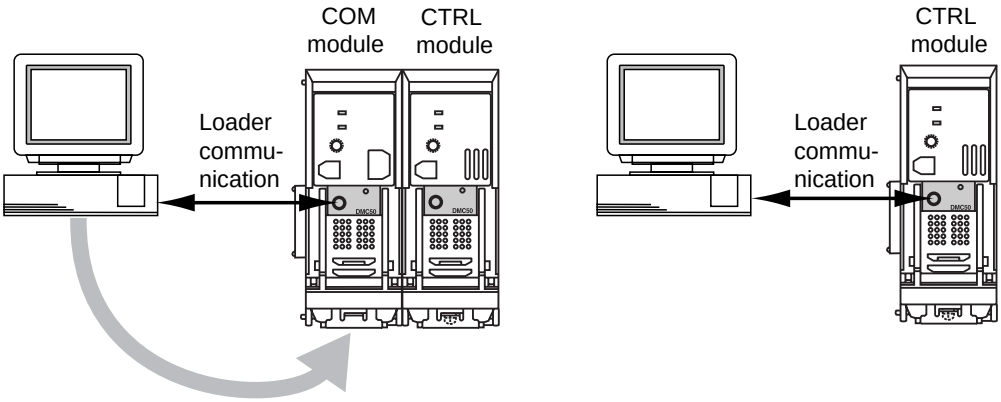


Handling Precautions

The OIT port does not support multidrop. Do not connect the OIT port to a multidrop network with the RJ11 connector customized for multidrop. Doing so might cause incorrect communication.

- **Loader port connections**

SLP-D50 connected to the loader port can access the CTRL module through the COM module. Additionally, SLP-D50 can directly access the CTRL module through the own loader port on the CTRL module.



Handling Precautions

Performing multiple loader communications or CPL communications at the same time with indirect connections can cause a timeout error due to internal processing load.

If timeout errors occur while monitoring with SLP-D50, connect SLP-D50 to the loader port of the target controller directly and stop the communication from other communication ports.

■ Brief summary of communication facilities

The following shows the communication facilities of DMC50:

- Loader communication
- CPL communication
- Clock synchronization

● Loader communication

For loader communication, the protocol specially designed for loader communication is used to connect SLP-D50 to DMC50. The loader communication can be made through the loader port of each controller, or RS-485 or Ethernet port of the COM module.

● CPL communication

This CPL communication is used to connect a personal computer or an operator interface terminal supporting to the CPL protocol to DMC50.
The CPL communication can be made through the OIT port of each controller, or RS-485 or Ethernet port of the COM module.

● Clock synchronization

When Date and Time Setup of the COM module is changed through the CPL communication or loader communication, the clock of the CTRL modules linked through the back-plane are automatically adjusted to that of the COM module.

! Handling Precautions

After adjusting clocks using the clock synchronization facility, clock deviations among modules will be ±1 sec.

● Ports and protocols

The following shows the relationship between communication ports and their compatible protocols:

Port name	Protocol
Loader port	Loader communication
OIT port	CPL communication
RS-485 port 1	CPL/Loader communication
RS-485 port 2	CPL/Loader communication
Ethernet IP port 1	Loader communication
Ethernet IP port 2	Loader communication
Ethernet IP port 3	CPL communication
Ethernet IP port 4	CPL communication

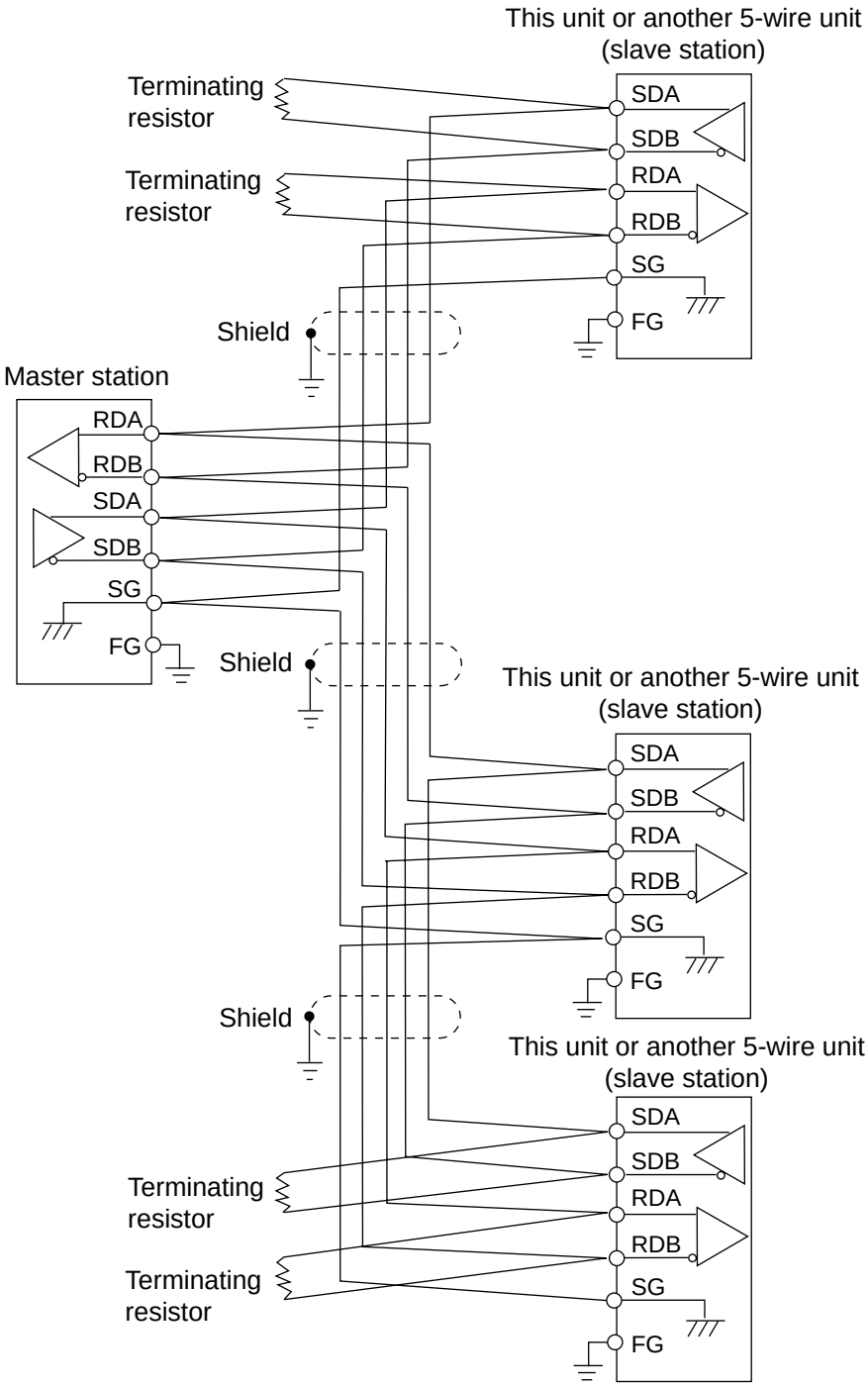
Chapter 2. CONNECTIONS AND COMMUNICATIONS

2 - 1 Cable connections

Cables should be connected as shown below.

● **5-wire RS-485 connections**

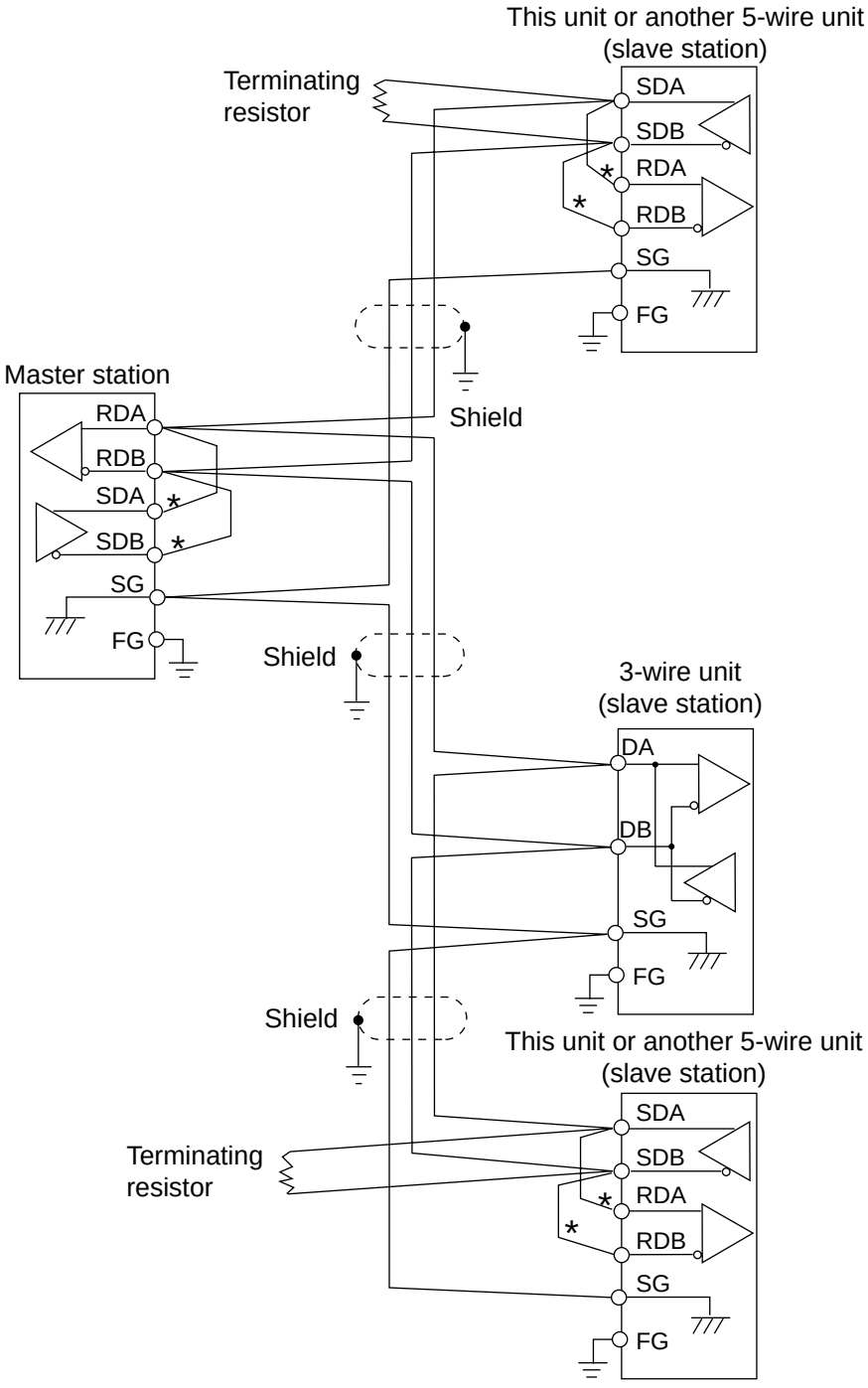
To connect from a 5-wire unit, make the wiring as shown in the Figure below.



Connect a terminating resistor with a resistance of 150Ω ± 5% and a power consumption of 1/2W or more to both ends of the communication path.
Do not connect both ends of the shield to FG (frame ground). Connect only one end of the shield to FG (frame ground).

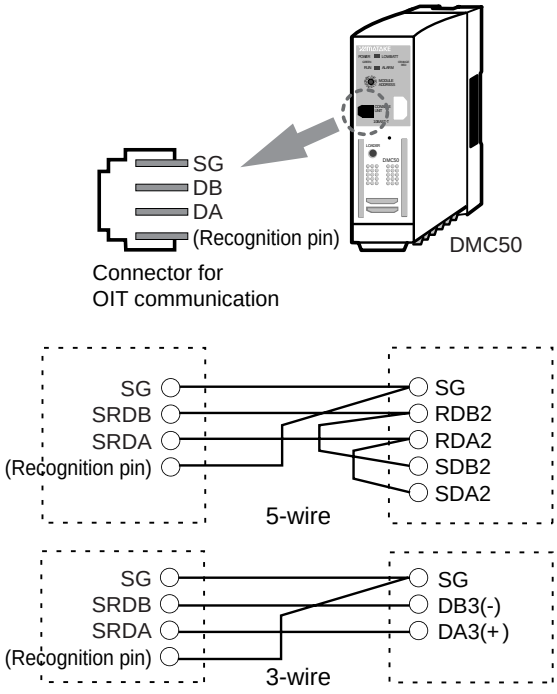
● 3-wire RS-485 connections

To connect to a 3-wire unit, make the wiring as shown in the Figure below.



Connect a terminating resistor with a resistance of 150Ω ± 5% and a power consumption of 1/2W or more to both ends of the communication path.
Do not connect both ends of the shield to FG (frame ground). Connect only one end of the shield to FG (frame ground).
Make the wiring marked with asterisks (*) outside the controller.

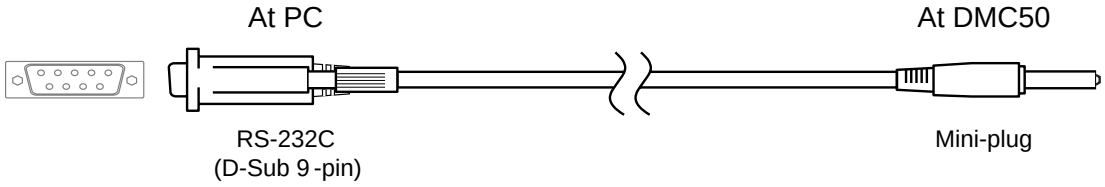
● **Connections through OIT port**



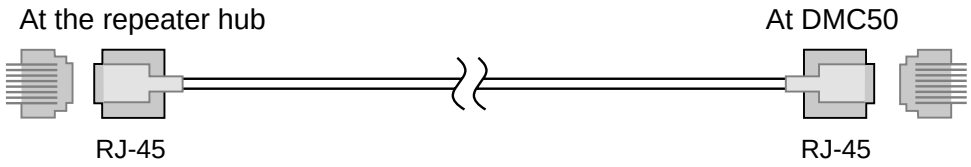
! Handling Precaution

Always connect the recognition pin to SG.

● **Connections through loader port (Loader cable: 81440793-001)**



● **Connections through Ethernet port (10BASE-T)**



! Handling Precaution

When making connections through the 100BASE-T/10BASE-T auto-sensing/auto-negotiation switching hub, this might cause a faulty connection.

Do not use the 100BASE-T/10BASE-T auto-sensing / auto-negotiation switching hub, but use the repeater hub.

Do not make direct connection between units with a 10BASE-T cross-cable. Doing so might cause incorrect recognition.

2 - 2 Communication settings

■ Communication settings for RS-485 port

The following shows the possible communication settings for the RS-485 port 1 and 2:

Transmission speed	9 6 00 bps, 19 200 bps or 38400 bps
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits
Flow control	XON control/No S-parameter

■ Communication settings for OIT port

The following shows the possible communication settings for the OIT port:

Transmission speed	9 6 00 bps, 19 200 bps or 38400 bps
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits
Flow control	XON control / No S-parameters

■ Communication settings for loader port

The following shows the possible communication settings for the loader port. The loader port settings are not modifiable:

Transmission speed	9 6 00 bps
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits
Flow control	XON control / No S-parameters

2 - 3 Practical procedures for connections

The following lists the different ways of connection to DMC50:

- Loader communication through loader port
- Loader communication through RS-485 port
- Loader communication through Ethernet
- CPL communication through OIT port
- CPL communication through RS-485 port
- CPL communication through Ethernet

There are two connection methods, direct connection and indirect connection.
The indirect connection means that a host connects to a CTRL module linked with a COM module through a communication port of the COM module.
A stand alone CTRL module supports only direct connection.

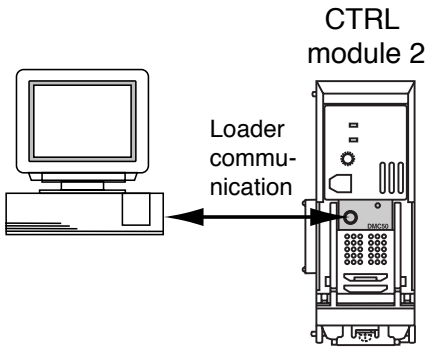
Note

After any settings were changed to a communication port, you may forget how they were set up, and you can not connect to the port again.
Even in this case, you can connect to the loader port if only you match the module address, because the loader port does not have communication settings.

■ Loader communication through loader port (direct connection)

This subsection illustrates the procedure for establishing connection to a module through its loader port with an example:

Example: To connect to a CTRL module with module address 2.



● Preparation

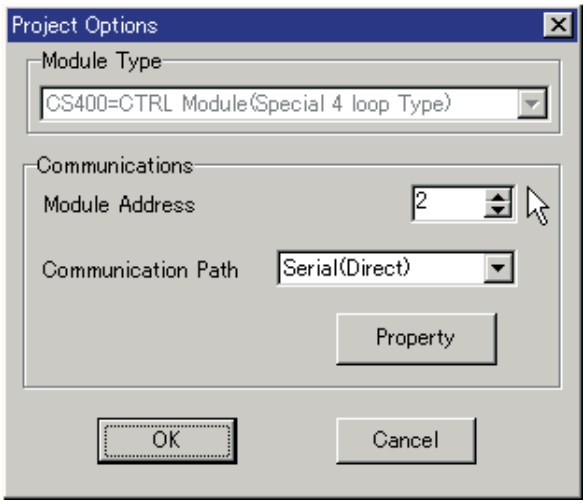
Connect the serial port of your personal computer to the loader port of the CTRL module with the loader cable "81440793-001".

● Connecting to a module

(1) Invoke SLP-D50.

(2) Open the project for the CTRL module to connect to.
If no projects are created, select [New Project].

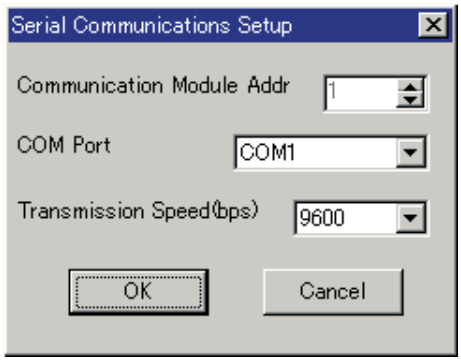
(3) Select [Options] from the [Project Options] menu.
>>The Project Options window will be opened.



(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module. (Here, the module address is set to 2.)

(5) Set [Communication Path] to [Serial (Direct)].

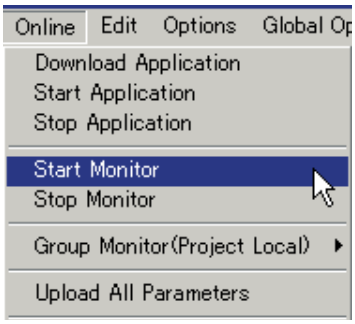
(6) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



Note

When connecting through the loader port, the transmission speed is fixed at 9600 bps.

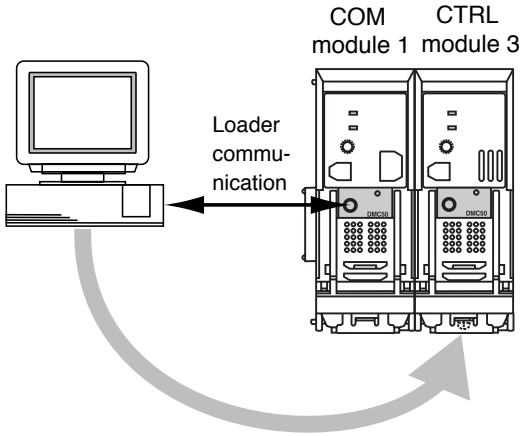
(7) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) To read information from the CTRL module, select [Online] → [Start Monitor].



■ Loader communication through loader port (indirect connection)

This subsection illustrates the procedure for establishing connection to a CTRL module through the loader port of a COM module with an example:

Example: To connect to a CTRL module with module address of 3 through a COM module with module address of 1.



● Preparation

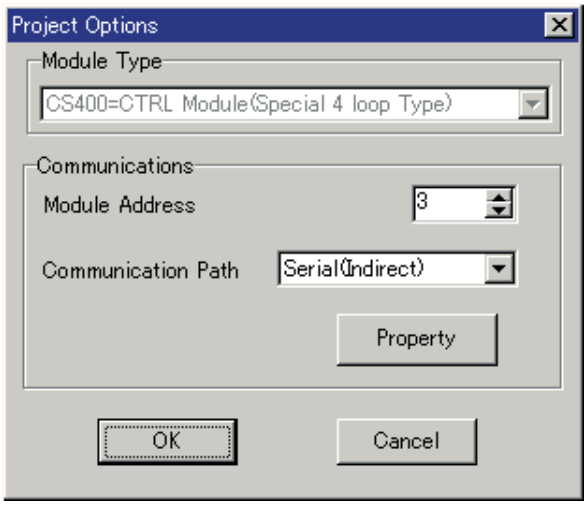
(1) Connect the CTRL module to the COM module with their multilink connectors.

(2) Connect the serial port of your personal computer to the loader port of the COM module with a loader cable.

● Connecting to the CTRL module

(1) Open the project for the CTRL module to connect to.
If no projects are created, select [New Project].

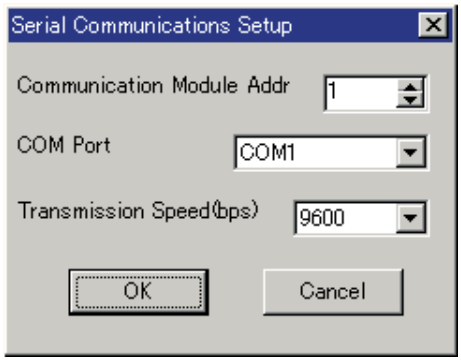
(2) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(3) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module.

(4) Set [Communication Path] to [Serial(Indirect)].

(5) Click the [Property] button.
>>The Serial Communications Setup window will be opened.

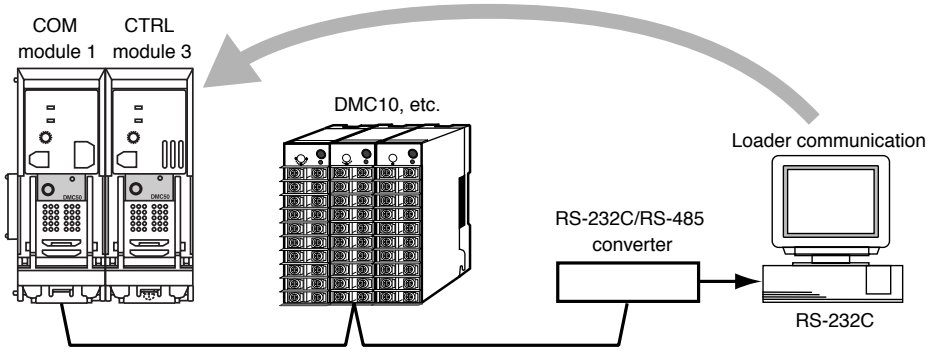


(6) Set the (decimal) value in [Communication Module Address] to the same as the (hexadecimal) rotary switch position number on the COM module to connect to.
(7) Set [COM Port] to the serial port name of the personal computer to which the loader cable is connected. Click the [OK] button.
>>The Communication Module Address window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) To read information from the CTRL module, select [Online] → [Start Monitor].

■ Loader communication through RS-485 port

This subsection illustrates the procedure for establishing connection to a CTRL module through the RS-485 port of a COM module with an example:

Example: To connect to a CTRL module (CS400) with module address of 3 through a COM module with module address of 1.



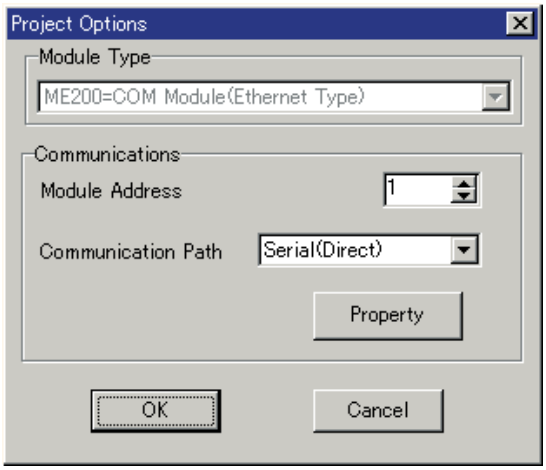
● Preparation

(1) Connect the CTRL module to the COM module with their multilink connectors.
(2) Connect a RS-232C cross-cable to the serial port of your personal computer.
(3) Connect other end of the RS-232C cross-cable to a RS-232C/RS-485 converter (Yamatake's CMC10L001A000 or its equivalent).
(4) Set up the RS-232C/RS-485 converter, and then connect the converter to the RS-485 port on the COM module with a RS-485 cable. Set up the converter as in the following table:

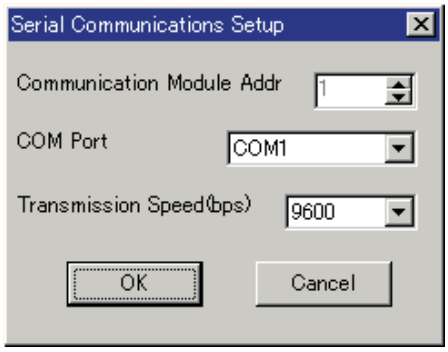
Transmission speed	9600 bps
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits
Flow control	XON control / No S-parameters

● Setting up the RS-485 port settings of the COM module

(1) Invoke SLP-D50.
(2) Open the project for the COM module.
If no projects are created, select [New Project].
(3) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module.
(5) Set [Communication Path] to [Serial (Direct)].
(6) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



(7) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) Select [Online] → [Start Monitor] to put SLP-D50 online. Modify the settings of the RS-485 port to connect to. Here, change the protocol of the port. The following screen example shows that [Protocol (RS-485 port 1)] is being changed:

Instance Header | Instance Body

			1
1	Trans. Speed (RS-485 port1)	[bps]	9600
2	Protocol (RS-485 port1)	--	2=Loader
3	Trans. Speed (RS-485 port2)	[bps]	
4	Protocol (RS-485 port2)	--	
5	IP Address	[IPAddr]	192.168.1.1
6	Subnet Mask	[IPAddr]	255.255.255.0
7	Default Router	[IPAddr]	0.0.0.0
8	KeepAliveTime	--	7200
9	IP Port1	--	1250
10	Protocol (IP Port1)	--	2
11	IP Port2	--	1251
12	Protocol (IP Port2)	--	2
13	IP Port3	--	1252
14	Protocol (IP Port3)	--	1
15	IP Port4	--	1253
16	Protocol (IP Port4)	--	1

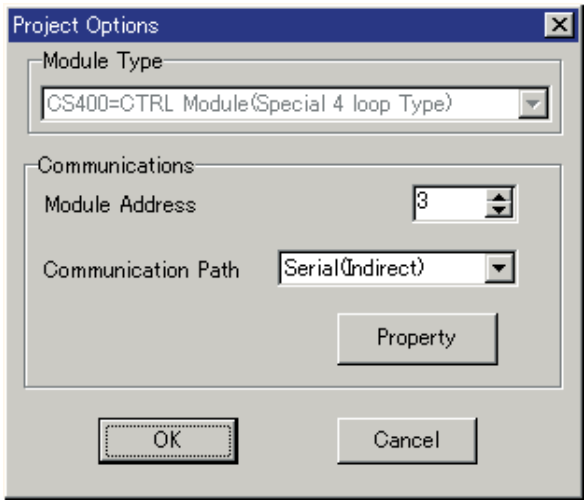
0=None
1=CPL
2=Loader

(10) Turn OFF and ON the power to the COM module again to allow the modification to take effect.

● Connecting to the CTRL module

(1) Open the project for the CTRL module to connect to.
If no projects are created, select [New Project].

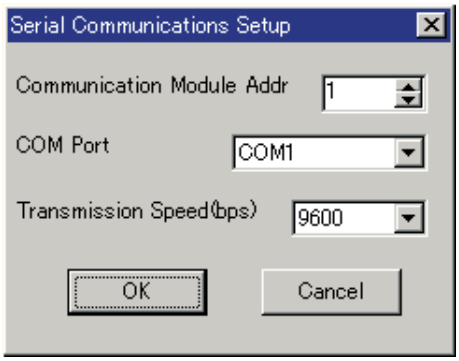
(2) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(3) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module.

(4) Set [Communication Path] to [Serial(Indirect)].

(5) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



(6) Set [Communication Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module.
Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.

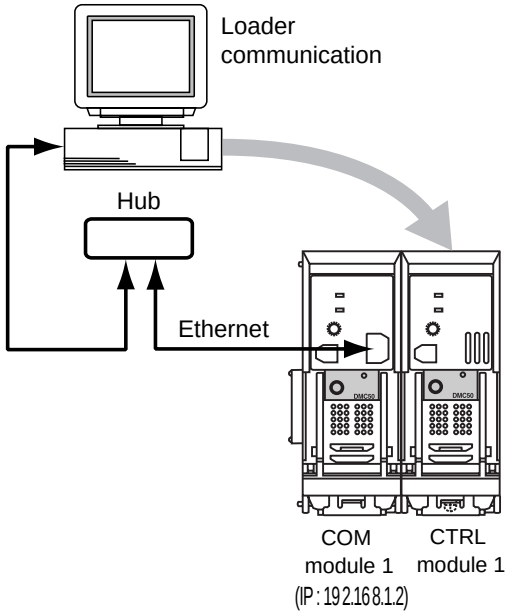
(7) Click the [OK] button in the Project Options window.

(8) To read information from the CTRL module, select [Online] → [Start Monitor].

■ Loader communication through Ethernet

This subsection illustrates the procedure for establishing connection to a CTRL module through the Ethernet port of a COM module with an example:

Example: To connect to a CTRL module with module address of 1 through a COM module with module address of 1 and IP address of 192.168.1.2:



● Preparation

(1) Set up the TCP/IP settings of the personal computer. It may be required to specify the default router, depending on your environment.
For details, consult the network administrator for your network.

(2) Connect the personal computer to a repeater hub with a 10 BASE-T cable, and connect the COM module to the same repeater hub with another cable.

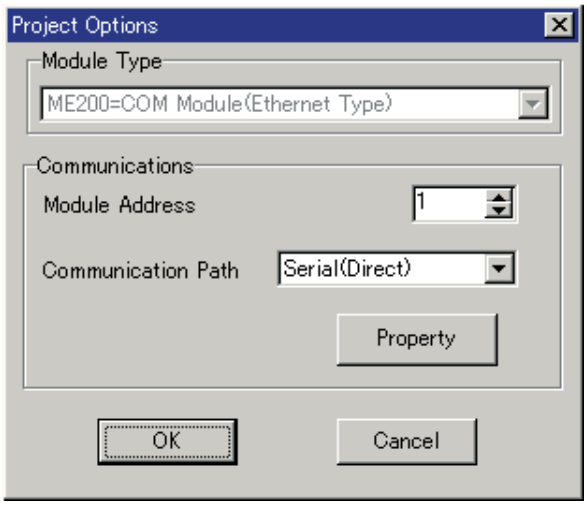
Handling Precautions

When making the connection through the 100BASE-T/10BASE-T auto-sensing/auto-negotiation switching hub, this might cause a faulty connection.
Do not use the 100BASE-T/10BASE-T auto-sensing/auto-negotiation switching hub, but use the repeater hub.

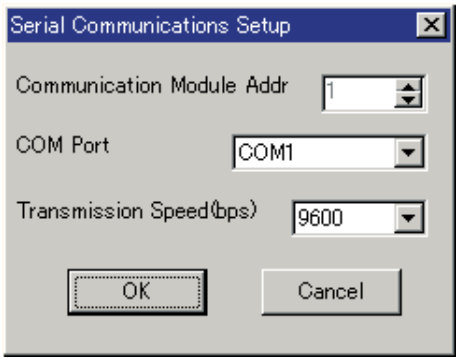
(3) Connect the serial port of the personal computer to the loader port of the COM module with a loader cable.

● **Setting up the COM module to connect directly to**

(1) Invoke SLP-D50.
(2) Open the project for the ME200, a COM module to use.
If no projects are created, select [New Project].
(3) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module.
(5) Set [Communication Path] to [Serial(Direct)].
(6) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



(7) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) Select [Online] → [Start Monitor] to put SLP-D50 online. Change the communication settings related to Ethernet. Set IP address, subnet mask, default router, and keep alive time for the COM module. Actual setting values may vary depending on your network environment. For details, consult the network administrator for your network.

The following screen example shows that [IP Address] is set to "192.168.1.2", [Subnet Mask] is set to "255.255.255.0", and [Default Router] is set to "192.168.1.37":

Instance Header | Instance Body

			1
1	Trans. Speed (RS-485 port1)	[bps]	9600
2	Protocol (RS-485 port1)	--	2
3	Trans. Speed (RS-485 port2)	[bps]	9600
4	Protocol (RS-485 port2)	--	1
5	IP Address	[IPAddr]	192.168.1.2
6	Subnet Mask	[IPAddr]	255.255.255.0
7	Default Router	[IPAddr]	192.168.1.37
8	KeepAliveTime	--	7200
9	IP Port1	--	1250
10	Protocol (IP Port1)	--	2
11	IP Port2	--	1251
12	Protocol (IP Port2)	--	2
13	IP Port3	--	1252
14	Protocol (IP Port3)	--	1
15	IP Port4	--	1253
16	Protocol (IP Port4)	--	1

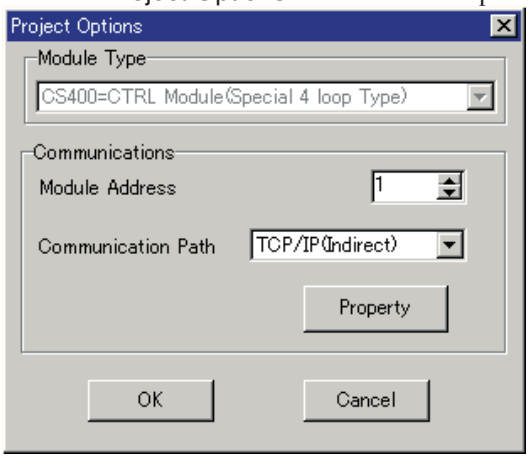
Unless otherwise required particularly, it is not necessary to change the [KeepAliveTime] and [IP Port 1 - 4] settings.

(10) Turn OFF and ON the power to the COM module again to allow the modifications to take effect.

● Connecting to the CTRL module indirectly

(1) Open the project for the CTRL module to connect to.
If no projects are created, select [New Project].

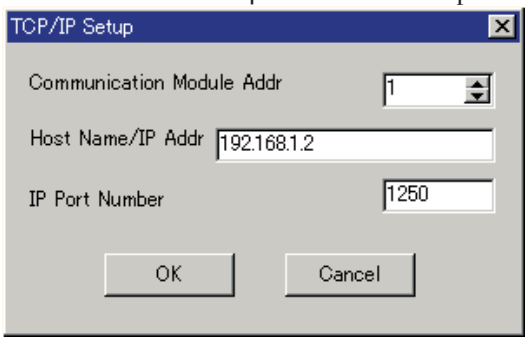
(2) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(3) In the Project Options window, set the [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module.

(4) Set [Communication Path] to [TCP/IP(Indirect)].

(5) Click the [Property] button.
>>The TCP/IP Setup window will be opened.



(6) Set [Communication Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module to connect directly.

(7) Set [Host Name/IP Address] to the same IP address as assigned to the COM module ("192.168.1.2" is assigned in the above example.), and then click the [OK] button.
>>The TCP/IP Setup window will be closed.

Handling Precautions

When multiple projects are opened on the SLP-D50 and the connections are made through Ethernet at the same time, do not use IP port 1 and IP port 2 at the same time. It is thought that one IP port must be used for backup. If SLP-D50 cannot shut down the communication correctly caused by that the personal computer gets hung up or other incident occurs, the power needs to be turned OFF and ON again or the period of time specified in [Keep Alive Time] must elapse to enable reconnection.

(8) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module.

(9) Click the [OK] button in the window.
(10) To read information from the CTRL module, select [Online] → [Start Monitor].

■ CPL communication through OIT port

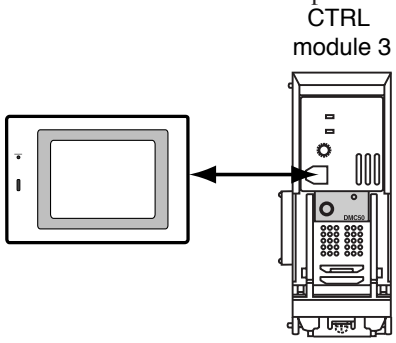
This subsection illustrates the procedure for establishing direct connection from a Yamatake's operator interface terminal to the OIT port of a CTRL module with an example:

Example: To connect from an operator interface terminal to a CTRL module with module address of 3.

Note

To issue CPL commands from a personal computer, you must write a program for this purpose.

Available CPL commands are explained in Chapter 4. CPL Command Reference.

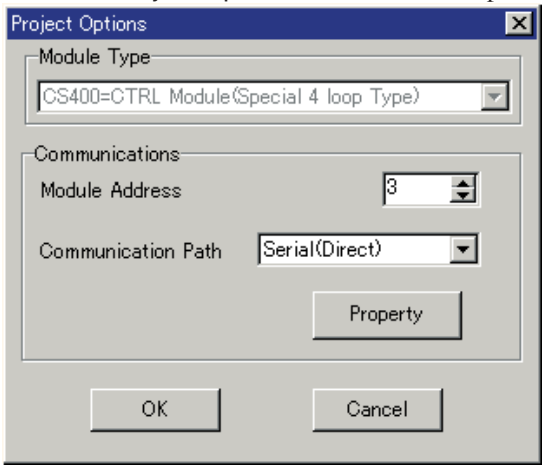


● Prepararion

Connect the serial port of you personal computer to the loader port of the CTRL module with a loader cable.

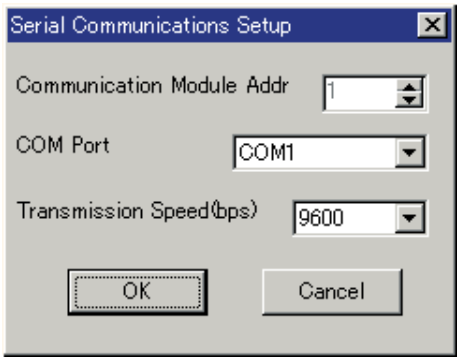
● Setting up the OIT port settings

(1) Invoke SLP-D50.
(2) Open the project for the CTRL module.
If no projects are created, select [New Project].
(3) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.

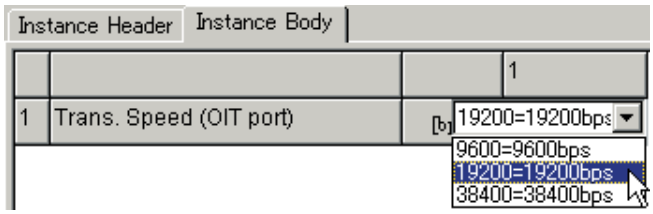


(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the CTRL module.

(5) Set [Communication Path] to [Serial(Direct)].
(6) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



(7) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) Select [Online] → [Start Monitor] to put SLP-D50 online. Modify the communication settings Front Port Communication Setup of the CTRL module.
The following screen example shows that [Trans. Speed (OIT port)] is being changed to "19200 bps":



(10) Turn OFF and ON the power to the COM module again to allow the modifications to take effect.

● **Preparation**

(1) Select [Online] → [Stop Monitor] to put SLP-D50 offline. Disconnect the plug of the loader cable from the CTRL module.
(2) Connect one end of the OIT communication cable to the OIT port of the CTRL module and the other end of the cable to the RS-485 port of the operator interface terminal.
(3) Set up the communication settings of the operator interface terminal as shown in the following:

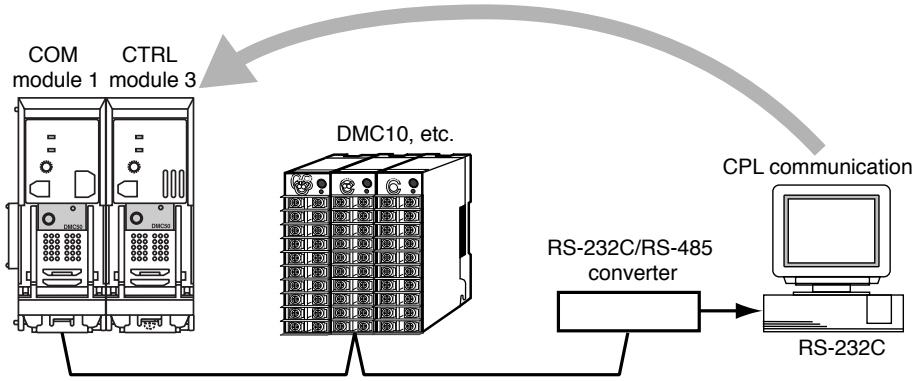
Transmission speed	Setting value (19200 bps in this case)
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits
Flow control	XON control / No S-parameters

! Handling Precaution

The loader port and OIT port cannot be used at the same time. If the plug for the loader communication and the connector for the OIT communication are connected at the same time, the communication will not work correctly.

■ CPL communication through RS-485 port

To issue CPL commands from a host computer (e.g. a personal computer), you must write a program for this purpose.
Available CPL commands are explained in Chapter 4. CPL Command Reference.
This subsection illustrates only the procedure for establishing connection to a CTRL module through the RS-485 port of a COM module with an example:
Example: To connect to a CTRL module with module of address 3 through a COM module with module address of 1.



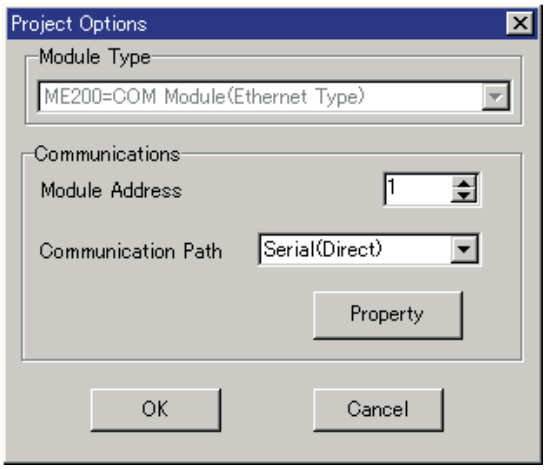
● Preparation

(1) Connect the CTRL module to the COM module with their multilink connectors.
(2) Connect the RS-485 port of a RS-232C/RS-485 converter to the RS-485 port of the COM module.
(3) Set up the communication settings of the host computer as shown in the following:

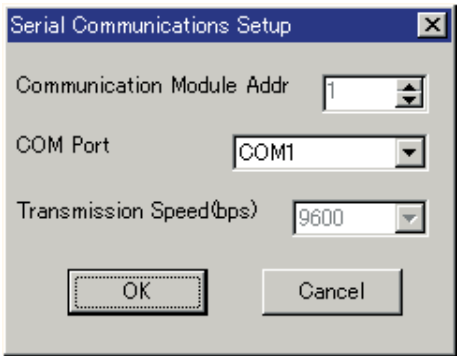
Transmission speed	Setting value (19200 bps in this case)
Parity	Even parity
Stop bit length	1 stop bit
Data length	8 bits

● Setting up the RS-485 port settings of the COM module

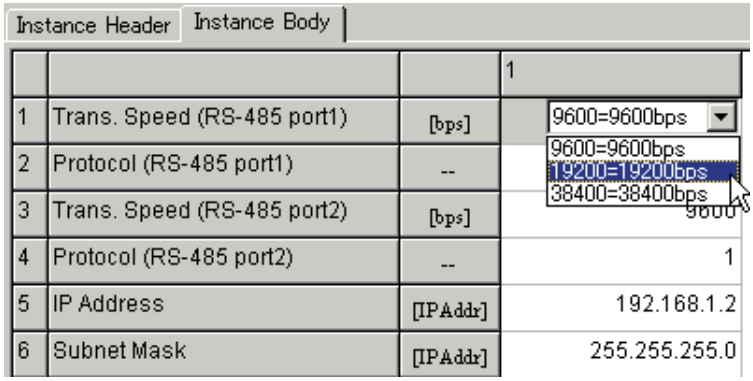
(1) Invoke SLP-D50.
(2) Open the project for the COM module.
If no projects are created, select [New Project].
(3) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module.
(5) Set [Communication Path] to [Serial(Direct)].
(6) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



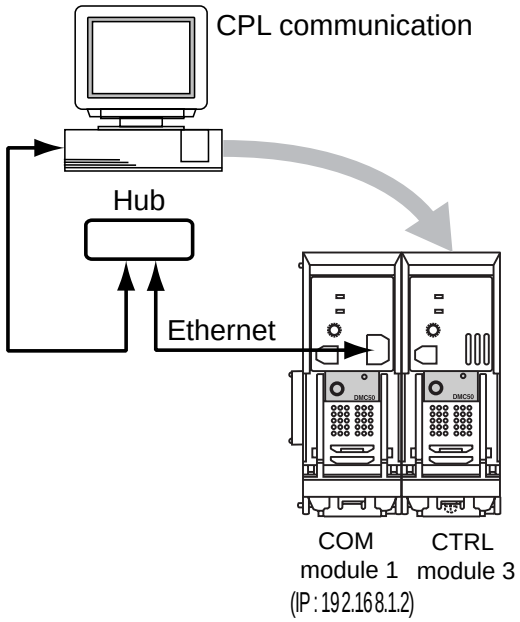
(7) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.
(8) Click the [OK] button in the Project Options window.
(9) Select [On line] → [Start Monitor] to put SLP-D50 online. Modify the settings of RS-485 port to connect to. Here, change the transmission speed of the port. The following screen example shows that [Trans. Speed (RS-485 port1)] is being changed to "19200 bps":



(10) Set [Protocol (RS-485 port1)] to "CPL".
Turn OFF and ON the power to the COM module again to allow the modifications to take effect.

■ CPL communication through Ethernet

To perform the CPL communication through Ethernet, you must write a program for this purpose.
Details about how to write such programs are described in the next chapter.
This subsection describes only the procedure for establishing connection to DMC50 with an example:
Example: To connect to a CTRL module with module address of 3 through a COM module with module address of 1 and IP address of 192.168.1.2:



● Preparation

(1) Set up the TCP/IP settings of your personal computer (or the host computer that is used). It may be required to specify the default router, depending on your environment.
For details, consult the network administrator for your network.
(2) Connect the personal computer to a repeater hub with a 10BASE-T cable, and connect the COM module to the same repeater hub with another cable.

Handling Precaution

When making the connection through the 100BASE-T/10BASE-T auto-sensing/auto-negotiation switching hub, this might cause a faulty connection.
Do not use the 100BASE-T/10BASE-T auto-sensing/auto-negotiation switching hub, but use the repeater hub.

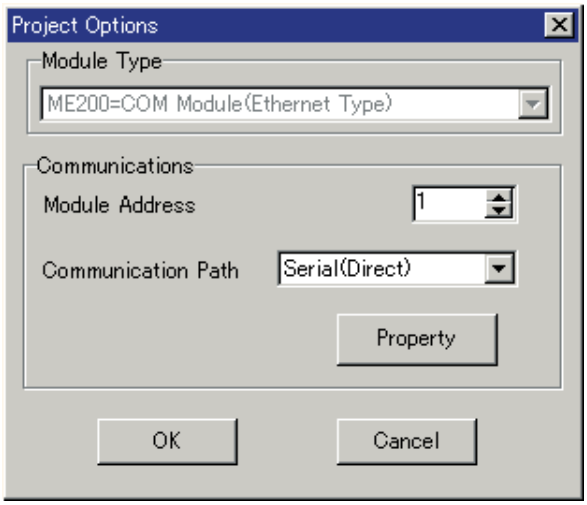
(3) Connect the serial port of the personal computer to the loader port of the COM module with a loader cable.

● **Ethernet port settings of COM module**

(1) Invoke SLP-D50.

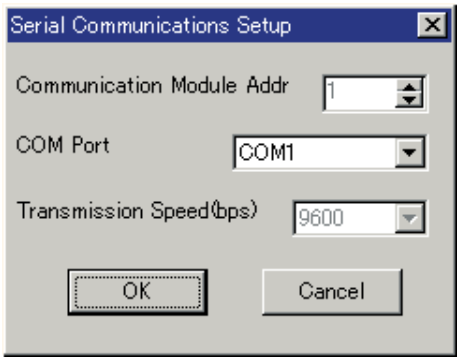
(2) Open the project for the COM module.
If no projects are created, select [New Project].

(3) Select [Project Options] from the [Options] menu.
>>The Project Options window will be opened.



(4) In the Project Options window, set [Module Address] to the (decimal) value the same as the (hexadecimal) rotary switch position number of the COM module.

(5) Click the [Property] button.
>>The Serial Communications Setup window will be opened.



(6) Set [COM Port] to the serial port name of the personal computer, to which the loader cable is connected. Click the [OK] button.
>>The Serial Communications Setup window will be closed.

(7) Click the [OK] button in the Project Options window.

(8) Select [Online] → [Start Monitor] to put SLP-D50 online. Modify the communication settings related to Ethernet. Set IP address, subnet mask, default router, and keep alive time for the COM module. Actual setting values may vary depending on your network environment. For details, consult the network administrator for your network.
The following screen example shows that [IP Address] is set to "192.168.1.2", the [Subnet Mask] is set to "255.255.255.0", and [Default Router] is set to "192.168.1.37":

Instance Header | Instance Body

			1
1	Trans. Speed (RS-485 port1)	[bps]	9600
2	Protocol (RS-485 port1)	--	2
3	Trans. Speed (RS-485 port2)	[bps]	9600
4	Protocol (RS-485 port2)	--	1
5	IP Address	[IPAddr]	192.168.1.2
6	Subnet Mask	[IPAddr]	255.255.255.0
7	Default Router	[IPAddr]	192.168.1.37
8	KeepAliveTime	--	7200
9	IP Port1	--	1250
10	Protocol (IP Port1)	--	2
11	IP Port2	--	1251
12	Protocol (IP Port2)	--	2
13	IP Port3	--	1252
14	Protocol (IP Port3)	--	1
15	IP Port4	--	1253
16	Protocol (IP Port4)	--	1

Unless otherwise required particularly, it is not necessary to change the [KeepAliveTime] and [IP Port 1 - 4] settings.

(9) Turn OFF and ON the power to the COM module again to allow the modification to take effect.

Chapter 3. ETHERNET COMMUNICATION

3 - 1 TCP/IP stack specifications

The following describes the TCP/IP stack specifications of the COM module:

■ IP ports and connections

● IP ports

Four IP ports are provided on the ME200. The following table shows the relationship between each IP port number and protocol in the factory settings: Two ports can be used for each of the loader commutation and CPL communication. However, one port must be used for backup in case a timeout error occurs. Do not use two ports at the same time.

IP port No.	Port No.	Protocol
1	1250	Loader communication server
2	1251	Loader communication server
3	1252	CPL communication server
4	1253	CPL communication server

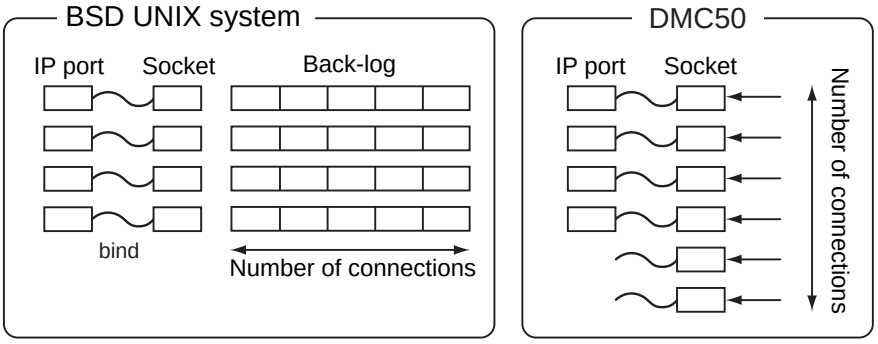
Note

IANA (Internet Assigned Numbers Authority) registers port numbers ranging from 1024 to 49151 as registered ports and opens the list of registered port numbers to public. Port numbers 1250 to 1253 used for the DMC50 factory setting are registered ports, and you should check yourself the availability in your environment.

● Connections

At maximum, six TCP/IP connections can be established for all IP ports of the ME200 in total. Each IP port of the ME200 does not have its own backlog (the queue size of clients waiting for connection), which is provided on the BSD (Berkeley Software Distribution) UNIX system. The number of connections is controlled by the number of socket descriptors so that six connections are maximum for all IP ports in total.

The socket descriptor is a method, which is widely used in systems, such as in the BSD UNIX. This socket descriptor is used to communicate with the internal or external process. The socket descriptor is a kind of extension to the file descriptor.



3 - 2 Client application

To perform the CPL communication over Ethernet, you must write a program for this purpose. The following describes how to write such a program in detail:

■ Error handling

● CPL error

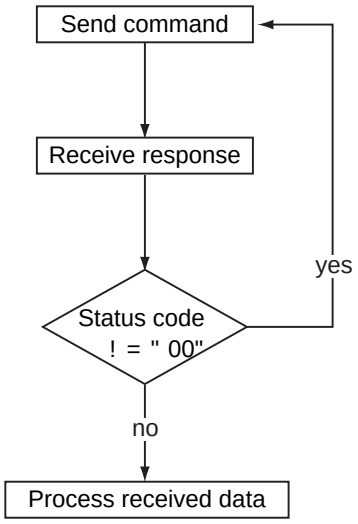
In the CPL communications, the following status codes are returned:

Status code	Results
00	Normal termination
13	Error during execution
80	CPL Ack response

- If the status code is "13" or "80", the retry should be performed.
- If a status code other than those listed in the above table returned, the CPL frame may be incorrect. In this case, do not handle the errors at the application level, such as retries, but remove the cause of such errors.

Handling Precautions

If the normal status code are not obtained even after retrying a number of times (for example, 20 times or more), an error may have occured in DMC50.



● TCP level error

If timeouts occur, reconsider the timeout setting.

The timeout time should be set to 3 sec. However, it is necessary to add RTT (Round Trip Time) of the TCP connection to this time.

tmout = 3(sec) + RTT

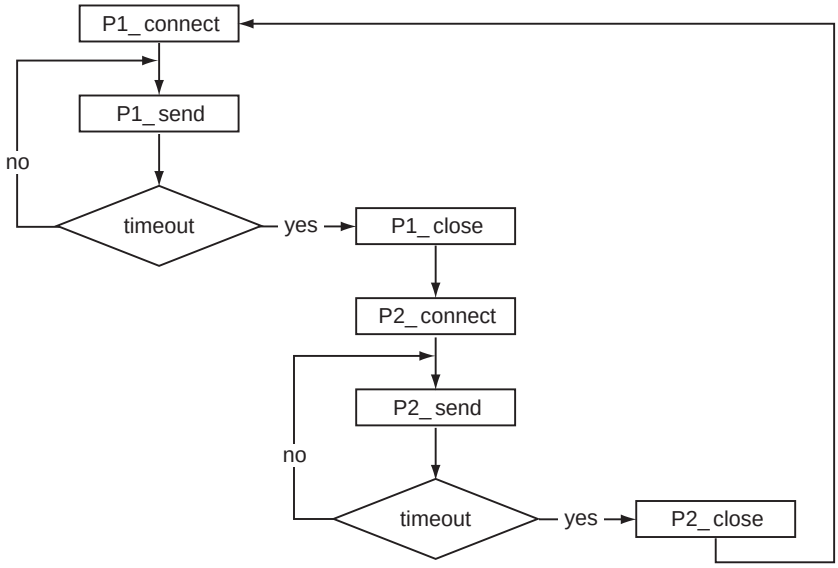
Actually, if Ethernet is used as LAN, no problems may arise even though the time-out time is set to 3 sec.

Note

A system supporting 4.4 BSD socket layer has "select( )" function to monitor the set of descriptors. A timeout used in the TCP/IP program on the system is specified as an argument to this function call.

● **Error handling if a timeout error occurs at the TCP level.**

If a timeout error occurs, two CPL ports must be switched as shown in the following flowchart, where two ports are named "P1" and "P2", respectively:



■ Sample program

! Handling Precautions

Yamatake Corporation shall not be responsible for any loss or damage caused, directly or indirectly, by your use of this sample program.

```
/*************************************************************************
 *                                                                       *
 *  DMC50 TCP/IP commnication test client program                        *
 *                                                                       *
 *************************************************************************/
#ifdef HAVE_CONFIG_H
# include "config.h"
#endif
#include    <ctype.h>
#include    <sys/types.h>   /* basic system data types */
#include    <sys/socket.h>  /* basic socket definitions */
#include    <sys/time.h>    /* timeval{} for select() */
#include    <time.h>        /* timespec{} for pselect() */
#include    <netinet/in.h>  /* sockaddr_in{} and other Internet defns */
#include    <arpa/inet.h>   /* inet(3) functions */
#include    <errno.h>
#include    <fcntl.h>       /* for nonblocking */
#include    <netdb.h>
#include    <signal.h>
#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>
#include    <sys/stat.h>    /* for S_xxx file mode constants */
#include    <sys/uio.h>     /* for iovec{} and readv/writev */
#include    <unistd.h>
#include    <sys/wait.h>
#include    <sys/un.h>      /* for Unix domain sockets */

#ifdef  HAVE_SYS_SELECT_H
# include   <sys/select.h>  /* for convenience */
#endif

#ifdef  HAVE_POLL_H
# include   <poll.h>        /* for convenience */
#endif

#ifdef  HAVE_STRINGS_H
# include   <strings.h>     /* for convenience */
#endif

#include    <stdarg.h>      /* ANSI C header file */
#include    <syslog.h>      /* for syslog() */

/* Three headers are normally needed for socket/file ioctl's:
 * <sys/ioctl.h>, <sys/filio.h>, and <sys/sockio.h>.
 */
#ifdef  HAVE_SYS_IOCTL_H
# include   <sys/ioctl.h>
#endif

#ifdef  HAVE_SYS_FILIO_H
# include   <sys/filio.h>
#endif
#ifdef  HAVE_SYS_SOCKIO_H
# include   <sys/sockio.h>
#endif

#ifdef  HAVE_PTHREAD_H
```

```
# include   <pthread.h>
#endif

static void err_doit(int errnoflag, int level, const char *fmt, va_list ap) ;
void err_quit(const char *fmt, ...) ;
void err_msg(const char *fmt, ...) ;
void err_sys(const char *fmt, ...) ;
void usage(const char *msg) ;
void dump_buf( char *ptr, int nrcv ) ;
int recv_data( int so, char *vptr, int len, int pause ) ;

int daemon_proc=0 ;

#define TIMEOUT    3                    /* cpl default timeout sec */
/* default test argumets */
#define DEFAULTARG1 "0101xRN00LL24300202243002040E6001020E6001040E600106"
#define DEFAULTARG2 "0100XRGLL00100101000A"
#define LOOPS_DEFAULT 1
#define PAUSE_DEFAULT 0
#define NON_SUM 0
#define WITH_SUM 1
#define WITH_SUM_ILL 2
#define CMD_SEND_DEFAULT 1
#define CMD_SEND_USER 0
#define WITH_LINT_BEFORE 1
#define WITH_LINT_AFTER 2

char cplComAppBuf[512];

#ifdef  notdef                          /* delete if <stdlib.h> doesn't define these for getopt() */
extern char *optarg;
extern int  optind, opterr, optopt;
#endif

#define MAXLINE 4096                    /* max line length */
char *host;
char *port;

#define STX 0x02
#define ETX 0x03
#define CR 0x0a
#define LF 0x0d
#define CPLLENGTH 512

unsigned char cplFrame[1024];  /* buffer for cpl command frame */
unsigned char rb[1024];        /* buffer for cpl response frame */

/* set CPL frame */
/*
    char *cplp      buffer for CPL command to be stored
    char *seq       CPL command string
    int mode        checksum mode
    int vmode       display mode
    int lint        frame with lint flag
*/

int Set_cplFrame(char *cplp, char *seq, int mode, int vmode, int lint )
{
    int len;                            /* length of CPL command string */
    int len2;                           /* length of CPL command string +2 */
    int st=0 ;                          /* frame start pointer */
    char sum;                           /* temp var to calculate checksum */
    char chksum;                        /* checksum value */
    unsigned char *checkptr;            /* pointer to checksum value */
    char chksumStr[10];                 /* checksum string */
    int i;
```

```
    len = strlen(seq);
    cplp[st++] = STX;
    len2 =1 ;

    if( vmode == 0 ) {
        /*    printf("COMMAND := %sIn", seq); */
    }

    strncpy(&cplp[st], seq, len);
    cplp[st + len] = ETX;
    len2 = len + st + 1 ;

    if( mode == WITH_SUM ) {
        /* generate checksum of this frame */
        sum = 0;
        for (i=st-1;i < len2; i++) {
            sum += cplp[i];
        }

        chksum = -sum;
        checkptr = (unsigned char *)&chksum;
        sprintf(chksumStr, "%02X", *checkptr);
        strncpy(&cplp[len2], &chksumStr[strlen(chksumStr)-2],2);
        len2 += 2 ;
        cplp[len2] = 0x0d;
        cplp[len2+1] = 0x0a;
        len2 += 2 ;
    }
    else {
        cplp[len2] = 0x0d ;
        cplp[len2+1] = 0x0a ;
        len2+=2 ;
    }

    /* return the length of the CPL command frame */
    return (len2);
}


/* check CPL response frame */
/*
    char *cplrp     buffer for CPL command response
    int rlen        length of CPL command response
    int smode       sum mode
    int vmode       verbose mode flag
*/
int Check_cplResponse(char *cplrp, int rlen, int smode, int vmode)
{

    char pbuf[1024];                    /* buffer for print */
    int content;                        /* content flag for CPL response */
    int pi;                             /* counter for print buffer */
    int exitfor;                        /* flag to escape for loop */
    int ri;                             /* counter for response buffer */
    char sum;                           /* temp var to calculate checksum */
    char chksum;                        /* checksum value */
    char rchksum;                       /* checksum value in response packet */

    if ( rlen == 0 )
      return rlen ;
    /* reset conters and flags */
    pi = 0;
    content = 0;
    exitfor = 0;

    for (ri = 0; ri < rlen; ri++) {
```

```
        switch(cplrp[ri]) {
        case STX:                       /* must be rb[0] */
            content = 1;
            sum = STX;
            continue;
        case ETX:
            exitfor= 1;
            sum += cplrp[ri];           /* ETX */
            break;
        default:
            if (content == 1) {
                pbuf[pi++] = cplrp[ri];
                sum += cplrp[ri];
            }
        }
        if (exitfor)
          break;
    }
    if (exitfor == 0) {
        /* if there is no ETX */
        printf("ERROR : broken response");
        exit(1);
    }

    /* check response checksum */
    chksum = -sum;
    ri++;
    cplrp[ri+2] = 0;
    rchksum = strtol(&cplrp[ri], NULL, 16);
    if( smode ) {
        if (chksum != rchksum) {
            /* if checksums do NOT match */
            pbuf[pi] = 0;
            printf("checksum ERROR %02X, %02X, %sIn",chksum,rchksum, pbuf);
            exit(1);
        }
    }

    /* print the contents of the response */
    pbuf[pi]= 0;
    if (pbuf[5] == '8' && pbuf[6] == '0') {
        printf("InACK RESPONSE:= %sInInIn", pbuf);
        /* exit(1); */
    } else if (pbuf[5] != '0' || pbuf[6] != '0') {
        printf("InERROR RESPONSE:= %sInInIn", pbuf);
        /* exit(1); */
    } else {
        /* printf("InRESPONSE:= %sInInIn", pbuf);*/
    }
    return rlen ;
}


/* send data in the specified buffer to the specified socket */

int S_write(int so, unsigned char *buf, int len, int verbose)
{
    int wpsiz, wsiz;
    static int counter = 1;
    time_t tm;
    char *tbuf ;
    tbuf = buf ;
    for (wpsiz = len; wpsiz > 0; wpsiz -= wsiz, buf += wsiz) {
        if ((wsiz = send(so, buf, wpsiz,0)) < 0) {
            printf("Insend ERROR");
            exit(1);
        }
```

```
    }
    tm = time(NULL);
    printf( "InSend[%d] %.24s %d BytesIn", counter++ , ctime(&tm), wsiz ) ;
    if( verbose != 0 )
      dump_buf( tbuf, wsiz ) ;
    return(len - wpsiz);
}

/* recieve data in the specified buffer from the specified socket */

int recv_data( int so, char *vptr, int len, int pause )
{
    int n, rc ;
    char *ptr ;
    int flg = 0 ;

    ptr = vptr ;
    n = 0 ;
    while(1) {
        if( ioctl(so, FIONREAD, &rc ) == -1 ) { /* number of bytes in socket */
            err_quit( "ioctl failed!!In" ) ;
        }
        else {
            if( rc == 0 ) {             /* socket empty */
                if( ( flg==0 ) && ( pause <= 0) ) {
                    n=0 ;
                    break ;
                }
                else if( flg ) {
                    break ;
                }
                else {
                    sleep(1) ;
                    pause-- ;
                }
            }
            if ( rc > len )
              rc = len ;
        }

        if( ( rc = recv( so, (void *)vptr, rc, NULL ) ) == EOF ) {
            printf( "recv EOFIn" ) ;
            break ;
        }
        else if ( rc != -1 ) {
            flg = 1 ;
            ptr += rc ;
            ptr++ ;
            *ptr = 0 ;
            n = n+rc ;
        }
        else {
            err_msg( "Error: recv err!" ) ;
        }
    }

    return n ;
}
/* Entry point of this demo program */

int main(int argc, char *argv[])
{
    struct sockaddr_in name;            /* structure for socket */
    int so;                             /* socket for connection to dmc50 */

    int cx;                             /* CPL command  index */
    int cpllen;                         /* length of CPL command frame */
```

```
int rsiz;                           /* received byte from the socket */
int ret;                            /* return value of select() */
struct timeval timeout, *ptimeout;  /* timeout for select */

fd_set  readmask;                   /* read mask backup for select() */
fd_set  readfds;                    /* array of file descripors */

char ch;                            /* chracter received from the socket */
char lch;                           /* last chracter received from the socket */
int ri;                             /* cpl response buffer counter */
int reslen;                         /* length of a cpl response frame */

int loops=LOOPS_DEFAULT ;           /* number of loops issueing the commands */
int pause=PAUSE_DEFAULT ;           /* interval between commandes to send */

int summode = NON_SUM ;             /* cpl frame default sum mode is non-sum*/
int nmulti = 0 ;

int cmddef = CMD_SEND_USER ;        /* default command send mode */
int count = 0 ;                     /* How many loops executed */
int rfrmcnt = 1 ;                   /* recev frame count */
int tmout = TIMEOUT ;               /* cpl timeout */
int verbose = 0;
int lint = 0 ;                      /* lint data flag */

int i ;
int c ;
unsigned long inaddr;
struct servent *sp;
struct hostent *hp;
char *protocol ;
short pnum,pnum1,pnum2 ;

setbuf(stdout, NULL);               /* stop buffering if stdout is connected to pipe */
if (argc < 2) {
    usage("") ;
}

while ( (c = getopt(argc, argv, "hi:p:o:vSslLmc")) != EOF) {
    switch (c) {
    case 'h':
        usage("") ;
        break ;
    case 'i':
        if( (loops = atoi(optarg)) <= 0 )
          usage("") ;
        break ;
    case 'p':
        if( (pause = atoi(optarg)) <= 0 )
          pause = PAUSE_DEFAULT ;
        break ;
    case 'o':
        if( (tmout = atoi(optarg)) <= 0 )
          tmout = TIMEOUT ;
        break ;
    case 'v':
        verbose = 1 ;
        break ;
    case 'S':
        summode = WITH_SUM ;
        break ;
    case 'm':
        nmulti = 1 ;
        break ;
    case '?':
        printf( "unrecognized option" ) ;
```

```
            break ;
        }
    }

    if( optind > (argc-2) ) {
        usage("missing serverip, portnum and countIn") ;
    }
    else if( optind == (argc-2) ) {     /* no frame input by user */
        cmddef = CMD_SEND_USER ;
        argc += 2 ;
        argv[optind+2] = DEFAULTARG1 ;
        argv[optind+3] = DEFAULTARG2 ;
    }
    else if( (optind == (argc-3)) && ( nmulti ) ) {
        argc ++ ;
        argv[optind+3] = DEFAULTARG2 ;
    }
    host = argv[optind] ;
    port = argv[optind+1] ;

    if( nmulti ) {
        tmout = 0 ;                     /* select timeout is forever */
    }

    /* initialize socket address structure */
    bzero( (char *)&name, sizeof(name) ) ;

    /*
     * First try to convert the host name as a dotted-decimal number.
     * Only if that fails do we call gethostbyname().
     */
    if ( (inaddr = inet_addr(host)) != INADDR_NONE) { /* it's dotted-decimal */
        bcopy((char *) &inaddr, (char *) &name.sin_addr, sizeof(inaddr));
    }
    else {
        if ( (hp = gethostbyname(host)) == NULL) {
            printf("gethostbyname() error for: %s", host);
            exit(1) ;
        }
        bcopy(hp->h_addr, (char *) &name.sin_addr, hp->h_length);
    }

    /* see if "port" is a service name or number */
    protocol = "tcp";
    if ( (i = atoi(port)) == 0) {
        if ( (sp = getservbyname(port, protocol)) == NULL) {
            printf("getservbyname() error for: %s/%s", port, protocol);
            exit(1) ;
        }
        pnum1 = sp->s_port;
        pnum2 = pnum1+1;
    }
    else{
      pnum1 = htons(i) ;
      pnum2 = pnum1+1;
    }

    printf( "port1=%dIn", pnum1 ) ;
    printf( "port2=%dIn", pnum2 ) ;
    pnum = 0;

    /* set sockaddr_in structure */
    /* create socket */
connectagain:

    if (pnum == pnum1)      /* change service port to another port */
        pnum = pnum2;
```

```
else
    pnum = pnum1;

if ((so = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
    printf("Incan't get socket");
    exit(1);
}
printf("Insocket opened %d In", so ) ;

name.sin_family = AF_INET;
/*  name.sin_len = ( u_char )sizeof(name); */
name.sin_port = pnum ;
/* connect to the server */
if (connect( so, (struct sockaddr *)&name, sizeof(name)) < 0) {
    printf("Inconnection ERROR = %dIn", errno);
    exit(1) ;
}
printf( "connected on %s:%d In", inet_ntoa(name.sin_addr), ntohs(name.sin_port));
while(1) {
    for(cx = (optind+2); cx <argc; cx++) {
        /* Issue a CPL command supplied with a command argument */

        /* set CPL command frame to cplFrame */
        if( nmulti ) {              /* set -m option :multi frame contain one TCP segment */
            int scl ;
            cpllen = Set_cplFrame(cplFrame, argv[cx], summode, verbose, lint );
            scl = cpllen ;
            cx++ ;
            cpllen = Set_cplFrame(&cplFrame[cpllen], argv[cx], summode, verbose, lint ) ;
            cpllen = scl + cpllen ;
        }
        else
          cpllen = Set_cplFrame(cplFrame, argv[cx], summode, verbose ,lint );
        /* send cplFrame to TCP/IP */
        S_write(so, cplFrame,cpllen, verbose );

        ri = 0;
        ch = lch = 0;
        /* socket for cpl comm is set for readmask */
        FD_ZERO(&readmask);
        FD_SET(so, &readmask);

        while(1) {
            /* timeout for select() is set to tmout */
            timeout.tv_sec = tmout ;
            timeout.tv_usec = 0;
            if( tmout )
              ptimeout = &timeout ;
            else
              ptimeout = NULL ;

            readfds = readmask;

            /* waiting for inputs */
            if ( (ret = select(so+1, (fd_set *)&readfds, NULL, NULL, ptimeout)) < 0 )
              err_sys("ERROR: select failedIn") ;
            else if (ret == 0) {
                printf("WARNING : CPL TIMEOUTIn");
                close(so) ;
                sleep(1) ;
                goto connectagain ;

            } else if (FD_ISSET(so, &readfds)) { /* There is data at socket for cpl */
                /* just receive one byte from the socket */
                rsiz = recv_data( so, rb, sizeof(rb)-1 , 5 ) ;
                if ( rsiz < 0) {
                    printf("In ERROR : recv error no reply error %dIn", errno);
```

```
                        reslen = 0 ;
                        close(so) ;
                        sleep(1) ;
                        goto connectagain ;
                    }
                    else if (rsiz == 0 ) {
                        /* closed connection by peer */
                        printf("InWARNING : connection closed by peerIn" ) ;
                        close(so) ;     /* socket close */
                        goto connectagain ; /* and connect again to peer */
                    }
                    else {
                        time_t tm;
                        tm = time(NULL) ;
                        printf( "Recv[%d] %.24s %d BytesIn", rfrmcnt++, ctime(&tm), rsiz ) ;
                        if( verbose ) {
                            dump_buf( rb, rsiz ) ;
                        }
                        break ;
                    }
                } /* end of else if */
            } /* end of while(1) */
            Check_cplResponse(rb, reslen, summode, verbose );
            if( reslen != 0 )
              sleep( pause ) ;
        } /* end of for */
        count++;
        if (count >= loops)
          break;
    }
    exit(0); // close socket and sucessfully exit from this program
}


void err_sys(const char *fmt, ...)
{
    va_list     ap;

    va_start(ap, fmt);
    err_doit(1, LOG_ERR, fmt, ap);
    va_end(ap);
    exit(1);
}
void err_msg(const char *fmt, ...)
{
    va_list     ap;

    va_start(ap, fmt);
    err_doit(0, LOG_INFO, fmt, ap);
    va_end(ap);
    return;
}
void err_quit(const char *fmt, ...)
{
    va_list     ap;

    va_start(ap, fmt);
    err_doit(0, LOG_ERR, fmt, ap);
    va_end(ap);
    exit(1);
}

static void err_doit(int errnoflag, int level, const char *fmt, va_list ap)
{
    int     errno_save, n;
    char    buf[MAXLINE];
```

```
    errno_save = errno;                 /* value caller might want printed */

    vsnprintf(buf, sizeof(buf), fmt, ap);
    n = strlen(buf);
    if (errnoflag)
      snprintf(buf+n, sizeof(buf)-n, ": %s", strerror(errno_save));
    strcat(buf, "In");

    if (daemon_proc) {
        syslog(level, buf);
    } else {
        fflush(stdout);                 /* in case stdout and stderr are the same */
        fputs(buf, stderr);
        fflush(stderr);
    }
    return;
}


/* dump output ptr */
void dump_buf( char *ptr, int nrcv )
{
    int j,ch ;
    u_char cbuf[17], d ;

    for( j = 0; j < 16; j++ ) {
        printf( "+%02X", j ) ;
    }
    printf( " ASCII In" ) ;
    while(nrcv>0) {
        for(j = 0; j < ( (nrcv < 16) ? nrcv :16 ); j++ ) {
            d = (u_char)*ptr ;
            printf( " %02x", d ) ;
            ch = (int)d ;
            if( isprint( ch ) )
              cbuf[j] = d ;
            else
              cbuf[j] = ' ' ;

            cbuf[j+1] = 0 ;
            ptr++ ;
        }
        if( nrcv < 16 ) {
            for( j = 0; j < (16-nrcv); j++ ) {
                printf( "   " ) ;
            }
        }
        nrcv = nrcv - 16 ;
        printf( " %sIn", cbuf ) ;
    }
}

void usage(const char *msg)
{
    printf( "DMC50 TCP/IP commnication test client programInIn" ) ;
    err_msg(
            "usage: cpltest [options] <host> <port> [CPL command 1] .. [CPL command n]¥n"
            "options: -h      this help¥n"
            "         -i n    continuous counts ¥n"
            "         -p n    seconds to pause after each access ¥n"
            "         -o n    CPL timeout seconds¥n"
            "         -v      verbose¥n"
            "         -S      with check sum¥n"
            "         -m n    num of frame include over one TCP segment¥n"
            "         -c      connect again after connection closed by peer¥n"
            "InInnote: <port> must be 1st service port nummber¥n"
            "      (factory setting of dmc50 is 1252) ¥n"
```

```
                );

        if (msg[0] != 0)
          err_quit("%s", msg);
        exit(1);
    }

    /*
     * $Log: cplsample.c,v $
     * Revision 1.1  2000/12/05 08:00:09
     */
```

3 - 3 Glossary

- **Ethernet**

  Ethernet is a local area network widely used throughout the world.

  Various units equipped with Ethernet, such as personal computers, workstations, and printers are available on the market. Use of such units makes it possible to construct a versatile network.

  If a CPL communication client program, as a TCP/IP application, is written on a general purpose personal computer or workstation, it can easily communicate with DMC50 over Ethernet.

- **TCP (Transmission Control Protocol)**

  TCP supports connection type communication services. TCP has necessary features to support the reliable services between two communication units such as:
  - Acknowledgement response
  - Flow control
  - Timer
  - Connection management

- **IP (Internet Protocol)**

  IP is the most basic protocol of TCP/IP. IP provides the following major features:
  - IP address handling (packet routing)
  - Fragmentation and reconstruction of datagrams

  Each host or gateway on the way to the final destination checks the destination IP address of the packet (or datagram) transferred to it, and determines a route to the next gateway or host, and then sends this datagram to the network toward the next gateway or host.

  Fragmentation of datagram is that, if the length of the received datagram is larger than the allowable maximum length of the next network, the datagram is fragmented into multiple datagrams to meet the allowable maximum length, and then the contents of the header information attached to the source datagram are used to construct the header of each fragmented datagram.

  Reconstruction of datagram is that at the final destination, the original datagram is reconstructed from multiple fragmented datagrams according to all the header information. There are some auxiliary protocols that are used with the IP protocol. The protocols, which the ME200 supports, are ARP and ICMP.

- **ARP (Address Resolution Protocol)**

  ARP is a protocol, which binds MAC addresses to IP addresses.
  If the IP address of the destination is given, but if the MAC address of the destination is not known, an ARP request with the IP address of the destination is broadcasted. If a unit having the IP address of the destination in its ARP table receives the ARP request, the MAC address information is sent to the ARP requestor.

● ICMP (Internet Control Message Protocol)

ICMP provides a way of network diagnosis.
Error notification of IP packets or network status is checked.
For example, the following operations are performed:

- If it is unable to reach the IP address or port number of the destination, the sender is notified that the delivery to the destination cannot be performed.
- The echo request to check the status of the mating host is sent or the answer to this request is returned.

The TCP/IP protocol stack automatically generates and interprets the ICMP.

● CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

CSMA/CD is a method, used in Ethernet, allowing simultaneous access from multiple carrier sense stations with collision detection.

If no sending data exists on the communication path, any stations can start sending data. When multiple stations send data, and if a collision occurs, they will wait a random amount of time before retrying.

● MAC address (Ethernet address)

MAC address is a 6-byte address assigned to all Ethernet units. This MAC address is a unique address assigned for each unit, and cannot be changed.
Therefore, there are no units having duplicated MAC addresses throughout the world.
A unique MAC address has been assigned to each DMC50 COM module at shipment from the factory.
TCP/IP applications use the IP addresses. They can perform communications without consideration of MAC addresses.

● IP address

IP address is an address the user can assign and change. The user performs the communication using the IP address.
An IP address (currently, 4-byte length) is assigned to a unit equipped with the IP protocol.
An appropriate IP address is given by the system administrator of the place (department) that the controller is used.
TCP/IP allows construction of a large-scale network through the bridge or modem. Therefore, care must be taken so that no IP addresses are duplicated.
The private addresses (local addresses) are specified by RFC1918 as listed below:

Class A: 10.0.0.0 to 10.255.255.255
Class B: 172.16.0.0 to 172.31.255.255
Class C: 192.168.0.0 to 192.168.255.255

● **Class A**

0[7bit].[8bit].[8bit].[8bit]
(Most significant 8 bits represent the IP network address part and rest 24 bits represent the IP host address part.)
Addresses that the IP network address part becomes 0 and 127 are reserved.
Addresses that all bits of the IP host address part become 0 and 1 are reserved.

● **Class B**

10[6bit].[8bit].[8bit].[8bit]
(Most significant 16 bits represent the IP network address part and rest 16 bits represent the IP host address part.)
Addresses that the IP network address part becomes 128.0 and 191.255 are reserved.
Addresses that all bits of the IP host address part become 0 or 1 are reserved.

● **Class C**

110[5bit].[8bit].[8bit].[8bit]
(Most significant 24 bits represent the IP network address part and rest 8 bits represent the IP host address part.)
Addresses that the IP network address part becomes 192.0.0 and 223.255.255 are reserved.
Addresses that all bits of the IP host address part become 0 or 1 are reserved.

● **Subnet mask**

Subnet mask subdivides one IP network address into multiple sub IP network addresses.

The upper bits representing the sub IP network must be set to 1 and all lower bits must be set to 0.
It is necessary that sub-networks are seen as one IP network from the outside.

The settings that all bit values of the sub-network part become 0 or 1 are not allowed.

Note

- When 1-bit subnet mask is defined for a class C IP network address 192.168.1.0 (Most significant 25 bits become the IP network part), the subnet mask is as follows:

  11111111.11111111.11111111.10000000
  (=255.255.255.128)

- In the above example, "192.168.1.1" and "192.168.1.129" belong to the same IP network, but they are separated in sub-networks.

- **Port number**

  Port number is a number that identifies the service type on TCP.

  An IP address specifies a unit or station. On the contrary, a port number specifies a service type in the unit or station.
  Different data sent to a same unit but to different ports are handled separately.

- **Well-known ports**

  IP port numbers 1 to 1023 are called as well-known ports.
  These port numbers are totally controlled and assigned by IANA (Internet Assigned Numbers Authority).

  For example, "21" for ftp service and "23" for telnet service are well-known ports.

- **Registered port**

  IP port numbers 1024 to 49151 are not controlled by IANA. However, these port numbers are registered as registered ports by IANA and the list of registered ports is opened to public.

  Port numbers 1250 to 1253 used for the DMC50 factory settings are registered ports, and you should check yourself the availability in your environment.

- **KeepAliveTime**

  KeepAliveTime is a cycle time, at intervals of which ME200 checks inactive TCP/IP connections caused by, for example, when a unit connected to the ME200 performs power-down, reboot, or process-kill operation without shutting down the connection to the ME200.

  With the factory setting of 7,200 sec. (= 2 hrs.), such inactive condition is not informed for 2 hrs. and the connection is kept connected.

Chapter 4. CPL COMMAND REFERENCE

■ Introduction

Application layer commands of Yamatake's existing CPL communication protocol are commands that can access 16-bit data and 16-bit address space. Since 32-bit data and 32-bit space is provided on DMC50, 32-bit extended CPL commands are required to access all data.

⚠ Handling Precautions

To connect from existing CPL units, carefully check the compatibility.

■ Basic frame format

XX of [XX] is a token of command (response).
XX of (XX) is the number of bytes. Example: (2)
X of 'X' is an ASCII code expressing the character. Example: 'X'

Command:

[STX][sta(2)][sub(2)]['X'][cmd(2)]{[args]..}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)]{[data]..}[ETX][sum(2)][CR][LF]

Token expression	Name	Contents
[STX]	Start code	02h
[sta(2)]	CPL station address	Rotary switch on the module to which a outside system directly connects. *1
[sub(2)]	CPL sub-address	•Rotary switch (00h..0Fh) on the CTRL module to communication with for indirect communication. *1 •00h for direct communication
['X']	CPL device distinction code	The code of DMC50 is fixed at X.
[cmd(2)]~	CPL application layer	A command [args].. are the arguments of each command.
[ETX]	Status code	03h
[sum(2)]	Checksum	Two's complement of addition of bytes from [STX] to [ETX] represented as ACH
[CR]	Frame end code 1	0Dh
[LF]	Frame end code 2	0Ah
[rc(2)]	Status code	*2
[arg]	Command argument	ASCII code (Details depend on each command.)
[data]	Response data	ASCII code (Details depend on each command.)

*1: ACH (Ascii Coded Hexadecimal) 01h..FFh
*2: ACD (Ascii Coded Decimal) 01..99

⚠ Handling Precautions

- Uppercase letters must be used for "A" to "F" of the ACH data.
- {[XX]..} represents that [XX] is repeated.

■ Error handling

In DMC50, the CPL communication errors are classified into the following categories:

- Communication setup errors
- Communication transmission direction errors
- CPL communication frame errors
- CPL application layer errors

● Communication setup errors

If an error exists in the communication setup, the communication is not established and DMC50 does not respond.
The communication setup includes the following:

- Rotary switch on DMC50 COM module
- Rotary switch on DMC50 CTRL module
- Transmission speed

If the communication cannot be performed at all, check the communication setup first.

● Communication transmission direction errors

The CPL communication is half-duplex communication.

When multiple command frames are received, frames other than the last one are disposed of.

If the response is late, the sender of the command should retransmit the same command frame (or send a different command frame) after the timeout error occurs.

The timeout time is recommended 3 sec.

When the response timeout error occurs as a result that the internal process of DMC50 has taken 3 sec. or longer, and if the sender of the command performs a retry (retransmission of the same frame), the COM module will send back an Ack response immediately. This notifies that the last command frame is not disposed of, but the timeout error occurs because it takes too long time. The last command frame (the same command as this time) is still being processed. As soon as the process is completed, the normal response is returned.

● CPL communication frame errors

If a CPL frame is corrupted and cannot be extracted, or if the destination address of a CPL frame is wrong and DMC50 does not accept it, such frame is disposed of. In these cases, no response is sent. If retries due to timeout error are performed a number of times, check if there are any errors in the communication frame of the command.

● **CPL application layer errors**

If an error occurs in the application layer, error handling depends on each command type and the error. The error handling should be based on the returned response.

If multiple CPL errors occur at the same time, the returned status code is determined with the following priority: (Error codes with 2nd or subsequent priority are not reflected on the status code.)

The priority of status codes ['99'] to ['13'] may vary depending on where the error occurs in the frame parsing.
(A frame is parsed one byte by one byte from the top.)

Status code priority for write commands:

(['99']>['10']>['13'])≥['40']>['21']>['23']>['22']>['43']

Status code priority for read commands:

(['99']>['10']>['13'])≥['40']>['21']>['23']>['22']

Additionally, if an error may exist in any specified address or in any data range in a command, parsing continues with invalid parts ignored. When reading multiple data, data "00000000" is returned for each invalid data specification. When writing multiple data, if the returned status code is not '00', this indicates that all of data may not be written as intended.
(However, which data is not written will not be notified.)

! Handling Precautions

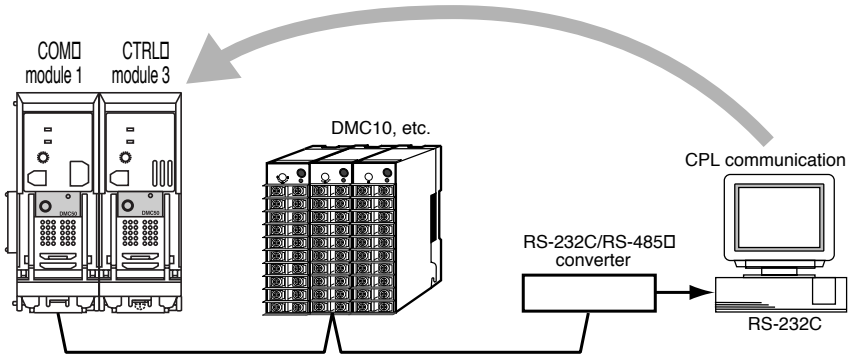
Before the real operation, all the communication errors must be eliminated so that the status code of every communication frame becomes '00'.

■ CPL address

Two kinds of addresses must be specified on the CPL frame, "sta(2)" and "sub(2)". These addresses are used to identify a communication destination unit.

• Address specification for indirect access via COM module

Token expression	Name	Description
sta(2)	CPL station address	• This address identifes a COM module in the host CPL communication. • The value must be same as the rotary switch position number (1..F) of the COM module.
sub(2)	CPL sub-address	• It is used to identify a module in the DMC50 backplane communication. • This CPL sub-address is the same as the rotary switch position number of each module (1..F). • When the sub-address is "0", this specifies the COM module itself.



Example: Specifying a CTRL module with module address 2 connected to a COM module with module address 10:

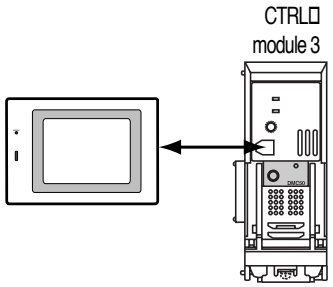
```
sta='0A' sub='02'
```

Specifying a COM module with module address 2:

```
sta='02' sub='00'
```

• Address specification for direct access

Token expression	Name	Description
sta(2)	CPL station address	• This address Identifes the module to connect to. • The value must be same as the rotary switch position number (1..F) of the module.
sub(2)	CPL sub-address	This must be specified "00". (Any value other than "00" is ignored.)



Example: Specifying a CTRL module with module address 3

```
sta='03' sub='00''
```

■ Details of commands

DMC50 supports the following commands:

The response to each command includes the status code of the application layer in [rc(2)].

The possible status codes are determined by each command. See the subsection on each command for details.

Command	Description
RG	Read Data – 32-bit, continuous (ASCII HEX)
WG	Write Data – 32-bit, continuous (ASCII HEX)
RN	Read Data – 32-bit, separate (ASCII HEX)
WN	Write Data – 32-bit, separate (ASCII HEX)
RD	Read Data – 16-bit, continuous (ASCII HEX)
WD	Write Data – 16-bit, continuous (ASCII HEX)
RS	Read Data – 16-bit, continuous (ASCII decimal)
WS	Write Data – 16-bit, continuous (ASCII decimal)
RU	Read Data – 16-bit, separate (ASCII decimal)
WU	Write Data – 16-bit, separate (ASCII decimal)

■ 16-bit commands

When using a 16-bit access command, it is necessary to copy 32-bit addressed Parameter data values to the 16-bit address space in the ISaGRAF application. This 16-bit address space is an area where both the ISaGRAF application and 16-bit CPL commands can access.

! Handling Precautions

If you newly write a communication program, use 32-bit access commands.

16-bit access commands are provided for connections from existing Yamatake's units.

RG

■ RG command (Read Data – 32-bit, continuous (ASCII HEX))

This command reads the specified number of 32-bit data values from the specified start address. The maximum number of data elements to be read is 50.

Command:

[STX][sta(2)][sub(2)]['X']['RG']['LL'][adr(8)][len(4)][ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)]{[data(8)]..}[ETX][sum(2)][CR][LF]

[adr(8)] → Start data address (Parameter Address or Network Address) in ACH

[len(4)] → Number of data elements (upto 50 data elements) (in ACH)

[data(8)] → A data value stored at one data address (in ACH)

['LL'] → Size specifier ('LL' must be specified.)

The RG command's response has any of the following status codes:

Code	Error Name	Possible cause
['10']	Parameter error	A value other than ['LL'] is specified for the size specifier [ss(2)]. The bit length of the 32-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['23']	Calculation parameter error	A Calculation Parameter element at address "20100001h" or above is accessed when access is prohibited (such as during updates).
['40']	Number of read data error	The number of read data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you receive this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'RG'.)

! Handling Precautions

REAL type data used in DMC50 is floating point data in the IEEE754 format. On the other hand, DINT type data is fixed decimal point data (no decimal point) that a negative value is in two's complement form.

Care should be taken on data types when accessing data using a 32-bit command.

■ WG command (Write Data – 32-bit, continuous (ASCII HEX))

This command writes the specified number of 32-bit data values from the specified start address. The maximum number of data elements to be written is 50.

Command:

[STX][sta(2)][sub(2)]['X']['WG']['LL'][adr(8)]{[arg(8)]..}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][ETX][sum(2)][CR][LF]

[adr(8)] → Start data address (Parameter Address or Network Address) (in ACH)

[arg(8)] → A data value to be stored at one data address. (upto 50 data elements) (in ACH)

['LL'] → Size specifier (Only 'LL' is supported.)

The WG command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	A value other than ['LL'] is specified for the size specifier [ss(2)]. The bit length of the 32-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present..
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Write data out of range	Data to be written at the given address does not match the data type. Data to be written to the Date and Time Setup Parameter is in incorrect format. Data to be written at the specified address is out of range.
['23']	Write prohibited	Data is tried to be written into a write prohibited Parameter element. Data is tried to be written into a Parameter element, to which access is prohibited in Operating mode.
	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of write data error	The number of write data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you receive this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of sending 'WG'.)

! Handling Precautions

REAL type data used in DMC50 is floating point data in the IEEE754 format. On the other hand, DINT type data is fixed decimal point data (no decimal point) that a negative value is in two's complement form.
Care should be taken on data types when accessing data using a 32-bit command.

RN

■ RN command (Read Data – 32-bit, separate (ASCII HEX))

This command reads 32-bit data values from the specified addresses. As an address is specified for each data element, it is possible to read multiple data values from separate addresses. The maximum number of data elements to be read is 50.

Command:

[STX][sta(2)][sub(2)]['X']['RN']['00']['LL']{[adr(8)]…}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)]{[data(8)]..}[ETX][sum(2)][CR][LF]

['00'] → Registration for repeat (Not supported; '00' must be specified.)
[adr(8)] → Data address for one element (upto 50 data elements) (in ACH)
[data(8)] → Data value stored at one data address (in ACH)
['LL'] → Size specifier ('LL' must be specified.)

The RN command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	A value other than ['LL'] is specified for the size specifier [ss(2)]. The bit length of the 32-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['23']	Calculation parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of read data error	The number of read data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'RN'.)

! Handling Precautions

REAL type data used in DMC50 is floating point data in the IEEE754 format. On the other hand, DINT type data is fixed decimal point data (no decimal point) that a negative value is in two's complement form.
Care should be taken on data types when accessing data using a 32-bit command.

■ WN command (Write Data – 32-bit, separate (ASCII HEX))

This command writes 32-bit data values at the specified addresses. As an address is specified for each data element, it is possible to write multiple data values to separate addresses. The maximum number of data elements to be written is 25.

Command:

[STX][sta(2)][sub(2)]['X']['WN']['00']['LL']{[adr(8)][arg(8)]...}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][ETX][sum(2)][CR][LF]

['00'] → Registration for repeat (Not supported; '00' must be specified.)
[adr(8)] → Data address for one element (upto 25 data elements) (in ACH)
[arg(8)] → Data value to be stored at one data address. (upto 25 data elements) (in ACH)
['LL'] → Size specifier ('LL' must be specified.)

The WN command's response has any of following status codes:

Code	Error name	Possible cause
['10']	Parameter error	A value other than ['LL'] is specified for the size specifier [ss(2)]. The bit length of the 32-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Write value out of range	Data to be written at a given address does not match the data type. Data to be written to the Date and Time Setup Parameter is in incorrect format. Data to be written at a specified address is out of range.
['23']	Write prohibited	Data is tried to be written into a write prohibited Parameter element. Data is tried to be written into a Parameter element, to which access is prohibited in operating mode.
	Calculation Parameter error	A Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of write data error	The number of write data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'WN'.)

⚠ Handling Precautions

REAL type data used in DMC50 is floating point data in the IEEE754 format. On the other hand, DINT type data is fixed decimal point data (no decimal point) that a negative value is in two's complement form.
Care should be taken on data types when accessing data using a 32-bit command.

RD

■ RD command (Read Data – 16-bit, continuous (ASCII HEX))

This command reads the specified number of 16-bit data values from the specified start address. The maximum number of data elements to be read is 50.

Command:

[STX][sta(2)][sub(2)]['X']['RD'][adr(4)][len(4)][ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)]{[data(4)]..}[ETX][sum(2)][CR][LF]

[adr(4)] → Start data address (Network Address) (in ACH)
[len(4)] → Number of data elements (upto 50 data elements) (in ACH)
[data(4)] → Data value stored at one data address (in ACH)

The RD command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	The bit length of the 16-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Read data out of range	Data value to be read is out of the range from -32768 to +32767. Data value out of this range is read as a maximum (32767: 7FFFh) or minimum value (-32768: 8000h).
['23']	Calculation parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of read data error	The number of read data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'RD'.)

! Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

■ WD command (Write Data – 16-bit, continuous (ASCII HEX))

This command writes the specified number of 16-bit data values from the specified start address. The maximum number of data elements to be written is 50.

Command:

[STX][sta(2)][sub(2)]['X']['WD'][adr(4)]{[dat(4)]...}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][ETX][sum(2)][CR][LF]

[adr(4)] → Start data address (Network Address) (in ACH)

[arg(4)] → Data value to be stored at one data address. (upto 25 data elements) (in ACH)

The WD command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	The bit length of the 16-bit data/address is incorrect (the number of bytes is incorrect.). Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command is accessed.
['23']	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during updates).
['40']	Number of write data error	The number of write data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'WD'.)

⚠ Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

RU

■ RU command (Read Data – 16-bit, separate (ASCII HEX))

This command reads 16-bit data values from the specified addresses. As an address is specified for each data element, it is possible to read multiple data values from separate addresses. The maximum number of data elements to be read is 50.

Command:

[STX][sta(2)][sub(2)]['X']['RU']['00']{[adr(4)]...}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)]{[data(4)]..}[ETX][sum(2)][CR][LF]

['00'] → Registration for repeat (Not supported; '00' must be specified.)
[adr(4)] → Data address for one element (upto 50 data elements) (in ACH)
[arg(4)] → Data value stored at one data address. (in ACH)

The RU command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	A value other than ['00'] is specified for the registration for repeat ['00']. Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped.
	Access data type error	STRING type data not supported by this command (address below 10000h) is accessed.
['22']	Read data out of range	Data value to be read is out of the range from -32768 to +32767. Data value out of this range is read as a maximum (32767: 7FFFh) or minimum value (-32768: 8000h).
['23']	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of read data error	The number of read data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent insteat of 'RU'.)

! Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

■ WU command (Write Data – 16-bit, separate (ASCII HEX))

This command writes 16-bit data values at the specified addresses. As an address is specified for each data element, it is possible to write multiple data values to separate addresses. The maximum number of data elements to be written is 25.

Command:

[STX][sta(2)][sub(2)]['X']['WU']['00']{[adr(4)][arg(4)]…}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][ETX][sum(2)][CR][LF]

['00'] → Registration for repeat (Not supported; '00' must be specified.)
[adr(4)] → Data address for one element (in ACH)
[arg(4)] → Data value to be stored at one data address. (upto 25 data elemets) (in ACH)

The WU command's response has any of the following end codes:

Code	Error name	Possible cause
['10']	Parameter error	A value other than '00' is specified for the registration for repeat. Data other than alphanumeric data ('0' to '9' and 'A' to 'F') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped. An address at which no parameters exist is accessed.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Write value out of range	Data to be written at a given address does not match the data type. Data to be written to the Date and Time Setup Parameter is in incorrect format. Data to be written at a specified address is out of range.
['23']	Write prohibited	Data is tried to be written into a write prohibited Parameter element. Data is tried to be written into a Parameter element, to which access is prohibited in operating mode.
	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of write data error	The number of write data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'WU'.)

! Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

RS

■ RS command (Read Data – 16-bit, continuous (ASCII decimal))

This command reads the specified number of 16-bit data values from the specified start address. The maximum number of data elements to be read is 50.

Command:

[STX][sta(2)][sub(2)]['X']['RS'][','][adr(a)]['W'][','][len(b)][ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][',']{[data(c)][','].}[ETX][sum(2)][CR][LF]

[adr(a)] → Start data address (32 bits) 0 to 4,294,967,295 (in ACD)
[len(b)] → Number of data elements (upto 50 data elements) (in ACD)
[data(c)] → Data value at one data address is signed 16-bit data (in ACD)

The RS command's response has any the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	Data other than numeric data ('0' to '9') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed while operation of ISaGRAF is stopped. An address at which no parameters exist is accessed.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Read data out of range	Data value to be read is out of the range from -32768 to +32767. Data value out of this range is read as a maximum (32767: 7FFFh) or minimum value (-32768: 8000h).
['23']	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of read data error	The number of read data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'RS'.)

! Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

■ WS command (Write Data – 16-bit, continuous (ASCII decimal))

This command writes the specified number of 16-bit data values from the specified start address. The maximum number of data elements to be written is 50.

Command:

[STX][sta(2)][sub(2)]['X']['WS'][','][adr(a)]['W'][',']{[arg(b)][',']..}[ETX][sum(2)][CR][LF]

Response:

[STX][sta(2)][sub(2)]['X'][rc(2)][ETX][sum(2)][CR][LF]

[adr(a)] → Start data address (32 bits) 0 to 4,294,967,295 (in ACD)

[arg(b)] → Data value to be stored at one data address (upto 50 data elements); signed 16-bit data (in ACD)

The WS command's response has any of the following status codes:

Code	Error name	Possible cause
['10']	Parameter error	Data other than numeric data ('0' to '9') is used.
['13']	Command execution error	Timeout error occurs while the data is being transferred internally. The cycle timing setting of ISaGRAF is too short. The command is sent to a CTRL module that is not present.
['21']	Address error	A variable (address below 10000h) is accessed when operation of ISaGRAF is stopped. An address at which no parameters exist is accessed.
	Access data type error	STRING type data not supported by this command is accessed.
['22']	Write value out of range	Data to be written at a given address does not match the data type. Data to be written to the Date and Time Setup Parameter is in incorrect format. Data to be written at a specified address is out of range.
['23']	Write prohibited	Data is tried be written into a write prohibited Parameter element. Data is tried to be written into a Parameter element, to which access is prohibited in operating mode.
	Calculation Parameter error	Calculation Parameter is accessed when access is prohibited (such as during data updates).
['40']	Number of write data error	The number of write data is too many or is 0.
['80']	CPL Ack response	When the last issued command is retransmitted, and if you get this response, it indicates the last command is still being processed.
['99']	Undefined command	Unsupported command is received. (Wrong command may be sent instead of 'WS'.)

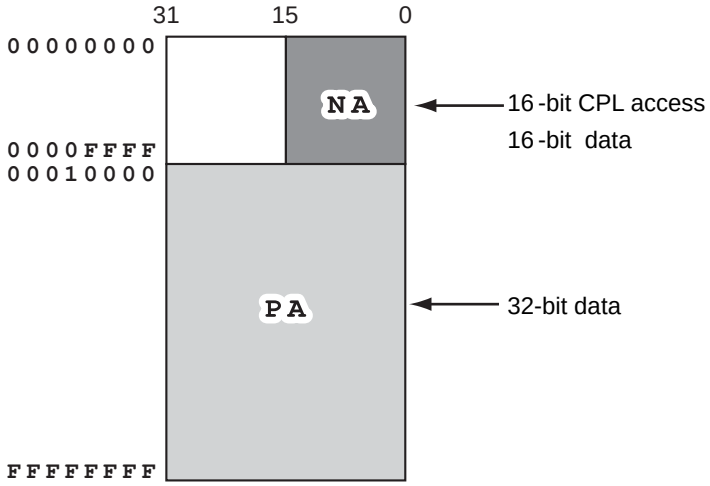
⚠ Handling Precautions

- 16-bit commands are used only when connecting from Yamatake's units.
- When using 16-bit commands, always use "Integer Conversion Wizard" to create data, which can be accessed by 16-bit commands.

Chapter 5. CONNECTIONS WITH EST-Z Series

■ Integer Conversion Wizard

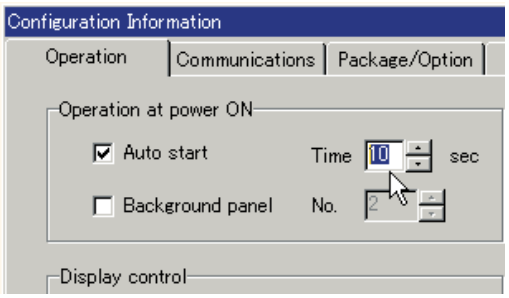
Yamatake's operator interface terminal, smart terminal EST-Z Series, does not support 32-bit CPL commands. 16-bit CPL commands are automatically used to connect from the EST-Z Series.



In DMC50, it is necessary to prepare data for access from EST in the area (NA area), from which the EST-Z Series can read data.
Use of "Integer Conversion Wizard" of the smart loader package SLP-D50 makes it possible to omit the ISaGRAF programming for data creation.
For details,
☞ refer to Smart Loader Package SLP-D50 for Module Type Controller DMC50 User's Manual, CP-SP-1122E.

■ About [Auto Start Time]

It takes more time to start up DMC50 than to start up EST. Set up [Auto Start Time] for EST with WinAPE . Normmaly, you need to specify around 10 to 15 seconds.
[Auto Start Time] is found in [Configuration Information] → [General] → [Operation] → [Operation at Power ON]



! Handling Precaution

It takes more time to start up DMC50 than the EST's default [Auto Start Time] .
EST will not wait until DMC50 starts operation, and will repeat retries while DMC50 is still in the startup process and cannot respond to commands.
If EST receives no responses after some retries, EST will limit the CPL commands used for the following communications to RS and WS commands.
It needs longer time for DMC50 to respond to these commands, which can cause timeout errors.
It is strongly recommended that you set up [Auto Start Time] in EST appropriately.

Appendix

■ ISaGRAF variable data types

The following shows the ISaGRAF variable data types:

Data type	Contents	Range
BOOL	Boolean value	0 or 1 (0: FALSE, 1: TRUE)
DINT	Double-precision integer	-2147483647 to +2147483647
REAL	Real number	Approximately 10^{-38} to 10^{38} (Maximum precision of 7-digits in decimal)
TIME	Time type data	0 to T#23h59m59s999ms
STRING *	Character string	Variable length, maximun of 255 single byte characters

* This type of data cannot be accessed through the CPL communication.

Note

- The ISaGRAF variables are available only on the CTRL module and not available on the COM module.
- You must assign a data type for each variable to be used in your programs with the dictionary editor of the ISaGRAF workbench while editing your project.
- The BOOL type is also refered to as "Boolean type" in ISaGRAF.
- The DINT type is 32-bit signed integer data, which is also refered to as "integer type" in ISaGRAF. Note that the range starts from "-2147483647".
- The REAL type is 32-bit floating-point (single-precision) data defined in IEEE754.
- The TIME type is 32-bit unsigned integer data (in units of millisecond), which is also refered to as "timer type" in ISaGRAF.
- The STRING type is also refered to as "variable-length character string type" (or character string type or MESSAGE type) in ISaGRAF. This data cannot be accessed through the CPL communication.

■ Parameter data types

The following shows the data types for each element of Parameter, data specially for DMC50 only:

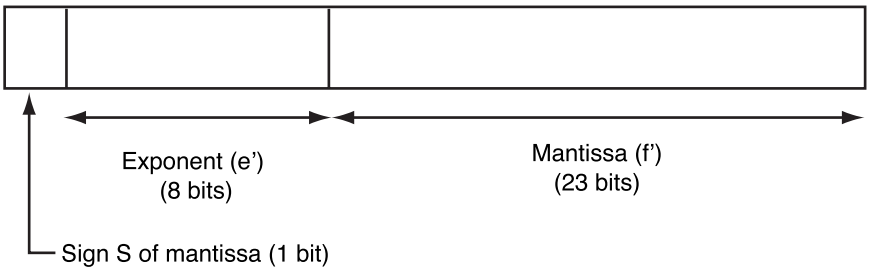
Data type	Contents	Range
BOOL	Boolean value	0 or 1 (0: FALSE, 1: TRUE)
DINT	Double-precision integer	-2147483647 to +2147483647
DWORD	32-bit binary data	0 to 16#FFFFFFFF
REAL	Real number	Approximately 10^{-38} to 10^{38} (Maximum of 7-digits in decimal)

Note

- Parameters are classified into as follows:
  - · System Parameters
  - · Calculation Parameters
  - · System Monitor Parameters
  - · Calculation Monitor Parameters
  - · User defined Parameters
- For the data type of a specific item, refer to the corresponding Parameter table.
- The DINT type is 32-bit signed integer data. It starts from "-2147483647".
- The DWORD type is 32-bit data. This data is normally used for handling bit data, etc. as binary images.
- The REAL type is 32-bit floating-point (single precision) data defined in IEEE 754.

■ Floating point format: IEEE754 format

Floating point format is defined by IEEE754 (Institute of Electrical and Electronics Engineers); 64-bit and 32-bit formats are provided.
DMC50 uses the 32-bit format.
A numeric value is represented in the exponential format "$f \times (2^e)$" where $1 \leq f < 2$ and “f” and “e” are stored in relevant set of bits in a 32 bit entity.
(f: Mantissa, e: Exponent)
The following Figure shows the meaning of each set of bits:



The exponent (e’) is a value that "127" is added to the value of "e".
The mantissa (f’) is a value that "1" is removed from "f" and the binary fraction is left-justified in the mantissa.

Example: 100 in decimal
100 (decimal)
= 64 (hexadecimal)
= 1100100 (binary)
1.1001 (binary) x (2 (decimal)) 6 (decimal)

The exponent is:
6 (decimal) + 127 (decimal)
= 133 (decimal)
= 10000101 (binary).

The mantissa is:
10010000000000000000 (binary).

Consequently, the total expression is:
01000010110010000000000000000000 (binary)
= 42C80000h (hexadecimal)

● Non-numeric value

Normally, the calculation result does not become non-numeric value. However, if the data is damaged by the external cause (e.g. overwritten by the communication), the result may become non-numeric value. Non-numeric value occurrence conditions are that "e" equals "255" and "f" does not equal "0" as described in the following:

- e’ = 255, f’ ≠ 0 : Non-numeric value (NAN: Not-a-Number)
- e’ = 255, f’ = 0 : $(-s) \cdot \infty$ (Infinity)
- $0 < e < 255$: $(-s) \cdot (1.f') \cdot 2^{e+127}$ (Normal number)
- e’ = 0, f’ ≠ 0 : $(-s) \cdot (0.f') \cdot 2^{e-126}$ (Subnormal number)
- e’ = 0, f’ = 0 : $(-s) \cdot (0.f') \cdot 0$ (Zero)

■ Operations of 16-bit commands

Command	Action			Description
RD, RU	Accessible data address range			NA only: 0000h to FFFFh
	Operations	Integer type data read		Data at the specified address is converted into 16-bit data. If data value out of the range from -32768 to +32767 is read, status code of the response is set to error 22.
		Real type data read (Not supported)		Real type data cannot be read. If real type data is read, undefined value is returned.
		Example (for Integer type)		Inside of DMC50 → In a communication frame FFFF8000h (-32768) → 8000h 00007000h (28672) → 7000h 00008000h (32768) → 7FFFh (Out of range error 22 occurs.) FFFF7000h (-36864) → 8000h (Out of range error 22 occurs.)
WD, WU	Accessible data address range			NA only: 0000h to FFFFh
	Operations	Integer type data write		If integer type data out of the range form -32768 to +32767 is specified for the write data, error 10 occurs and this data cannot be written.
		Real type data write (Not supported)		Real type data cannot be written. When the data in DMC50 is real type data, the data will be corrupted. Never attempt to write to real type data.
		Example (for Integer type)		In a communication frame → Inside of DMC50 8000h (-32768) → FFFF8000h 7000h (28672) → 00007000h
RS	Accessible data address range			NA, PA: 0W to 4294967295W (00000000h to FFFFFFFFh)
	Operations	Reading of data from NA area (00000000h to 0000FFFFh)	Integer type data read	Data at the specified address is converted into 16-bit data. If data value out of the range from -32768 to +32767 is read, status code of the response is set to error 22.
			Real type data read (Not supported)	Real type data cannot be read. If real type data is read, undefined value is returned.
			Example (for Integer type)	Inside of DMC50 → In a communication frame FFFF8000h (-32768) → -32768 00007000h (28672) → 28672 00008000h (32768) → 32767 (Out of range error 22 occurs.) FFFF7000h (-36864) → -32768 (Out of range error 22 occurs.)
		Reading of data from PA area (00010000h to FFFFFFFFh)	Integer type data read	Data type at the specified address is checked whether it is integer or real, and the data is converted into 16-bit data. If data value out of the range from -32768 to +32767 is read, status code of the response is set to error 22.

Command	Action			Description
RS	Operations	Reading of data from PA area (00010000h to FFFFFFFFh)	Real type data read	Data type at the specified address is checked whether it is integer or real, and the data is converted into 16-bit data. If data out of the range from -32768 to +32767 is read, status code of the response is set to error 22. If real type data is read, the data as a fixed point decimal with one decimal place.
			Example (for Real type)	Inside of DMC50 → In a communication frame C4FA0000h (-2000.0000) → -20000 44FA0000h (2000.0000) → 20000
WS	Accessible data address range			NA, PA: 0W to 4294967295W (00000000h to FFFFFFFFh)
	Operations	Writing of data to NA area (00000000h to 0000FFFFh)	Integer type data write	Data at the specified address is converted into 16-bit data. If data out of a range from -32768 to +32767 is read, status code of the response is set to error 10.
			Real type data write (Not supported)	Real type data cannot be written. When the data in DMC50 is real type data, the data will be corrupted. Never attempt to write real type data.
			Example (for Integer type)	In a communication frame → Inside of DMC50 -32768 → FFFF8000h 28672 → 00007000h
		Writing of data to PA area (00010000h to FFFFFFFFh)	Integer type data write	Data type at the specified address is checked whether it is integer or real, and the data is converted into 16-bit data. If data out of the range from -32768 to +32767 is handled read, status code of the response is set to error 10.
			Real type data write	Data type at the specified address is checked, and the data is converted into 16-bit data. If data out of the range from -32768 to +32767 is read, status code of the response is set to error 10. If real type data is wrriten, the data is written as a fixed point decimal with one decimal place.
			Example (for Integer type)	In a communication frame → Inside of DMC50 10000 → 447A0000h (1000.0000) -10000 → C47A0000h (-1000.0000)

Revision History

Printed Date	Manual Number	Edition	Revised pages	Description
02-12	**CP-SP-1093E**	**1st Edition**		

Specifications are subject to change without notice.

YAMATAKE

Yamatake Corporation
Control Products Division
Head office : Totate International Building
2-12-19 Shibuya Shibuya-ku Tokyo 150-8316 Japan
Inquiries to : International Business Division
Phone : 81-3-3486-2331, Fax : 81-3-3486-2300 (Sales)
Phone : 81-466-20-2307, Fax : 81-466-27-9264 (Customer Service)
http://www.yamatake.com

Printed in Japan.
1st Edition: Issued in Dec., 2002(W)

This has been printed on recycled paper.