SCARA Robot

KSL3000 Communication Manual

INSTRUCTION MANUAL

SM-A20051-A

- Read this Instruction Manual before using the product.
- Read the safety notes carefully.
- Keep this Instruction Manual in a safe and convenient place for future reference.

CKD Corporation

Preface

This manual describes serial communication and Ethernet communication between the robot controller and peripheral devices. It covers such subjects as connecting communication channels and setting communication modes. It also describes communication protocols, communication commands and data format, and presents information on how to operate the robot with data communication and how to handle communication dialogue with controller programs.
Before reading this manual, we ask that you first read and understand the contents of the following user manuals.

- Operator's Manual
- Robot Language Manual
- Interface Manual

This manual is divided into following six (6) sections:

Section 1 Introduction
This section presents and introduce to the communication functions provided by the TS3000 Series robot system.

Section 2 COM1 Port and HOST Port Specifications and Settings
This section describes the RS-232C port hardware interfaces, communication modes, and other information.

Section 3 Ethernet Specifications and Settings
This section describes the Ethernet hardware interfaces, communication modes, and other information.

Section 4 Non-Protocol Communication
This section describes the protocol for communication between an external device and the robot program only.

Section 5 Simple Protocol Communication
This section describes the communication protocol, commands, and robot program files for communication with the host computer.

Section 6 Robot Operation Sequence Using External Communication
This section shows examples of methods for operating the robot.

Table of Contents

Page

Section 1

Introduction

This manual describes serial communication and Ethernet communication between the robot controller and an external computer. There are two procedures for data transmission: non-protocol communication and simple protocol communication, each of which can achieve the following functions.

Non-protocol communication

(1) Input variables into the program from the external device.
(2) Output variables or messages in the program.

Simple protocol communication

(1) Start and stop operation of the robot
(2) Program reset, step reset, cycle reset and output signal reset
(3) Servo ON/OFF
(4) Upload and download program files
(5) Erase program files
(6) Select programs
(7) Monitor the status of the robot
(8) Monitor for robot errors

The communication interface configuration is as shown below.

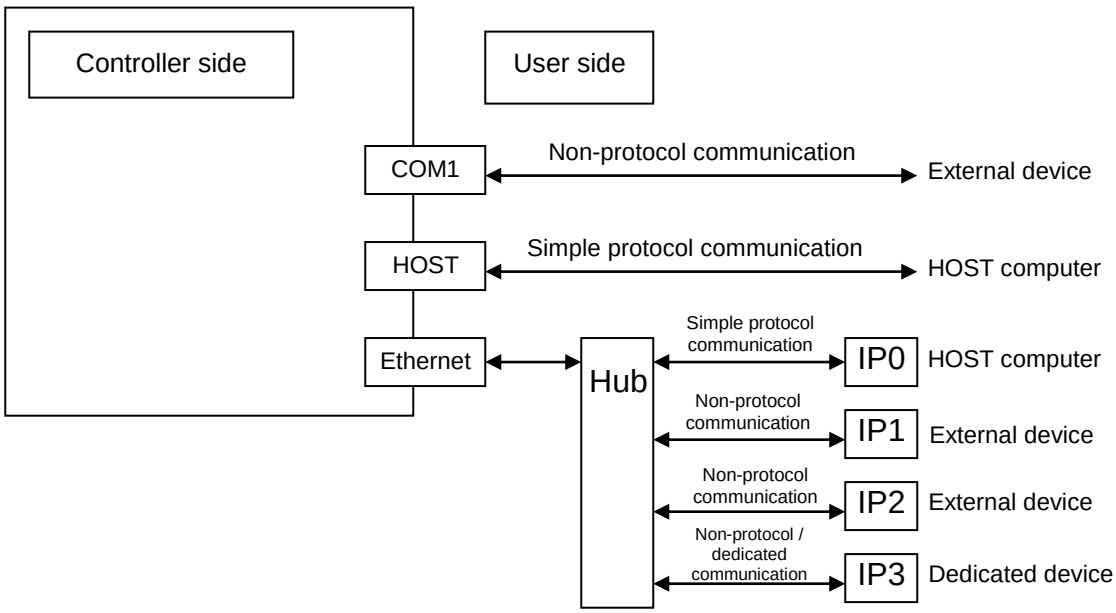


Fig. 1.1 Communication interface configuration

1.1 Status Transitions

The status transition diagram of the controller is as shown below in Figure 1.2.
According to the status of the controller, the status of a command changes between valid and invalid.
Refer to Section 5.3.2 to find out whether each command is valid or invalid for specific status of the controller.

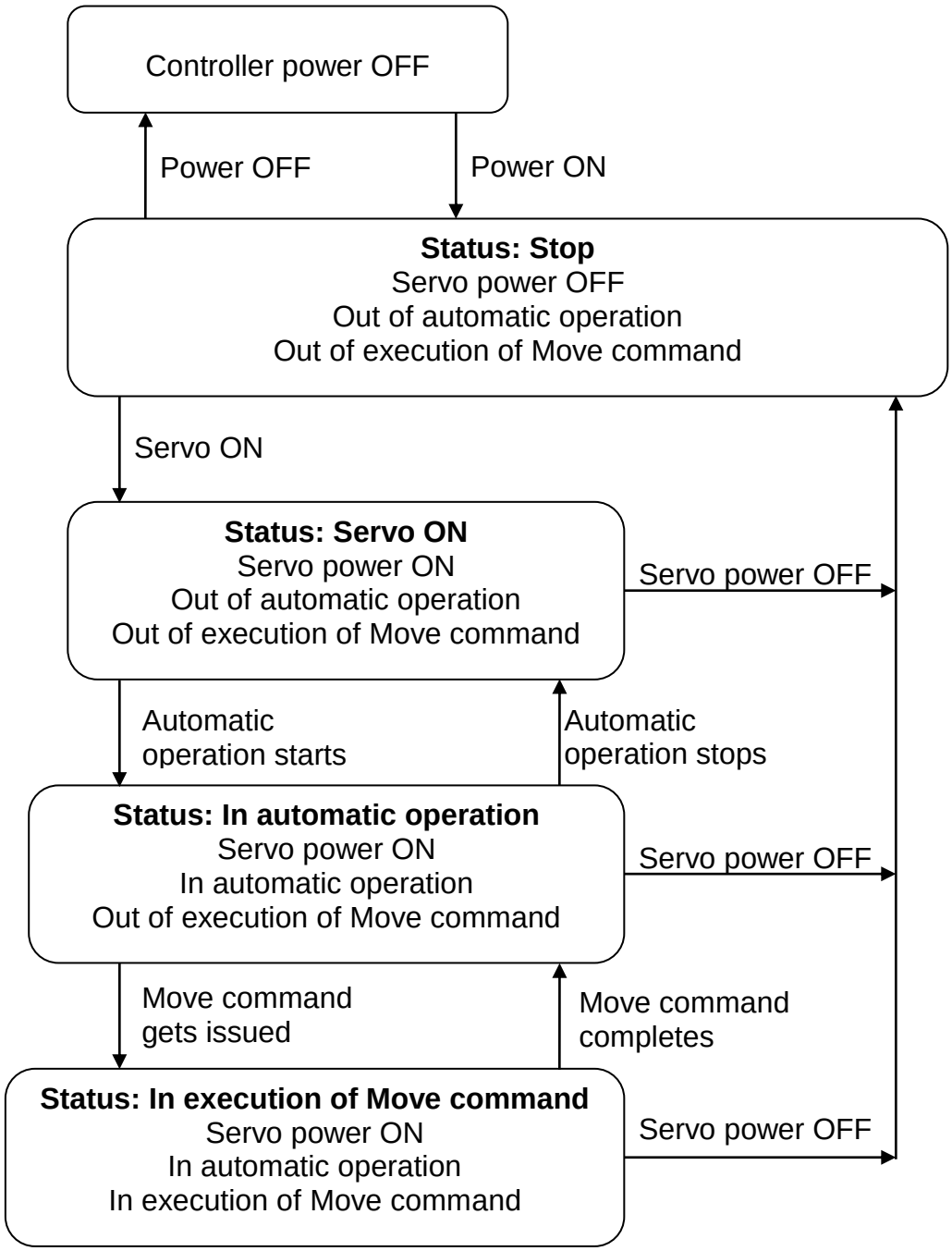


Figure 1.2 Status transition diagram

Section 2

COM1 Port and HOST Port Specifications and Settings

The KSL3000 controller has two (2) serial communication ports. The COM1 port is used for non-protocol communication, being connected to external devices, and the HOST port is used for protocol communication and communicates with the HOST computer in simple protocol communication.

2.1 Connections

The COM1 port and HOST port should be connected, using an RS-232C cross connect cable (D-SUB: 9-pin). (For details, see the Interface Manual.)

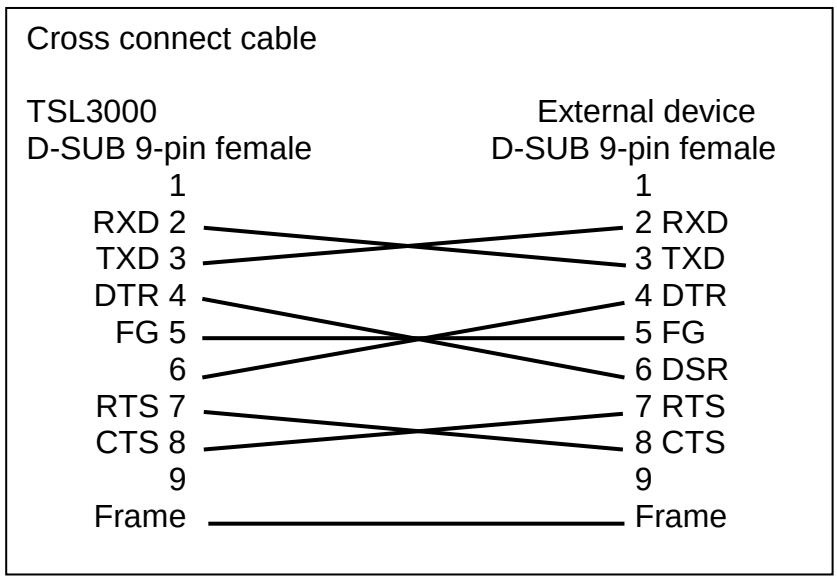


Fig. 2.1 Pin arrangement for cross connect cable

2.2 Communication Specifications

2.2.1 COM1 Port

Table 2.1 COM1 port communication specifications

Item	Specifications
Interface	RS-232C
Synchronous system	Start-stop synchronization system
Communication system	Full duplex system
Communication rate	300, 600, 1200, 2400, 4800, 9600, 19200, 38400 bps
Data format	ASCII code
No. of transfer bytes	Max. 256 bytes
Transfer data structure	Character length: 7 or 8 bits Parity: None, odd, even Stop bit: 1 or 2 bits
Flow control	None
Protocol	Non-protocol communication
Timeout	No timeout
Fault recovery	No special protocols are used.

Note: The communication rate and transfer data structure are specified in the user parameter file. (For details, see Section. 2.3.)

2.2.2 HOST Port

Table 2.2 HOST port communication specifications

Item	Specifications
Interface	RS-232C
Synchronous system	Start-stop synchronization system
Communication system	Half-duplex system
Communication rate	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 115200 bps
Data format	ASCII code
No. of transfer bytes	Max. 255 bytes
Transfer data structure	Character length: 7 or 8 bits Parity: None, odd, even Stop bit: 1 or 2 bits
Flow control	None
Protocol	Simple protocol communication
Timeout	10 sec. (character receiving intervals)
Fault recovery	Command is re-sent from the host computer. No special protocols are used.

Note: The communication rate and transfer data structure are specified in the user parameter file. (For details, see Section. 2.3.)

2.3 Communication Parameters

The communication speed and transmission data configuration of the COM1 and HOST ports are defined in the user parameter file (USER.PAR) stored in the robot controller. Communication is performed according to the contents defined under [U06] of the user parameter file (USER.PAR) in the RAM drive.

[U06] Serial port setting

Specify the communication ports [COM1 port, HOST port].

KSL3000

```
[U06] Serial port setting
{Speed } (38400, 19200, 9600, 4800, 2400, 1200)
{Character} (7, 8)
{Parity } (0: Without, 1: Odd, 2: Even)
{Stop bit } (1, 2)
{COM1}
= 9600   8   0   1
{HOST}
= 9600   8   0   1
```

{COM1}

= (Speed) (Character length) (Parity) (Stop bit length) ← COM1 port setting

{HOST}

= (Speed) (Character length) (Parity) (Stop bit length) ← HOST port setting

(Speed) : Select the speed for data communication from the following six (6) rates.

115200	:	115200 bps	* Host port only
38400	:	38400 bps	
19200	:	19200 bps	
9600	:	9600 bps	
4800	:	4800 bps	
2400	:	2400 bps	
1200	:	1200 bps	

(Character length) : Specify the length of characters to be transferred.

8 : 8 bits

7 : 7 bits

(Parity) : Specify the parity of characters to be transferred.

0 : Without parity

1 : Odd-number parity

2 : Even-number parity

(Stop bit length) : Specify the stop bit length of characters to be transferred.

1 : Stop bit 1

2 : Stop bit 2

Set value

Example: KSL3000

{COM1}
= 9600 8 0 1
{HOST}
= 38400 8 1 1

"Speed 9,600 bps, character length 8 bits, without parity and stop bit 1" are specified for the COM1 port.

"Speed 38,400 bps, character length 8 bits, even-number parity and stop bit 1" are specified for the HOST port.

Section 3

Ethernet Specifications and Settings

The KSL3000 controller has one Ethernet connector.
IP0 can communicate with the host computer using simple protocol, IP1 and IP2 can communicate with an external device using non-protocol communication, and IP3 can communicate with the host computer or a dedicated device using non-protocol or dedicated communication.
This function uses TCP/IP protocol to enable communication on the same local area network with computers and visual equipment on the Ethernet.

3.1 Connections

If a hub is used to connect to an external device, connect using a straight cable. Use a cross connect cable when connecting a computer directly to the controller. Some devices have the automatic recognition function so that either a straight cable or cross connect cable can be used, and so check the specifications of the device that is used.

3.2 Communication Specifications

Table 3.1 Communication specifications

Item	Specification
Interface	10BASE-T
Communication system	TCP
Communication rate	10 Mbps
Maximum packet size	512 bytes/packet
Number of simultaneous connectable ports	4
Applications	IP0: Simple protocol communication IP1: Non-protocol communication IP2: Non-protocol communication IP3: Non-protocol / dedicated communication

3.3 Communication Parameters

The Ethernet communication settings are defined in the Ethernet parameter file (ETHERNET.PAR) stored in the robot controller.

[E00] Open mode IP0 IP1 IP2 IP3

```
[E00] Open mode IP0 IP1 IP2 IP3
{ 0: non   1: Robot is TCP server   2: Robot is TCP client   }
=   1   0   0   0
```

IP0 is for simple protocol only, and so the server mode must be specified.
If the external device is a server, the controller uses client settings, and if the external device is a client, the controller uses server settings.

=(IP0) (IP1) (IP2) (IP3)

This enables or disables the Ethernet function.
0: Not used
1: Operates as a server
2: Operates as a client

[E01] Robot controller IP address

```
[E01] Robot controller IP address
=   192.168.0.124
```

This sets the IP address of the robot controller itself.
The IP address is an address for identifying the controller.
The IP address consists of four numbers ranging from 0 to 255, which are separated by periods (“.”). If the controller is connected directly to a computer on the network, the IP address may remain at the initial value (any setting), but if it is connected to a LAN, the IP address must be set by following the instructions of the user's LAN system administrator. This function will not work properly if any IP addresses are duplicated. The robot controller and computer to communicate with it must be connected on the same network. When sending data to devices on the network, the controller can be operated as a client or a server.
If the external device is a server, use the client setting for the controller, and if the external device is a client, use the server setting for the controller.

= (Robot controller IP address)

[E02] Robot controller name

(Planned for use in function expansion)

[E02] Robot controller name
= “NO1-TS3K”

This parameter specifies the name of the robot controller.
This is planned for use when the Ethernet function is expanded.

[E03] Subnet mask

[E03] Subnet mask
= 255.255.255.0

This parameter specifies the subnet mask of the connected network.
When connecting to a LAN, set the subnet mask by following the instructions of the user's LAN system administrator.
= (Subnet mask)

[E04] Default gateway

[E04] Default gateway
= 192.168.0.1

This parameter specifies the default gateway address.
When connecting to a LAN, set the default gateway by following the instructions of the user's LAN system administrator.
= (Default gateway address)

[05] Own port no

[E05] Own port no
{ IP0 }
= 1000
{ IP1 }
= 1001
{ IP2 }
= 1002
{ IP3 }
= 1003

This parameter specifies each port number that is used by the controller. A port number is similar to a radio frequency. The numbers IP0 to IP3 indicate the port numbers. Be careful that no port numbers are duplicated.

{IP0}
=(Port number used)
{IP1}
=(Port number used)
{IP2}
=(Port number used)
{IP3}
=(Port number used)

[06] Port number of destination

[E06] Port number of destination
{ IP0 }
= 1000
{ IP1 }
= 1001
{ IP2 }
= 1002
{ IP3 }
= 1003

This parameter specifies each port number that is used by the destination. This is used in client mode only. This must match the port number used by the server at the destination.

{IP0}
=(Port number of destination)
{IP1}
=(Port number of destination)
{IP2}
=(Port number of destination)
{IP3}
=(Port number of destination)

[E07] IP address of destination

```
[E07]  IP address of destination
{IP0}
= 192.168.0.150
{IP1}
= 192.168.0.151
{IP2}
= 192.168.0.152
{IP3}
= 192.168.0.153
```

This parameter specifies the IP address of the destination where each port is connected.
This is used in client mode only.

{IP0}
=(IP address of destination)
{IP1}
=(IP address of destination)
{IP2}
=(IP address of destination)
{IP3}
=(IP address of destination)

[E08] Robot community name
(Planned for use in function expansion)

```
[E08] Robot community name
= “NO1-TS3K”
```

This parameter specifies the community name of the network.
This is planned for usage when the Ethernet function is expanded.

[E09] DHCP
(Planned for use in function expansion)

```
[E09] DHCP
= 0
```

This parameter specifies if an address is obtained from the DHCP server.
This is planned for usage when the Ethernet function is expanded.

[E10] TCP status alarm IP0 IP1 IP2 IP3 Ethernet communication alarm

```
[E10] TCP status alarm IP0 IP1 IP2 IP3
{0: NON 1: Lv_1 2: Lv_2 4: LV_4 8: Lv_8}
=  0  0  0  0
```

For the port being used, the alarm level (1Lv, 2Lv, 4Lv, 8Lv) generated when the Ethernet connection is cut off during running of a program can be set.

When the value is 0 (zero), no alarm is generated.

*Parameter (ETHERNET.PAR) setting example

```
[E00] OPEN MODE IP0 IP1 IP2 IP3
{0: NON 1: ROBOT IS TCP SERVER 2: ROBOT IS CLIENT}
= 1    0    0    0
[E01] ROBOT CONTROLLER IP ADDRESS
= 192.168.0.124
[E02] ROBOT CONTROLLER NAME
= "NO1-TS3K"
[E03] SUBNET MASK
= 255.255.255.0
[E04] DEFAULT GATEWAY
= 192.168.0.1
[E05] OWN PORT NO
{IP0}
= 1000
{IP1}
= 1001
{IP2}
= 1002
{IP3}
= 1003
[E06] PORT NUMBER OF DESTINATION
{IP0}
= 1000
{IP1}
= 1001
{IP2}
= 1002
{IP3}
= 1003
[E07] IP ADDRESS OF DESTINATION
{IP0}
= 192.168.0.150
{IP1}
= 192.168.0.151
{IP2}
= 192.168.0.152
{IP3}
= 192.168.0.153
[E08] ROBOT COMMUNITY NAME
= "NO1-TS3K"
[E09] DHCP
= 0
[E10] TCP status alarm IP0 IP1 IP2 IP3
{0: NON    1: Lv_1    2: Lv_2    4: Lv_4    8: Lv_8}
= 0    0    0    0
[END]
```

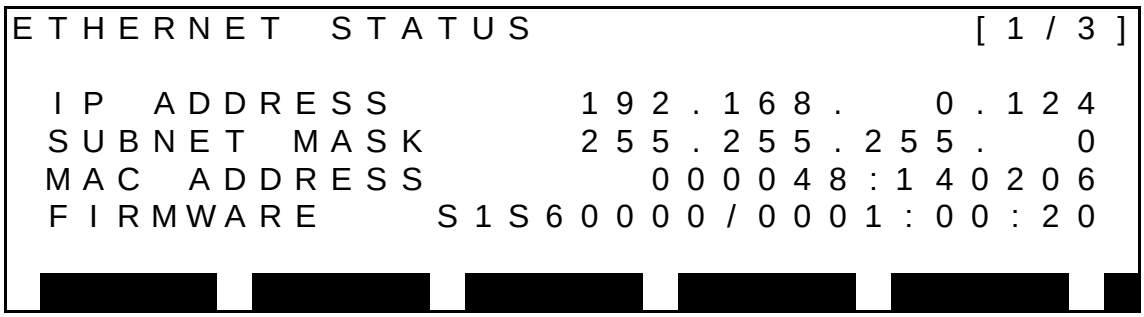
3.3.1 Ethernet Status Display [ETHER]

(1) Function

Displays the Ethernet status.

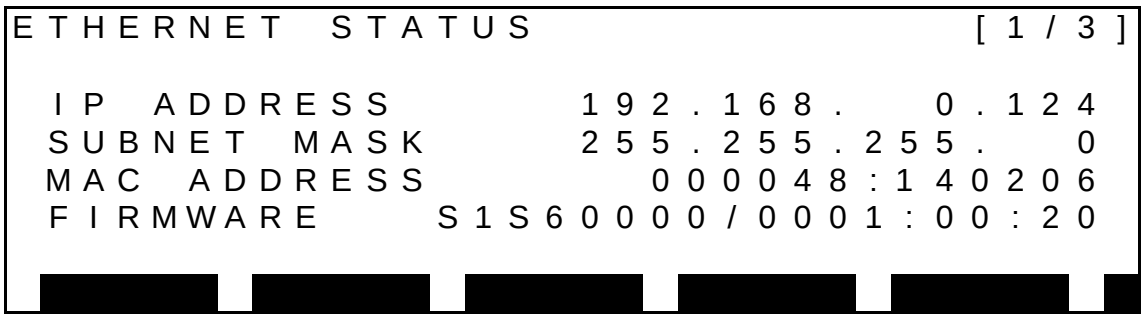
(2) Procedures

1. Press the NEXT key repeatedly until [ETHER] is displayed in the menu, and then press [ETHER].
   The screen shown below is displayed.



2. The display items can be changed by pressing ALT + ⇓ and the NEXT key (or ALT + ⇑ and the ALT + NEXT key).
   The information displayed on each page is shown below.

(Page 1)



- ♦ IP ADDRESS: IP address of the robot controller itself that is written in the "ETHERNET.PAR" [E01] file
- ♦ SUBNET MASK: Subnet mask of the network that is written in the "ETHERNET.PAR" [E03] file
- ♦ MAC ADDRESS: MAC address of the robot controller itself
- ♦ FIRMWARE: Version of the Ethernet unit firmware

(Page 2)

```
ETHERNET STATUS                                    [ 2 / 3 ]
      OwnPort  Connect IP          Port
IP0     1000   000.000.000.000    00000
IP1     0000   000.000.000.000    00000
IP2     0000   000.000.000.000    00000
IP3     0000   000.000.000.000    00000
```

- ♦ IP0 to IP3: Ethernet channel number
- ♦ OwnPort: Robot port number
- ♦ Port: Port number of the destination
- ♦ Connect IP: IP address of the destination

(Page 3)

```
ETHERNET STATUS                                    [ 3 / 3 ]
      App.      Mode          Status
IP0   HOST      TCP/S         LISEN
IP1   SCOL      TCP/C         CLOSED
IP2   SCOL      TCP/S         CLOSED
IP3   SYSTEM    TCP/S         CLOSED
```

- ♦ IP0 to IP3: Ethernet channel number
- ♦ App.: Name of application using the channel

  The connection in the control status from robot language uses simple protocol when HOST is shown and uses non-protocol communication when SCOL is shown.
- ♦ Mode: Protocol type, open status

  TCP/S indicates a server connection using TCP protocol, and TCP/C indicates a client connection using TCP protocol.
- ♦ Status: Ethernet port status

  The statuses below are displayed. For the specific meaning of each status, refer to any commercially-available reference that provides explanations of TCP/IP.

  Closed
  Listen
  Syn-Sent
  Syn-Received

Established
Fin-Wait1
Fin-Wait2
Close wait
Closing
Last ACK
Time Wait

(3) Cautions

- Check that the Ethernet cable between the controller and external device is connected correctly.
- Use a cross connect cable if connecting the robot controller directly to an external device, and use a straight cable if the connection passes through a hub.
- After making any parameter settings, turn the power off and then on again to enable the new settings.

3.4 Confirmation with a Windows Computer

This describes how to confirm the connection using the DOS command "ping".
From the Start menu in Windows, go to the Command Prompt, and specify the IP address of the robot controller as shown below.

C: ¥> ping Robot controller IP address

If there is a transmission problem, the result "unreachable" or "Request Time out" is displayed as shown in Figure 3.1.
If transmission is performed normally, the result "Reply from …" is displayed as shown in Figure 3.2.

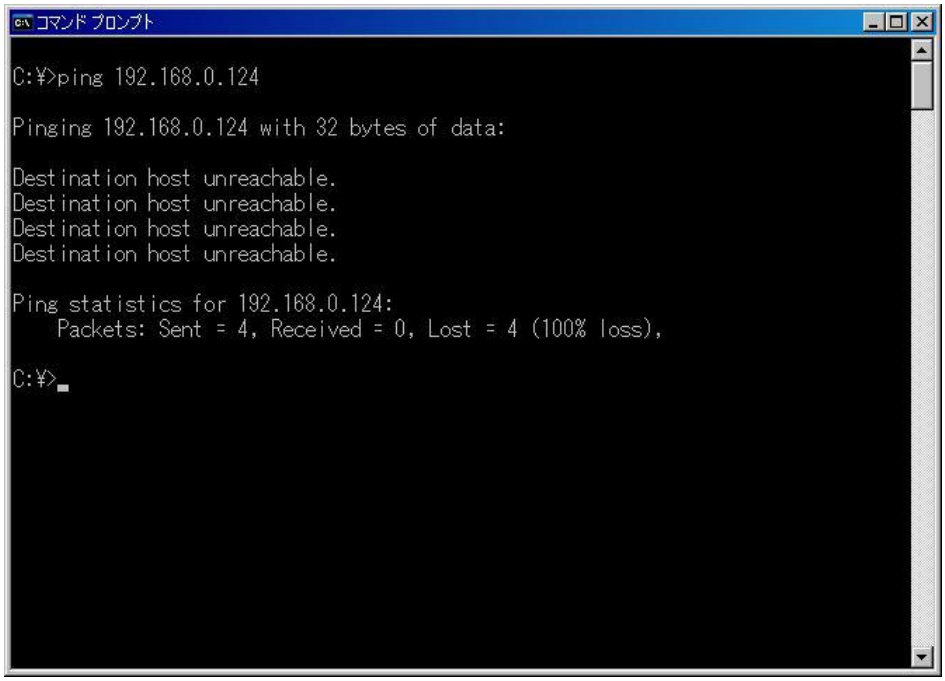


Fig. 3.1　Ping execution screen (Problem result)

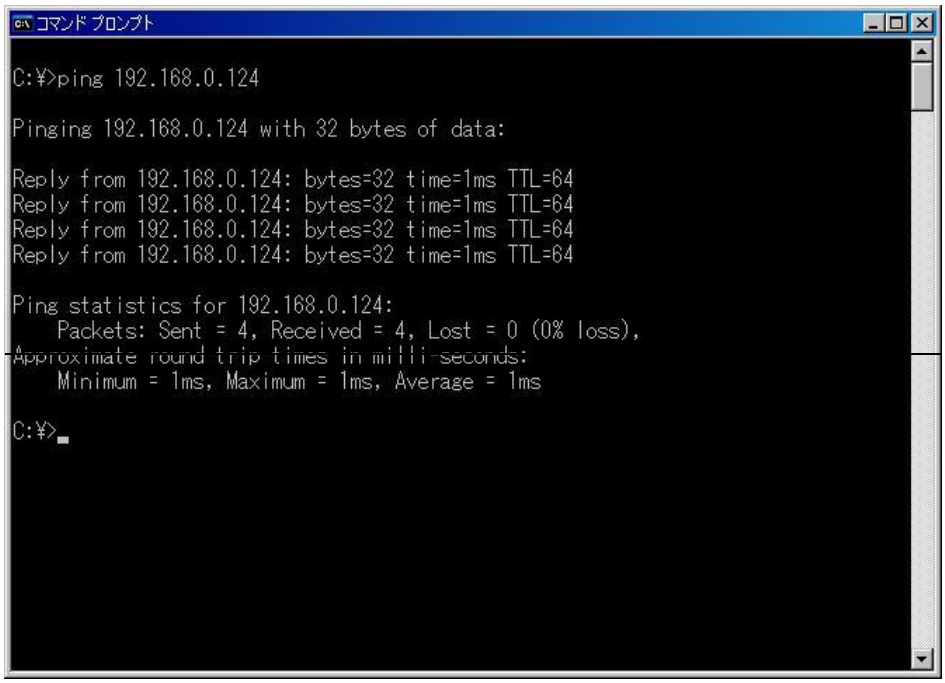


Fig. 3.2　Ping execution screen (Normal result)

Section 4

Non-Protocol Communication

Non-protocol communication is used for exchanging data with a robot language program and for printing out files. The COM1 (RS-232C) port, and Ethernet IP1 and IP2 ports are used for non-protocol communication.
The following is the communication data format for sending data to the robot controller using non-protocol communication.

The symbol [ ] in this section represents a control character in ASCII code. For ASCII code, refer to Section 7.1.

Character string + CR Character string, character string + CR	These character strings are based on ASCII format. At the end of the character strings, add CR, which indicates the end of the character strings. Separating character strings with commas enables you to send more than one kind of character strings.

The functions as described in the simple protocol cannot be used for the non-protocol data communication. Also, response to data received by the controller and timeout check are not processed at all.

4.1 Communication with a Robot Language Program

Data communication with a robot language program can be performed using the PRINT and INPUT commands. For more information on the robot language, see "Robot Language Manual".

4.1.1 PRINT Command

The PRINT command allows you to send out a specified character string from the controller. The controller transfers character strings or values of variable specified by the PRINT command. The format of the PRINT command is shown below:

PRINT ␣ [<channel>,] {<character string>|<expression>|}[,{<character string>|<expression>}] .....[,CR]

<channel>:

Specifies the communication port over which the data is to be transmitted. One of the following should be specified as the channel.

COM1	: COM1 port
IP1	: Ethernet IP1 port
IP2	: Ethernet IP2 port
TP	: Screen output to teach pendant

Unless <channel> is specified, data is transmitted to the teach pendant.

<character string>:

The character string to be transmitted is specified by enclosing that string in double quotation marks (").

<expression>:

Expressions made up of constants, variables, arithmetic operands and functions may be specified.

CR :

CR is used when the record end code (0x0D) is added to the last of sending data.

```
Ex.:  PRINT COM1, “INPUT DATA =”, -1000.0/3, CR
      PRINT IP1, “INPUT DATA =”, -1000.0/3, CR
```

Commas are used to separate any multiple character strings or expressions specified in the PRINT command. Character strings are enclosed inside double quotation marks, and everything inside of those double quotation marks is transmitted in ASCII code. Expressions are first solved, and the result is sent as a 12-character block of

fixed length with the result pushed over to the right of that block. Should the result of the expression be an integer, that result is sent as a Base 10 number having a maximum of ten places (digits).

Should the result of the expression be a real number, the result is sent as a number having an integral part with a maximum of four (4) digits and a decimal part with a maximum of three (3) digits (for a maximum of eight (8) places counting the decimal point). One space in front of the number is allocated for the sign (+ or -) of that number, although the sign is omitted if it is plus (+). The number is sent in a 12-character block, with the number pushed over to the right. The remaining spaces are filled with space codes (0x20). The number itself is sent in ASCII code. The commas used to separate character strings or expressions in a PRINT command are themselves not transmitted. When multiple character strings and/or expressions are specified with the PRINT command, the controller will send out these character strings and/or expressions as a single text.

Ex.:	When the above example program is executed, the following data are transmitted.
	INPUT DATA = -333.333 CR

* To left-justify the output display by the PRINT command without space, set 1 in the value of (FUNCTION 14), the 2nd column in 3rd row of user parameter [U25] FUNCTION SELECT SWITCH. Data can be left-justified without displaying space. Refer below for an example.

[U25] FUNCTION SELECT SWITCH					
= 0	0	0	0	0	0
= 0	0	0	0	0	0
= 0	1	0	0	0	0
= 0	0	0	0	0	0

* To use the PRINT command to output multiple character strings and/or expressions delimited by a comma ‘,’, add a comma enclosed by double quotation marks ‘”’ separately from delimiter ‘,’. They can be output delimited by a comma.

Ex.: PRINT COM1, A,“,”, B, CR

4.1.2 INPUT Command

As opposed to the PRINT command, which is used for transmitting data, the INPUT command is used for receiving data. The only data which may be received by the controller are integers and real numbers. Data received by the controller is put into a variable (in a robot language program) specified by the INPUT command. This data can be referred to later in the program to operate the robot.

INPUT ␣ [<channel>,] <variable>, [<variable>] ...

<channel>:
Specifies the communication channel over which the data is to be received. One of the following should be specified as the channel.

COM1	:	COM1 port
IP1	:	Ethernet IP1 port
IP2	:	Ethernet IP2 port
TP	:	Key input from teach pendant

Unless <channel> is specified, the controller receives data from the teach pendant.

<variable>:
Specifies the variable in the robot program into which the data is to be entered.

```
Ex.: INPUT COM1, N1, N2
     INPUT IP1, N1, N2
```

Commas are used to separate two (2) or more variables specified in the INPUT command. The controller waits until data comes in over the communication channel specified by the INPUT command. Add a record end code (0x0D) to the end of data which is sent to the controller. When multiple units of data are to be sent to the controller, the individual data units should be separated with commas before being transmitted. When more units of data are received by the controller than was specified with the INPUT command, the surplus data are ignored and used for the next INPUT command. If fewer units of data are received by the controller than was specified with the INPUT command, the controller waits until the short data reach.

Note 1)

The controller starts reading any data sent to it only after an INPUT command is executed. Data are set in the ring buffer, and the input data are picked up according to the request of the INPUT statement.
Any data received before the INPUT command is executed may be ignored. Take careful precautions when determining the data transmission timing.

Note 2)

Should program execution be suspended while the controller is waiting for data to come in (as directed by an INPUT command), the execution of the INPUT command will be cancelled. Should the program be resumed, program execution will start from the step following the INPUT command. Any variables for which data was not yet received when the program was suspended will be treated as 0.
You should keep this in mind when writing your robot language program, i.e., you should arrange your program in such a way that it will still function properly even should the controller (mistakenly) treat input values as 0. One way to do this is to have the controller ask the host computer for confirmation (retransmission) whenever the controller receives a 0. Another way to do this is to add on check-sum data (to the data to be transmitted) and check the validity of the received data. (For programming examples, see Section. 4.2, "Programming examples for communication with a robot language program.")

Note 3)

In the step operation mode, which is one of the test operation modes, when the INPUT command is executed, the program enters the wait state until data are received. The same holds true when an INPUT command is executed directly.

Note 4)

When an INPUT command is to be used to receive multiple variables transmitted as one text from the host computer, individual numerals should be separated in the text with commas.

Note 5)

The INPUT command cannot be executed in the step operation mode.

4.1.3 Clearing Communication Buffer

If the number of data the controller has received is larger than the number of data specified by the INPUT command, such data are stored in the communication buffer and used at request of the next INPUT command. If data is left in the communication buffer, unexpected data will reach by the INPUT command. To clear the data left in the communication buffer, output character string "BUFFRESET" to relevant communication port.

Example)
When clearing the communication buffer of the COM1 port:

```
PRINT␣COM1, "BUFFRESET"
```

Example)
When clearing the communication buffer of the IP1 port:

```
PRINT␣IP1,“BUFFRESET”
```

4.2 Programming Examples for Communication with a Robot Language Program

It is possible to utilize communication functions in a robot language program in order to do such things as specify program branches, correct the position of the robot to be performed, etc. Listed below are several programming examples showing how to do this.

4.2.1 Program Branching

The INPUT command can be utilized to specify the number of times a certain action is to be repeated, to select a task for execution, etc.

Specifying the number of times an action is to be repeated (COM1)

```
PROGRAM REPEAT
    N=0
START:
    PRINT COM1,“REQ”
    INPUT COM1,N
    IF N==0 THEN GOTO
    FOR K=1 TO N
        MOVE A1
        MOVE A2
        • • •
    NEXT K
    PRINT COM1,“END”
END
```

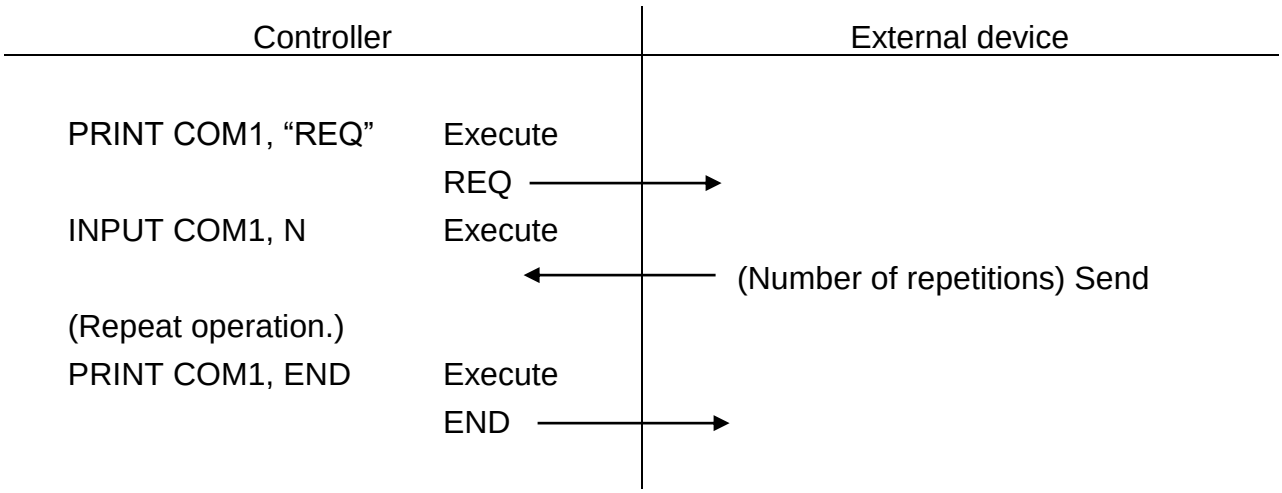
In this program, the robot will repeat a certain series of motions as many times as the external device tells it to.

First, the controller will send the character string REQ to the external device. The external device will reply by sending back the number of times the operation is to be repeated.

The controller will read in this number as the variable N, and will use this variable in the FOR statement.

Should the program execution have been suspended while the controller was waiting receiving data in the INPUT command, the variable N would be (mistakenly) taken as 0 when the program is resumed. In such a case, branch is executed to the top address of the program by asking the external device for a retransmission. When the task is completed, the controller will send the character string END to the external device.

Data exchange

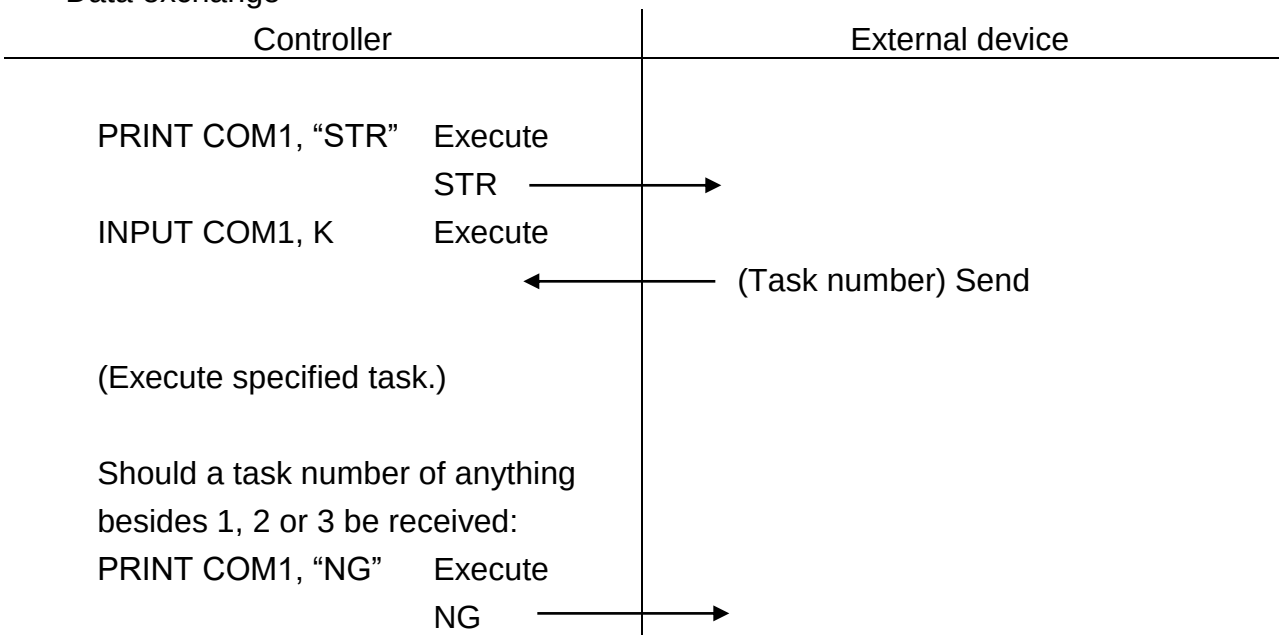


Selecting a task to be performed (COM1)

```
PROGRAM SELECT
    K=0
START:
    PRINT COM1,“STR”
    INPUT COM1,K
    GOTO(K) L1,L2,L3
    PRINT COM1,“NG”
    GOTO FIN
L1:
    (Task 1)
    GOTO FIN
L2:
    (Task 2)
    GOTO FIN
L3:
    (Task 3)
    GOTO FIN
FIN:
END
```

In this program, the external device tells the controller what task is to be performed. First, the controller will send the character string STR to the external device. The external device will send back a number which specifying task the controller (robot) is to perform. The controller reads in this number as variable K, which is then used in the GOTO ( ) command to branch the program to the appropriate task. Task 1 will be performed if the value of K is 1, Task 2 if the value is 2, and Task 3 if the value is 3. If the value of K is anything else, the controller will send the character string NG (non-acknowledge) back to the external device. Should program execution have been suspended while the controller was waiting for the number to come in, variable K will be taken as 0 when the program is resumed. In this case also, the controller will send the character string NG back to the host computer.

Data exchange



4.2.2 Correcting the Position of the Robot

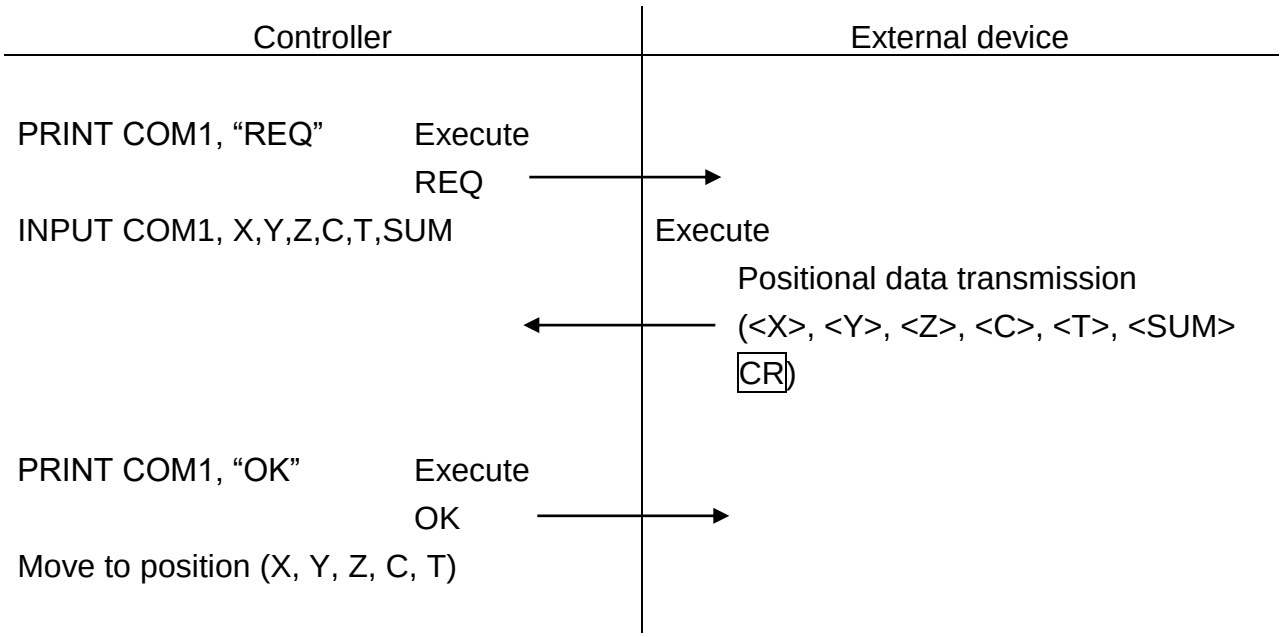
Data received from the host computer can be used to correct the position of the robot.

Directly specifying position (COM1)

```
PROGRAM DIRECT
  X=0.0
  Y=0.0
  Z=0.0
  C=0.0
  T=0.0
  PRINT COM1,"REQ"
  INPUT COM1,X,Y,Z,C,T,SUM
  IF SUM==X+Y+Z+C+T+1
    THEN GOTO ACTION
  PRINT COM1,"NG"
  GOTO FIN
ACTION:
  PRINT COM1,"OK"
  P1=POINT(X,Y,Z,C,T)
  MOVE P1
FIN:
END
```

In this program, the controller moves the robot to a position specified by the external device.
The controller sends the character string REQ to the external device. The external device will respond by sending back the values for positional data X, Y, Z, C, and T, and also the sum of these values plus 1 (as check-sum data). The controller will then see if the received data is correct by adding all the positional data, adding 1 to the result, and determining if check-sum value agrees with the value. If it does agree, the controller will send the character string OK.
If the received data is wrong, the controller will send the character string NG back to the external device and quit the program.
The controller will create positional data P1 using that data and then tell the robot to move to that point.

Data exchange

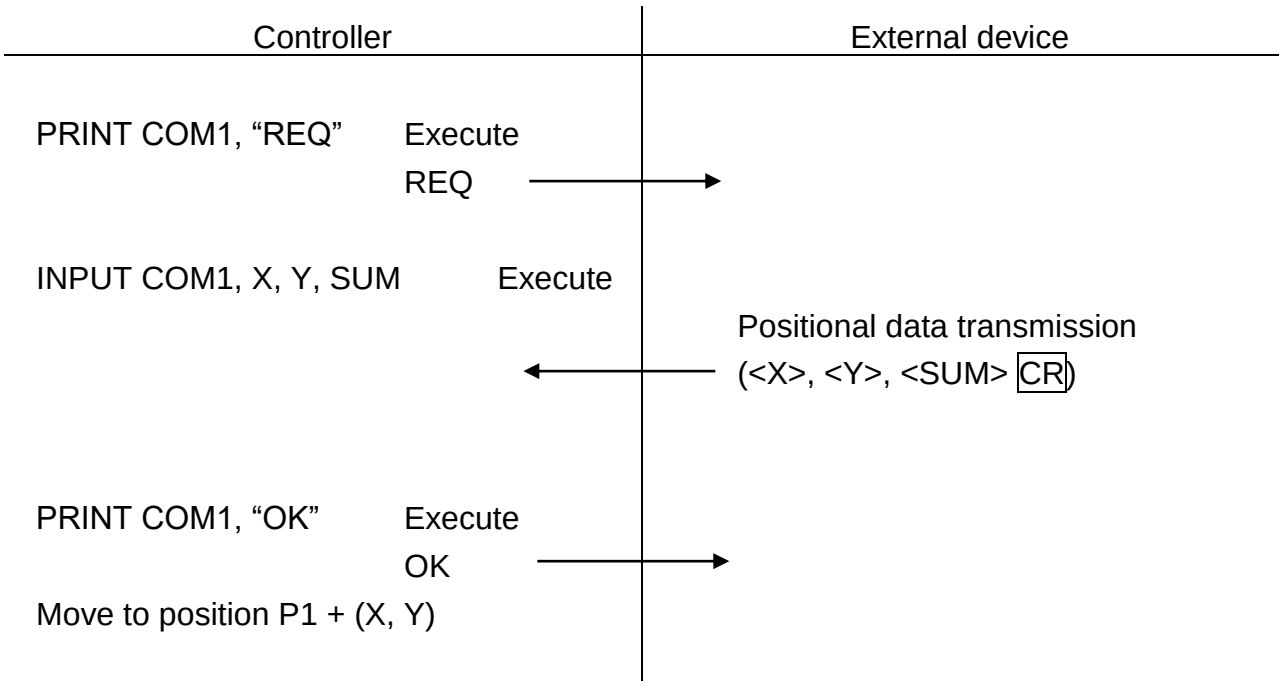


Specifying a relative position (COM1)

```
PROGRAM RELATIV
   X=0.0
   Y=0.0
   SUM=0.0
   MOVE P1
   PRINT COM1,“REQ”
   INPUT COM1,X,Y,SUM
   IF SUM==X+Y+1   THEN GOTO
ACTION
   PRINT COM1,“NG”
   GOTO FIN
ACTION:
   PRINT COM1,“OK”
   MOVE P1+POINT(X,Y)
FIN:
END
```

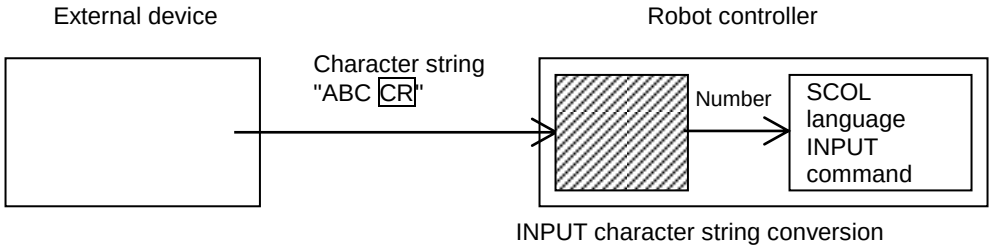
In this program, the controller moves the robot to a position specified by the external device.
The controller sends the character string REQ to the external device.
The external device will respond by sending back the values for coordinate positions X and Y, and also the sum of these values plus 1 (as check-sum data). The controller will then see if the received data is correct by adding X and Y, adding 1 to the result, value agrees with the check-sum value. If it does agree, the controller will send back the character string OK to the external device.
If the received data is wrong, the controller will send the character string NG back to the external device and quit the program.
The controller will add the X and Y values of the transmission to the X and Y values of positional data P1 and make the robot move to the new position thus created.

Data exchange



4.3 INPUT Character String Conversion Function

The INPUT command in SCOL language cannot receive a character string. If a nonnumeric character string is received, 2-046 Invalid Channel error occurs. A character string may be sent depending on the specification of the external device. This function converts the character string received by the robot into a numeric value.



The value of (FUNCTION 1), the 1st column in 1st row of user parameter [U25] FUNCTION SELECT SWITCH is set to the following to switch the processing if a nonnumeric character string is received.

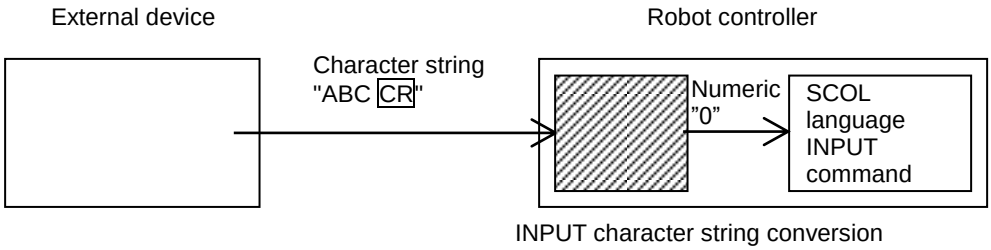
0: Function disabled (if a nonnumeric character string is received, Invalid Channel error occurs.)
1: A nonnumeric character string is converted to 0 if received.
2: A nonnumeric character string is converted to any numeric value if received.

4.3.1 Converting a Character String to 0

The value of (FUNCTION 1), the 1st column in 1st row of user parameter [U25] FUNCTION SELECT SWITCH is set to 1.

[U25] FUNCTION SELECT SWITCH					
= 1	0	0	0	0	0
= 0	0	0	0	0	0
= 0	0	0	0	0	0
= 0	0	0	0	0	0

When the value is set to 1, a nonnumeric character string is converted to 0 if received.

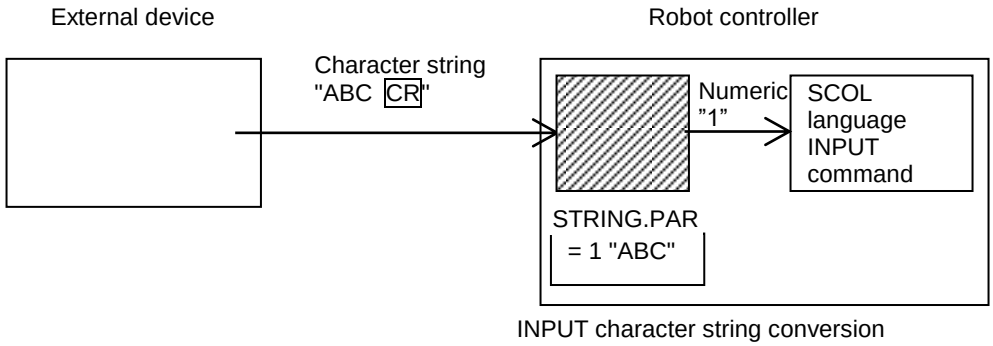


4.3.2 Converting a Character String to Any Numeric Value

The value of (FUNCTION 1), the 1st column in 1st row of user parameter [U25] FUNCTION SELECT SWITCH is set to 2.

```
 [U25] FUNCTION SELECT SWITCH
= 2   0   0   0   0   0
= 0   0   0   0   0   0
= 0   0   0   0   0   0
= 0   0   0   0   0   0
```

When the value is set to 2, a nonnumeric character string is compared with the character string in the previously created parameter file “STRING.PAR” and the conversion number of the matching character string is returned if received.



INPUT character string conversion

STRING.PAR should be created in the following format.

```
     ①        ②
=   1   "▲▲▲▲▲"
```

① Conversion number (0 to 32767)

② Character string to be received (up to 32 characters)

Example of STRING.PAR:

```
= 0   "OK000"
= 1   "OK001"
= 2   "OK002"
      ⋮
= 127   "OK127"
```

The number of data rows can be specified to be up to 128. If 128 rows are exceeded, data is truncated by 128 rows.

If the first data is other than a number (integer), the second data has 33 characters or more, and the number of data elements is other than 2, 8-015 Parameter error occurs.

* If a matching character string is not found, Invalid Channel error occurs without numeric conversion.
* If the received character string can be determined to be numeric, conversion using this function is not performed.
* STRING.PAR becomes valid when the power is turned OFF and then turned ON.

Section 5

Simple Protocol Communication

The simple protocol communication issues a command to the robot controller and performs start/stop operation of the robot, transfer of program files, and monitoring of the status.
This allows program creation and editing, debugging start support, and maintenance using the computer programmer TSPC.
Data can be sent and received by the robot program for exchanging numerical and positional data.
INPUT and PRINT statements from SCOL language can be used to perform data input and output.
Example: INPUT IP1, INDATA, PRINT IP1, OUTDATA, CR, etc.

The symbol ☐ in this section represents a control character in ASCII code. For ASCII code, refer to Section 7.1.

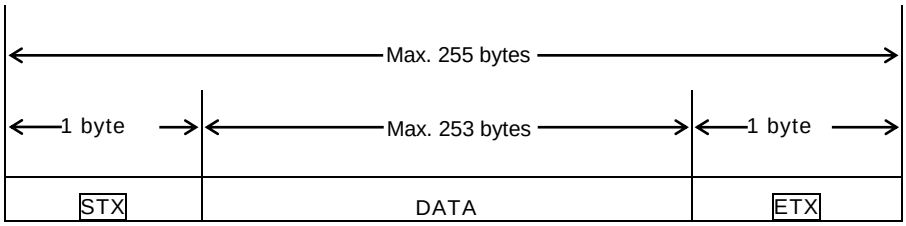
5.1 Transmission Format

Transmissions are made with the text unit shown below. A maximum of 253 bytes of actual data can be transmitted as a single text. Actual data in amounts over 253 bytes will be transmitted in one of the following two ways.

(1) Files (robot programs, positional data, parameters) will be broken down into multiple texts and transmitted as described in Para. 5.3.3. The receiving station will send an answer signal back for each block of text transmitted. File upload (or download) commands are not necessary to transmit the second and following blocks of text.

(2) Messages to be transmitted
Messages will be sent with multiple transmissions. In other words, the data will be broken down into multiple texts and each text will be transmitted independently. The station that received the data will reconstruct the multiple texts back into a single message.

5.1.1 Text



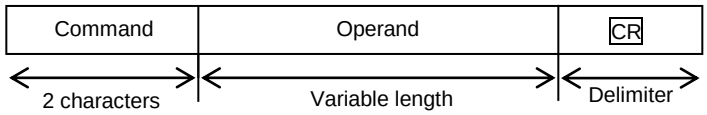
Text length: Max. 255 byte (from STX through ETX)

Text content:

- STX Start of text code (0x02) –1 byte
- DATA Data section (Max. 253 bytes)
- ETX End of text code (0x03) –1 byte

5.1.2 Data Section Format

The basic format of data is shown below.



(1) Command

Command consists of two alphabetical letters which signify the type of command. See Table 5.1 for command types and descriptions.

(2) Operand

The form of the operand varies depending on the type of command. For more information, read the description for the command in question.

(3) Delimiter

Add CR as delimiter at the end of the data. In the description of commands in Para 5.3.3, CR is used.

(4) Characters

The characters to be used should be ASCII code alphanumeric characters and symbols.

Alphanumeric characters:

a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0 1 2 3 4 5 6 7 8 9

Special symbols:

“ ‘ ( ) + - * / , . < > = ! [ ] { } % ^ & ? ;
: # $ _ ~ | ␣ (Space)

5.2 Transmission Protocol

The RS-232C HOST port and Ethernet IP0 port wait for requests from the host computer after the power is supplied to the robot controller.
Basically, the host computer is a master station and the robot controller is a slave station. The robot controller sends back necessary data to the host computer in reply to commands reached from the host computer.

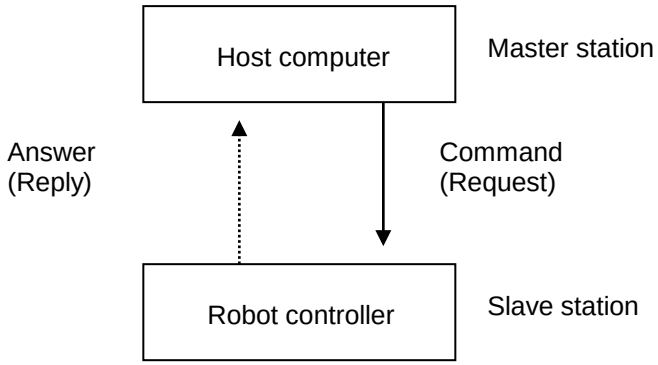


Fig. 5.1 Basic communication

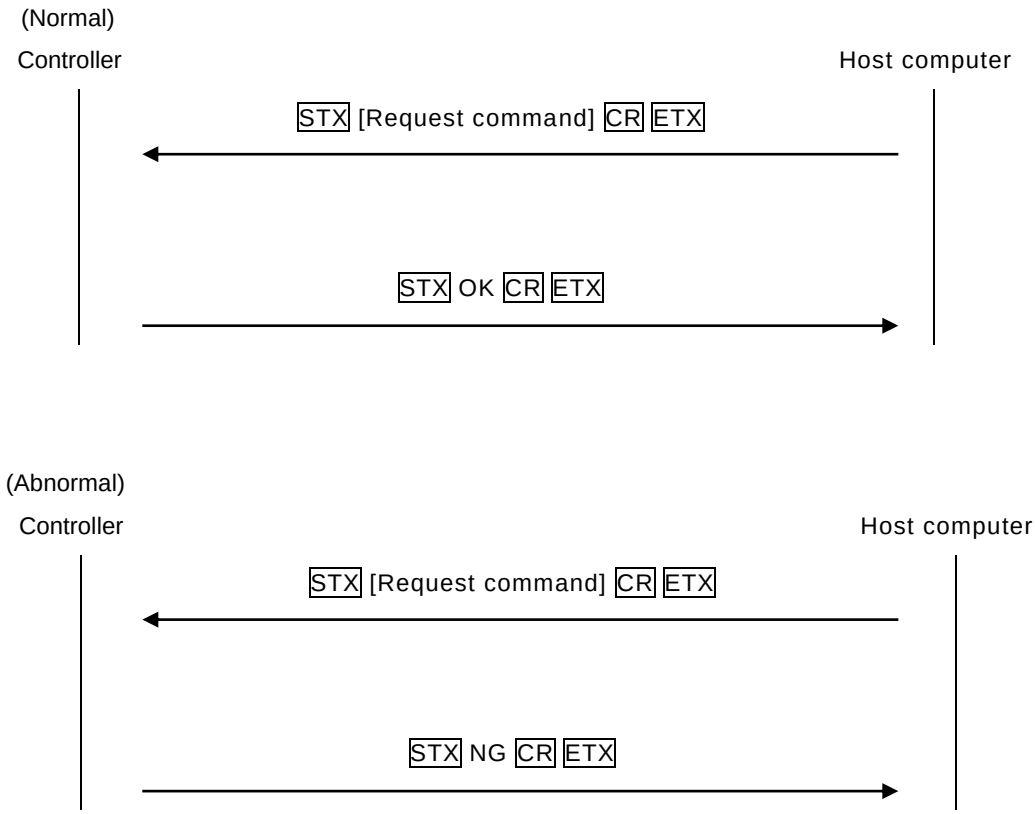
Should the robot controller return an NG (Non-acknowledge) code in response to a command from the host computer, or should the robot controller give no response at all, resend the command from the host computer. Even should an error have occurred while transmitting a file, resend the file upload (or download) command from the host computer.

5.2.1 Details of Communication Protocols

There are three protocols for the simple protocol communication: command, file read, and file write communication protocols.

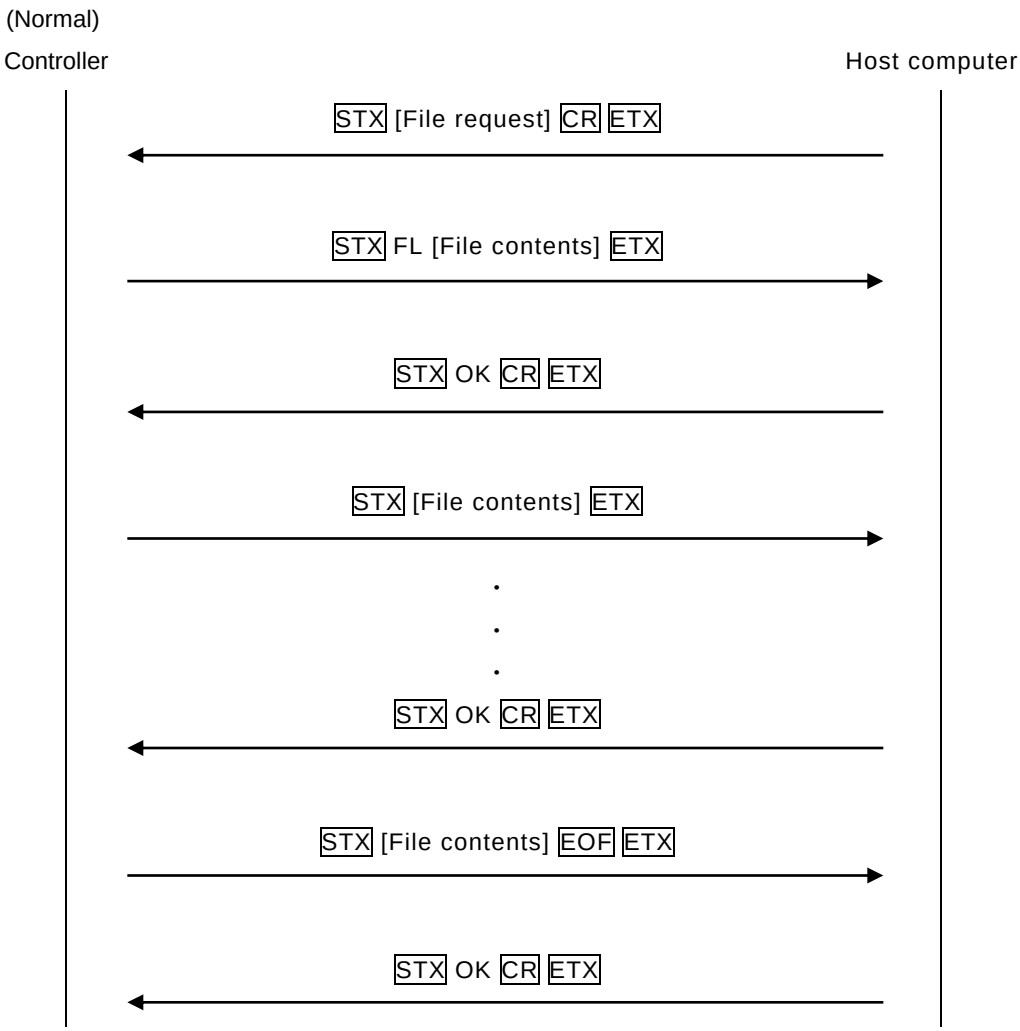
5.2.1.1 Command Communication

The controller responds to a request command sent from the host computer.

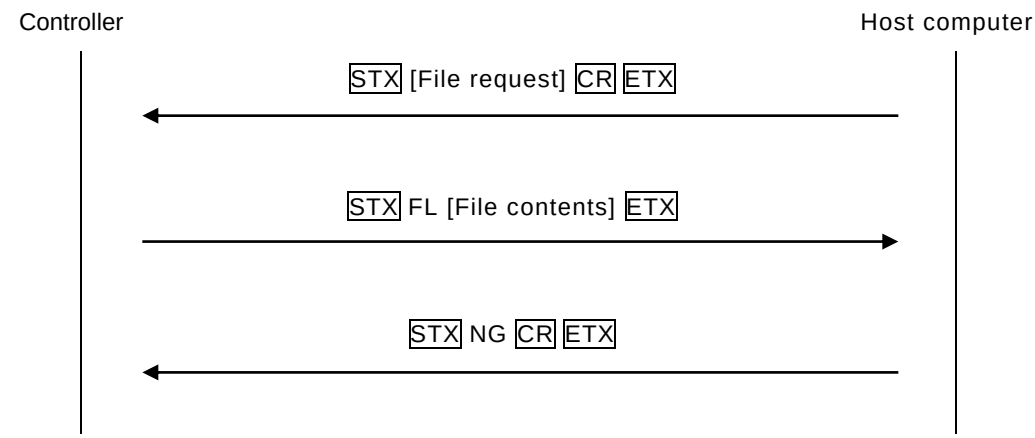


5.2.1.2 File Read Communication

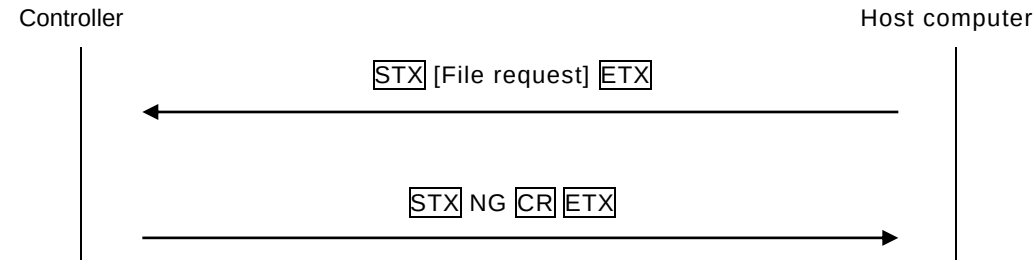
The controller returns the file contents added to the character string “FL” in response to a request command sent from the host computer. If the file contents include EOF at the end of the file, it is suggested that the read processing of the file is complete.
If the file contents do not include EOF, which suggests that the read processing of the file is not complete, the host computer is supposed to send the OK command to perform the read processing of the rest of the file contents. If the controller does not receive the OK command within 10 seconds, it sends the NG command to the host computer.



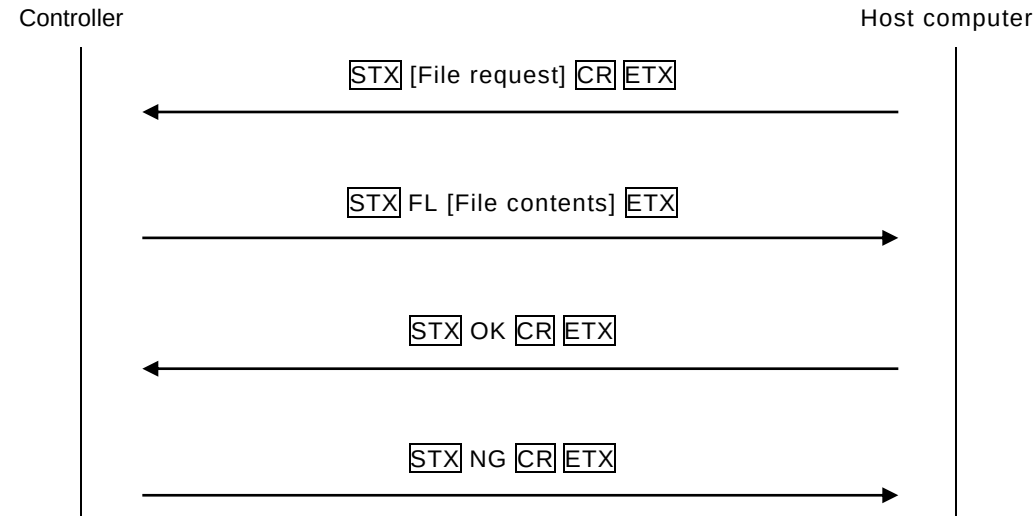
(Interrupted)



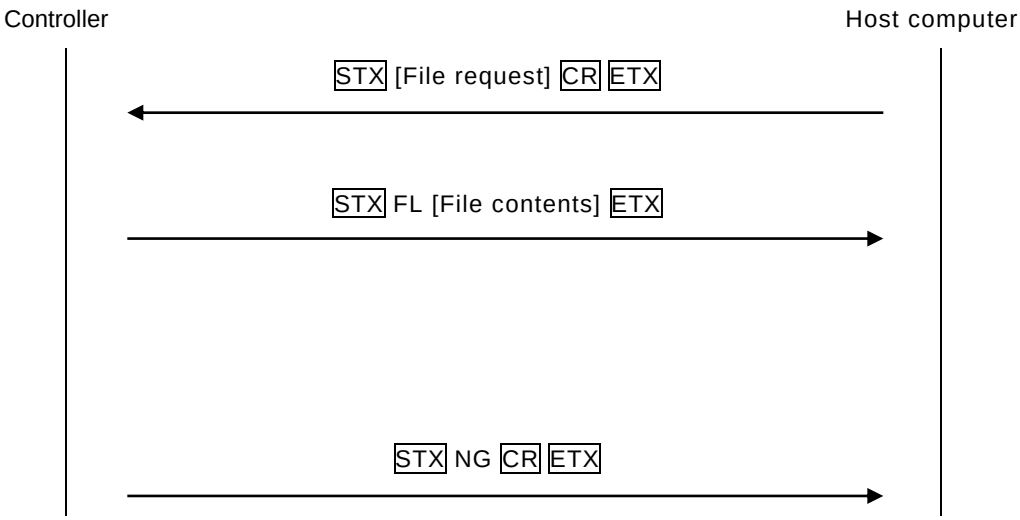
(Abnormal 1)



(Abnormal 2)

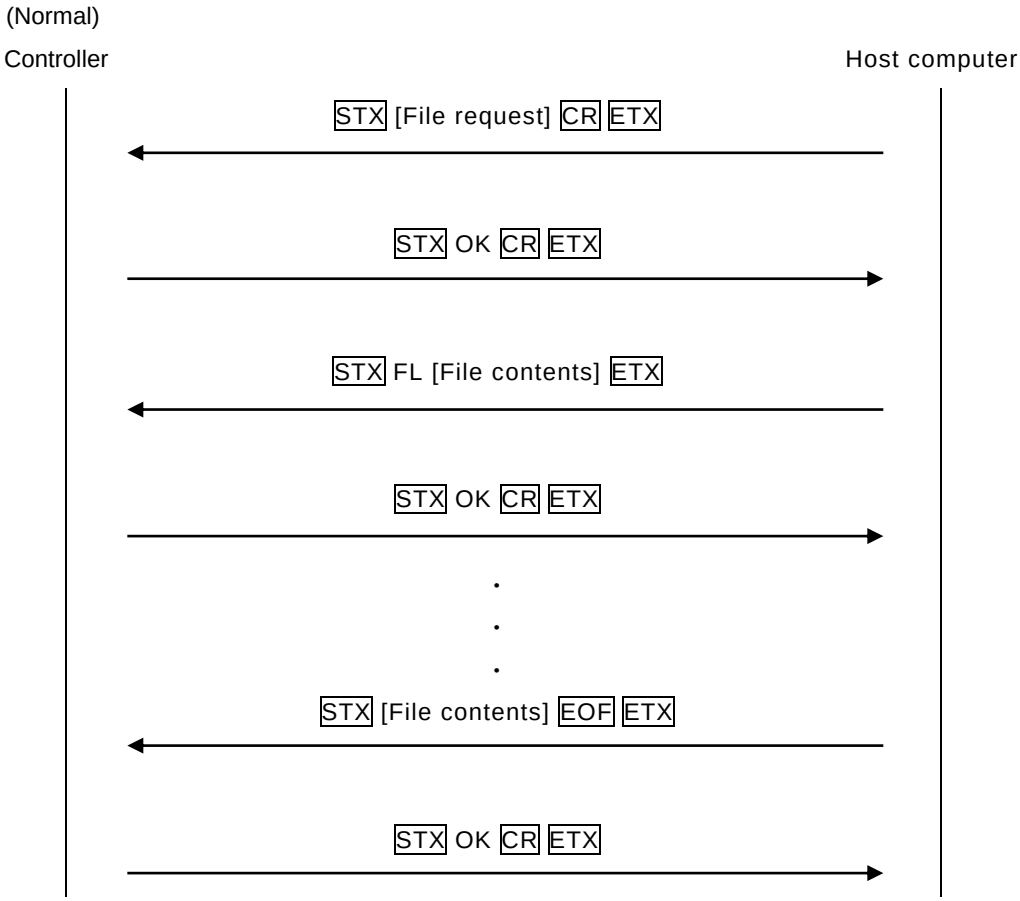


(Abnormal 3) On the occurrence of timeout

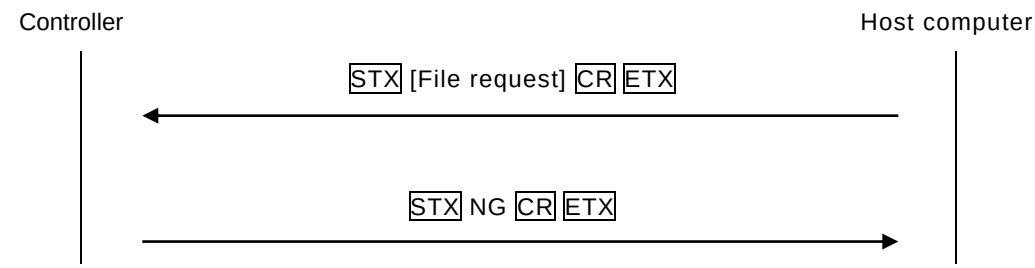


5.2.1.3 File Write Communication

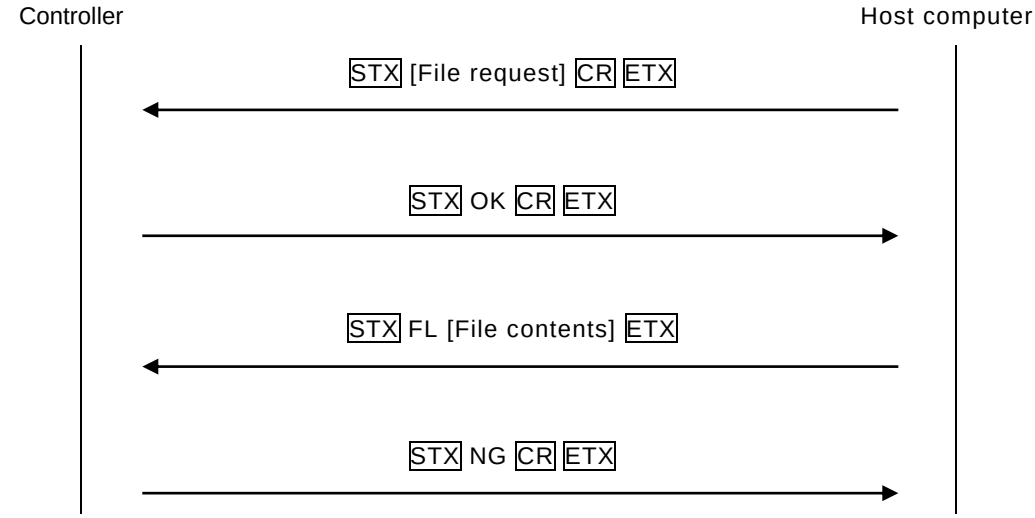
The controller responds to a request command sent by the host computer. After receiving a normal response from the controller, the host computer sends the file contents to be written added to the character string “FL,” and then the controller responds to it. To complete the write processing of the file, the host computer sends the file contents with EOF at the end of the contents. When the host computer receives a response to it, the write processing of the file completes.
If the file contents do not include EOF, which suggests that file transfer is not complete, the host computer is supposed to perform the write processing of the rest of the file contents. If the controller does not receive the contents within 10 seconds, it sends the NG command to the host computer.



(Abnormal 1)

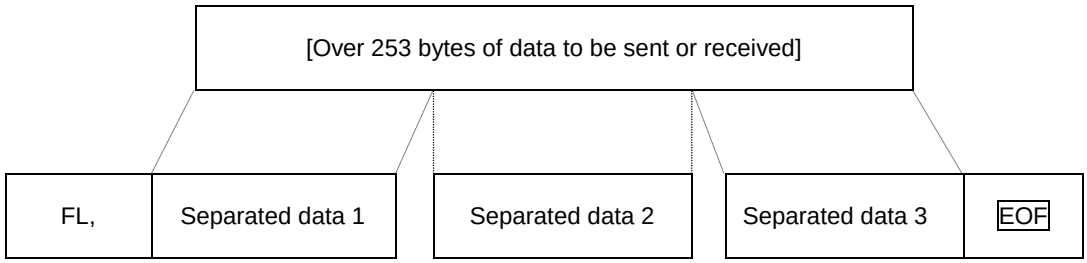


(Abnormal 2)



5.2.2 File

If the data exceeds one text field or 253 bytes, the data is transmitted in two or more texts, as shown below.



* EOF (End of File) : File end code (0x1A) — 1 byte
The STX and ETX are attached to each text.

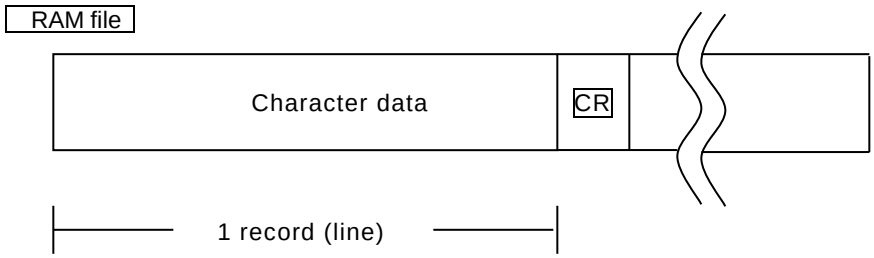
5.2.2.1 File types

The description of each command is as follows.

Description	Corresponding command
RAM file	UL, DL
File directory	CA
Status	SU
Error history	EU
System total status	SF
Version information	VR
Memory read data	MR
Currently occurring alarm	AC
Current position information	PS
Individual current position information	PR
Maintenance information	MN
IO information	IO
Alarm history	AH
Robot tip speed information	VL
Motion status	SM
Hexadecimal-coding and reading of input signal	HI

5.2.2.2 File contents

A file is composed of one or more records. The following are the records that compose a file.



Record contents

NO.	Name	Size (byte)	Description
1	Character data	Max 252 Variable length	ASCII code alphanumeric characters and symbols
2	CR	1 Fixed length	Record end code (0x0D)

This record corresponds to one line of a program or one line of positional data.

5.3 Commands

5.3.1 List of Commands

Commands which may be used with this communication protocol are shown in Table 5.1.

Table 5.1 List of commands

NO.	Command	Function
1	AC	Transfers up to 10 currently occurring alarms with messages to the host computer.
2	AH	Transfers the alarm history with messages to the host computer.
3	BR	Directs the controller to turn off the servo power.
4	CA	Transfers the directory information in the controller RAM drive to the host computer.
5	DL	Transfers a file from the host computer to the controller RAM drive.
6	DO	Directly executes a command from the host computer.
7	EC	Executes an internal command of the controller.
8	EM	Directs the controller to change the EXTERNAL mode.
9	ER	Deletes a specified file in the controller RAM drive.
10	EU	Transfers the error history data to the host computer.
11	FD	Informs the controller that the FEED HOLD pushbutton switch has been pressed.
12	HI	Transfers the general input signal information of the controller to the host computer.
13	HO	Directs the controller to overwrite general output signals.
14	IO	Transfers the I/O information of a specified model to the host computer.
15	IW	Directs the controller to overwrite general output signals.
16	JG	Directs the controller to start jog.
17	MD	Sets the guide mode of the controller.
18	MN	Transfers specified maintenance information to the host computer.
19	MP	Directs the controller to start teaching point movement.
20	MR	Transfers the read data of a defined global variable to the host computer.
21	MW	Writes data to a global variable defined in the controller.
22	PR	Transfers the current position information for a specified coordinate system to the host computer.
23	PS	Transfers the current position information (motion status, execution line number, current position) to the host computer.
24	RM	Directs the controller to reset specified maintenance information.
25	RN	Directs the controller to start up a program.
26	RS	Directs the controller to reset a reset target, such as a program.
27	RT	Sets the guide rate of the controller.
28	SC	Sets the guidance coordinate of the controller.
29	SF	Transfers the internal status of the controller to the host computer.
30	SL	Selects a program to run automatically.
31	SM	Transfers the motion status of the controller to the host computer.

NO.	Command	Function
32	SO	Directs the controller to turn on the servo power.
33	SP	Directs the controller to stop operation.
34	SU	Transfers the internal status of the controller to the host computer.
35	UL	Transfers a specified file in the controller RAM drive to the host computer.
36	VL	Transfers specified robot tip speed information to the host computer.
37	VR	Transfers the system version information to the host computer.
38	WD	Sets the watchdog timer of the controller.
39	ZS	Sets the tool and work coordinates of the controller.

* The following files can be sent with the FL command.

(1) RAM files
- User file (program and positional data)
- Parameter file

(2) File directories
(3) Status files
(4) Error history files
(5) System total status files
(6) Version information files
(7) Variable read data files
(8) Currently occurring alarm

5.3.2 Commands and Operation Modes

Tables 5.2 and 5.3 show the operation modes in which each command is operative.

5.3.2.1 The Commands and Operation Modes of the Host Port

The communications through the HOST port are effective irrespective of master modes, but their function is limited by a master mode. For details, see the following table.

Table 5.2 HOST port (RS232C) commands and operation modes

Master mode		TEACHING INTERNAL(KSL3000) EXT.SIG EXT.ETHER				EXT.RS-232C			
Status		Stop	Servo ON	In automatic operation	In execution of Move command	Stop	Servo ON	In automatic operation	In execution of Move command
Command	Description								
AC	Currently occurring alarm acquisition	○				○			
AH	Alarm history acquisition	○				○			
BR	Servo OFF	○				○			
CA	Directory request	○				○			
DL	File download	○		×		○		×	
DO	Execution of DO statement	×				×	○	×	
EC	Internal command	×				○			
EM	Controller mode change execution	×				○	×		
ER	File erase	○		×		○		×	
EU	Error history upload	○				○			
FD	Feed hold	○				○			
HI	Input signal read	×				○			
HO	Output signal write	×				○			
IO	IO information acquisition	○				○			
IW	I/O write	×				○			
JG	JOG execution	×				×	○	×	
MD	Guide mode setting	×				○		×	
MN	Maintenance information acquisition	○				○			
MP	Teaching point movement execution	×				×	○	×	
MR	Variable read	○				○			
MW	Variable write	○				○			
PR	Individual current information acquisition	○				○			
PS	Current position acquisition	○				○			
RM	Maintenance information reset execution	○				○			
RN	Automatic operation start	×				×	○	×	
RS	Program reset	×				○		×	
RT	Guide rate setting	×				○		×	
SC	Guidance coordinate setting	×				○		×	

Master mode		TEACHING INTERNAL(KSL3000) EXT.SIG EXT.ETHER				EXT.RS-232C			
Status		Stop	Servo ON	In automatic operation	In execution of Move command	Stop	Servo ON	In automatic operation	In execution of Move command
Command	Description								
SF	System total status	○				○			
SL	Program selection	○		×		○		×	
SM	Motion status acquisition	○				○			
SO	Servo ON	×				○	×		
SP	Stop	×		○		×		○	
SU	Status request	○				○			
UL	File upload	○		×		○		×	
VL	Robot tip speed information acquisition	○				○			
VR	Version read	○				○			
WD	Watchdog execution	○				○			
ZS	Coordinate setting	*×				○		×	

○: Valid, ×: Invalid, △: Valid when the robot is stopped

Commands transmitted from the host computer to the controller can be received by the controller in the modes marked "○".
Commands transmitted from the controller to the host computer can be transmitted by the controller in the mode marked "○".

Note: All external control input signals are also limited by a master mode.
For details, see the Interface Manual.

5.3.2.2 The Commands and Operation Modes of the IP0 Port

The communications through the IP0 port are effective irrespective of master modes, but their function is limited by a master mode. For details, see the following table.

Table 5.3 IPO port (Ethernet) commands and operation modes

Master mode		TEACHING INTERNAL(KSL3000) EXT.SIG EXT.RS-232C				EXT. ETHER			
Status		Stop	Servo ON	In automatic operation	In execution of Move command	Stop	Servo ON	In automatic operation	In execution of Move command
Command	Description								
AC	Currently occurring alarm acquisition	○				○			
AH	Alarm history acquisition	○				○			
BR	Servo OFF	○				○			
CA	Directory request	○				○			
DL	File download	○		×		○		×	
DO	Execution of DO statement	×				×	○	×	
EC	Internal command	×				○			
EM	Controller mode change execution	×				○	×		
ER	File erase	○		×		○		×	
EU	Error history upload	○				○			
FD	Feed hold	○				○			
HI	Input signal read	×				○			
HO	Output signal write	×				○			
IO	IO information acquisition	○				○			
IW	I/O write	×				○			
JG	JOG execution	×				×	○	×	
MD	Guide mode setting	×				○		×	
MN	Maintenance information acquisition	○				○			
MP	Teaching point movement execution	×				×	○	×	
MR	Variable read	○				○			
MW	Variable write	○				○			
PR	Individual current information acquisition	○				○			
PS	Current position acquisition	○				○			
RM	Maintenance information reset execution	○				○			
RN	Automatic operation start	×				×	○	×	
RS	Program reset	×				○		×	
RT	Guide rate setting	×				○		×	
SC	Guidance coordinate setting	×				○		×	

Master mode		TEACHING INTERNAL(KSL3000) EXT.SIG EXT.RS-232C				EXT. ETHER			
Status		Stop	Servo ON	In automatic operation	In execution of Move command	Stop	Servo ON	In automatic operation	In execution of Move command
Command	Description								
SF	System total status	○				○			
SL	Program selection	○		×		○		×	
SM	Motion status acquisition	○				○			
SO	Servo ON	×				○	×		
SP	Stop	×		○		×		○	
SU	Status request	○				○			
UL	File upload	○		×		○		×	
VL	Robot tip speed information acquisition	○				○			
VR	Version read	○				○			
WD	Watchdog execution	○				○			
ZS	Coordinate setting	*×				○		×	

○: Valid, ×: Invalid, △: Valid when the robot is stopped

Commands transmitted from the host computer to the controller can be received by the controller in the modes marked "○".
Commands transmitted from the controller to the host computer can be transmitted by the controller in the mode marked "○".

Note: All external control input signals are also limited by a master mode.
For details, see the Interface Manual.

5.3.3 Details of Commands

AC Currently occurring alarm acquisition

Function

The AC command is used to transfer up to 10 currently occurring alarms with messages to the host computer.
In the absence of alarms, STX FL, 0 CR ETX is returned.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX AC CR ETX

Response format

STX FL, [Size], Alarm contents(1/n) ETX
STX Alarm contents(2/n) ETX
:
STX Alarm contents (n/n) EOF ETX

Alarm contents contain [Size] records, each of which is as follows.
[Alarm number], [Message], [Date] ␣ [Time] CR

* A record is defined in the following format: [Alarm number], [Message], [Date] ␣ [Time] CR.

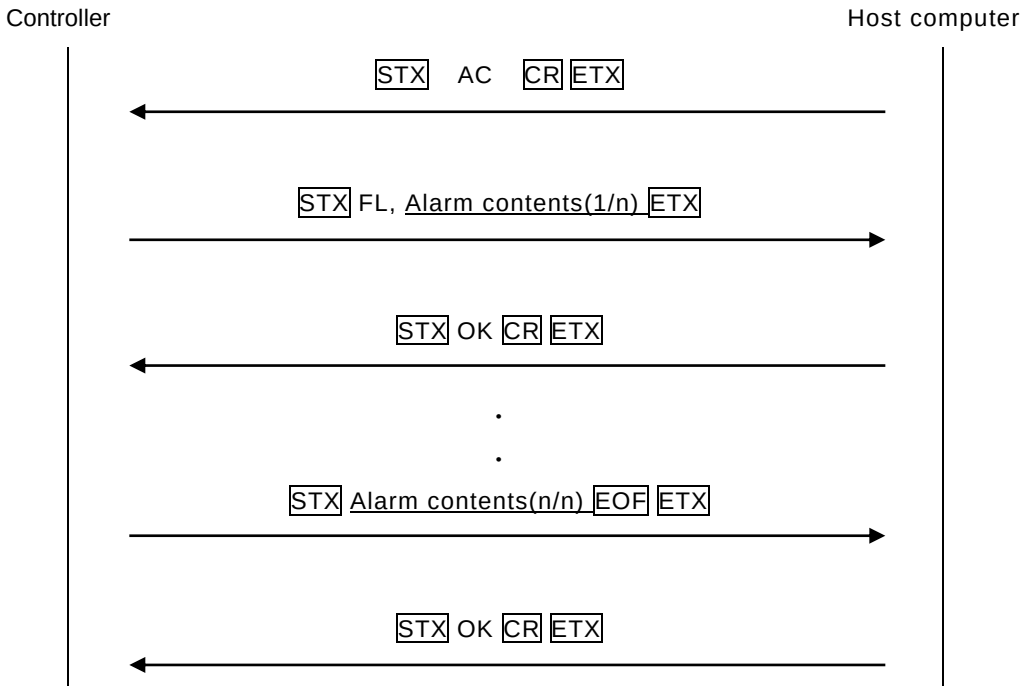
NO.	Name	Size (byte)	Description
1	Size	Variable length	Number of alarm histories (decimal) This is added to only the beginning of the file.
2	Alarm number	7 Fixed length	Shows the occurring alarm number in the following format. “XXX-YYY” XXX : Main code YYY : Sub code For details on the alarm contents, see the “Alarm Manual.”
3	Message	Variable length	Shows the message of occurring alarm.
4	Date	8 Fixed length	The date on which the alarm occurred is shown in the following format. “YY-MM-DD” YY : Year (Last two numbers only) MM : Month DD : Day
5	Time	8 Fixed length	The time on which the alarm occurred is shown in the following format. “HH: MM: SS” HH : Hours (In 24-hour"military time.") MM : Minutes SS : Seconds
6	CR	1 Fixed length	Record end code

Response example

On the occurrence of two alarms, “Emergency Stop SW ON” and “Safety SW ON,” the controller’s response to the AC command is as follows.

STX FL, 2, 008-014, Emergency Stop SW ON, 17-06-15 ␣ 10: 32: 18 CR008-017, Safety SW ON, 17-06-15 ␣ 10: 29: 26 CR EOF ETX

Communication example



Note

* On the occurrence of many alarms, all alarms may not be able to be received by one communication. Then, to receive the rest of the alarms, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the alarms.
* The host computer's receipt of EOF ensures that all the alarms are received.

AH Alarm history message acquisition

Function

The AH command is used to transfer the alarm history with message of the controller to the host computer.
The number of alarm histories to be sent can be set using the AH command with [U29] COMMUNICATION SIZE OF ALARM HISTORY: 0 for 256 alarm histories to be sent and 1 for 128 alarm histories.
In the absence of alarms, STX FL, 0 CR ETX is returned.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX AH CR ETX

Response format

STX FL, [Size], Alarm history (1/n) ETX
STX Alarm history (2/n) ETX
:
STX Alarm history (n/n) EOF ETX

Alarm history contains [Size] records, each of which is as follows.
[Alarm number], [Message], [Date] ␣ [Time] CR

* A record is defined in the following format: [Alarm number], [Message], [Date] ␣ [Time] CR.

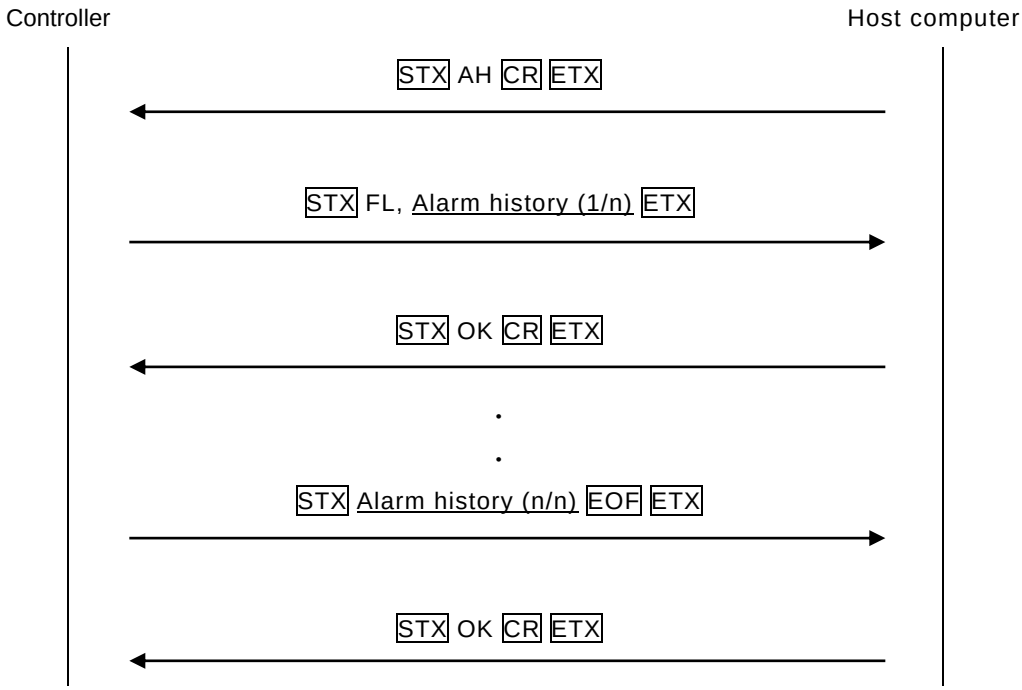
NO.	Name	Size (byte)	Description
1	Size	Variable length	Number of alarm histories (decimal) This is added to only the beginning of the file.
2	Alarm number	7 Fixed length	Shows the occurring alarm number in the following format. “XXX-YYY” XXX : Main code YYY : Sub code For details on the alarm contents, see the “Alarm Manual.”
3	Message	Variable length	Shows the message of occurring alarm.
4	Date	8 Fixed length	The date on which the alarm occurred is shown in the following format. “YY-MM-DD” YY : Year (Last two numbers only) MM : Month DD : Day
5	Time	8 Fixed length	The time on which the alarm occurred is shown in the following format. “HH: MM: SS” HH : Hours (In 24-hour"military time.") MM : Minutes SS : Seconds
6	CR	1 Fixed length	Record end code

Response example

On the occurrence of two alarms, “Emergency Stop SW ON” and “Safety SW ON,” the controller’s response to the AC command is as follows.

STX FL, 2, 008-014, Emergency Stop SW ON, 17-06-15 ␣ 10: 32: 18 CR008-017, Safety SW ON, 17-06-15 ␣ 10: 29: 26 CR EOF ETX

Communication example



Note

* When there are many alarm histories, all the alarm histories may not be able to be received by one communication. Then, to receive the rest of the alarm histories, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the alarm histories.

* The host computer's receipt of EOF ensures that all the alarm histories are received.

BR Servo OFF

Function

The BR (Servo OFF) command is given by the host computer to the controller telling the controller to turn off the servo power.
When the request cannot be accepted, STX NG CR ETX is returned.

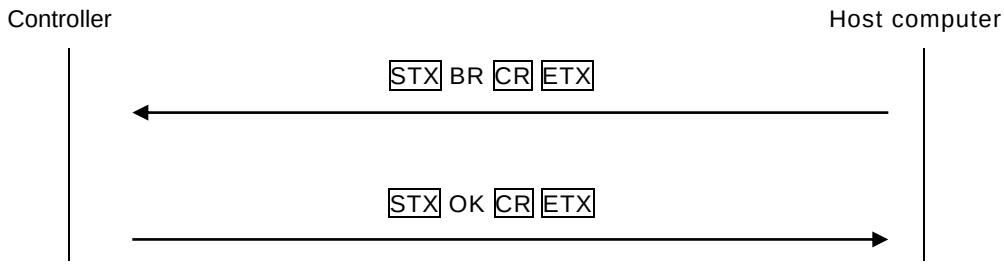
Transmission format

STX BR CR ETX

Response format

STX OK CR ETX

Communication example



CA File directory request

Function

The CA command is used to transmit the directory of files in the RAM drive from the controller to the host computer.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX CA CR ETX

Response format

STX FL, Directory information (1/n) ETX
STX Directory information (2/n) ETX
:
STX Directory information (n/n) EOF ETX

Directory information contains multiple records, each of which is as follows.
[File name] ␣ [Size] CR

* A record is defined in the following format: [File name] ␣ [Size] CR.

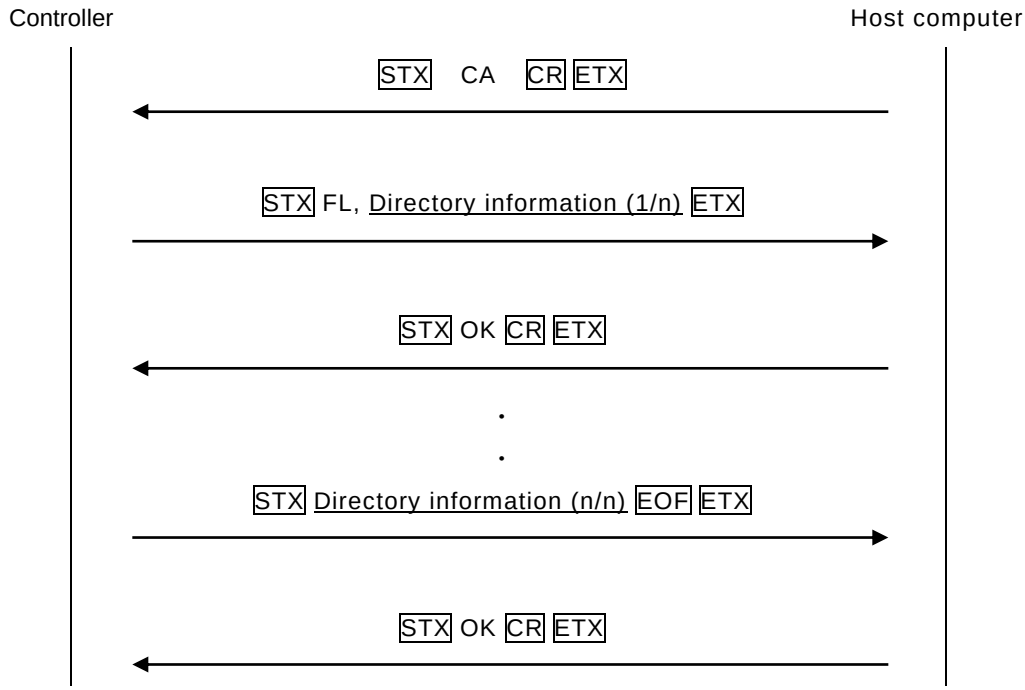
NO.	Name	Size (byte)	Description
1	File name	Variable length	An alphanumeric file name beginning with an alphabetical letter
2	Size	Variable length	File size in bytes
3	CR	1 Fixed length	Record end code

Response example

The following is the controller's response to the CA command for the 20-byte file "PRG1" and 30-byte file "PRG2" in the controller RAM drive.

STX FL, PRG1, 20, CR PRG2, 30, CR EOF ETX

Communication example



Note

* When there is much directory information, it may not be able to be all received by one communication. Then, to receive the rest of the directory information, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the directory information.
* The host computer's receipt of EOF ensures that all the directory information is received.

DL File download request

Function

The DL command is used to download (transmit) a specified file from the host computer to the controller RAM drive.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX DL, [File name] CR ETX
STX FL, Download file contents (1/n) ETX
STX Download file contents (2/n) ETX
:
STX Download file contents (n/n) EOF ETX

Download file contents contain [File contents].

NO.	Name	Size (byte)	Description
1	File name	Variable length	The name of a file to be downloaded The file name needs to be defined in either of the following formats. ・ Name.Extension ・ Name The file name must be one to eight characters. The extension must be zero to three characters. When a file name is defined without extension, do not include a period (“.”) in the file name.
2	File contents	Variable length	The contents of the file to be downloaded to the controller: ASCII code is the only valid character code, while CR is the only valid control code.

Transmission example

To transfer the file “PRG1” shown below to the controller RAM drive, use the DL command as follows.

“PRG1” File contents

```
PROGRAM␣MAIN[CR]
␣SUB1[CR]
[CR]
END[CR]
```

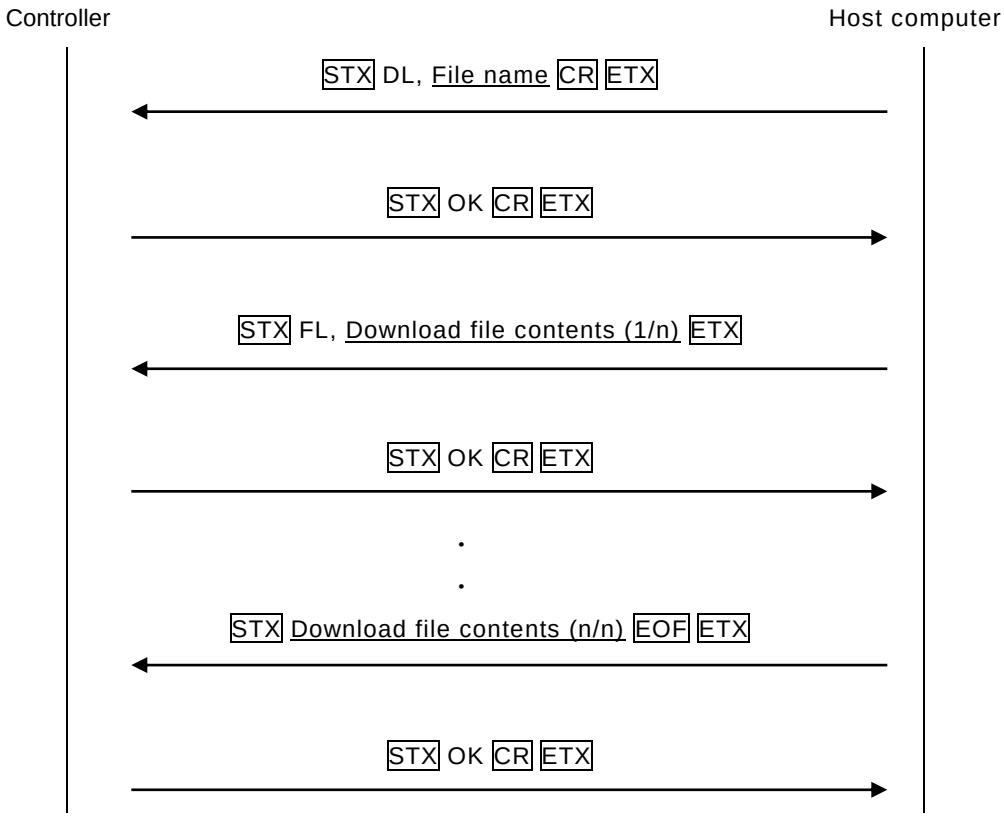
[STX] DL, PRG1 [CR] [ETX]

[STX] FL, PROGRAM␣MAIN [CR] ␣SUB1 [CR] [CR] END [CR] [EOF] [ETX]

Response format

[STX] OK [CR]

Communication example



Note

* When the contents of the file to be downloaded are large, they may not be able to be all sent from the host computer by one communication. Then, to send the rest of the file contents, send them after each receipt of acknowledge. If the download file contents are not sent from the host computer, the controller returns "STX NG CR ETX," not "STX OK CR ETX."

* The controller's receipt of EOF ensures that all the download file contents are received.

DO Execution of DO statement

Function

The DO command is used to directly execute a command from the host computer.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX DO, [Command statement] CR ETX

NO.	Name	Size (byte)	Description
1	Command statement	Variable length	For details on the command, see the Robot Language Manual. In the feed hold status or servo power OFF status, negative response NG is sent back from the controller.

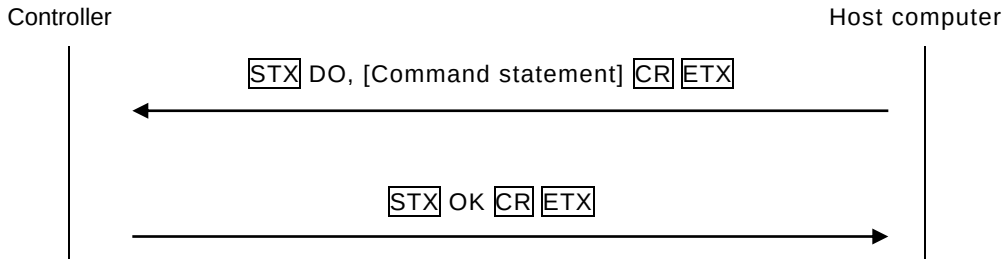
Transmission example

STX DO, MOVEA␣1, 90 CR ETX

Response format

STX OK CR ETX

Communication example



EC Internal command

Function

The internal command can be executed from the host computer.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX EC, [Command] CR ETX

NO.	Name	Size (byte)	Description
1	Command	Variable length	The executable internal commands are as follows. MODE␣CONT: Executes another program cycle after a cycle is complete, and repeats the program until operation is stopped. MODE␣CYCLE: Stops the program after a program cycle is complete. MODE␣STEP: Stops the program after a line of the program is executed. MODE␣SEG: Stops the program each time a Move command is executed. OVRD␣Set value: Sets the operation speed, which can be specified between 1 and 100. BREAK : Immediately makes the robot decelerate to a stop.

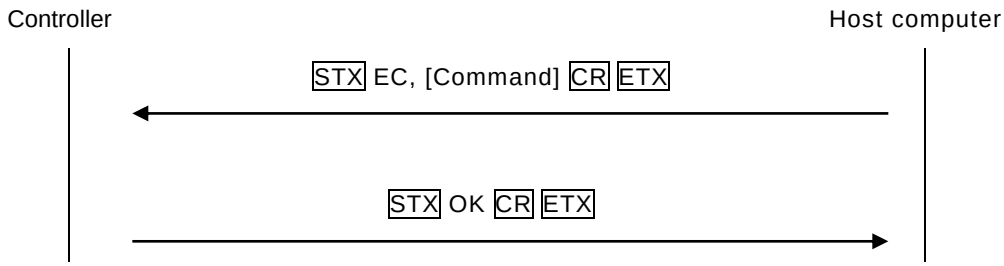
Transmission example

STX EC, MODE␣CONT CR ETX

Response format

STX OK CR ETX

Communication example



Note

* The BREAK command can be accepted only in execution of a SCOL program (status: RUN).

EM Controller mode change request

Function

The EM command is used to direct the controller to change the EXTERNAL mode from the host computer.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX EM, [EXTERNAL mode] CR ETX

NO.	Name	Size (byte)	Description
1	EXTERNAL mode	1 Fixed length	Specify either of the following numbers for the EXTERNAL mode. ・ 2. EXT.ETHER ・ 1: EXT.RS232C

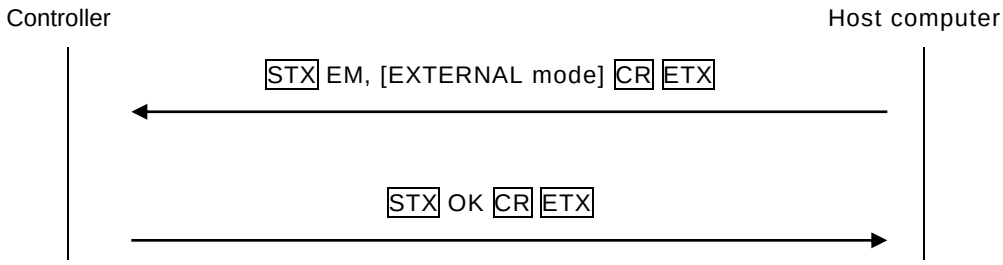
Transmission example

STX EM, 1 CR ETX

Response format

STX OK CR ETX

Communication example



ER File erase

Function

The ER command is used to delete a specified file in the controller RAM drive.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX ER, [File name] CR ETX

NO.	Name	Size (byte)	Description
1	File name	Variable length	The name of a file in the RAM drive to be deleted The file name needs to be defined in either of the following formats. ・ Name.Extension ・ Name The file name must be one to eight characters The extension must be zero to three characters. When a file name is defined without extension, do not include a period (“.”) in the file name.

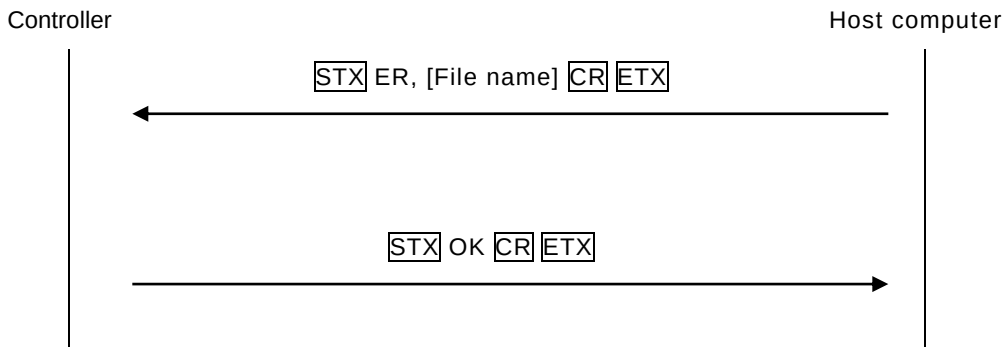
Transmission example

STX ER, PGR1 CR ETX

Response format

STX OK CR ETX

Communication example



EU Error history request

Function

The EU command is used to send the controller error history data to the host computer
In the absence of errors, STX FL, 0 CR ETX is returned.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX EU CR ETX

Response format

STX FL, [Size], Error history (1/n) ETX
STX Error history (2/n) ETX
:
STX Error history (n/n) EOF ETX

Error history contains [Size] records, each of which is as follows.
[Error number] [Date]␣[Time] CR

* A record is defined in the following format: [Error number] [Date]␣[Time] CR.

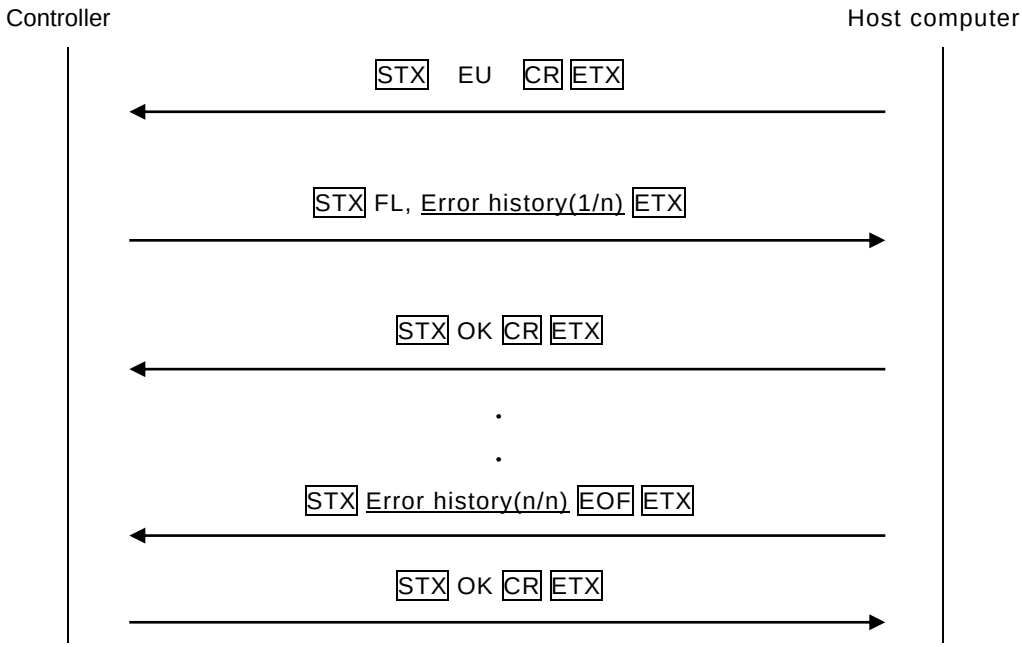
NO.	Name	Size (byte)	Description
1	Size	Variable length	Number (in base 10) of error histories. Placed only at the beginning of the file.
2	Error code	7 Fixed length	The error code of an error which occurred is shown in the following format. “XXX-YYY” XXX : Main code YYY : Sub code For more information on errors, see the "Operator’s Manual."
4	Date	8 Fixed length	The date on which the error occurred is shown in the following format. “YY-MM-DD” YY : Year (Last two numbers only) MM : Month DD : Day
5	Time	8 Fixed length	The time on which the error occurred is shown in the following format. “HH: MM: SS” HH : Hours (In 24-hour"military time.") MM : Minutes SS : Seconds
6	CR	1 Fixed length	Record end code

Response example

When the error history contains two alarms, “Emergency Stop SW ON” and “Safety SW ON,” the controller’s response to the EU command is as follows.

STX FL, 2, 008-014 ␣ 17-06-15 ␣ 10: 32: 18 CR 008-017, ␣ 17-06-15 ␣ 10: 29: 26 CR EOF ETX

Communication example



Note

* When there are many error histories, all the error histories may not be able to be received by one communication. Then, to receive the rest of the error histories, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the error histories.

* The host computer's receipt of EOF ensures that all the error histories are received.

FD Feed hold

Function

The FD command is used to inform from the host computer to the controller that the FEED HOLD pushbutton switch has been pressed.
When the request cannot be accepted, STX NG CR ETX is returned.

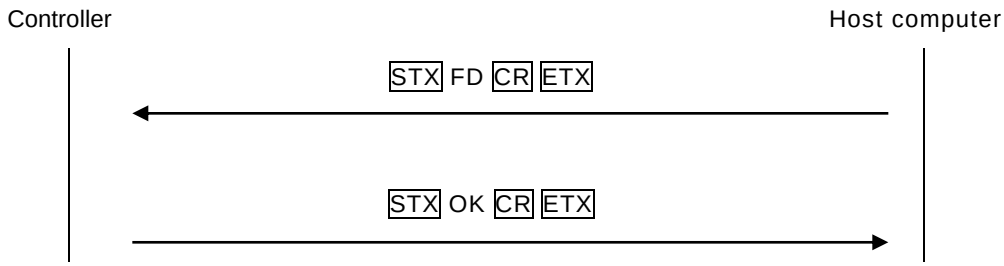
Transmission format

STX FD CR ETX

Response format

STX OK CR ETX

Communication example



HI Hexadecimal-coding and reading of Input signal

Function

The HI command is used to transfer the general input signal information of the controller to the host computer.
More specifically, it is used to transfer to the host computer the part of the general input signal status data that starts from [Start signal number] and has a length of [Signal length] .

Transmission format

STX HI, [Start signal number] ␣ [Signal length] CR ETX

NO.	Name	Size (byte)	Description
1	Start signal number	Variable length	Specify one of the following numbers for the start signal number. 1 to 64 101 to 164 201 to 264 301 to 364 401 to 464
2	Signal length	Variable length	Specify one of the following numbers for the signal length. 1 to 32

Transmission example

STX HI, 2 ␣ 4 CR ETX

Response format

STX FL, General input signal status (1/1) EOF ETX

General input signal status contains the following.

[Input value]

NO.	Name	Size (byte)	Description
1	Input value	Variable length	A decimal value for the general input signal status

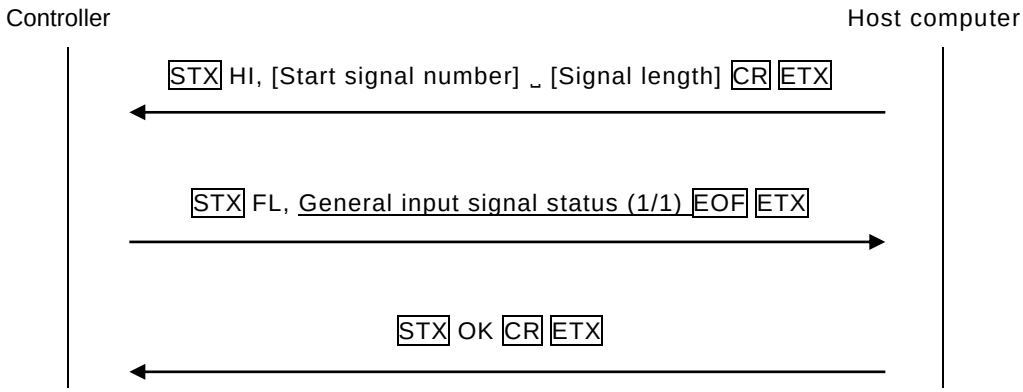
Response example

For the general input signal status given below, specifying 2 for the start signal number and 4 for the signal length leads to the following response from the controller to the HI command.

Signal No.	1	**2**	**3**	**4**	**5**	6	7	8
Signal status	OFF	**ON**	**ON**	**OFF**	**ON**	OFF	OFF	ON

STX FL, 11 EOF ETX

Communication example



HO Hexadecimal-coding and writing of output signal

Function

The HO command is used to direct the controller to overwrite the general output signals.
It also transfers to the controller the part of [Output value] that starts from [Start signal number] and has a length of [Signal length].

Transmission format

STX HO, [Start signal name] ␣ [Signal length] ␣ [Output value] CR ETX

NO.	Name	Size (byte)	Description
1	Start signal number	Variable length	Specify one of the following numbers for the start signal number. 1 to 64 101 to 164 201 to 264 301 to 364 401 to 464
2	Signal length	Variable length	Specify one of the following numbers for the signal length. 1 to 32
3	Output value	Variable length	Specify one of the following numbers for the output value. Decimal number of more than 0

Transmission example

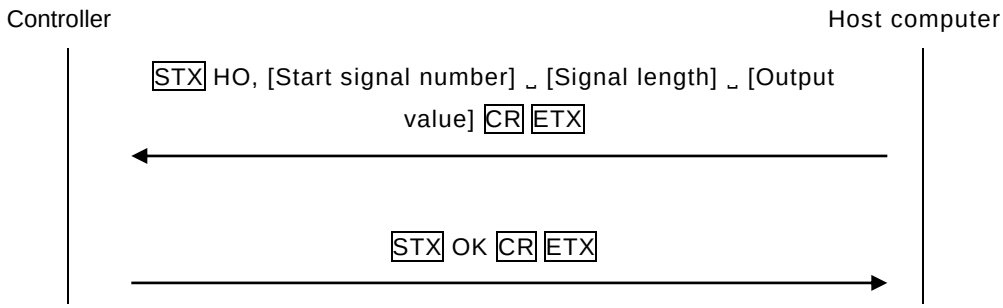
When the HO command is used to set the start signal number to 2, the signal length to 4, and the output value to 11 and sent to the controller, the resulting general output signals are as follows.
STX HO, 2 ␣ 4 ␣ 11 CR ETX

Signal No.	1	**2**	**3**	**4**	**5**	6	7	8
Signal status	-	**ON**	**ON**	**OFF**	**ON**	-	-	-

Response format

STX OK CR ETX

Communication example



IO I/O information acquisiton

Function

The IO command is used to transfer the I/O information of a specified model to the host computer.

Transmission format

STX IO, [Model code] CR ETX

NO.	Name	Size (byte)	Description
1	Model code	1 Fixed length	Specify the following number for the model code. 1: TS3000 series robot controller

Transmission example

STX IO, 1 CR ETX

Response format

STX FL, I/O information (1/1) EOF ETX

I/O information contains the following.
[General input 1] [General input 2] [General input 3] [General input 4] [Extension input 1] [Extension input 2] [Extension input 3] [Extension input 4] [System input 1] [System input 2] [System input 3] [System input 4] [Field bus input 1] [Field bus input 2] [Field bus input 3] [Field bus input 4] [Field bus input 5] [Field bus input 6] [Field bus input 7] [Field bus input 8] [General output 1] [General output 2] [General output 3] [General output 4] [Extension output 1] [Extension output 2] [Extension output 3] [Extension output 4] [System output 1] [System output 2] [System output 3] [System output 4] [Field bus output 1] [Field bus output 2] [Field bus output 3] [Field bus output 4] [Field bus output 5] [Field bus output 6] [Field bus output 7] [Field bus output 8]

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>1</td><td>General input 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din1</td><td>General input</td></tr>
<tr><td>1</td><td>Din2</td><td>General input</td></tr>
<tr><td>2</td><td>Din3</td><td>General input</td></tr>
<tr><td>3</td><td>Din4</td><td>General input</td></tr>
<tr><td>4</td><td>Din5</td><td>General input</td></tr>
<tr><td>5</td><td>Din6</td><td>General input</td></tr>
<tr><td>6</td><td>Din7</td><td>General input</td></tr>
<tr><td>7</td><td>Din8</td><td>General input</td></tr>
<tr><td>8</td><td>Din9</td><td>General input</td></tr>
<tr><td>9</td><td>Din10</td><td>General input</td></tr>
<tr><td>10</td><td>Din11</td><td>General input</td></tr>
<tr><td>11</td><td>Din12</td><td>General input</td></tr>
<tr><td>12</td><td>Din13</td><td>General input</td></tr>
<tr><td>13</td><td>Din14</td><td>General input</td></tr>
<tr><td>14</td><td>Din15</td><td>General input</td></tr>
<tr><td>15</td><td>Din16</td><td>General input</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>General input 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din17</td><td>General input</td></tr>
<tr><td>1</td><td>Din18</td><td>General input</td></tr>
<tr><td>2</td><td>Din19</td><td>General input</td></tr>
<tr><td>3</td><td>Din20</td><td>General input</td></tr>
<tr><td>4</td><td>Din21</td><td>General input</td></tr>
<tr><td>5</td><td>Din22</td><td>General input</td></tr>
<tr><td>6</td><td>Din23</td><td>General input</td></tr>
<tr><td>7</td><td>Din24</td><td>General input</td></tr>
<tr><td>8</td><td>Din25</td><td>General input</td></tr>
<tr><td>9</td><td>Din26</td><td>General input</td></tr>
<tr><td>10</td><td>Din27</td><td>General input</td></tr>
<tr><td>11</td><td>Din28</td><td>General input</td></tr>
<tr><td>12</td><td>Din29</td><td>General input</td></tr>
<tr><td>13</td><td>Din30</td><td>General input</td></tr>
<tr><td>14</td><td>Din31</td><td>General input</td></tr>
<tr><td>15</td><td>Din32</td><td>General input</td></tr>
</table>
</td></tr>
</table>

3	General input 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din33	General input
1	Din34	General input
2	Din35	General input
3	Din36	General input
4	Din37	General input
5	Din38	General input
6	Din39	General input
7	Din40	General input
8	Din41	General input
9	Din42	General input
10	Din43	General input
11	Din44	General input
12	Din45	General input
13	Din46	General input
14	Din47	General input
15	Din48	General input

4	General input 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din49	General input
1	Din50	General input
2	Din51	General input
3	Din52	General input
4	Din53	General input
5	Din54	General input
6	Din55	General input
7	Din56	General input
8	Din57	General input
9	Din58	General input
10	Din59	General input
11	Din60	General input
12	Din61	General input
13	Din62	General input
14	Din63	General input
15	Din64	General input

<table>
<tr>
<td>5</td>
<td>Extension input 1
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din101</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din102</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din103</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din104</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din105</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din106</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din107</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din108</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din109</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din110</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din111</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din112</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din113</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din114</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din115</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din116</td><td>Extension input</td></tr>
</table>
</td>
</tr>
<tr>
<td>6</td>
<td>Extension input 2
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din117</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din118</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din119</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din120</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din121</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din122</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din123</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din124</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din125</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din126</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din127</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din128</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din129</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din130</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din131</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din132</td><td>Extension input</td></tr>
</table>
</td>
</tr>
</table>

7	Extension input 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din133	Extension input
1	Din134	Extension input
2	Din135	Extension input
3	Din136	Extension input
4	Din137	Extension input
5	Din138	Extension input
6	Din139	Extension input
7	Din140	Extension input
8	Din141	Extension input
9	Din142	Extension input
10	Din143	Extension input
11	Din144	Extension input
12	Din145	Extension input
13	Din146	Extension input
14	Din147	Extension input
15	Din148	Extension input

8	Extension input 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din149	Extension input
1	Din150	Extension input
2	Din151	Extension input
3	Din152	Extension input
4	Din153	Extension input
5	Din154	Extension input
6	Din155	Extension input
7	Din156	Extension input
8	Din157	Extension input
9	Din158	Extension input
10	Din159	Extension input
11	Din160	Extension input
12	Din161	Extension input
13	Din162	Extension input
14	Din163	Extension input
15	Din164	Extension input

9	System input 1 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din201	Hand input
1	Din202	Hand input
2	Din203	Hand input
3	Din204	Hand input
4	Din205	Hand input
5	Din206	Hand input
6	Din207	Hand input
7	Din208	Hand input
8	Din209	
9	Din210	
10	Din211	
11	Din212	
12	Din213	
13	Din214	
14	Din215	
15	Din216	

10	System input 2 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din217	Alarm of level 8
1	Din218	Alarm of level 8
2	Din219	Alarm of level 8
3	Din220	Alarm of level 8
4	Din221	Alarm of level 4
5	Din222	Alarm of level 4
6	Din223	Alarm of level 4
7	Din224	Alarm of level 4
8	Din225	Alarm of level 2
9	Din226	Alarm of level 2
10	Din227	Alarm of level 2
11	Din228	Alarm of level 2
12	Din229	Alarm of level 1
13	Din230	Alarm of level 1
14	Din231	Alarm of level 1
15	Din232	Alarm of level 1

11	System input 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din233	
1	Din234	
2	Din235	
3	Din236	
4	Din237	
5	Din238	
6	Din239	
7	Din240	
8	Din241	
9	Din242	
10	Din243	
11	Din244	
12	Din245	
13	Din246	
14	Din247	
15	Din248	

12	System input 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din249	STROBE
1	Din250	PRG_RST
2	Din251	STEP_RST
3	Din252	CYC_RST
4	Din253	DO_RST
5	Din254	ALM_RST
6	Din255	RUN
7	Din256	EX_SVON
8	Din257	STOP
9	Din258	CYCLE
10	Din259	LOW_SPD
11	Din260	BREAK
12	Din261	SVOFF
13	Din262	
14	Din263	
15	Din264	

<table>
<tr>
<td>13</td>
<td>Field bus input 1
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>DIN301</td><td></td></tr>
<tr><td>1</td><td>DIN302</td><td></td></tr>
<tr><td>2</td><td>DIN303</td><td></td></tr>
<tr><td>3</td><td>DIN304</td><td></td></tr>
<tr><td>4</td><td>DIN305</td><td></td></tr>
<tr><td>5</td><td>DIN306</td><td></td></tr>
<tr><td>6</td><td>DIN307</td><td></td></tr>
<tr><td>7</td><td>DIN308</td><td></td></tr>
<tr><td>8</td><td>DIN309</td><td></td></tr>
<tr><td>9</td><td>DIN310</td><td></td></tr>
<tr><td>10</td><td>DIN311</td><td></td></tr>
<tr><td>11</td><td>DIN312</td><td></td></tr>
<tr><td>12</td><td>DIN313</td><td></td></tr>
<tr><td>13</td><td>DIN314</td><td></td></tr>
<tr><td>14</td><td>DIN315</td><td></td></tr>
<tr><td>15</td><td>DIN316</td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td>14</td>
<td>Field bus input 2
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>DIN317</td><td></td></tr>
<tr><td>1</td><td>DIN318</td><td></td></tr>
<tr><td>2</td><td>DIN319</td><td></td></tr>
<tr><td>3</td><td>DIN320</td><td></td></tr>
<tr><td>4</td><td>DIN321</td><td></td></tr>
<tr><td>5</td><td>DIN322</td><td></td></tr>
<tr><td>6</td><td>DIN323</td><td></td></tr>
<tr><td>7</td><td>DIN324</td><td></td></tr>
<tr><td>8</td><td>DIN325</td><td></td></tr>
<tr><td>9</td><td>DIN326</td><td></td></tr>
<tr><td>10</td><td>DIN327</td><td></td></tr>
<tr><td>11</td><td>DIN328</td><td></td></tr>
<tr><td>12</td><td>DIN329</td><td></td></tr>
<tr><td>13</td><td>DIN330</td><td></td></tr>
<tr><td>14</td><td>DIN331</td><td></td></tr>
<tr><td>15</td><td>DIN332</td><td></td></tr>
</table>
</td>
</tr>
</table>

15	Field bus input 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din333	
1	Din334	
2	Din335	
3	Din336	
4	Din337	
5	Din338	
6	Din339	
7	Din340	
8	Din341	
9	Din342	
10	Din343	
11	Din344	
12	Din345	
13	Din346	
14	Din347	
15	Din348	

16	Field bus input 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din349	
1	Din350	
2	Din351	
3	Din352	
4	Din353	
5	Din354	
6	Din355	
7	Din356	
8	Din357	
9	Din358	
10	Din359	
11	Din360	
12	Din361	
13	Din362	
14	Din363	
15	Din364	

17	Field bus input 5 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	DIN401	
1	DIN402	
2	DIN403	
3	DIN404	
4	DIN405	
5	DIN406	
6	DIN407	
7	DIN408	
8	DIN409	
9	DIN410	
10	DIN411	
11	DIN412	
12	DIN413	
13	DIN414	
14	DIN415	
15	DIN416	

18	Field bus input 6 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	SignalNo.	Signal name
0	DIN417	
1	DIN418	
2	DIN419	
3	DIN420	
4	DIN421	
5	DIN422	
6	DIN423	
7	DIN424	
8	DIN425	
9	DIN426	
10	DIN427	
11	DIN428	
12	DIN429	
13	DIN430	
14	DIN431	
15	DIN432	

19	Field bus input 7 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din433	
1	Din434	
2	Din435	
3	Din436	
4	Din437	
5	Din438	
6	Din439	
7	Din440	
8	Din441	
9	Din442	
10	Din443	
11	Din444	
12	Din445	
13	Din446	
14	Din447	
15	Din448	

20	Field bus input 8 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Din449	
1	Din450	
2	Din451	
3	Din452	
4	Din453	
5	Din454	
6	Din455	
7	Din456	
8	Din457	
9	Din458	
10	Din459	
11	Din460	
12	Din461	
13	Din462	
14	Din463	
15	Din464	

21	General output 1 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout1	General output
1	Dout2	General output
2	Dout3	General output
3	Dout4	General output
4	Dout5	General output
5	Dout6	General output
6	Dout7	General output
7	Dout8	General output
8	Dout9	General output
9	Dout10	General output
10	Dout11	General output
11	Dout12	General output
12	Dout13	General output
13	Dout14	General output
14	Dout15	General output
15	Dout16	General output

22	General output 2 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout17	General output
1	Dout18	General output
2	Dout19	General output
3	Dout20	General output
4	Dout21	General output
5	Dout22	General output
6	Dout23	General output
7	Dout24	General output
8	Dout25	General output
9	Dout26	General output
10	Dout27	General output
11	Dout28	General output
12	Dout29	General output
13	Dout30	General output
14	Dout31	General output
15	Dout32	General output

23	General output 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout33	General output
1	Dout34	General output
2	Dout35	General output
3	Dout36	General output
4	Dout37	General output
5	Dout38	General output
6	Dout39	General output
7	Dout40	General output
8	Dout41	General output
9	Dout42	General output
10	Dout43	General output
11	Dout44	General output
12	Dout45	General output
13	Dout46	General output
14	Dout47	General output
15	Dout48	General output

24	General output 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout49	General output
1	Dout50	General output
2	Dout51	General output
3	Dout52	General output
4	Dout53	General output
5	Dout54	General output
6	Dout55	General output
7	Dout56	General output
8	Dout57	General output
9	Dout58	General output
10	Dout59	General output
11	Dout60	General output
12	Dout61	General output
13	Dout62	General output
14	Dout63	General output
15	Dout64	General output

<table>
<tr>
<td>25</td>
<td>Extension output 1
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout101</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout102</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout103</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout104</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout105</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout106</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout107</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout108</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout109</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout110</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout111</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout112</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout113</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout114</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout115</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout116</td><td>Extension output</td></tr>
</table>
</td>
</tr>
<tr>
<td>26</td>
<td>Extension output 2
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout117</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout118</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout119</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout120</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout121</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout122</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout123</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout124</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout125</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout126</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout127</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout128</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout129</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout130</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout131</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout132</td><td>Extension output</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr>
<td>27</td>
<td>Extension output 3
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout133</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout134</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout135</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout136</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout137</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout138</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout139</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout140</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout141</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout142</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout143</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout144</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout145</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout146</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout147</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout148</td><td>Extension output</td></tr>
</table>
</td>
</tr>
<tr>
<td>28</td>
<td>Extension output 4
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout149</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout150</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout151</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout152</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout153</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout154</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout155</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout156</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout157</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout158</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout159</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout160</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout161</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout162</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout163</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout164</td><td>Extension output</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr>
<td>29</td>
<td>System output 1
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout201</td><td>Hand output</td></tr>
<tr><td>1</td><td>Dout202</td><td>Hand output</td></tr>
<tr><td>2</td><td>Dout203</td><td>Hand output</td></tr>
<tr><td>3</td><td>Dout204</td><td>Hand output</td></tr>
<tr><td>4</td><td>Dout205</td><td>Hand output</td></tr>
<tr><td>5</td><td>Dout206</td><td>Hand output</td></tr>
<tr><td>6</td><td>Dout207</td><td>Hand output</td></tr>
<tr><td>7</td><td>Dout208</td><td>Hand output</td></tr>
<tr><td>8</td><td>Dout209</td><td></td></tr>
<tr><td>9</td><td>Dout210</td><td></td></tr>
<tr><td>10</td><td>Dout211</td><td></td></tr>
<tr><td>11</td><td>Dout212</td><td></td></tr>
<tr><td>12</td><td>Dout213</td><td></td></tr>
<tr><td>13</td><td>Dout214</td><td></td></tr>
<tr><td>14</td><td>Dout215</td><td></td></tr>
<tr><td>15</td><td>Dout216</td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td>30</td>
<td>System output 2
(Binary value)
(0000 to FFFF)</td>
<td>2
Fixed length</td>
<td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout217</td><td>Seq. parameter</td></tr>
<tr><td>1</td><td>Dout218</td><td>Seq. parameter</td></tr>
<tr><td>2</td><td>Dout219</td><td>Seq. parameter</td></tr>
<tr><td>3</td><td>Dout220</td><td>Seq. parameter</td></tr>
<tr><td>4</td><td>Dout221</td><td>Seq. parameter</td></tr>
<tr><td>5</td><td>Dout222</td><td>Seq. parameter</td></tr>
<tr><td>6</td><td>Dout223</td><td>Seq. parameter</td></tr>
<tr><td>7</td><td>Dout224</td><td>Seq. parameter</td></tr>
<tr><td>8</td><td>Dout225</td><td></td></tr>
<tr><td>9</td><td>Dout226</td><td></td></tr>
<tr><td>10</td><td>Dout227</td><td></td></tr>
<tr><td>11</td><td>Dout228</td><td></td></tr>
<tr><td>12</td><td>Dout229</td><td></td></tr>
<tr><td>13</td><td>Dout230</td><td></td></tr>
<tr><td>14</td><td>Dout231</td><td></td></tr>
<tr><td>15</td><td>Dout232</td><td></td></tr>
</table>
</td>
</tr>
</table>

31	System output 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout233	
1	Dout234	
2	Dout235	
3	Dout236	
4	Dout237	
5	Dout238	
6	Dout239	
7	Dout240	
8	Dout241	
9	Dout242	
10	Dout243	
11	Dout244	
12	Dout245	
13	Dout246	
14	Dout247	
15	Dout248	

32	System output 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout249	EMG_ST
1	Dout250	SV_RDY
2	Dout251	ACK
3	Dout252	TEACH
4	Dout253	INT
5	Dout254	EXT_SIG
6	Dout255	EXT_232C
7	Dout256	SYS_RDY
8	Dout257	AUTORUN
9	Dout258	CYC_END
10	Dout259	LOW_ST
11	Dout260	CYC_ST
12	Dout261	BT_ALM
13	Dout262	ALARM
14	Dout263	EXT_ETHER
15	Dout264	

33	Field bus output 1 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout301	
1	Dout302	
2	Dout303	
3	Dout304	
4	Dout305	
5	Dout306	
6	Dout307	
7	Dout308	
8	Dout309	
9	Dout310	
10	Dout311	
11	Dout312	
12	Dout313	
13	Dout314	
14	Dout315	
15	Dout316	

34	Field bus output 2 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout317	
1	Dout318	
2	Dout319	
3	Dout320	
4	Dout321	
5	Dout322	
6	Dout323	
7	Dout324	
8	Dout325	
9	Dout326	
10	Dout327	
11	Dout328	
12	Dout329	
13	Dout330	
14	Dout331	
15	Dout332	

35	Field bus output 3 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout333	
1	Dout334	
2	Dout335	
3	Dout336	
4	Dout337	
5	Dout338	
6	Dout339	
7	Dout340	
8	Dout341	
9	Dout342	
10	Dout343	
11	Dout344	
12	Dout345	
13	Dout346	
14	Dout347	
15	Dout348	

36	Field bus output 4 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout349	
1	Dout350	
2	Dout351	
3	Dout352	
4	Dout353	
5	Dout354	
6	Dout355	
7	Dout356	
8	Dout357	
9	Dout358	
10	Dout359	
11	Dout360	
12	Dout361	
13	Dout362	
14	Dout363	
15	Dout364	

37	Field bus output 5 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout401	
1	Dout402	
2	Dout403	
3	Dout404	
4	Dout405	
5	Dout406	
6	Dout407	
7	Dout408	
8	Dout409	
9	Dout410	
10	Dout411	
11	Dout412	
12	Dout413	
13	Dout414	
14	Dout415	
15	Dout416	

38	Field bus output 6 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout417	
1	Dout418	
2	Dout419	
3	Dout420	
4	Dout421	
5	Dout422	
6	Dout423	
7	Dout424	
8	Dout425	
9	Dout426	
10	Dout427	
11	Dout428	
12	Dout429	
13	Dout430	
14	Dout431	
15	Dout432	

39	Field bus output 7 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout433	
1	Dout434	
2	Dout435	
3	Dout436	
4	Dout437	
5	Dout438	
6	Dout439	
7	Dout440	
8	Dout441	
9	Dout442	
10	Dout443	
11	Dout444	
12	Dout445	
13	Dout446	
14	Dout447	
15	Dout448	

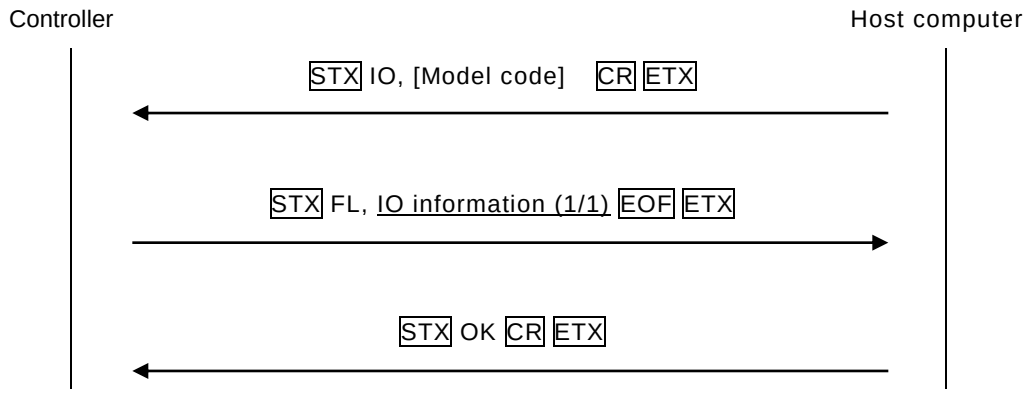
40	Field bus output 8 (Binary value) (0000 to FFFF)	2 Fixed length	

Bit	Signal No.	Signal name
0	Dout449	
1	Dout450	
2	Dout451	
3	Dout452	
4	Dout453	
5	Dout454	
6	Dout455	
7	Dout456	
8	Dout457	
9	Dout458	
10	Dout459	
11	Dout460	
12	Dout461	
13	Dout462	
14	Dout463	
15	Dout464	

Response example

STX FL, 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3F 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 ETX

Communication example



Note

* All responses to the IO command are returned in binary code.

IW I/O Information write

Function

The IW command is used to direct the controller to overwrite up to 32 general output signals from the host computer.

Transmission format

For one general output signal, use the IW command as follows.
STX IW, [Signal No.] ␣ [ON/OFF command] CR ETX

For more than one general output signal, use the IW command as follows.
STX IW, [Signal No.] ␣ [ON/OFF command] ␣ [Signal No.] ␣ [ON/OFF command] ␣ [Signal No.] ␣ [ON/OFF command] … ␣ [Signal No.] ␣ [ON/OFF command] CR ETX

NO.	Name	Size (byte)	Description
1	Signal No.	Variable length	Specify one of the following numbers for the signal number.
2	ON/OFF command	1 Fixed length	Specify either of the following numbers for the ON/OFF command 0: OFF command 1: ON command

Transmission example

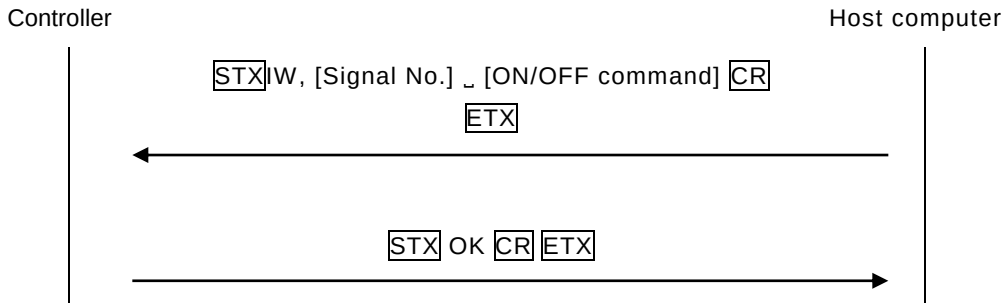
When the IW command is used to set the general output signal No. 2 to ON and is transferred to the host computer, the resulting general output signals are as follows.
STX IW, 2 ␣ 1 CR ETX

Signal No.	1	**2**	3	4	5	6	7	8
Signal status	-	**ON**	-	-	-	-	-	-

Response format

STX OK CR ETX

Communication example



JG JOG execution

Function

The JG command is used to direct the controller to start jog from the host computer.

Transmission format

STX JG, [Axis] ␣ [Direction] CR ETX

NO.	Name	Size (byte)	Description
1	Axis	1 Fixed length	Specify one of the following numbers for the axis. 1 to 5
2	Direction	1 Fixed length	Specify one of the following items for the direction +: Plus direction jog -: Minus direction jog !: Jog motion stop

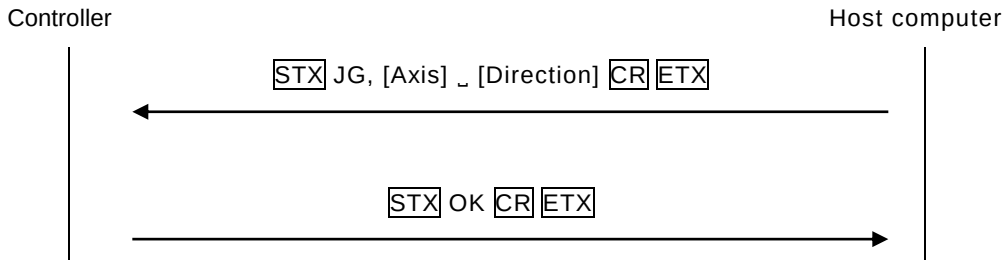
Transmission example

STXJG, 1 ␣ + CR ETX

Response format

STX OK CR ETX

Communication example



Note

* Be sure to keep your hand close to the emergency stop switch while using the JG command so that you can press the emergency stop switch at any time.

MD Guide mode setting

Function

The MD command is used to set the guide mode of the controller from the host computer.

Transmission format

STX MD, [Guide mode] CR ETX

NO.	Name	Size (byte)	Description
1	Guide mode	1 Fixed length	Specify one of the following numbers for the guide mode. 0: JOG 1: INCHING 2: FREE

Transmission example

STX MD, 0 CR ETX

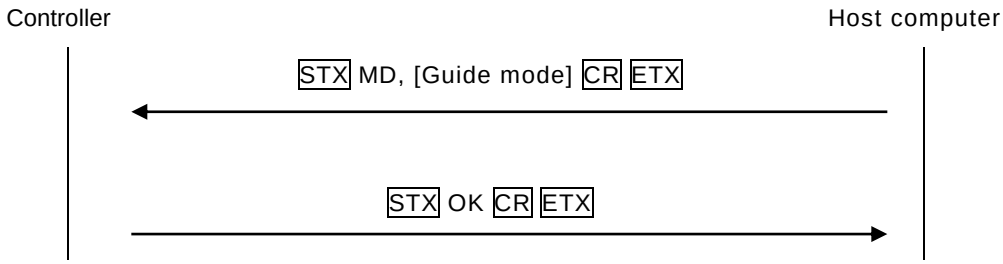
Response format

STX OK CR ETX

Response example

STX OK CR ETX

Communication example



MN Maintenance information acquisition

Function

The MN command is used to transfer the specified maintenance information to the host computer.

Transmission format

STX MN, [Maintenance information number] CR ETX

NO.	Name	Size (byte)	Description
1	Maintenance information number	Variable length	Specify one of the following numbers for the maintenance information number. 1: Total rotational amount of motor (rev) 2: Movement amount of joint axis (mm or deg) 3: Integrated amount of motor torque (%) 4: Integrated amount of motor torque exceeding rating (%) 5: Peak torque (%) (cleared by communication request) 6: 3-second load factor (%) 7: 1-minute load factor (%) 8: 15-minute load factor (%) 9: 2-hour load factor (%) 10: Total time 11: Energizing time 12: Power OFF time 13: Motion time 14: Fault time 15: Servo information (servo OFF count) 16: Total servo ON time 17: Peak torque (%) (Not cleared by communication request)

Transmission example

STX MN, 1 CR ETX

Response format

STX FL, Maintenance information (1/1) ETX

Maintenance information contains the following information; the controller's response to the MN command is returned in Format 1 for the maintenance information number 1 to 9, or 17, and in Format 2 for 10 to 16.

- Format 1

[Maintenance information number], [Axis 1 information], [Axis 2 information], [Axis 3 information], [Axis 4 information], [Axis 5 information], [Axis 6 information]

- Format 2

[Maintenance information number], [Current information]

- Format 1

NO.	Name	Size (byte)	Description
1	Maintenance information number	Variable length	Maintenance information number of request
2 to 7	Axis information	Variable length	Axis 1 to axis 6 information This value is sent as a real number to one decimal point. * The contents of response vary depending on the maintenance information number. 1: Total rotational amount of motor (rev) 2: Movement amount of joint axis (mm or deg) 3: Integrated amount of motor torque (%) 4: Integrated amount of motor torque exceeding rating (%) 5: Peak torque (%) 6: 3-second load factor (%) 7: 1-minute load factor (%) 8: 15-minute load factor (%) 9: 2-hour load factor (%) 17: Peak torque (%)

・ Format 2

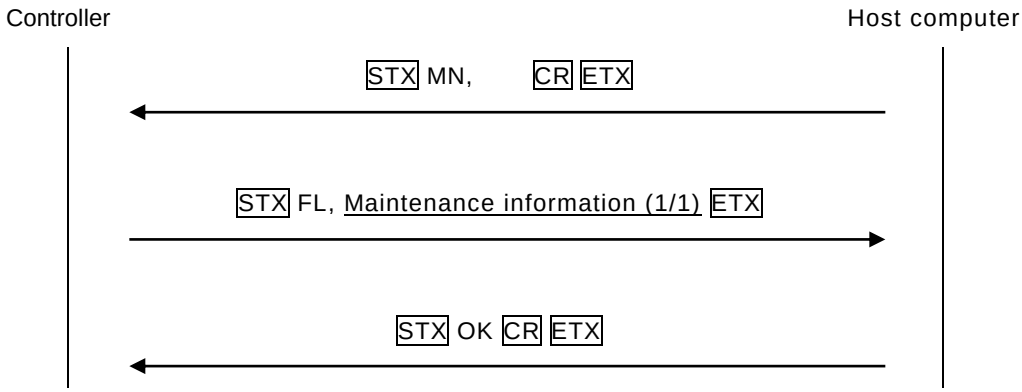
NO.	Name	Size (byte)	Description
1	Maintenance information number	Variable length	Maintenance information number of request
2	Information	Variable length	Information This value is sent as a real number to one decimal point. * The contents of response vary depending on the maintenance information number. 10: Total time (seconds) 11: Energizing time (seconds) 12: Power OFF time (seconds) 13: Motion time (seconds) 14: Fault time (seconds) 15: Servo information (servo OFF count) 16: Total servo ON time (seconds)

Response example

The controller's response to the MN command for obtaining the total rotational amount of motor is as follows.

STX FL, 1, 46.4, 0.0, 0.0, 0.0, 0.0, 0.0 ETX

Communication example



MP Teaching point movement

Function

The MP command is used to direct the controller to start teaching point movement from the host computer.

Transmission format

STX MP, Teaching point information CR ETX

Teaching point information contains the following.
[Coordinate identification code] ␣ [X coordinate value] ␣ [Y coordinate value] ␣
[Z coordinate value] ␣ [C coordinate value] ␣ [T coordinate value] ␣ [Configuration]

NO.	Name	Size (byte)	Description
1	Coordinate identification code	1 Fixed length	Specify one of the following numbers for the coordinate identification code. 0: Normal movement motion 1: Move the Z axis at the current position
2 to 6	Coordinate value	Variable length	Specify real numbers to three decimal places for the coordinate values of the teaching point to be moved. The unit for each axis is as follows. X axis: mm Y axis: mm Z axis: mm C axis: deg T axis: Check the specification of the machine.
7	Configuration	5 Fixed length	Specify one of the following character strings for the teaching configuration. FREE LEFTY RIGHTY

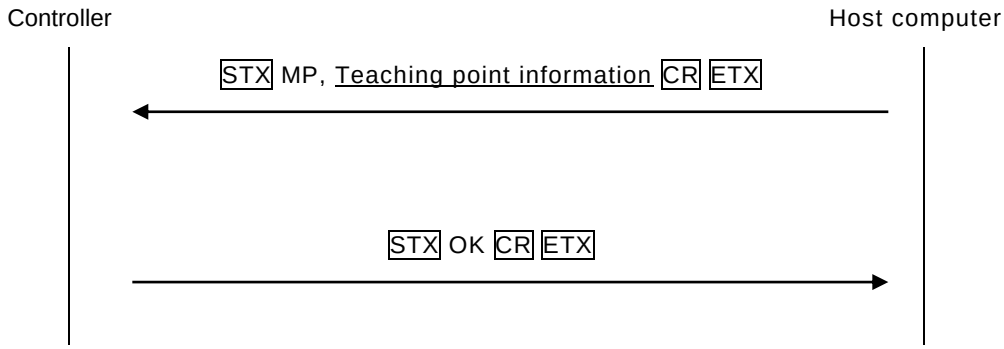
Transmission example

STX MP, 0 ␣ 100.0 ␣ 200.0 ␣ 100.0 ␣ 0.0 ␣ 0.0 ␣ FREE CR ETX

Response format

STX OK CR ETX

Communication example



Note

* Be sure to keep your hand close to the emergency stop switch while using the MP command so that you can press the emergency stop switch at any time.

MR Read global variable

Function

The MR command is used to transfer the read data of a defined global variable to the host computer.

Transmission format

STX MR, [Variable name] ␣ [Variable type] CR ETX

NO.	Name	Size (byte)	Description
1	Variable name	Variable length	Specify the name of a global variable to read. For the array variable, specify the array elements ("Variable name (*, *, •••) ").
2	Variable type	Variable length	Specify one of the following numbers for the variable type data. 0 : Integer type 5 : Array integer type 1 : Real number type 6 : Array real number type 2 : Load type 7 : Array load type 3 : Coordinate type 8 : Array coordinate type 4 : Position type 9 : Array position type -1 : Unclarified

Transmission example

STX MR, X␣0 CR ETX

Response format

STX FL, Read data (1/1) ETX

Read data contains the following.

[Read data 1] ␣ [Read data 2] ␣ [Read data 3] ␣ ･･･

NO.	Name	Size (byte)	Description
1	Read data	– Variable length	The variable Read data must be specified in one of the following formats depending on its variable type.

Type	Read data		
Integer type	Format	long type data	
	No. of pcs	1 pc.	
	Type	long type data	long
Real number type	Format	float type data	
	No. of pcs	1 pc.	
	Type	float type	float
Load type	Format	Mass ␣ Center of gravity offset	
	No. of pcs	2 pcs.	
	Type	Mass	float
		Center of gravity offset	float
Coordinate type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value	
	No. of pcs	4 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
Position type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value ␣ T coordinate value ␣ Configuration	
	No. of pcs	6 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
		T coordinate value	float
		Configuration	Specify one of the following numbers for the configuration. 0.000: FREE 1.000: LEFTY 2.000: RIGHTY

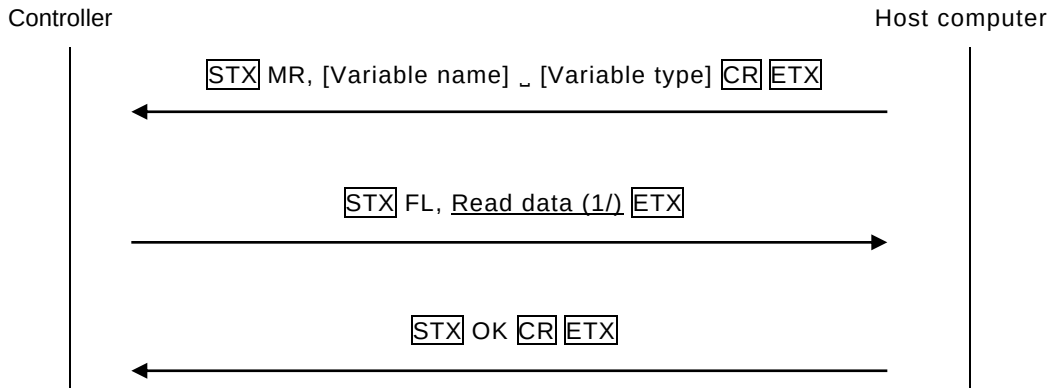
Type	Read data		
Array integer type	Format	long type data	
	No. of pcs	1 pc.	
	Type	long type data	long
Array real number type	Format	float type data	
	No. of pcs	1 pc.	
	Type	float type	float
Array load type	Format	Mass ␣ Center of gravity offset	
	No. of pcs	2 pcs.	
	Type	Mass	float
		Center of gravity offset	float
Array coordinate type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value	
	No. of pcs	4 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
Array position type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣Configuration	
	No. of pcs	6 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
		T coordinate value	float
		Configuration	Specify one of the following numbers for the configuration. 0.000: FREE 1.000: LEFTY 2.000: RIGHTY

The float data is expressed by a value calculated to three decimal places.

Response example

STX FL, 100 EOF ETX

Communication example



Note

* When there is much read data, it may not be able to be all received by one communication. Then, to receive the rest of the read data, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the read data.

* The host computer's receipt of EOF ensures that all the read data is received.

MW Write global variable

Function

The MW command is used to write data to a global variable defined in the controller from the host computer,

Transmission format

STX MW, [Flag] ␣ [Variable name] ␣ [Variable type] ␣ [Write data] CR ETX

NO.	Name	Size (byte)	Description
1	Flag	1 Fixed length	The flag is a control flag for restoring or non-restoring of data in the program file, as shown below. 0: Restoring of data in the program file is not executed. 1: Restoring of data in the program file is executed.
2	Variable name	Variable length	Specify the name of a global variable to write. For the array variable, specify the array elements ("Variable name (*, *, •••) ").
3	Variable type	1 Fixed length	Specify one of the following numbers for the variable type data. 0 : Integer type 5 : Array integer type 1 : Real number type 6 : Array real number type 2 : Load type 7 : Array load type 3 : Coordinate type 8 : Array coordinate type 4 : Position type 9 : Array position type
4	Write data	Variable length	The variable Write data must be specified in one of the following formats depending on its variable type.

Type	Write data		
Integer type	Format	long type data	
	No. of pcs	1 pc.	
	Type	long type data	long
Real number type	Format	float type data	
	No. of pcs	1 pc.	
	Type	float type	float
Load type	Format	Mass ␣ Center of gravity offset	
	No. of pcs	2 pcs.	
	Type	Mass	float
		Center of gravity offset	float
Coordinate type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value	
	No. of pcs	4 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
Position type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ Configuration	
	No. of pcs	6 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
		T coordinate value	float
		Configuration	Specify one of the following numbers for the configuration. 0.000: FREE 1.000: LEFTY 2.000: RIGHTY

Type	Write data		
Array integer type	Format	long type data	
	No. of pcs	1 pc.	
	Type	long type data	long
Array real number type	Format	float type data	
	No. of pcs	1 pc.	
	Type	float type	float
Array load type	Format	Mass ␣ Center of gravity offset	
	No. of pcs	2 pcs.	
	Type	Mass	float
		Center of gravity offset	float
Array coordinate type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value	
	No. of pcs	4 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
Array position type	Format	X coordinate value ␣ Y coordinate value ␣ Z coordinate value ␣ C coordinate value ␣ T coordinate value ␣ Configuration	
	No. of pcs	6 pcs.	
	Type	X coordinate value	float
		Y coordinate value	float
		Z coordinate value	float
		C coordinate value	float
		T coordinate value	float
		Configuration	Specify one of the following numbers for the configuration. 0.000: FREE 1.000: LEFTY 2.000: RIGHTY

Transmission example

- To write a value of “3” to the global variable “X” of integer type and execute the RESTORE command, use the MW command as follows.

  STX MW, 1␣X␣0␣3 CR ETX

- To write a position of “(10,20,30,0,0,FREE)” to the global variable “P” of coordinate type and not execute the RESTORE command, use the MW command as follows.

  STX MW, 0␣P␣4␣10␣20␣30␣0␣0␣FREE CR ETX

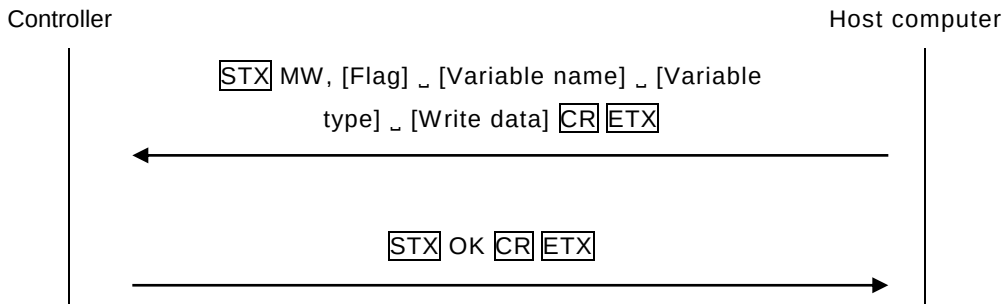
- To write load data of “(2.0,10)” to the global variable “L(1)” of array load type and execute the RESTORE command, use the MW command as follows.

  STX MW, 1␣L(1) ␣7␣2.0␣10 CR ETX

Response format

STX OK CR ETX

Communication example



Note

* The RESTORE command is used to overwrite a SCOL program. Do not turn off the controller power in execution of the RESTORE command, or the program file may be damaged.

PR Individual current position acquisition

Function

The PR command is used to transfer the current position information for the specified coordinate system to the host computer.

Transmission format

STX PR, [Coordinate system] CR ETX

NO.	Name	Size (byte)	Description
1	Coordinate system	1 Fixed length	Specify one of the following numbers for the coordinate system. 0: Joint coordinate 1: World coordinate 2: Work coordinate 3: Joint feedback coordinate 4: World feedback coordinate 5: Work feedback coordinate

Transmission example

STX PR,1 CR ETX

Response format

STX FL, Individual current position information (1/1) EOF ETX

Individual current position information contains the following.

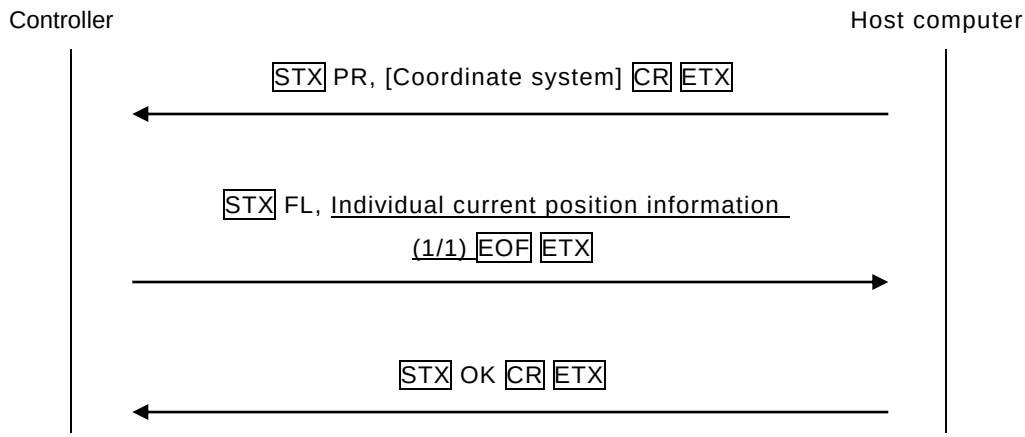
[The axis 1 (X) coordinate of the current position] ␣ [The axis 2 (Y) coordinate of the current position] ␣ [The axis 3 (Z) coordinate of the current position] ␣ [The axis 4 (C) coordinate of the current position] ␣ [The axis 5 (T) coordinate of the current position] ␣ [The axis 6 coordinate of the current position] ␣ [Configuration]

NO.	Name	Size (byte)	Description
1 to 6	Current position coordinate value	9 Variable length	Current position coordinate value of axis 1 to axis 6. This value is sent as a real number to three decimal points. * The response to the PR command varies depending on the requested coordinate system. 0: Joint coordinate 1: World coordinate 2: Work coordinate 3: Joint feedback coordinate 4: World feedback coordinate 5: Work feedback coordinate
7	Configuration	1 Fixed length	Specify one of the following numbers for the configuration. 0: FREE 1: LEFTY 2: RIGHTY

Response example

STX FL , ␣␣500.000 , ␣␣␣␣0.000 , ␣␣␣␣0.000 , ␣␣␣␣0.000 , ␣␣␣␣0.000 , ␣␣␣␣0.000 , 2EOF ETX

Communication example



PS Current position information acquisition

Function

The PS command is used to transfer current position information (motion status, execution line number, current position) to the host computer.

Transmission format

STX PS CR ETX

Response format

STX FL, Current position information (1/n) ETX
STX Current position information (2/n) ETX
:
STX Current position information (n/n) EOF ETX

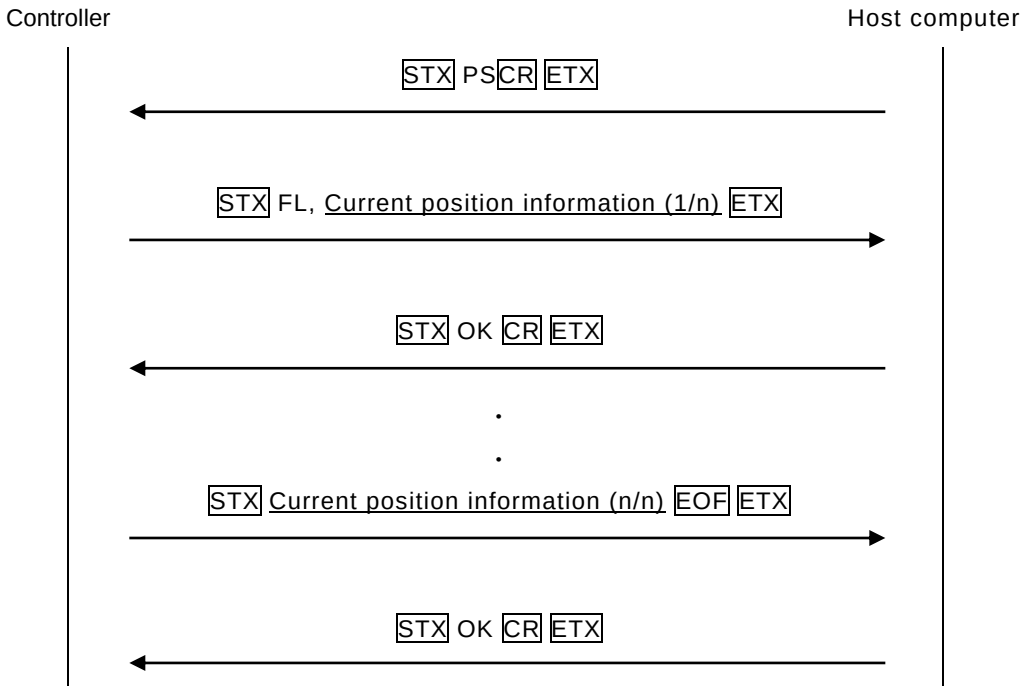
Current position information contains the following.
[Motion status] ␣ [Program execution line] ␣ [Axis 1 current position (integer number)] ␣ [Axis 2 current position (integer number)] ␣ [Axis 3 current position (integer number)] ␣ [Axis 4 current position (integer number)] ␣ [Axis 5 current position (integer number)] ␣ [Axis 6 current position (integer number)] ␣ [Axis 1 current position (real number)] ␣ [Axis 2 current position (real number)] ␣ [Axis 3 current position (real number)] ␣ [Axis 4 current position (real number)] ␣ [Axis 5 current position (real number)] ␣ [Axis 6 current position (real number)] ␣ [Axis 1 motor torque current value (real number)] ␣ [Axis 2 motor torque current value (real number)] ␣ [Axis 3 motor torque current value (real number)] ␣ [Axis 4 motor torque current value (real number)] ␣ [Axis 5 motor torque current value (real number)] ␣ [Axis 6 motor torque current value (real number)]

NO.	Name	Size (byte)	Description
1	Motion status	1 Fixed length	0: STOP(RESET) 1: RUN 2: STOP(RETRY) 3: STOP(CONTINUE)
2	Program execution line	Variable length	Line number during program execution
3 to 8	Current position (integer number)	Variable length	Joint coordinate value of axis 1 to axis 6 However, the decimal point is deleted and only integer part is sent.
9 to 14	Current position (real number)	Variable length	Joint coordinate value of axis 1 to axis 6 The value is sent as a real number to three decimal places.
15 to 20	Motor torque current value (real number) [%]	Variable length	Motor torque current value of axis 1 to axis 6 The value is sent as a real number to one decimal place.

Response example

STX FL, 1 ␣ 0 ␣ -18 ␣ 88 ␣ 67 ␣ -70 ␣ 0 ␣ 0 ␣ -17.731 ␣ 87.977 ␣ 66.745 ␣ -70.246 ␣ 0.000 ␣ 0.000 ␣ 0.0 ␣ 0.0 ␣ 0.0 ␣ 0.0 ␣ 0.0 ␣ 0.0 EOF ETX

Communication example



Note

* When there is much current position information, it may not be able to be all received by one communication. Then, to receive the rest of the current position information, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the current position information.

* The host computer's receipt of EOF ensures that all the current position information is received.

RM Maintenance information reset

Function

The RM command is used to direct the controller to reset the specified maintenance information from the host computer.

Transmission format

STX RM, [Reset number] CR ETX

NO.	Name	Size (byte)	Description
1	Reset number	Variable length	Specify one of the following numbers for the reset number. 1: Total rotational amount of motor (rev) 2: Movement amount of joint axis (mm or deg) 3: Integrated amount of motor torque (%) 4: Integrated amount of motor torque exceeding (%) 5: Peak torque (%) 6: 3-second load factor (%) 7: 1-minute load factor (%) 8: 15-minute load factor (%) 9: 2-hour load factor (%) 10: Total time 11: Energizing time 12: Power OFF time 13: Motion time 14: Fault time 15: Servo information (servo OFF count) 16: Total servo ON time

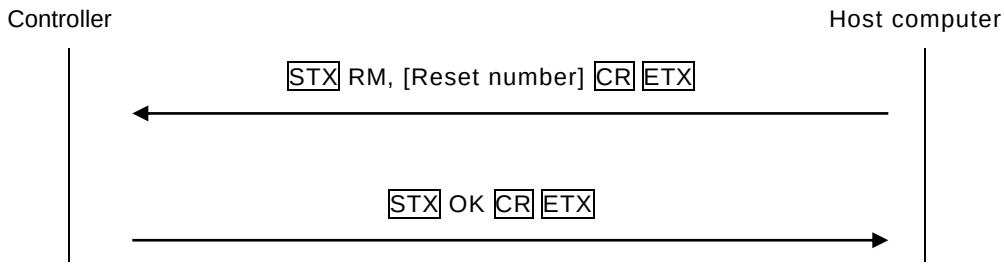
Transmission example

STX RM, 1 CR ETX

Response format

STX OK CR ETX

Communication example



RN Automatic Operation Start

Function

The RN (Automatic Operation Start) command is a command given by the host computer to the controller telling the controller to start up the program. If, after stopping the program with the SP (Automatic Operation Stop) command, one sends the RN command again, the robot will start up from the step immediately following the step at which it was stopped.

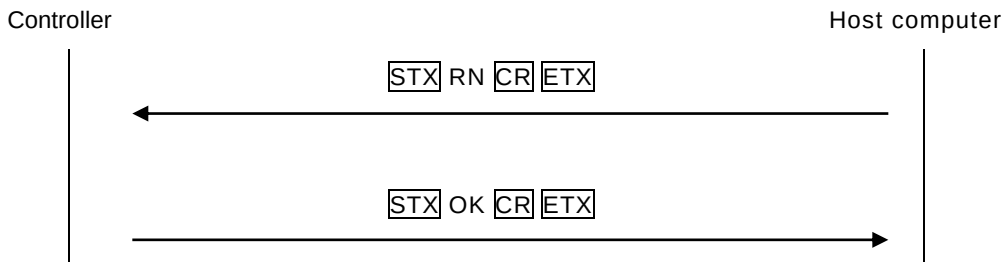
Transmission format

STX RN CR ETX

Response format

STX OK CR ETX

Communication example



Note

* Be sure to keep your hand close to the emergency stop switch while using the RN command so that you can press the emergency stop switch at any time.

RS Reset

Function

The RS command is used to direct the controller to reset a reset target, such as a program, from the host computer.

Transmission format

STX RS, [Reset target] CR ETX

NO.	Name	Size (byte)	Description
1	Reset target	3 Fixed length	Specify one of the following character strings for the reset target. PRG: PRG will reset the program to Step 1. All program data will be initialized. STP: STP will reset the program to Step 1. All program data (such as variables) will remain unchanged. CYC: CYC will specify step number to the specified label (RCYCLE). All program data will remain unchanged. SIG: SIG will put the digital outputs in the non-active state (OFF). SEL: SEL will reset the execution file. ERR: ERR will reset an alarm.

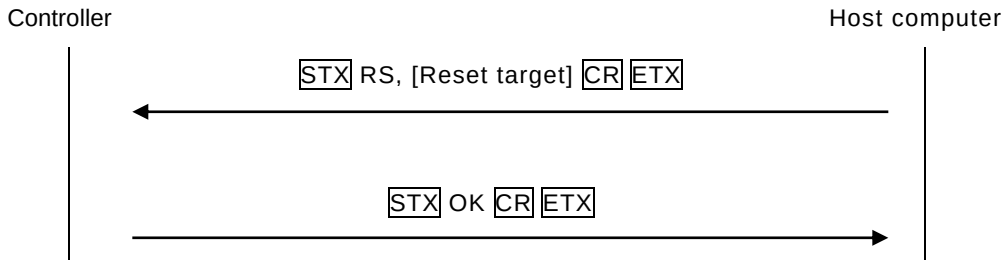
Transmission example

STX RS, PRG CR ETX

Response format

STX OK CR ETX

Communication example



Note

* The RS command will not be accepted unless the system is in a stop mode.

RT Guide rate setting

Function

The RT command is used to set the guide rate of the controller from the host computer.

Transmission format

STX RT, [Guide rate] CR ETX

NO.	Name	Size (byte)	Description
1	Guide rate	1 Fixed length	Specify one of the following numbers for the guide rate. 0: Slow speed 1: Mid speed 2: Fast speed

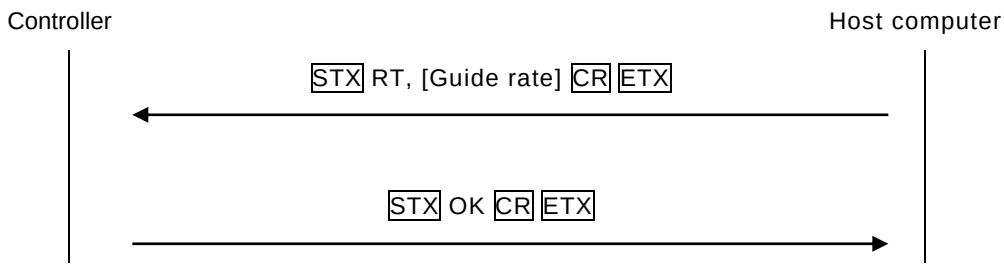
Transmission example

STX RT, 1 CR ETX

Response format

STX OK CR ETX

Communication example



SC Guidance coordinate setting

Function

The SC command is used to set the guidance coordinate of the controller from the host computer.

Transmission format

STX SC, [Guidance coordinate] CR ETX

NO.	Name	Size (byte)	Description
1	Guidance coordinate	1 Fixed length	Specify one of the following numbers for the guidance coordinate. 0: Joint coordinate 1: Tool coordinate 2: Work coordinate 3: World coordinate

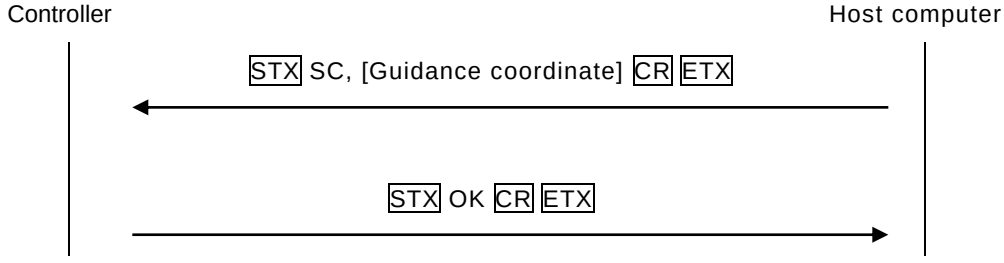
Transmission example

STX SC, 3 CR ETX

Response format

STX OK CR ETX

Communication example



SF System total status request

Function

The SF command is used to send the internal status (state) of the controller to the host computer.

Transmission format

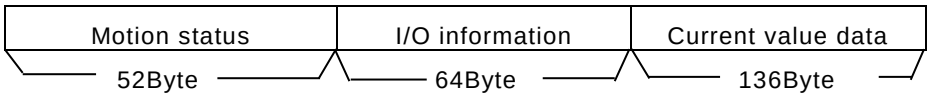
STX SF CR ETX

Response format

STX FL, System total status information (1/1) ETX

System total status information contains the following.
[Motion status] [I/O information] [Current value data]

The following information is transmitted as the system total status information in the binary notation (252 bytes).



Detailed data of each group are tabled below.

I) Motion status

NO.	Name	Size (byte)	Description
1	Servo power status	1	0: OFF, 1: ON
2	EMERGENCY stop switch status	1	0: OFF, 1: ON
3	Motion status	1	0: STOP(RESET) 1: RUN 2: STOP(RETRY) 3: STOP(CONT)
4	SU command request	1	0: Without request, 1: With request
5	Current alarm information	2×10 pcs.	Error of level 8: 1 to 367 Error of level 4: 368 to 511 Error of level 2: 512 to 735 Error of level 1: 736 to 895 Other than above: No error
6	Program execution line	2	Line number during program execution
7	Program analysis line	2	Line number during program analysis
8	Program execution task	2	Task number during program execution
9	Program analysis task	2	Task number during program analysis
10	Feed hold status	2	0: OFF, 1: ON
11	Guidance coordinate system status	2	0: Joint, 1: Tool, 2: Work, 3: World
12	Guide rate status	2	0: Slow, 1: Mid, 2: Fast
13	Guide mode status	2	0: Jog, 1: Inching, 2: Free
14	Master mode status	2	0: TEACHING mode, 1: INTERNAL mode 2: EXT (SIG) mode, 3: EXT (RS232C/ETHER) mode
15	Dummy	2	Reserved
16	Power ON time	4	Unit: Min.
17	Program run time	4	Unit: Min.
Total		52 Fixed length	

II) I/O INFORMATION

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>1</td><td>General input 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din1</td><td>General input</td></tr>
<tr><td>1</td><td>Din2</td><td>General input</td></tr>
<tr><td>2</td><td>Din3</td><td>General input</td></tr>
<tr><td>3</td><td>Din4</td><td>General input</td></tr>
<tr><td>4</td><td>Din5</td><td>General input</td></tr>
<tr><td>5</td><td>Din6</td><td>General input</td></tr>
<tr><td>6</td><td>Din7</td><td>General input</td></tr>
<tr><td>7</td><td>Din8</td><td>General input</td></tr>
<tr><td>8</td><td>Din9</td><td>General input</td></tr>
<tr><td>9</td><td>Din10</td><td>General input</td></tr>
<tr><td>10</td><td>Din11</td><td>General input</td></tr>
<tr><td>11</td><td>Din12</td><td>General input</td></tr>
<tr><td>12</td><td>Din13</td><td>General input</td></tr>
<tr><td>13</td><td>Din14</td><td>General input</td></tr>
<tr><td>14</td><td>Din15</td><td>General input</td></tr>
<tr><td>15</td><td>Din16</td><td>General input</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>General input 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din17</td><td>General input</td></tr>
<tr><td>1</td><td>Din18</td><td>General input</td></tr>
<tr><td>2</td><td>Din19</td><td>General input</td></tr>
<tr><td>3</td><td>Din20</td><td>General input</td></tr>
<tr><td>4</td><td>Din21</td><td>General input</td></tr>
<tr><td>5</td><td>Din22</td><td>General input</td></tr>
<tr><td>6</td><td>Din23</td><td>General input</td></tr>
<tr><td>7</td><td>Din24</td><td>General input</td></tr>
<tr><td>8</td><td>Din25</td><td>General input</td></tr>
<tr><td>9</td><td>Din26</td><td>General input</td></tr>
<tr><td>10</td><td>Din27</td><td>General input</td></tr>
<tr><td>11</td><td>Din28</td><td>General input</td></tr>
<tr><td>12</td><td>Din29</td><td>General input</td></tr>
<tr><td>13</td><td>Din30</td><td>General input</td></tr>
<tr><td>14</td><td>Din31</td><td>General input</td></tr>
<tr><td>15</td><td>Din32</td><td>General input</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>3</td><td>General input 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din33</td><td>General input</td></tr>
<tr><td>1</td><td>Din34</td><td>General input</td></tr>
<tr><td>2</td><td>Din35</td><td>General input</td></tr>
<tr><td>3</td><td>Din36</td><td>General input</td></tr>
<tr><td>4</td><td>Din37</td><td>General input</td></tr>
<tr><td>5</td><td>Din38</td><td>General input</td></tr>
<tr><td>6</td><td>Din39</td><td>General input</td></tr>
<tr><td>7</td><td>Din40</td><td>General input</td></tr>
<tr><td>8</td><td>Din41</td><td>General input</td></tr>
<tr><td>9</td><td>Din42</td><td>General input</td></tr>
<tr><td>10</td><td>Din43</td><td>General input</td></tr>
<tr><td>11</td><td>Din44</td><td>General input</td></tr>
<tr><td>12</td><td>Din45</td><td>General input</td></tr>
<tr><td>13</td><td>Din46</td><td>General input</td></tr>
<tr><td>14</td><td>Din47</td><td>General input</td></tr>
<tr><td>15</td><td>Din48</td><td>General input</td></tr>
</table>
</td></tr>
<tr><td>4</td><td>General input 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din49</td><td>General input</td></tr>
<tr><td>1</td><td>Din50</td><td>General input</td></tr>
<tr><td>2</td><td>Din51</td><td>General input</td></tr>
<tr><td>3</td><td>Din52</td><td>General input</td></tr>
<tr><td>4</td><td>Din53</td><td>General input</td></tr>
<tr><td>5</td><td>Din54</td><td>General input</td></tr>
<tr><td>6</td><td>Din55</td><td>General input</td></tr>
<tr><td>7</td><td>Din56</td><td>General input</td></tr>
<tr><td>8</td><td>Din57</td><td>General input</td></tr>
<tr><td>9</td><td>Din58</td><td>General input</td></tr>
<tr><td>10</td><td>Din59</td><td>General input</td></tr>
<tr><td>11</td><td>Din60</td><td>General input</td></tr>
<tr><td>12</td><td>Din61</td><td>General input</td></tr>
<tr><td>13</td><td>Din62</td><td>General input</td></tr>
<tr><td>14</td><td>Din63</td><td>General input</td></tr>
<tr><td>15</td><td>Din64</td><td>General input</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>5</td><td>Extension input 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din101</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din102</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din103</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din104</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din105</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din106</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din107</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din108</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din109</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din110</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din111</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din112</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din113</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din114</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din115</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din116</td><td>Extension input</td></tr>
</table>
</td></tr>
<tr><td>6</td><td>Extension input 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din117</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din118</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din119</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din120</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din121</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din122</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din123</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din124</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din125</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din126</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din127</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din128</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din129</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din130</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din131</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din132</td><td>Extension input</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>7</td><td>Extension input 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din133</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din134</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din135</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din136</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din137</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din138</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din139</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din140</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din141</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din142</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din143</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din144</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din145</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din146</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din147</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din148</td><td>Extension input</td></tr>
</table>
</td></tr>
<tr><td>8</td><td>Extension input 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din149</td><td>Extension input</td></tr>
<tr><td>1</td><td>Din150</td><td>Extension input</td></tr>
<tr><td>2</td><td>Din151</td><td>Extension input</td></tr>
<tr><td>3</td><td>Din152</td><td>Extension input</td></tr>
<tr><td>4</td><td>Din153</td><td>Extension input</td></tr>
<tr><td>5</td><td>Din154</td><td>Extension input</td></tr>
<tr><td>6</td><td>Din155</td><td>Extension input</td></tr>
<tr><td>7</td><td>Din156</td><td>Extension input</td></tr>
<tr><td>8</td><td>Din157</td><td>Extension input</td></tr>
<tr><td>9</td><td>Din158</td><td>Extension input</td></tr>
<tr><td>10</td><td>Din159</td><td>Extension input</td></tr>
<tr><td>11</td><td>Din160</td><td>Extension input</td></tr>
<tr><td>12</td><td>Din161</td><td>Extension input</td></tr>
<tr><td>13</td><td>Din162</td><td>Extension input</td></tr>
<tr><td>14</td><td>Din163</td><td>Extension input</td></tr>
<tr><td>15</td><td>Din164</td><td>Extension input</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>9</td><td>System input 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din201</td><td>Hand input</td></tr>
<tr><td>1</td><td>Din202</td><td>Hand input</td></tr>
<tr><td>2</td><td>Din203</td><td>Hand input</td></tr>
<tr><td>3</td><td>Din204</td><td>Hand input</td></tr>
<tr><td>4</td><td>Din205</td><td>Hand input</td></tr>
<tr><td>5</td><td>Din206</td><td>Hand input</td></tr>
<tr><td>6</td><td>Din207</td><td>Hand input</td></tr>
<tr><td>7</td><td>Din208</td><td>Hand input</td></tr>
<tr><td>8</td><td>Din209</td><td></td></tr>
<tr><td>9</td><td>Din210</td><td></td></tr>
<tr><td>10</td><td>Din211</td><td></td></tr>
<tr><td>11</td><td>Din212</td><td></td></tr>
<tr><td>12</td><td>Din213</td><td></td></tr>
<tr><td>13</td><td>Din214</td><td></td></tr>
<tr><td>14</td><td>Din215</td><td></td></tr>
<tr><td>15</td><td>Din216</td><td></td></tr>
</table>
</td></tr>
<tr><td>10</td><td>System input 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din217</td><td>Alarm of level 8</td></tr>
<tr><td>1</td><td>Din218</td><td>Alarm of level 8</td></tr>
<tr><td>2</td><td>Din219</td><td>Alarm of level 8</td></tr>
<tr><td>3</td><td>Din220</td><td>Alarm of level 8</td></tr>
<tr><td>4</td><td>Din221</td><td>Alarm of level 4</td></tr>
<tr><td>5</td><td>Din222</td><td>Alarm of level 4</td></tr>
<tr><td>6</td><td>Din223</td><td>Alarm of level 4</td></tr>
<tr><td>7</td><td>Din224</td><td>Alarm of level 4</td></tr>
<tr><td>8</td><td>Din225</td><td>Alarm of level 2</td></tr>
<tr><td>9</td><td>Din226</td><td>Alarm of level 2</td></tr>
<tr><td>10</td><td>Din227</td><td>Alarm of level 2</td></tr>
<tr><td>11</td><td>Din228</td><td>Alarm of level 2</td></tr>
<tr><td>12</td><td>Din229</td><td>Alarm of level 1</td></tr>
<tr><td>13</td><td>Din230</td><td>Alarm of level 1</td></tr>
<tr><td>14</td><td>Din231</td><td>Alarm of level 1</td></tr>
<tr><td>15</td><td>Din232</td><td>Alarm of level 1</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>11</td><td>System input 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din233</td><td></td></tr>
<tr><td>1</td><td>Din234</td><td></td></tr>
<tr><td>2</td><td>Din235</td><td></td></tr>
<tr><td>3</td><td>Din236</td><td></td></tr>
<tr><td>4</td><td>Din237</td><td></td></tr>
<tr><td>5</td><td>Din238</td><td></td></tr>
<tr><td>6</td><td>Din239</td><td></td></tr>
<tr><td>7</td><td>Din240</td><td></td></tr>
<tr><td>8</td><td>Din241</td><td></td></tr>
<tr><td>9</td><td>Din242</td><td></td></tr>
<tr><td>10</td><td>Din243</td><td></td></tr>
<tr><td>11</td><td>Din244</td><td></td></tr>
<tr><td>12</td><td>Din245</td><td></td></tr>
<tr><td>13</td><td>Din246</td><td></td></tr>
<tr><td>14</td><td>Din247</td><td></td></tr>
<tr><td>15</td><td>Din248</td><td></td></tr>
</table>
</td></tr>
<tr><td>12</td><td>System input 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Din249</td><td>STROBE</td></tr>
<tr><td>1</td><td>Din250</td><td>PRG_RST</td></tr>
<tr><td>2</td><td>Din251</td><td>STEP_RST</td></tr>
<tr><td>3</td><td>Din252</td><td>CYC_RST</td></tr>
<tr><td>4</td><td>Din253</td><td>DO_RST</td></tr>
<tr><td>5</td><td>Din254</td><td>ALM_RST</td></tr>
<tr><td>6</td><td>Din255</td><td>RUN</td></tr>
<tr><td>7</td><td>Din256</td><td>EX_SVON</td></tr>
<tr><td>8</td><td>Din257</td><td>STOP</td></tr>
<tr><td>9</td><td>Din258</td><td>CYCLE</td></tr>
<tr><td>10</td><td>Din259</td><td>LOW_SPD</td></tr>
<tr><td>11</td><td>Din260</td><td>BREAK</td></tr>
<tr><td>12</td><td>Din261</td><td>SVOFF</td></tr>
<tr><td>13</td><td>Din262</td><td></td></tr>
<tr><td>14</td><td>Din263</td><td></td></tr>
<tr><td>15</td><td>Din264</td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>13</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>14</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>15</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>16</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>17</td><td>General output 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout1</td><td>General output</td></tr>
<tr><td>1</td><td>Dout2</td><td>General output</td></tr>
<tr><td>2</td><td>Dout3</td><td>General output</td></tr>
<tr><td>3</td><td>Dout4</td><td>General output</td></tr>
<tr><td>4</td><td>Dout5</td><td>General output</td></tr>
<tr><td>5</td><td>Dout6</td><td>General output</td></tr>
<tr><td>6</td><td>Dout7</td><td>General output</td></tr>
<tr><td>7</td><td>Dout8</td><td>General output</td></tr>
<tr><td>8</td><td>Dout9</td><td>General output</td></tr>
<tr><td>9</td><td>Dout10</td><td>General output</td></tr>
<tr><td>10</td><td>Dout11</td><td>General output</td></tr>
<tr><td>11</td><td>Dout12</td><td>General output</td></tr>
<tr><td>12</td><td>Dout13</td><td>General output</td></tr>
<tr><td>13</td><td>Dout14</td><td>General output</td></tr>
<tr><td>14</td><td>Dout15</td><td>General output</td></tr>
<tr><td>15</td><td>Dout16</td><td>General output</td></tr>
</table>
</td></tr>
<tr><td>18</td><td>General output 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout17</td><td>General output</td></tr>
<tr><td>1</td><td>Dout18</td><td>General output</td></tr>
<tr><td>2</td><td>Dout19</td><td>General output</td></tr>
<tr><td>3</td><td>Dout20</td><td>General output</td></tr>
<tr><td>4</td><td>Dout21</td><td>General output</td></tr>
<tr><td>5</td><td>Dout22</td><td>General output</td></tr>
<tr><td>6</td><td>Dout23</td><td>General output</td></tr>
<tr><td>7</td><td>Dout24</td><td>General output</td></tr>
<tr><td>8</td><td>Dout25</td><td>General output</td></tr>
<tr><td>9</td><td>Dout26</td><td>General output</td></tr>
<tr><td>10</td><td>Dout27</td><td>General output</td></tr>
<tr><td>11</td><td>Dout28</td><td>General output</td></tr>
<tr><td>12</td><td>Dout29</td><td>General output</td></tr>
<tr><td>13</td><td>Dout30</td><td>General output</td></tr>
<tr><td>14</td><td>Dout31</td><td>General output</td></tr>
<tr><td>15</td><td>Dout32</td><td>General output</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>19</td><td>General output 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout33</td><td>General output</td></tr>
<tr><td>1</td><td>Dout34</td><td>General output</td></tr>
<tr><td>2</td><td>Dout35</td><td>General output</td></tr>
<tr><td>3</td><td>Dout36</td><td>General output</td></tr>
<tr><td>4</td><td>Dout37</td><td>General output</td></tr>
<tr><td>5</td><td>Dout38</td><td>General output</td></tr>
<tr><td>6</td><td>Dout39</td><td>General output</td></tr>
<tr><td>7</td><td>Dout40</td><td>General output</td></tr>
<tr><td>8</td><td>Dout41</td><td>General output</td></tr>
<tr><td>9</td><td>Dout42</td><td>General output</td></tr>
<tr><td>10</td><td>Dout43</td><td>General output</td></tr>
<tr><td>11</td><td>Dout44</td><td>General output</td></tr>
<tr><td>12</td><td>Dout45</td><td>General output</td></tr>
<tr><td>13</td><td>Dout46</td><td>General output</td></tr>
<tr><td>14</td><td>Dout47</td><td>General output</td></tr>
<tr><td>15</td><td>Dout48</td><td>General output</td></tr>
</table>
</td></tr>
<tr><td>20</td><td>General output 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout49</td><td>General output</td></tr>
<tr><td>1</td><td>Dout50</td><td>General output</td></tr>
<tr><td>2</td><td>Dout51</td><td>General output</td></tr>
<tr><td>3</td><td>Dout52</td><td>General output</td></tr>
<tr><td>4</td><td>Dout53</td><td>General output</td></tr>
<tr><td>5</td><td>Dout54</td><td>General output</td></tr>
<tr><td>6</td><td>Dout55</td><td>General output</td></tr>
<tr><td>7</td><td>Dout56</td><td>General output</td></tr>
<tr><td>8</td><td>Dout57</td><td>General output</td></tr>
<tr><td>9</td><td>Dout58</td><td>General output</td></tr>
<tr><td>10</td><td>Dout59</td><td>General output</td></tr>
<tr><td>11</td><td>Dout60</td><td>General output</td></tr>
<tr><td>12</td><td>Dout61</td><td>General output</td></tr>
<tr><td>13</td><td>Dout62</td><td>General output</td></tr>
<tr><td>14</td><td>Dout63</td><td>General output</td></tr>
<tr><td>15</td><td>Dout64</td><td>General output</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>21</td><td>Extension output 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout101</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout102</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout103</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout104</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout105</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout106</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout107</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout108</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout109</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout110</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout111</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout112</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout113</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout114</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout115</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout116</td><td>Extension output</td></tr>
</table>
</td></tr>
<tr><td>22</td><td>Extension output 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout117</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout118</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout119</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout120</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout121</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout122</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout123</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout124</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout125</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout126</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout127</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout128</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout129</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout130</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout131</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout132</td><td>Extension output</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>23</td><td>Extension output 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout133</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout134</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout135</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout136</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout137</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout138</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout139</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout140</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout141</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout142</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout143</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout144</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout145</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout146</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout147</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout148</td><td>Extension output</td></tr>
</table>
</td></tr>
<tr><td>24</td><td>Extension output 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout149</td><td>Extension output</td></tr>
<tr><td>1</td><td>Dout150</td><td>Extension output</td></tr>
<tr><td>2</td><td>Dout151</td><td>Extension output</td></tr>
<tr><td>3</td><td>Dout152</td><td>Extension output</td></tr>
<tr><td>4</td><td>Dout153</td><td>Extension output</td></tr>
<tr><td>5</td><td>Dout154</td><td>Extension output</td></tr>
<tr><td>6</td><td>Dout155</td><td>Extension output</td></tr>
<tr><td>7</td><td>Dout156</td><td>Extension output</td></tr>
<tr><td>8</td><td>Dout157</td><td>Extension output</td></tr>
<tr><td>9</td><td>Dout158</td><td>Extension output</td></tr>
<tr><td>10</td><td>Dout159</td><td>Extension output</td></tr>
<tr><td>11</td><td>Dout160</td><td>Extension output</td></tr>
<tr><td>12</td><td>Dout161</td><td>Extension output</td></tr>
<tr><td>13</td><td>Dout162</td><td>Extension output</td></tr>
<tr><td>14</td><td>Dout163</td><td>Extension output</td></tr>
<tr><td>15</td><td>Dout164</td><td>Extension output</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>25</td><td>System output 1
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout201</td><td>Hand output</td></tr>
<tr><td>1</td><td>Dout202</td><td>Hand output</td></tr>
<tr><td>2</td><td>Dout203</td><td>Hand output</td></tr>
<tr><td>3</td><td>Dout204</td><td>Hand output</td></tr>
<tr><td>4</td><td>Dout205</td><td>Hand output</td></tr>
<tr><td>5</td><td>Dout206</td><td>Hand output</td></tr>
<tr><td>6</td><td>Dout207</td><td>Hand output</td></tr>
<tr><td>7</td><td>Dout208</td><td>Hand output</td></tr>
<tr><td>8</td><td>Dout209</td><td></td></tr>
<tr><td>9</td><td>Dout210</td><td></td></tr>
<tr><td>10</td><td>Dout211</td><td></td></tr>
<tr><td>11</td><td>Dout212</td><td></td></tr>
<tr><td>12</td><td>Dout213</td><td></td></tr>
<tr><td>13</td><td>Dout214</td><td></td></tr>
<tr><td>14</td><td>Dout215</td><td></td></tr>
<tr><td>15</td><td>Dout216</td><td></td></tr>
</table>
</td></tr>
<tr><td>26</td><td>System output 2
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout217</td><td>Seq. parameter</td></tr>
<tr><td>1</td><td>Dout218</td><td>Seq. parameter</td></tr>
<tr><td>2</td><td>Dout219</td><td>Seq. parameter</td></tr>
<tr><td>3</td><td>Dout220</td><td>Seq. parameter</td></tr>
<tr><td>4</td><td>Dout221</td><td>Seq. parameter</td></tr>
<tr><td>5</td><td>Dout222</td><td>Seq. parameter</td></tr>
<tr><td>6</td><td>Dout223</td><td>Seq. parameter</td></tr>
<tr><td>7</td><td>Dout224</td><td>Seq. parameter</td></tr>
<tr><td>8</td><td>Dout225</td><td></td></tr>
<tr><td>9</td><td>Dout226</td><td></td></tr>
<tr><td>10</td><td>Dout227</td><td></td></tr>
<tr><td>11</td><td>Dout228</td><td></td></tr>
<tr><td>12</td><td>Dout229</td><td></td></tr>
<tr><td>13</td><td>Dout230</td><td></td></tr>
<tr><td>14</td><td>Dout231</td><td></td></tr>
<tr><td>15</td><td>Dout232</td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>27</td><td>System output 3
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout233</td><td></td></tr>
<tr><td>1</td><td>Dout234</td><td></td></tr>
<tr><td>2</td><td>Dout235</td><td></td></tr>
<tr><td>3</td><td>Dout236</td><td></td></tr>
<tr><td>4</td><td>Dout237</td><td></td></tr>
<tr><td>5</td><td>Dout238</td><td></td></tr>
<tr><td>6</td><td>Dout239</td><td></td></tr>
<tr><td>7</td><td>Dout240</td><td></td></tr>
<tr><td>8</td><td>Dout241</td><td></td></tr>
<tr><td>9</td><td>Dout242</td><td></td></tr>
<tr><td>10</td><td>Dout243</td><td></td></tr>
<tr><td>11</td><td>Dout244</td><td></td></tr>
<tr><td>12</td><td>Dout245</td><td></td></tr>
<tr><td>13</td><td>Dout246</td><td></td></tr>
<tr><td>14</td><td>Dout247</td><td></td></tr>
<tr><td>15</td><td>Dout248</td><td></td></tr>
</table>
</td></tr>
<tr><td>28</td><td>System output 4
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td>Dout249</td><td>EMG_ST</td></tr>
<tr><td>1</td><td>Dout250</td><td>SV_RDY</td></tr>
<tr><td>2</td><td>Dout251</td><td>ACK</td></tr>
<tr><td>3</td><td>Dout252</td><td>TEACH</td></tr>
<tr><td>4</td><td>Dout253</td><td>INT</td></tr>
<tr><td>5</td><td>Dout254</td><td>EXT_SIG</td></tr>
<tr><td>6</td><td>Dout255</td><td>EXT_232C</td></tr>
<tr><td>7</td><td>Dout256</td><td>SYS_RDY</td></tr>
<tr><td>8</td><td>Dout257</td><td>AUTORUN</td></tr>
<tr><td>9</td><td>Dout258</td><td>CYC_END</td></tr>
<tr><td>10</td><td>Dout259</td><td>LOW_ST</td></tr>
<tr><td>11</td><td>Dout260</td><td>CYC_ST</td></tr>
<tr><td>12</td><td>Dout261</td><td>BT_ALM</td></tr>
<tr><td>13</td><td>Dout262</td><td>ALARM</td></tr>
<tr><td>14</td><td>Dout263</td><td>EXT_ETHER</td></tr>
<tr><td>15</td><td>Dout264</td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>29</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>30</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>NO.</th><th>Name</th><th>Size (byte)</th><th>Description</th></tr>
<tr><td>31</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>32</td><td>System reservation
(Binary value)
(0000 to FFFF)</td><td>2
Fixed length</td><td>
<table>
<tr><th>Bit</th><th>Signal No.</th><th>Signal name</th></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td></tr>
<tr><td>9</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td></tr>
<tr><td>12</td><td></td><td></td></tr>
<tr><td>13</td><td></td><td></td></tr>
<tr><td>14</td><td></td><td></td></tr>
<tr><td>15</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

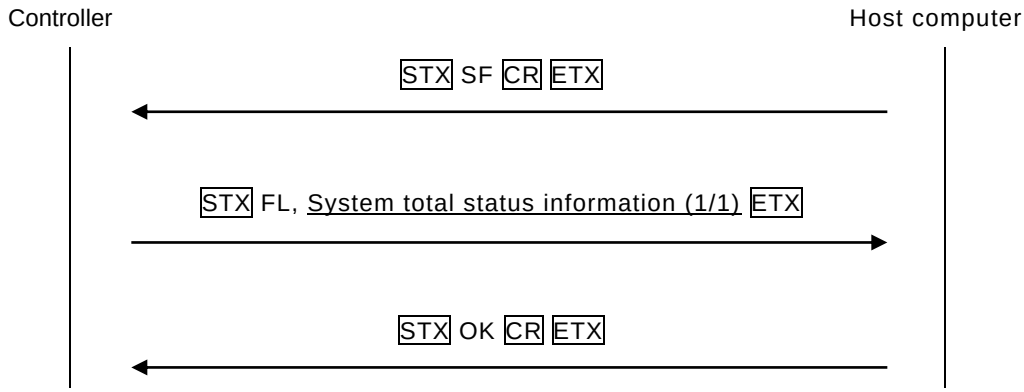
III) Current value data

NO.	Name	Size (byte)	Description
1	Joint coordinate value	4×6 axes	As shown below, values are set in the order of axis 1 to axis 6. Axis 1 (float) Axis 2 (float) Axis 3 (float) Axis 4 (float) Axis 5 (float) Axis 6 (float)
2	World coordinate value	4×6 axes	
3	Work coordinate value	4×6 axes	
4	Work coordinate name	20	Name of work coordinate system
5	Tool coordinate name	20	Name of tool coordinate system
6	Base coordinate name	20	Name of base coordinate system
7	Reserved	4	
Total		136 Fixed length	

Response example

STX FL, 00 00 00 00 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F 3F
3F 00 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 00 00 00 01 03 3F 01
1E 50 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1F
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 42 3F 00 00 00 00 00 00
44 61 00 00 43 34 00 00 00 00 00 00 43 34 00 00 00 00 00 00 00 00 00 00 42 3F 00
00 00 00 00 00 44 61 00 00 43 34 00 00 00 00 00 00 43 34 00 00 00 00 00 00 00 00
00 00 .world 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .system 00 00 00 00 00 00
00 00 00 00 00 00 00 .system 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2F 4F
00 ETX

Communication example



Note

* When there is much system total status information, it may not be able to be all received by one communication. Then, to receive the rest of the system total status information, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the system total status information.

* The host computer's receipt of EOF ensures that all the system total status information is received.

* All responses to the SF command are returned in binary code.

SL Program Selection

Function

The SL command selects a program to be executed under automatic operation.

Transmission format

STX SL, [File name] CR ETX

NO.	Name	Size (byte)	Description
1	File name	Variable length	The name of a program file to be executed The file name needs to be defined in either of the following formats. • Name.Extension • Name The file name must be one to eight characters. The extension must be zero to three characters. When a file name is defined without extension, do not include a period (“.”) in the file name.

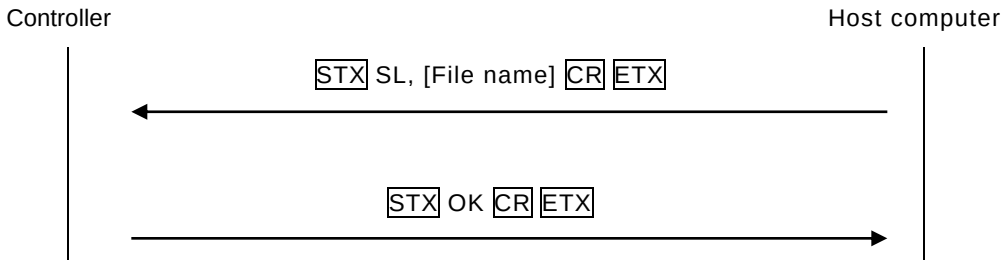
Transmission example

STX SL, PRG1 CR ETX

Response format

STX OK CR ETX

Communication example



SM Motion status request

Function

The SM command is used to transfer the motion status of the controller to the host computer.

Transmission format

STX SM, [Model code] CR ETX

NO.	Name	Size (byte)	Description
1	Model code	1 Fixed length	Specify the following number for the model code. 1: TS3000 series robot controller

Transmission example

STX SM, 1 CR ETX

Response format

STX FL, Motion status information EOF ETX

Motion status information contains the following.
EE [Emergency STOP Event] ␣ SE [Safety SW Event] ␣ SC [Stop command Event] ␣ BC [Break command Event] ␣ ES [Emergency Switch Status] ␣ SS [Safety Switch Status] ␣ SV [Servo Status] ␣ MM [Master Mode Status] ␣ RM [Run Mode Status] ␣ RS [Run Status] ␣ OV [Override] ␣ AL [Alarm] ␣ DC [Do Move Count] ␣ DS [Do Move Status]

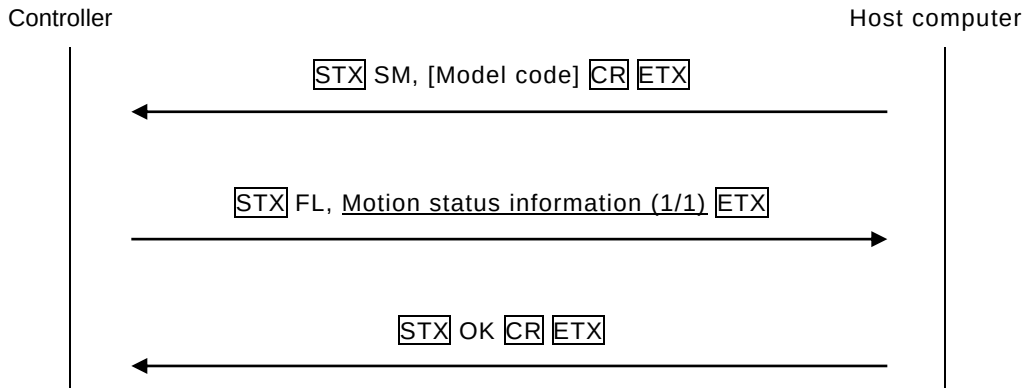
NO.	Name	Size (byte)	Description
1	Emergency STOP Event	1 Fixed length	0: Without event 1: With event
2	Safety SW Event	1 Fixed length	0: Without event 1: With event
3	Stop command Event	1 Fixed length	0: Without event 1: With event
4	Break command Event	1 Fixed length	0: Without event 1: With event
5	Emergency Switch Status	1 Fixed length	0: Safety Switch OFF 1: Safety Switch ON
6	Safety Switch Status	1 Fixed length	0: Safety Switch OFF 1: Safety Switch ON
7	Servo Status	1 Fixed length	0: Servo OFF 1: Servo ON
8	Master Mode Status	1 Fixed length	0: TEACHING 1: INTERNAL 2: EXT.SIG 4: EXT.RS232C 5: EXT.ETHER
9	Run Mode Status	1 Fixed length	0: CONTINUOUS 1: CYCLE 2: STEP 3: SEGMENT
10	Run Status	1 Fixed length	0: STOP(RESET) 1: RUN 2: STOP(RETRY) 3: STOP(CONT)
11	Override	Variable length	0 to 100
12	Alarm	1 Fixed length	0: No alarm 1: Level 1 alarm 2: Level 2 alarm 4: Level 4 alarm 8: Level 8 alarm * Alarm at the highest level among occurring alarms
13	Do Move Count	Variable length	0 to 65536

NO.	Name	Size (byte)	Description
14	Do Move Status	1 Fixed length	0: Motion completion 1: Motion in progress 2: Stop end 3: Break end

Response example

STX FL, EE0 ␣ SE1 ␣ SC1 ␣ BC1 ␣ ES0 ␣ SS0 ␣ SV1 ␣ MM4 ␣ RM0 ␣ RS0 ␣ OV100 ␣ AL0 ␣ DC114 ␣ DS2 EOF ETX

Communication example



Note

* When there is much motion status information, it may not be able to be all received by one communication. Then, to receive the rest of the motion status information, send acknowledge "STX OK CR ETX" for each communication. If STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the motion status information.

* The host computer's receipt of EOF ensures that all the motion status information is received.

SO Servo ON

Function

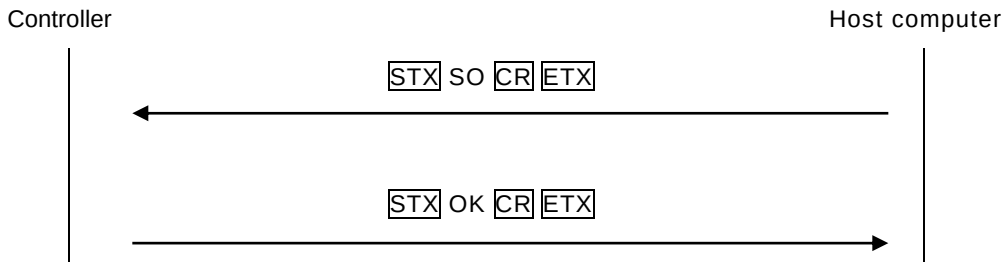
Transmission format

STX SO CR ETX

Response format

STX OK CR ETX

Communication example



Note

* When servo ON and OFF are repeatedly executed, the controller may return NG to prevent the servo from seizing up.
Then, wait a while and try sending the SO command again.

SP Automatic Operation Stop

Function

The SP (Automatic Operation Stop) command is given by the host computer to the controller telling the controller to stop automatic operation.

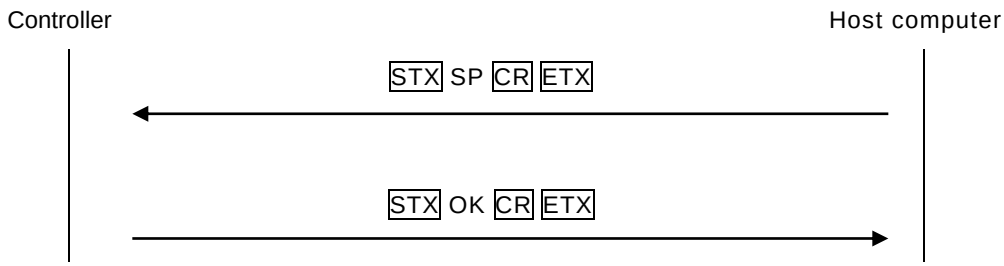
Transmission format

STX SP CR ETX

Response format

STX OK CR ETX

Communication example



SU Status request

Function

The SU command is used to send the internal status of the controller to the host computer.

Transmission format

STX SU CR ETX

Response format

STX FL, Status information (1/n) ETX
STX Status information (2/n) ETX
:
STX Status information (n/n) EOF ETX

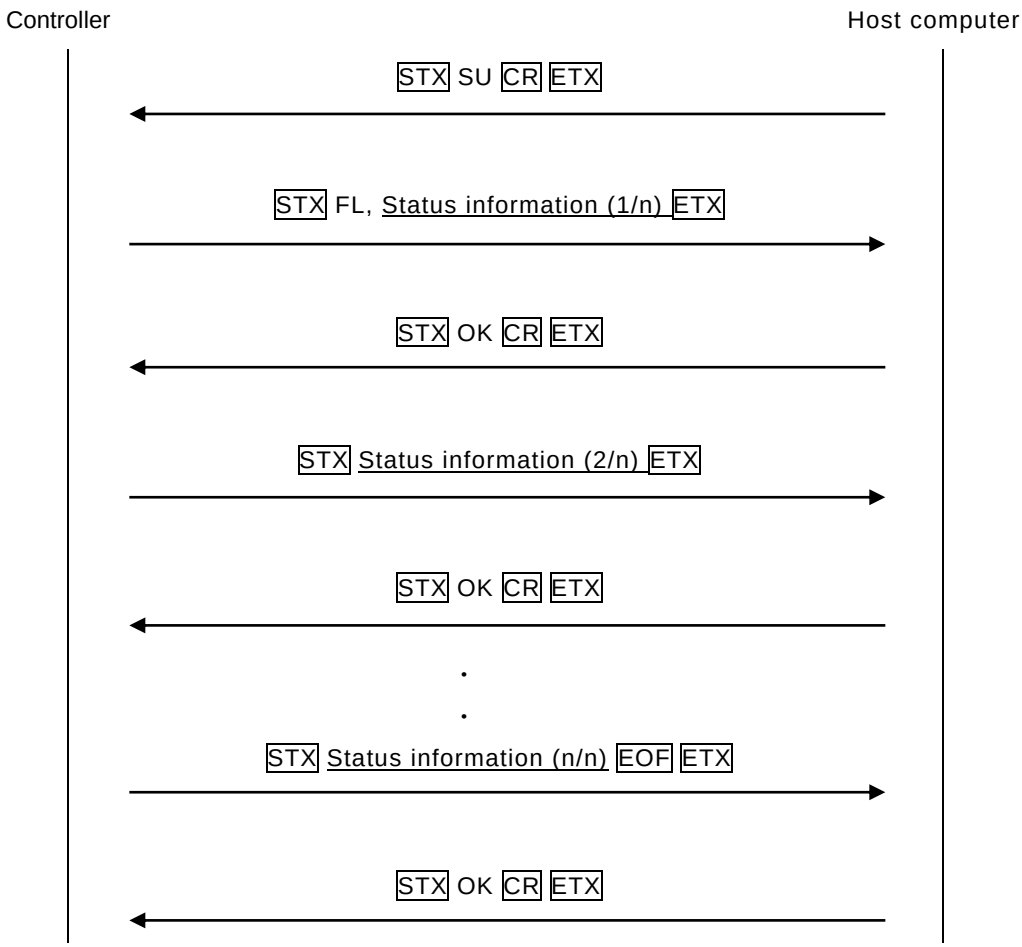
The status information contains the following.
MODE: [Mode] / [Operation mode] ␣ FILE: [File name] ␣ OVRD: [Speed override] %␣ LSPEED: [Speed limit] % ␣ MACHINE: [Machine status] ␣ STATUS: [Execution status]

NO.	Name	Size (byte)	Description
1	Mode	Variable length	Controller mode “external(sig)”: External Automatic Mode (External operation signal mode) “external(rs232C)”: External Automatic Mode (RS232C mode) “external(ethernet)”: External Automatic Mode (ethernet mode) “internal”: Internal Operation Mode “teaching”: Test run mode
2	Operation mode	Variable length	Automatic Operation Modes “step”: Step Operation Mode “continuous”: Continuous Operation Mode “cycle”: Cycle Operation Mode “segment”:
3	File name	Variable length	File name presently selected
3	Speed override	Variable length	Percentage for the speed override.
4	Speed limit	Variable length	Speed limit value Fixed at 100 for compatibility with our older models.
5	Machine status	Variable length	Machine status “free”: Status in which machine lock has been released (Mode in which the robot can move) “lock”: Status in which machine lock is in effect. (Status in which the robot cannot move)
6	Execution status	Variable length	Operation status “running”: In Automatic Operation “stop(reset)”: In Stop (Initialization Mode) (Same status as that for program Reset) “stop(retry)”: In Stop (Retry Mode) (Restart operation from the interrupted movement) “stop(continus)”: In Stop (Continuous Mode) (Continue program from the present step)

Response example

STX FL, MODE: external(RS232C) /continuous ␣ FILE: PRG1 ␣ OVRD: 100% ␣
LSPEED: 100% ␣ MACHINE: free ␣ STATUS: stop(continue) EOF ETX

Communication example



UL File upload request

Function

The UL command is used to upload (transmit) the specified RAM file in the controller to the host computer.
When the request cannot be accepted, STX NG CR ETX is returned.

Transmission format

STX UL, [File name] CR ETX

NO.	Name	Size (byte)	Description
1	File name	Variable length	The name of a file to be uploaded

Transmission example

STX UL, PRG1 CR ETX

Response format

STX FL, File contents (1/n) ETX
STX File contents (2/n) ETX
:
STX File contents (n/n) EOF ETX

File contents contain [File contents].

NO.	Name	Size (byte)	Description
1	File contents	Variable length	The contents of the file to be uploaded to the host computer ASCII code is the only valid character code.

Response example

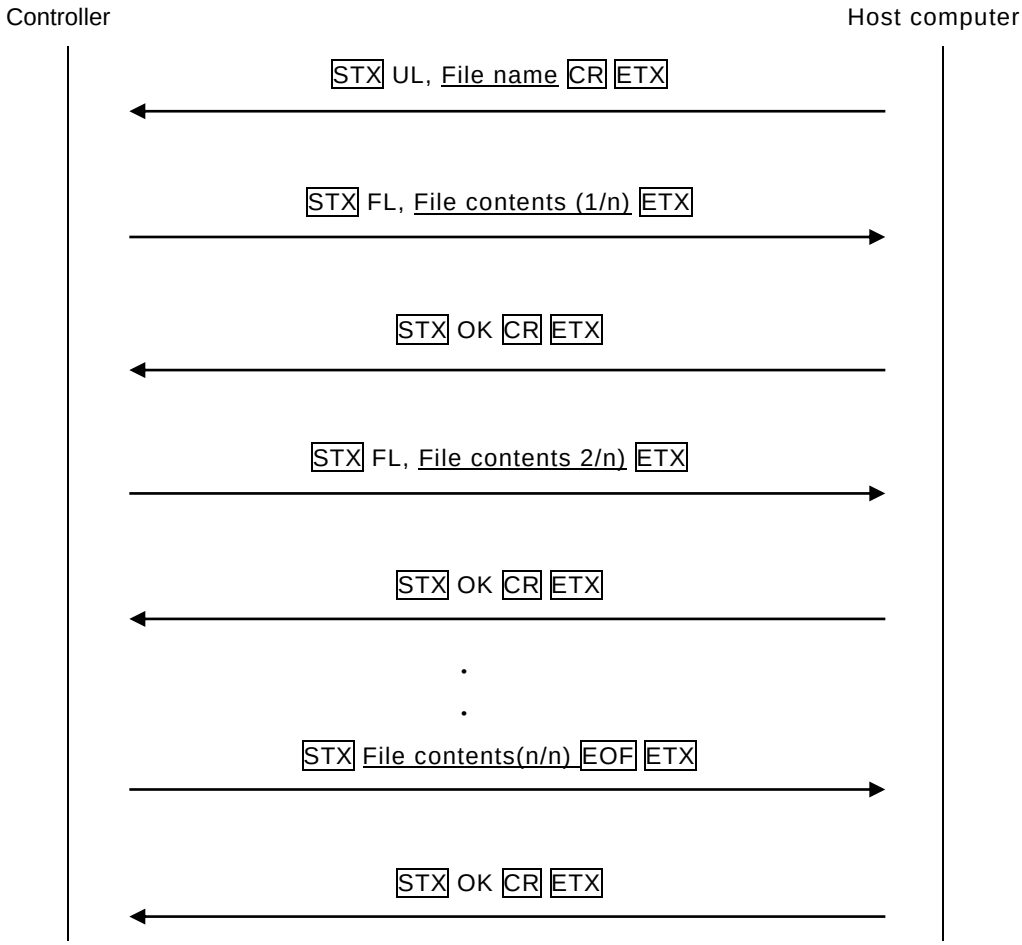
To transfer the file “PRG1” in the controller RAM drive to the host computer, use the UL command as follows.

“PRG1”File contents

```
PROGRAM␣MAIN[CR]
␣SUB1[CR]
[CR]
END[CR]
```

[STX] FL, PROGRAM␣MAIN [CR] ␣SUB1 [CR] [CR] END [CR] [EOF] [ETX]

Communication example



VL Robot tip speed acquisition

Function

The VL command is used to transfer the specified robot tip speed information to the host computer.

Transmission format

STX VL, [Speed type] CR ETX

NO.	Name	Size (byte)	Description
1	Speed type	1 Fixed length	Specify one of the following numbers for the speed type. 0: Robot tip speed (joint) 1: Robot tip speed (world)

Transmission example

STX VL, 0 CR ETX

Response format

STX ZS, Robot tip speed ETX

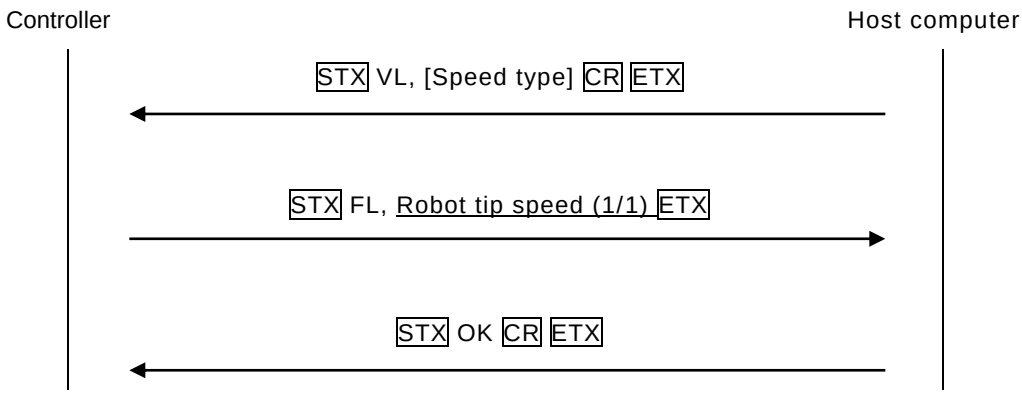
Robot tip speed contains the following.
FL, [Axis 1 (X) tip speed information], [Axis 2 (Y) tip speed information], [Axis 3 (Z) tip speed information], [Axis 4 (C) tip speed information], [Axis 5 (T) tip speed information], [Axis 6 tip speed information] EOF

NO.	Name	Size (byte)	Description
1 to 6	Axis tip speed information	Variable length	Axis 1 to axis 6 information This value is sent as a real number to three decimal points. * The speed type varies depending on the response contents. 0: Robot tip speed (joint) 1: Robot tip speed (world)

Response example

STX FL, -␣␣␣0.907, ␣␣␣␣4.501, ␣␣␣␣3.415, -␣␣␣3.595, ␣␣␣␣0.000, ␣␣␣␣0.000 EOF ETX

Communication example



Note

* When there is much robot tip speed information, it may not be able to be all received by one communication. Then, to receive the rest of the robot tip speed information, send acknowledge "STX OK CR ETX" for each communication. If "STX OK CR ETX" is not sent from the host computer, the controller returns "STX NG CR ETX," not the rest of the robot tip speed information.

* The host computer's receipt of EOF ensures that all the robot tip speed information is received.

VR Version read

Function

The VR command is used to send the system version information to the host computer.

Transmission format

STX VR CR ETX

Response format

STX FL, Version information (1/n) ETX
STX Version information (2/n) ETX
:
STX Version information (n/n) EOF ETX

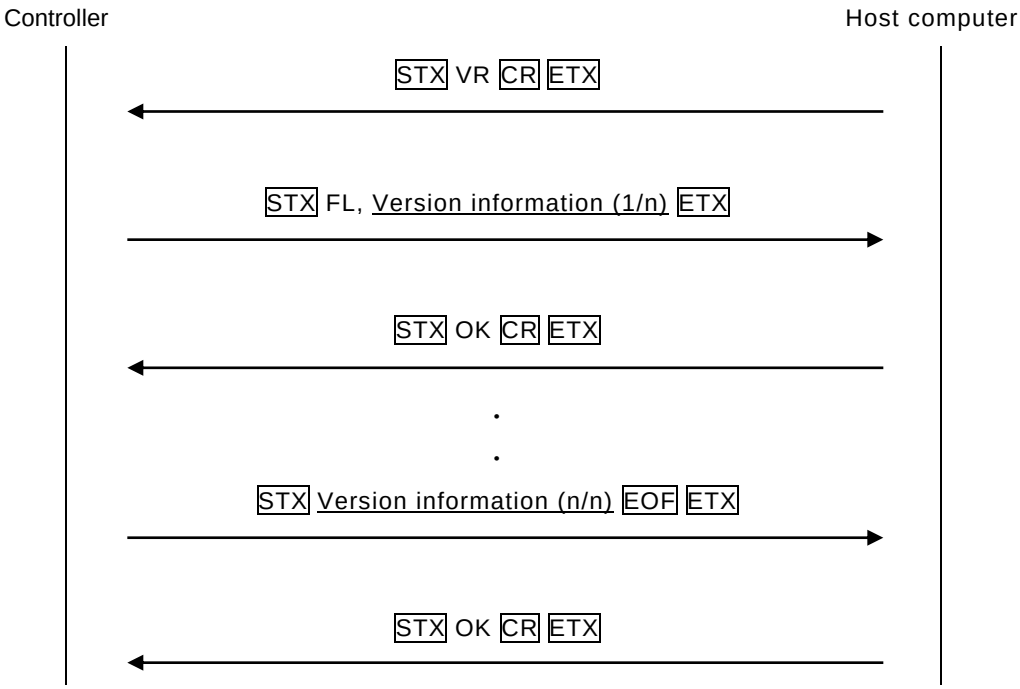
Version information contains the following.
[System name] ␣ [Date of creation] ␣ [Time of creation] ␣ [Checksum] CR

NO.	Name	Size (byte)	Description
1	System name	10 Fixed length	Shows the name of the system.
2	Date of creation	10 Fixed length	Shows the date on which the system was created. “YYYY-MM-DD” YYYY: Year (of the Christian era) MM: Month DD: Day
3	Time of creation	5 Fixed length	Shows the time on which the system was created. “HH: MM: SS” HH : Hours (In 24-hour"military time.") MM : Minutes
4	Checksum	4 Fixed length	Checksum value of the system file

Response example

STX FL,X8LBC-05B ␣ ␣ 2014-12-15 ␣ 08:40 ␣ BAC3 CR X8GCAS15E ␣ ␣
2018-07-27 ␣ 19:26␣ 3A93CR X8YCC-09A ␣ ␣ 2018-04-20 ␣ 17:35 ␣ 0027 CR
X8YCB-14A ␣ ␣ 2017-08-25 ␣ 09:00 ␣ FD58 CR EOF ETX

Communication example



WD Watchdog timer setting

Function

The WD command is used to set the watchdog timer of the controller from the host computer.

* An alarm is generated during the set timer value (msec) when communication is discontinued.

Transmission format

STX WD, [Timer value] ␣ [Alarm level] CR ETX

NO.	Name	Size (byte)	Description
1	Timer value	Variable length	Specify one of the following numbers for the timer value (msec). 0 to 99999 * However, if 0 is specified, it means that watchdog monitor is stopped.
2	Alarm level	1 Fixed length	Specify one of the following numbers for the alarm level. 0: No alarm is generated. 1: Level 1 alarm is generated. (001-200 Host Port Time Out) 2: Level 2 alarm is generated. (002-137 Host Port Time Out) 8: Level 8 alarm is generated. (008-352 Host Port Time Out)

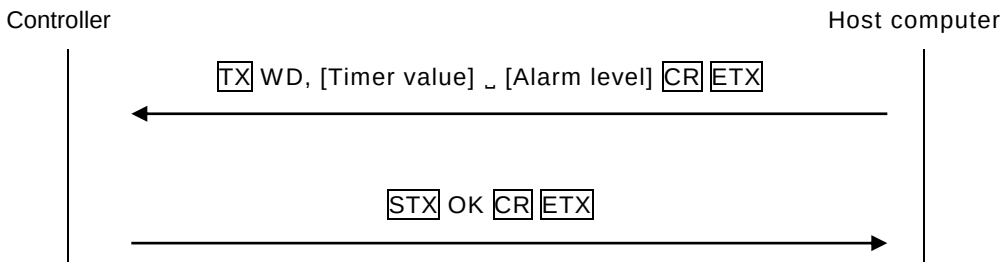
Response format

STX OK CR ETX

Transmission example

STX WD, 1000␣1 CR ETX

Transmission example



ZS Coordinate setting

Function

The ZS command is used to set the work and tool coordinates of the controller from the host computer.

Transmission format

STX ZS, Coordinate information CR ETX

Coordinate information contains the following.
[Coordinate identification] ␣ [Coordinate name] ␣ [X coordinate value] ␣ [Y coordinate value] ␣ [Z coordinate value] ␣ [C coordinate value]

NO.	Name	Size (byte)	Description
1	Coordinate identification	1 Fixed length	Specify one of the following numbers for the coordinate identification. 1: Tool coordinate 2: Work coordinate
2	Coordinate name	Variable length	A coordinate name consists of up to 20 characters.
3 to 6	Coordinate value	Variable length	Specify a real number to three decimal places for the coordinate value to be set.

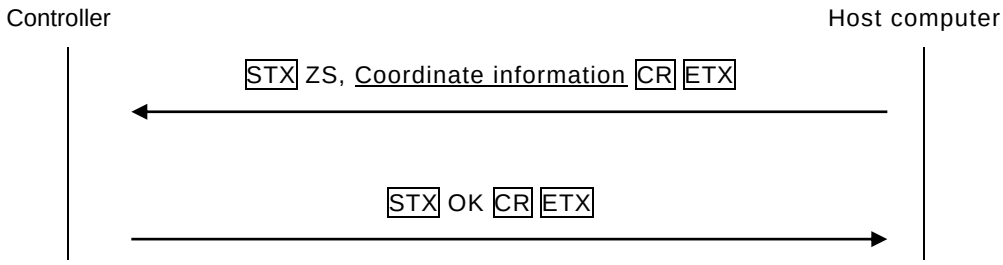
Transmission example

STX ZS, 1␣X␣10.0␣20.0␣0.0␣0.0 CR ETX

Response format

STX OK CR ETX

Communication example



Section 6

Robot Operation Sequence Using External Communication

Operating procedures for the robot in the External Control Mode are basically the same as those in the Internal Control Mode.

6.1 Basic Operation

Figure 6.1 presents a flow sequence in which a certain block is selected and executed repeatedly.

6.2 Program Download

Figure 6.2 presents a flow sequence in which a program is downloaded at the completion of each cycle and it is executed.

6.3 Reinitializing and Starting a Stopped Program

Figure 6.3 presents a flow sequence which reinitializes and starts a program which has been stopped.

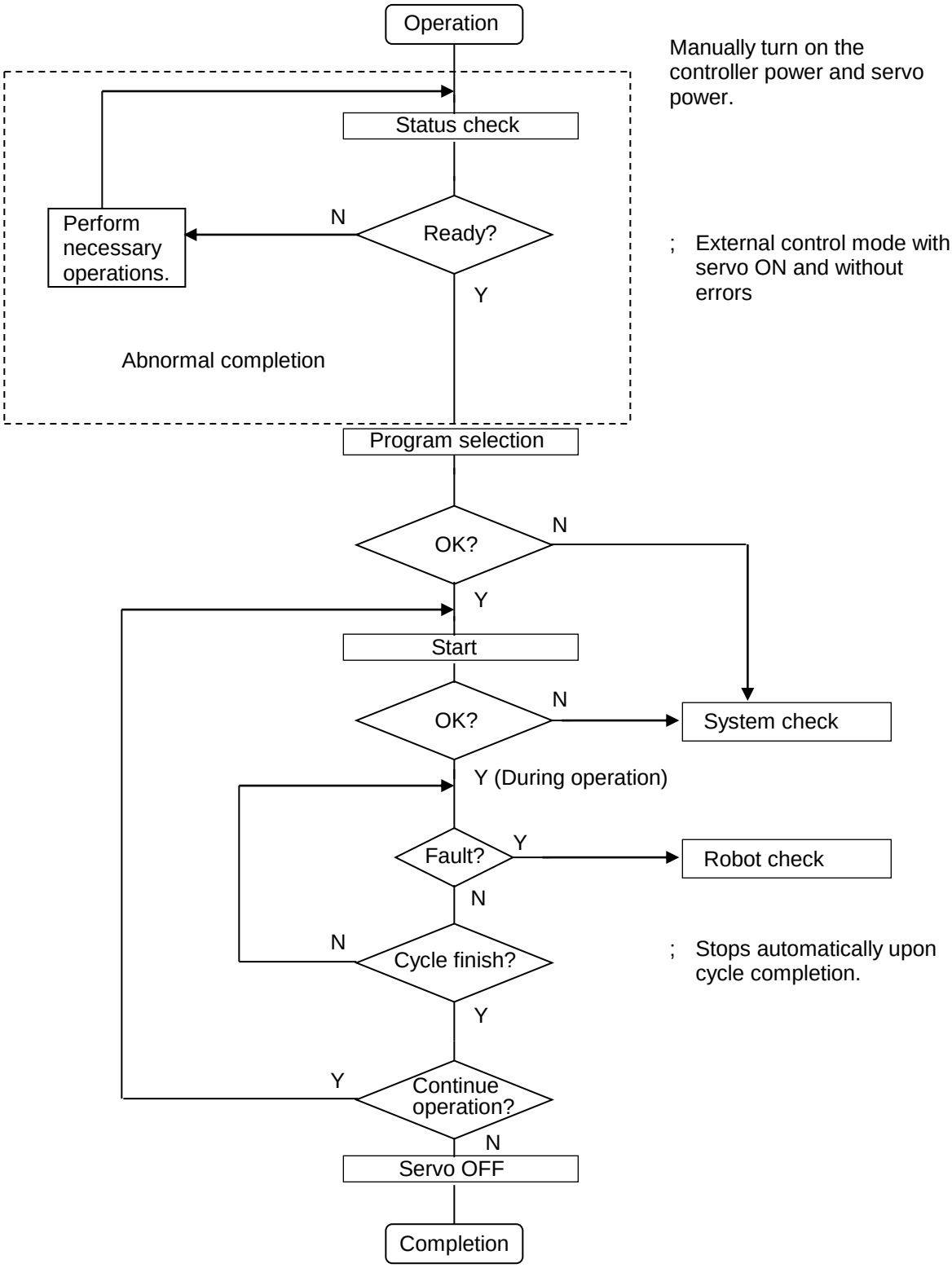


Fig. 6.1 Basic operation sequence

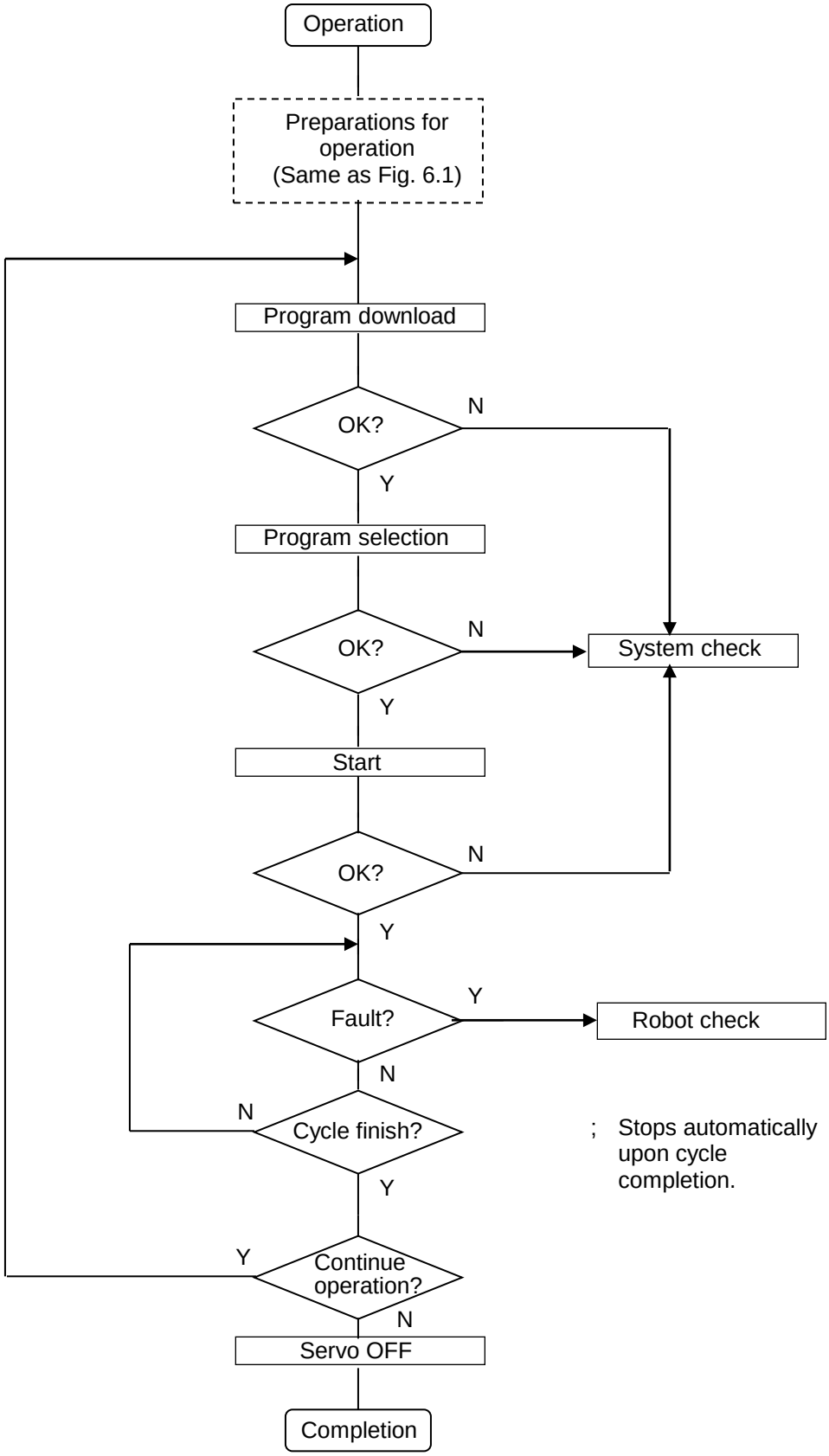


Fig. 6.2 Operation sequence including program download

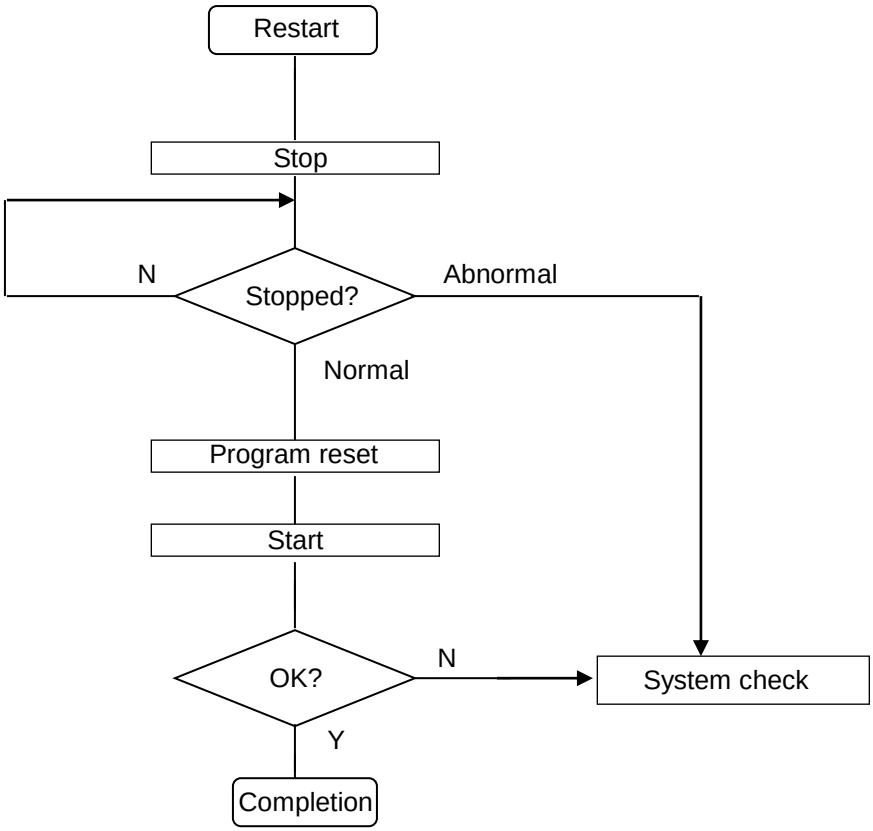


Fig. 6.3 Restart after program interruption

Section 7

Appendix

7.1 ASCII Code

Low-order 4 bits (Hexadecimal number)	High-order 4 bits (Hexadecimal number)							
	0	1	2	3	4	5	6	7
0	NUL Null	DLE Data link escape	␣ Space	0	@ At sign	P	` Grave accent	p
1	SOH Start of heading	DC1 Device control 1	! Exclamation mark	1	A	Q	a	q
2	STX Start of text	DC2 Device control 2	“ Quotation mark	2	B	R	b	r
3	ETX End of text	DC3 Device control 3	# Number sign	3	C	S	c	s
4	EOT End of transmission	DC4 Device control 4	$ Dollar sign	4	D	T	d	t
5	ENQ Enquiry	NAK Negative acknowledge	% Percent sign	5	E	U	e	u
6	ACK Acknowledge	SYN Synchronous idle	& Ampersand	6	F	V	f	v
7	BEL Bell	ETB End of transmission Block	‘ Single quote	7	G	W	g	w
8	BS Backspace	CAN Cancel	(Left parenthesis	8	H	X	h	x
9	HT Horizontal tab	EM End of medium	) Right parenthesis	9	I	Y	i	y
A	LF Line feed	EOF End of file	* Asterisk	: Colon	J	Z	j	z
B	VT Vertical tab	ESC Escape	+ Plus sign	; Semicolon	K	[Left square bracket	k	{ Left curly bracket
C	FF Form feed	FS File separator	, Comma	< Less-than sign; Inequality	L	¥ Yen sign	l	| Vertical line
D	CR Carriage return	GS Group separator	- Hyphen-minus	= Equals sign	M	] Right square bracket	m	} Right curly bracket
E	SO Shift out	RS Record separator	. Period	> Greater-than sign; Inequality	N	^ Caret	n	~ Tilde
F	SI Shift in	US Unit separator	/ Slash	? Question mark	O	_ Underscore	o	DEL Delete

* Codes 00 to 1F correspond to control characters. These characters will either be interpreted as a space or as a code having a specific meaning. The meaning of these codes is shown in the above table.