DENSO ROBOT

SUPPLEMENT

Main System Software Version 1.9

Preface

With the development of the HS-E series, DENSO WAVE has updated main system software designed for DENSO robot series from Version 1.8 to Version 1.9.

This book is a supplement to the DENSO robot manuals. It describes newly added and updated functions. Use this supplement together with other robot manuals.

As for the new HS-E series, refer to the "HS-E SERIES GENERAL INFORMATION ABOUT ROBOT" and "HS-E SERIES INSTALLATION & MAINTENANCE GUIDE."

Products covered by this manual

RC5 robot controller (Version 1.9 or later)

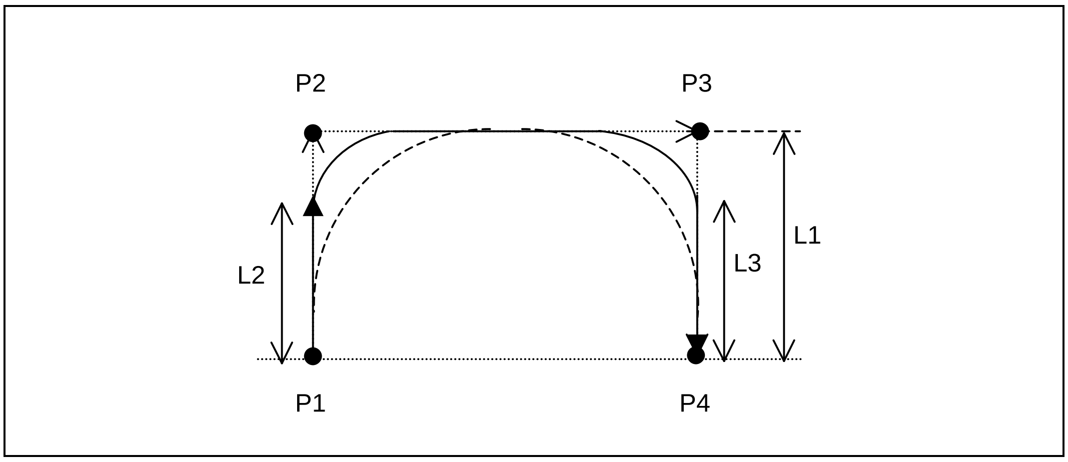
Contents

1. Enhanced Feature--Arch Motion Control (In Version 1.9 or later, only for 4-Axis Robot)

1.1 Introduction to Arch Motion Control

Refer to the PROGRAMMER'S MANUAL (I), p. 3-3.

The arch motion control facilitates an effective pick-and-place motion.



The above figure shows a pick-and-place motion from source position P1 to destination position P4. The pick-and-place motion can be usually accomplished by setting route positions P2 and P3. In the motion, specifying a pass motion in the vicinity of positions P2 and P3 may save the motion time of the arm endpoint.

Unlike a usual pass motion, the new arch motion control allows the arm endpoint to move from anywhere on the Z axis towards the next position, enabling more efficient pick & place motion.

Further, it is easy to program the arch motion control. You need to define only the move distance of the Z axis from P3 to P4 (L1), the distance from P1 to the arch start position (L2), and the distance from P4 to the arch end position (L3).

Note that the arch motion does not support the position control so that the path of the arm endpoint may vary depending upon the actual robot speed. Be careful with interference with the surrounding facilities when the robot is in arch motion. Under the arch motion control, all robot operations are performed via point-to-point (PTP) moves.

1.2 Arch Motion Library

ArchMove (Library) [In Version 1.9 or later]

Function

Executes an arch motion.

Syntax

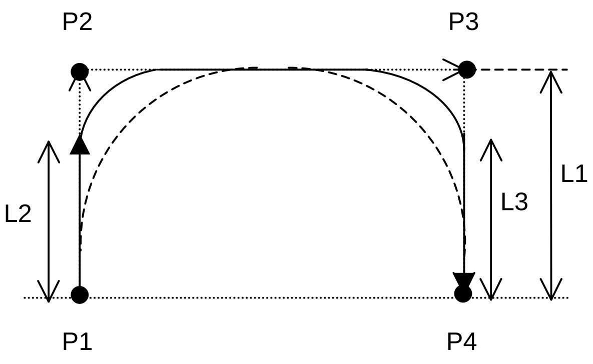
```
ArchMove (<destination (Position variable)>, <Z-axis move>)
```

Explanation

This library makes a pick-and-place motion from current position P1 (pick) to destination position P4 (place) via the vicinity of route positions P2 and P3.

The route position P3 is located above destination position P4 by the vertical move distance (L1) specified by `<Z-axis move>`.

The route position P2 is on the Z axis of source position P1 and has the same Z-axis value as P3.



Arch start position (L2) and end position (L3) should be defined in the SetArchParam library.

Macro Definition

Requires file <pacman.h>

Related Terms

SetArchParam (library)

Notes

(1) Execute this library in a TAKEARMed task that holds an arm semaphore.

(2) An arch motion does not support the position control so that the path of the arm endpoint may vary depending upon the actual robot speed. Be careful with interference with the surrounding facilities when the robot is in arch motion.

(3) The Z-axis value of the source position and that of the destination position are not identical with each other. Otherwise, this library will not function normally.

(4) This library is not available to the work coordinates system. Use it in the base coordinates system.

Sample Coding

```
CALL ArchMove (P10,100)     'Moves the arm endpoint to destination P10
                            'with 100 mm Z-axis move. The current arch
                            start and end positions will apply.
```

SetArchParam (Library) [In Version 1.9 or later]

Function

Defines the start position of a horizontal movement (Arch start position) in upward movement of the arm endpoint and the end position in downward movement (Arch end position).

Syntax

```
SetArchParam (<Arch start position>, <Arch end position>)
```

Explanation

This library configures an arch form.

`<Arch start position>` is the distance (mm) from the source position in the direction of the Z axis; `<Arch end position>` is that from the destination position.

The entry range is from 0 to "Maximum reach position (mm) in the Z-axis direction" in increment of 1 mm.

The defaults of `<Arch start position>` and `<Arch end position>` are 0 mm when the robot controller is turned on.

Values defined in this library will be effective until the robot controller is turned off.

Macro Definition

Requires file <pacman.h>

Related Terms

ArchMove

Notes

Specifying any invalid value will cause no syntax error, but it will result in a run-time error at the execution of the `ArchMove`.

Sample Coding

```
CALL SetArchParam (10,20)   'Set the arch form so that the arm endpoint
                            'moves upward 10 mm, starts horizontal move
                            'and ends it 20 mm above the destination.
<Program lines hidden>

Call ArchMove (P10,100)     'Execute arch motion towards destination P10
                            'with 100 mm Z-axis vertical move.
```

2. Local Variable-Related Enhancement

2.1 Referring To or Writing Into Local Variables

This new local variable-related enhancement allows you to refer to or write into local variables (integer, floating-point, double-precision, vector, position, joint, homogeneous transform matrix, string, and DEFIO variables) in a program.

This enhancement includes the following variable facilities:

(1) Quick reference

You may immediately refer to local variables defined in a program just by specifying a desired program line.

(2) Referring to registered variables

You may refer to local variables as well as global variables. Use this facility when you cannot designate a program line since the program is running or when you want to refer to variables in more than one program.

(3) Running a program with local variable arguments

You may run a program with local variable arguments independently.

NOTE: This facility will not be supported for a program with array variable arguments, e.g., `PROGRAM SUB1 (li%(10))`.

(4) Referring to or writing into local variables in WINCAPSII

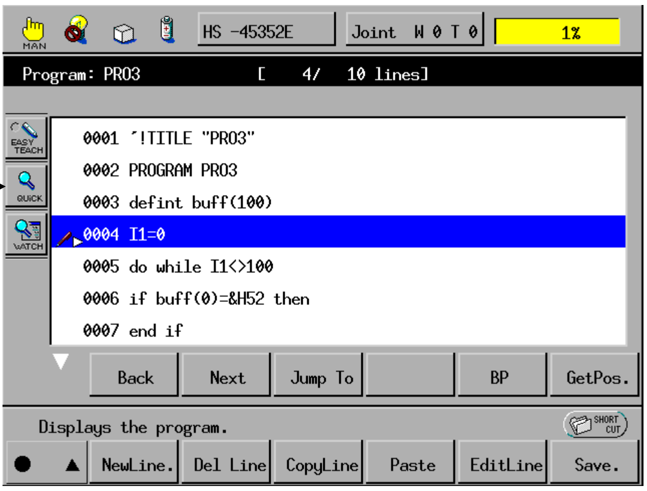
In WINCAPSII you may refer to or write into local variables.

NOTE: You may use the above facilities (1) through (3) with the teach pendant, not with the mini-pendant or operating panel.

2.2 Quick Reference

You may immediately refer to local variables defined in a program. To do so, specify a desired program line and press the **QUICK** reference button that is newly provided in the coding list window as shown below.

QUICK reference button



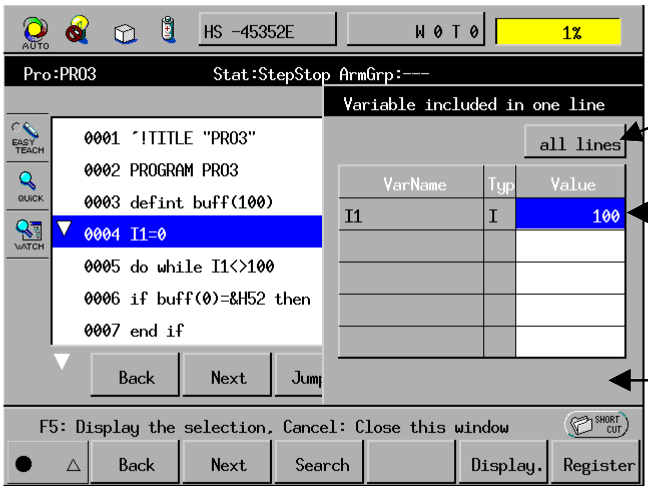
```
Program: PR03            [    4/   10 lines]
0001 '!TITLE "PR03"
0002 PROGRAM PR03
0003 defint buff(100)
0004 I1=0
0005 do while I1<>100
0006 if buff(0)=&H52 then
0007 end if
Displays the program.
```

NOTE: Only in manual mode, you can highlight a desired program line or move the cursor to a desired line.

The "Variables included in one line" window (see below) appears where local variables involved in the currently highlighted line and global variables are displayed. The sample window below displays variable "I1" in the STEP STOP program line.

Integer, floating-point, double-precision, or DEFIO variables, if any, will display with their values.

If DEFIO variables are referred to, "IO variable type," "Port address" and "Mask info" also appear.



```
Pro:PR03            Stat:StepStop ArmGrp:---
                    Variable included in one line
0001 '!TITLE "PR03"
0002 PROGRAM PR03
0003 defint buff(100)
0004 I1=0
0005 do while I1<>100
0006 if buff(0)=&H52 then
0007 end if
F5: Display the selection, Cancel: Close this window
```

VarName	Typ	Value
I1	I	100

Display switcher button

Values of integer, floating-point, double-precision, and DEFIO variables only will display

In the case of DEFIO variables, "IO variable type," "Port address" and "Mask info" also appear.

NOTE 1: If the index of the referred-to variable is out of range (Example 1 below) or not a numerical value (Example 2 below), then the index field of the variable name will show "?."

(Example 1) Although the number of integer variables defined is 200, you attempt to refer to integer variable I201 written in a program line.

(Example 2) You attempt to display a variable with macro name index like `I[slotnum]`.

If the index field shows "?," then no value will display even for integer, floating-point, double-precision, and DEFIO variables. Press the [Display.] and choose the index you want to refer to.

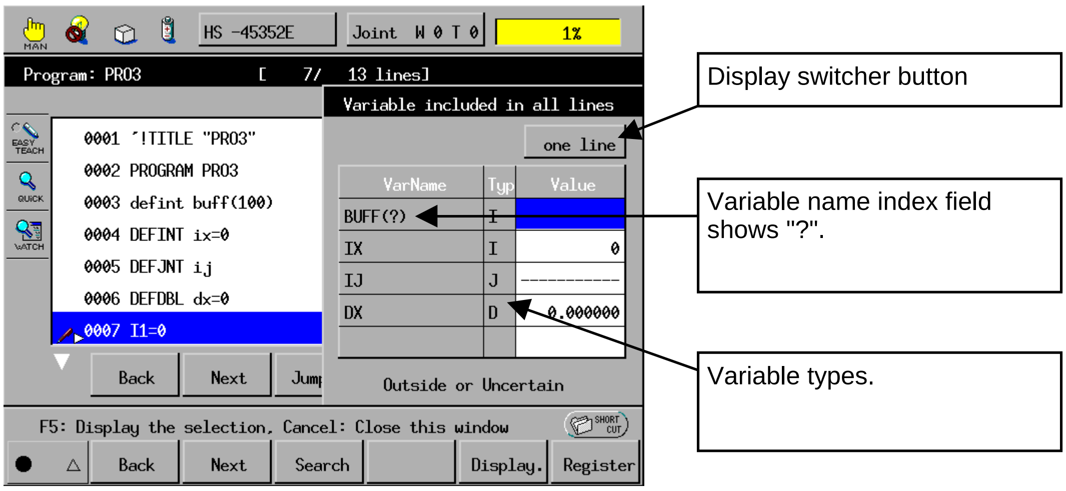
NOTE 2: If the port address of a referred-to DEFIO variable is out of the specified I/O range, then the DEFIO variable will display in gray.

NOTE 3: An array variable assigned to an argument cannot be displayed.

(Example) `PROGRAM SUB1 (li%, li2%(10))`

The `li2` cannot be displayed since the argument is an array variable.

With the display switcher button, you may switch from the "Variables included in one line" to "Variables included in all lines." The sample window below shows variables included in all program lines in the currently selected program.



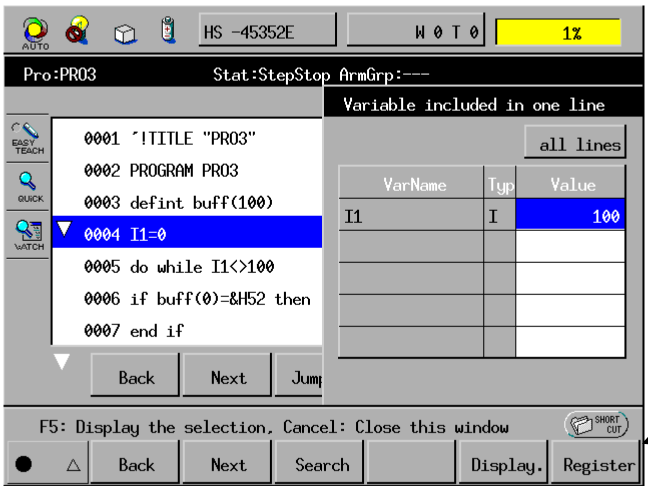
```
HS -45352E   Joint  W 0 T 0   1%
Program: PR03          [   7/   13 lines]
                              Variable included in all lines
0001 '!TITLE "PR03"                               one line
0002 PROGRAM PR03             VarName     Typ  Value
0003 defint buff(100)         BUFF(?)     I
0004 DEFINT ix=0              IX          I             0
0005 DEFJNT ij                IJ          J    ----------
0006 DEFDBL dx=0              DX          D      0.000000
0007 I1=0
     Back   Next   Jump            Outside or Uncertain
F5: Display the selection, Cancel: Close this window
●  △  Back   Next   Search        Display.  Register
```

NOTE 1: While the "Variables included in one line" window displays not only local variables but global variables, the "Variables included in all lines" window cannot display global variables.

NOTE 2: In the "Variables included in all lines" window, all array variables will display with "?" in their indexes. Press the [Display] and choose the index you want to refer to.

On this screen, you may register variables by pressing [Register]. It is possible to register a maximum of 50 variables.

TIP: You may refer to those registered variables with the display facility described in Section 2.3.



```
HS -45352E        W 0 T 0        1%
Pro:PR03                 Stat:StepStop ArmGrp:---
                              Variable included in one line
'!TITLE "PR03"                                  all lines
PROGRAM PR03                  VarName    Typ   Value
defint buff(100)              I1         I       100
I1=0
do while I1<>100
if buff(0)=&H52 then
end if
        Back    Next    Jump
F5: Display the selection, Cancel: Close this window
        Back    Next    Search          Display.   Register
```

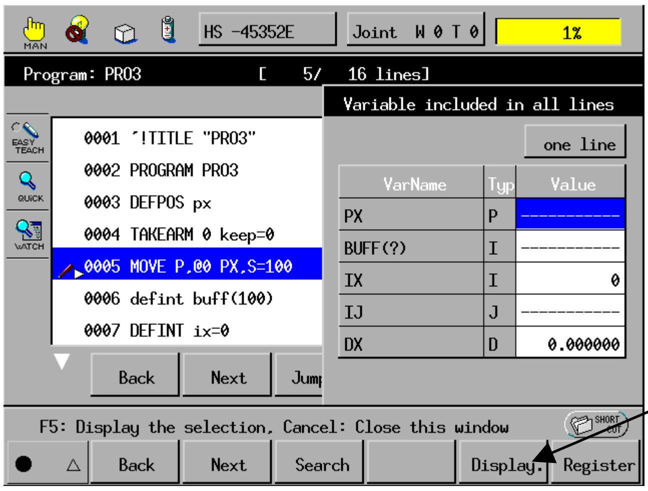
[Register] button

NOTE 1: If the index field of a variable name shows "?" on the above screen, the variable cannot be registered here.

NOTE 2: Any DEFIO variable whose port address is out of the specified range cannot be registered here.

Press [Display] shown below to display the values of the selected variable (see the next page)

NOTE: If you select a DEFIO variable whose port address is out of the specified range, its details cannot be displayed.

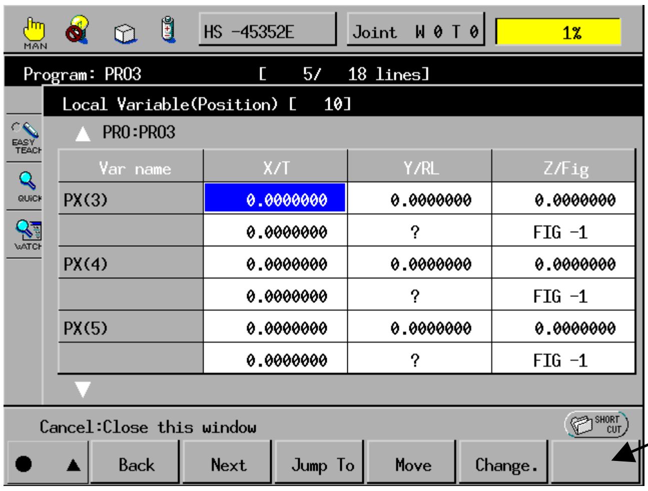


```
HS -45352E        Joint   W 0 T 0        1%
Program: PR03               [   5/   16 lines]
                              Variable included in all lines
'!TITLE "PR03"                                  one line
PROGRAM PR03                  VarName    Typ   Value
DEFPOS px                     PX         P     -----------
TAKEARM 0 keep=0              BUFF(?)    I     -----------
MOVE P,@0 PX,S=100            IX         I               0
defint buff(100)              IJ         J     -----------
DEFINT ix=0                   DX         D        0.000000
        Back    Next    Jump
F5: Display the selection, Cancel: Close this window
        Back    Next    Search          Display.   Register
```

[Display] button

The next sample screen shows the values of locally defined position variable PX (3).

On this screen, you may modify the local variable values or replace local variables as well as for global variables. To register the modified variables, press [F12 Register].



Program: PR03 [5/ 18 lines]

Local Variable(Position) [10]

PRO:PR03

Var name	X/T	Y/RL	Z/Fig
PX(3)	0.0000000	0.0000000	0.0000000
	0.0000000	?	FIG -1
PX(4)	0.0000000	0.0000000	0.0000000
	0.0000000	?	FIG -1
PX(5)	0.0000000	0.0000000	0.0000000
	0.0000000	?	FIG -1

Cancel:Close this window

Press the Shift button to shift the menu bar and show [F12 Register] button here.

NOTE 1: When a variable's index field is "?," pressing [Display] will display a variable whose index is 0. Move the cursor to that index.

NOTE 2: To modify the current value of a DEFIO variable, you need to hold down the deadman switch, same way as modifying I/Os.

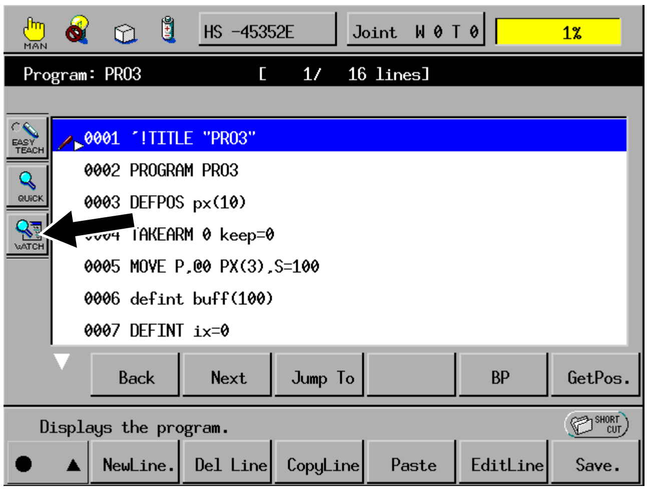
NOTE 3: This quick reference facility cannot take position data into local variables.

2.3 Referring to Registered Variables

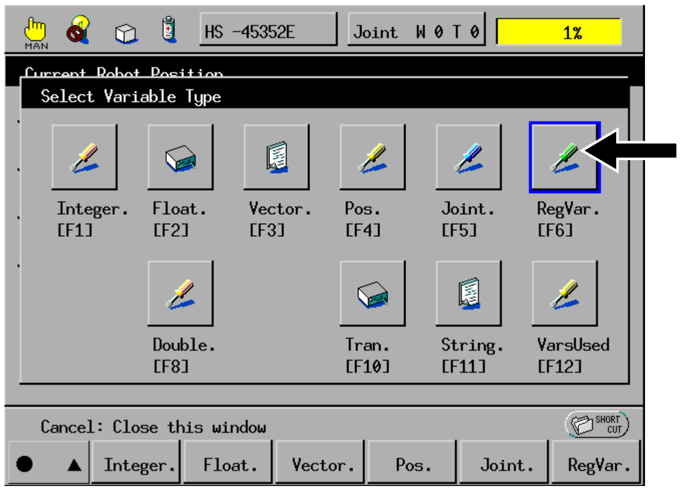
As described in Section 2.2, you may register variables with [Register] in the variable quick reference facility or [F12 Register] on the variable value screen.

This registered variable reference facility allows you to refer to previously registered variables. Use this facility when you cannot designate a program line since the program is running or when you want to refer to variables in more than one program.

To call up the registered variables window, press [WATCH] in the coding list window or [F6 RegVar.] in the Select Variable Type window.

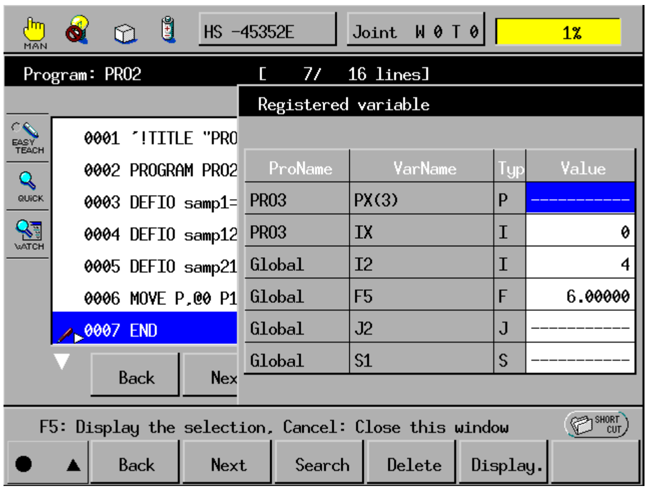


```
HS -45352E    Joint  W 0 T 0    1%
Program: PR03          [  1/   16 lines]
0001 '!TITLE "PR03"
0002 PROGRAM PR03
0003 DEFPOS px(10)
0004 TAKEARM 0 keep=0
0005 MOVE P,@0 PX(3),S=100
0006 defint buff(100)
0007 DEFINT ix=0
Displays the program.
```



```
HS -45352E    Joint  W 0 T 0    1%
Current Robot Position
Select Variable Type
Cancel: Close this window
```

The Registered variable list window will appear as shown below.



```
HS -45352E    Joint  W 0 T 0    1%
Program: PR02          [  7/   16 lines]
0001 '!TITLE "PRO
0002 PROGRAM PR02
0003 DEFIO samp1=
0004 DEFIO samp12
0005 DEFIO samp21
0006 MOVE P,@0 P1
0007 END
```

Registered variable

ProName	VarName	Typ	Value
PR03	PX(3)	P	-----------
PR03	IX	I	0
Global	I2	I	4
Global	F5	F	6.00000
Global	J2	J	-----------
Global	S1	S	-----------

F5: Display the selection, Cancel: Close this window

NOTE 1: Global variables registered will appear as "Global" in the program name column (ProName).

NOTE 2: If the program name, variable name, or the number of dimensions of a registered variable is modified and compiled, then the variable will appear in gray.

NOTE 3: If the program name, variable name, I/O variable type, port address, or mask information of a registered DEFIO variable is modified and compiled, then the DEFIO variable will appear in gray.

Press [Display.], and the values of the selected variable will appear where you may modify those values or replace local variables as well as for global variables.

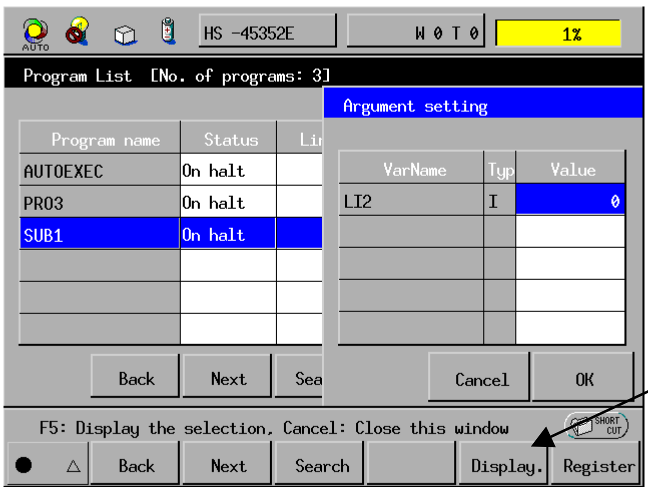
2.4 Running a Program with Local Variable Arguments

You may run a program containing local variable arguments in stand-alone mode or edit those arguments.

If you attempt to start an on-halt program containing arguments in the Program List screen, the Argument setting window will appear. The sample screen given below will appear if you attempt to start the PROGRAM SUB1 (li%).

NOTE: If a program has an argument containing an array variable, no Argument setting window will appear. Instead, a run-time error will occur with the error message 736F "Cannot start any programs with array argument(s)."

To modify argument values, press [Display.]. After modification, check the new argument values in the Argument setting window and then press [OK].



HS -45352E W 0 T 0 1%

Program List [No. of programs: 3]

Program name	Status	Li
AUTOEXEC	On halt	
PRO3	On halt	
SUB1	On halt	

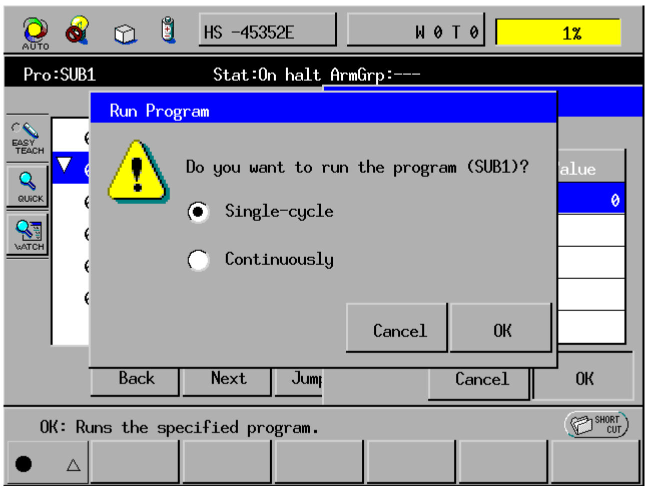
Argument setting

VarName	Typ	Value
LI2	I	0

F5: Display the selection, Cancel: Close this window

To modify arguments, press [Display]. You can modify it in the same way as for global variables.

As shown below, the same prompting dialog as for ordinary start will appear. After that, the same starting operation as usual will take place.



HS -45352E W 0 T 0 1%

Pro:SUB1 Stat:On halt ArmGrp:---

Run Program

Do you want to run the program (SUB1)?

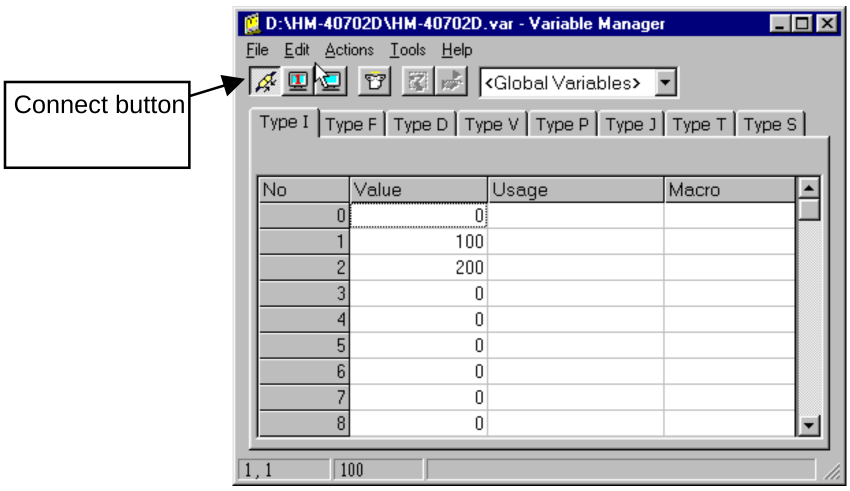
[✓ Single-cycle] Continuously

OK: Runs the specified program.

2.5 Referring To or Writing Into Local Variables in WINCAPSII

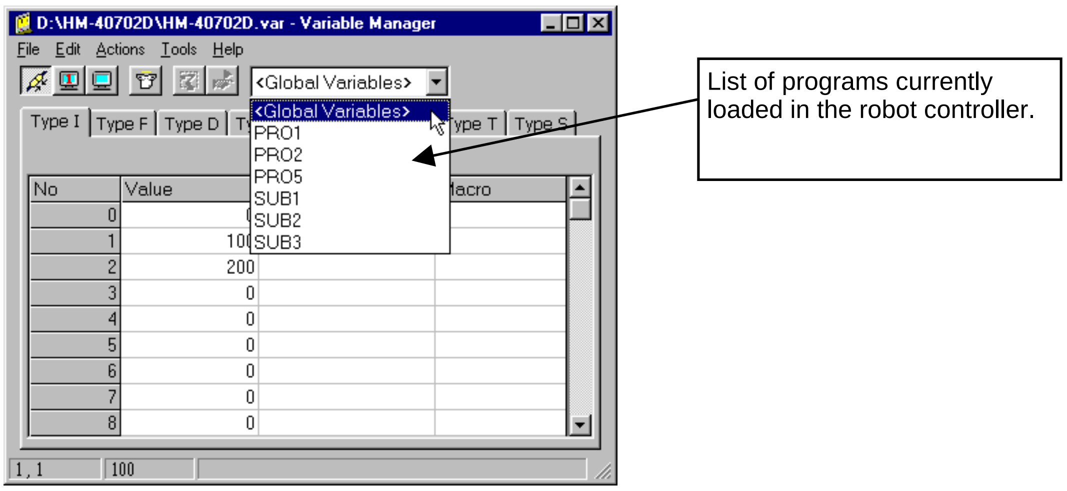
In WINCAPSII, you may refer to or write into local variables.

First, press the Connect button to make Variable Manager in WINCAPSII online with the robot controller.

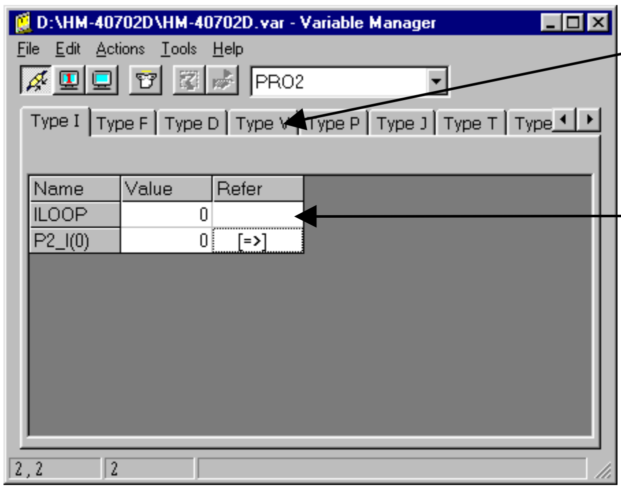

Connect button
D:\HM-40702D\HM-40702D.var - Variable Manager
File Edit Actions Tools Help
<Global Variables>
Type I Type F Type D Type V Type P Type J Type T Type S
1, 1 100

No	Value	Usage	Macro
0	0		
1	100		
2	200		
3	0		
4	0		
5	0		
6	0		
7	0		
8	0		

Next, open the pull-down menu and select a program whose local variables should be referred to or modified.

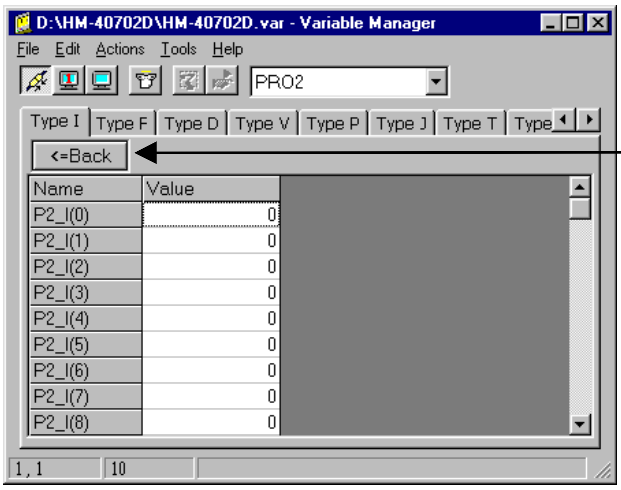

List of programs currently loaded in the robot controller.
D:\HM-40702D\HM-40702D.var - Variable Manager
File Edit Actions Tools Help
<Global Variables>
<Global Variables>
PRO1
PRO2
PRO5
SUB1
SUB2
SUB3
Type I Type F Type D Type T Type S
1, 1 100

The sample window below displays local variables defined in program PRO3 and the contents of the array argument of the variable named BUFF.


D:\HM-40702D\HM-40702D.var - Variable Manager
File Edit Actions Tools Help
PRO2
Type I Type F Type D Type V Type P Type J Type T Type
Name Value Refer
ILOOP 0
P2_I(0) 0 [=>]
2, 2 2

Select a local variable type to display its contents. In this example, local integer variables in program PRO3 are displayed.

If the local variable has an array argument, => icon will appear in the reference column. Double-click this icon to display its contents.


D:\HM-40702D\HM-40702D.var - Variable Manager
File Edit Actions Tools Help
PRO2
Type I Type F Type D Type V Type P Type J Type T Type
<=Back
1, 1 10

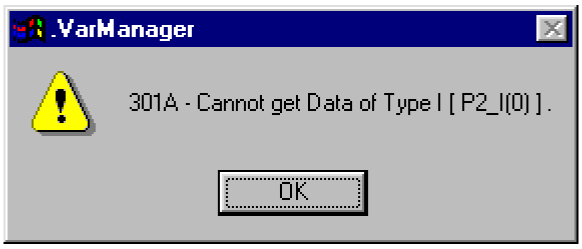
Name	Value
P2_I(0)	0
P2_I(1)	0
P2_I(2)	0
P2_I(3)	0
P2_I(4)	0
P2_I(5)	0
P2_I(6)	0
P2_I(7)	0
P2_I(8)	0

To return to the previous window, click here.

The subsequent value change procedure will be the same as for global variables.

NOTE 1: To modify DEFIO variables in WINCAPSII, you need to hold down the deadman switch of the teach pendant.

NOTE 2: When this facility is running in WINCAPSII online, if the controller loads any new programs, Variable Manager automatically goes offline, issues the following message, and no longer displays any variables.



301A - Cannot get Data of Type I [P2_I(0)] .

If so, make Variable Manager online again.

3. Enhanced Easy Teaching [TP]

Refer to the SETTING-UP MANUAL, p. 3-32

As an additional enhancement of the easy teaching facility implemented in Version 1.8 first, Version 1.9 enables you to:

- Specify global variables
- Write APPROACH and DEPART commands

This chapter covers an overall easy teaching including the above additional enhancement.

Easy teaching is a new facility that enables data entry, program edition, and teaching of point coordinates from the teach pendant with ease of operation.

The easy teaching allows you to:

(1) Insert a motion command and its parameters (including destination positions) to the desired program by one-touch operation.

(2) Edit motion commands and their parameters by one-touch operation.

(3) Execute and check programs you have edited before compiling them.

3.1 Inserting a Motion Command

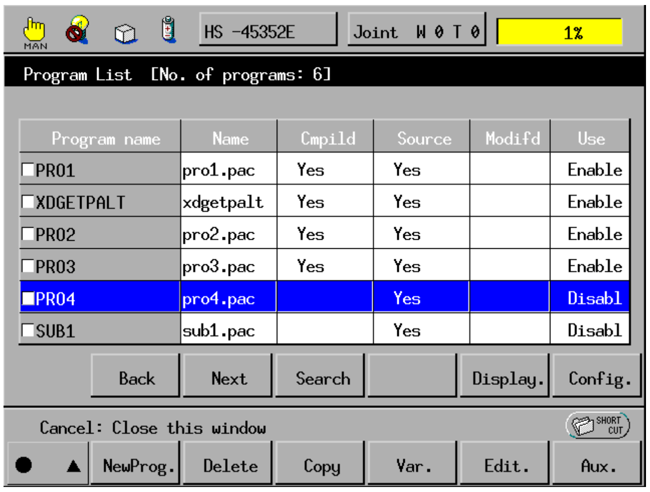
What follows is an operation flow for inserting a motion command by using the easy teaching facility.

■ Possible from the teach pendant only

Step 1 Set the mode selector switch to the MANUAL position.

Step 2 On the top screen, press [F1 Program].

The Program List window appears as shown below.



Program List [No. of programs: 6]

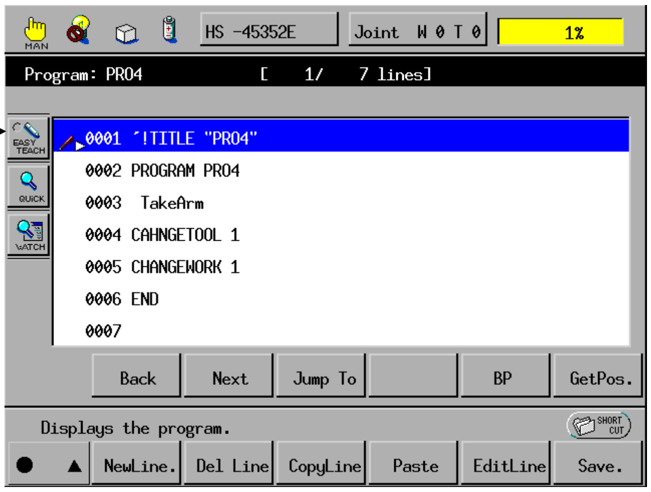
Program name	Name	Cmpild	Source	Modifd	Use
☐PRO1	pro1.pac	Yes	Yes		Enable
☐XDGETPALT	xdgetpalt	Yes	Yes		Enable
☐PRO2	pro2.pac	Yes	Yes		Enable
☐PRO3	pro3.pac	Yes	Yes		Enable
■PRO4	pro4.pac		Yes		Disabl
☐SUB1	sub1.pac		Yes		Disabl

F5

Select a program you want to edit and then press [F5 Edit.] or [Display.] (or [F1 NewProg.].

NOTE: To edit existing programs, use [F5 Edit.]; to check them, use [Display.]. To edit a new program, use [F1 NewProg.].

Step 3 The coding list of the selected program appears as shown below.

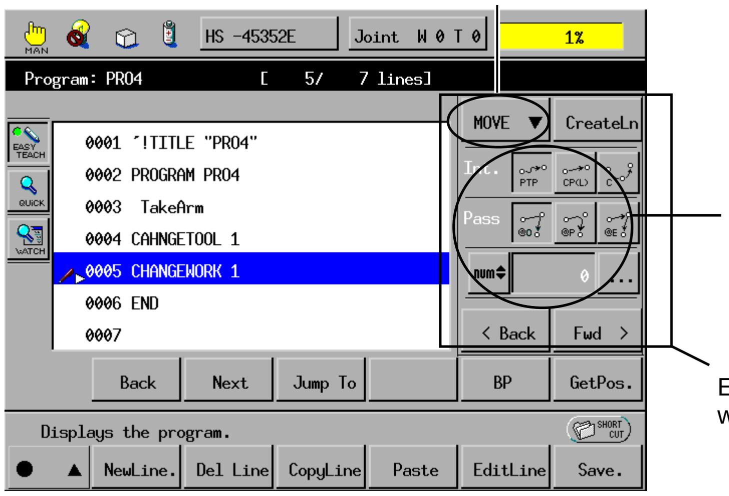


```
HS -45352E    Joint  W 0 T 0    1%
Program: PR04          [   1/    7 lines]
0001 '!TITLE "PR04"
0002 PROGRAM PR04
0003  TakeArm
0004 CAHNGETOOL 1
0005 CHANGEWORK 1
0006 END
0007
Back  Next  Jump To    BP  GetPos.
Displays the program.
NewLine.  Del Line  CopyLine  Paste  EditLine  Save.
```

Press the easy teaching icon to call up the easy teaching window.

Step 4 Press parameter icons to set up the motion command parameters.

TIP: The specifications of parameter icons that seem to be depressed will be set up as effective parameter values.


Command select button
(reserved for future expansion)
Parameter icons (See the table below.)
Easy teaching window

```
HS -45352E    Joint  W 0 T 0    1%
Program: PR04          [   5/    7 lines]
0001 '!TITLE "PR04"
0002 PROGRAM PR04
0003  TakeArm
0004 CAHNGETOOL 1
0005 CHANGEWORK 1
0006 END
0007
MOVE ▼  CreateLn
Inc.  PTP  CP(L)  C
Pass  @0  @P  @E
num  0  ...
< Back  Fwd >
Back  Next  Jump To    BP  GetPos.
Displays the program.
NewLine.  Del Line  CopyLine  Paste  EditLine  Save.
```

NOTE: The Command select button is reserved for future expansion. It is currently disabled.

Interpolation (Note 1)		Pass Start Position (Note 1)		Destination Position	
PTP	PTP control (Syntax: "P")	@0	End motion (@0 option)	num	Immediate value (Note 2)
CP(L)	CP control (Syntax: "L")	@P	Pass motion (@P option)	P	Global, position variable (Note 2)
C	Arc Motion (Syntax: "C")	@E	Encoder value check motion (@E option)	J	Global, joint variable (Note 2)
				T	Global, homogeneous transform matrix variable (Note 2)
				0 ...	Global variable number. Press this button to call up a numeric keypad where you can modify the current variable value.

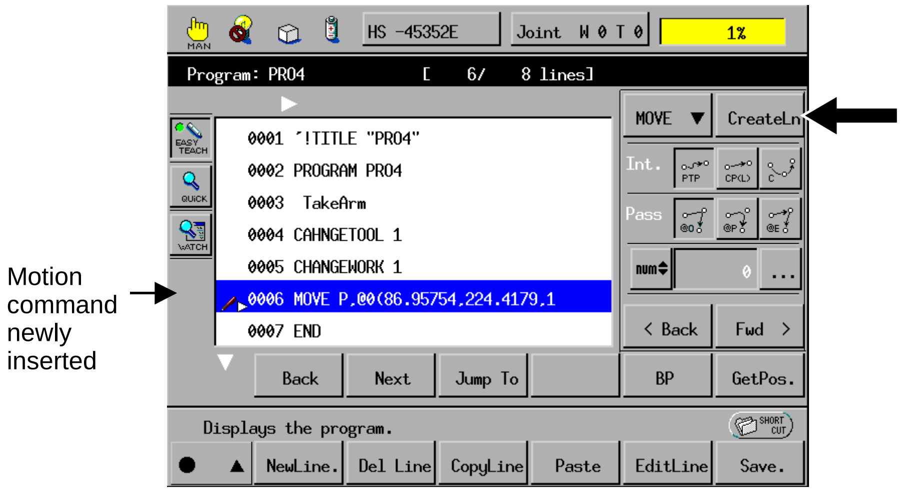
(Note 1) The feature implemented for the depressed icon takes effect.

(Note 2) Each time you press this button, the icon will cycle through [Immediate] -> [Position variable] -> [Joint variable] -> [Trans variable] in this order. The feature implemented for the displayed icon takes effect.

Step 5

Move the cursor to a line immediately preceding the line where you want to insert a motion command, then press [CreateLn].

By calculating the parameter icon information and the current robot position, the controller automatically creates an appropriate motion command. Then it inserts the command to the line next to the selected line.



```
Program: PR04            [   6/    8 lines]
0001 '!TITLE "PR04"
0002 PROGRAM PR04
0003  TakeArm
0004 CAHNGETOOL 1
0005 CHANGEWORK 1
0006 MOVE P,@0(86.95754,224.4179,1
0007 END
Displays the program.
```

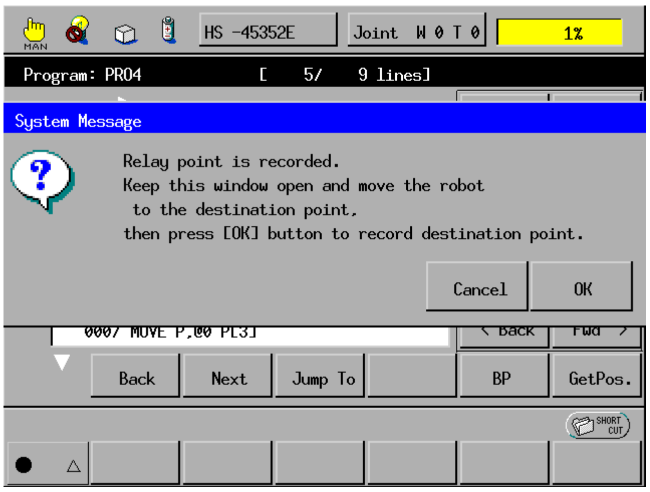
If you select an immediate value to specify the destination position, then the current position value will be written in the current motion command. For the procedure, go to Step 6.

If you select a global variable to specify the destination position, then the current position will be saved into the global variable. For the procedure, go to Step 7.

Step 6

If you insert such a motion command that defines interpolation along an arc (e.g., MOVE C) and destination position with an immediate value, then pressing [CreateLn] will display the following message.

Keep this dialog open and move the robot arm to the destination position, then press [OK]. The command will be inserted.

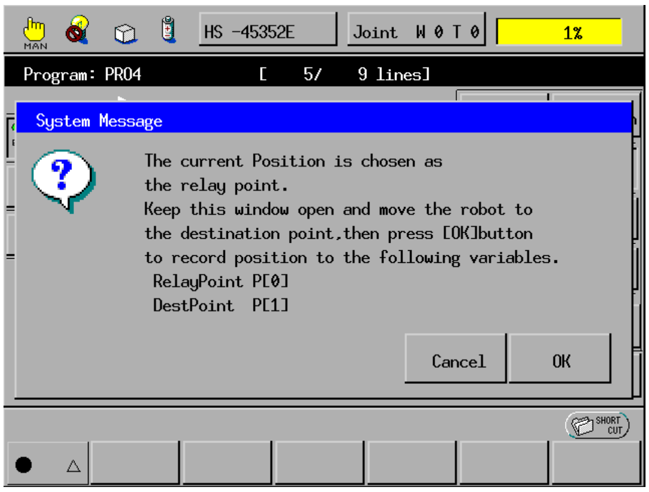


```
Program: PR04            [   5/    9 lines]
System Message
Relay point is recorded.
Keep this window open and move the robot
 to the destination point,
then press [OK] button to record destination point.
0007 MOVE P,@0 P[3]
```

Step 7

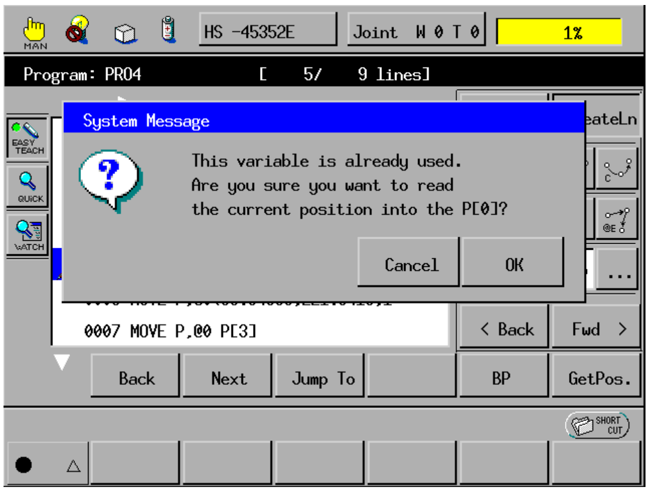
If you insert such a motion command that defines interpolation along an arc (e.g., MOVE C) and destination position with a global variable, then pressing [CreateLn] will display the following message.

Keep this dialog open and move the robot arm to the destination position, then press [OK].



```
System Message
The current Position is chosen as
the relay point.
Keep this window open and move the robot to
the destination point,then press [OK]button
to record position to the following variables.
 RelayPoint P[0]
 DestPoint  P[1]
```

The following confirmation massage may appear informing that the global variable has already contained any value.



```
System Message
This variable is already used.
Are you sure you want to read
the current position into the P[0]?
```

```
0007 MOVE P,@0 P[3]
```

If overwriting is OK, press [OK]. The variable will be updated.

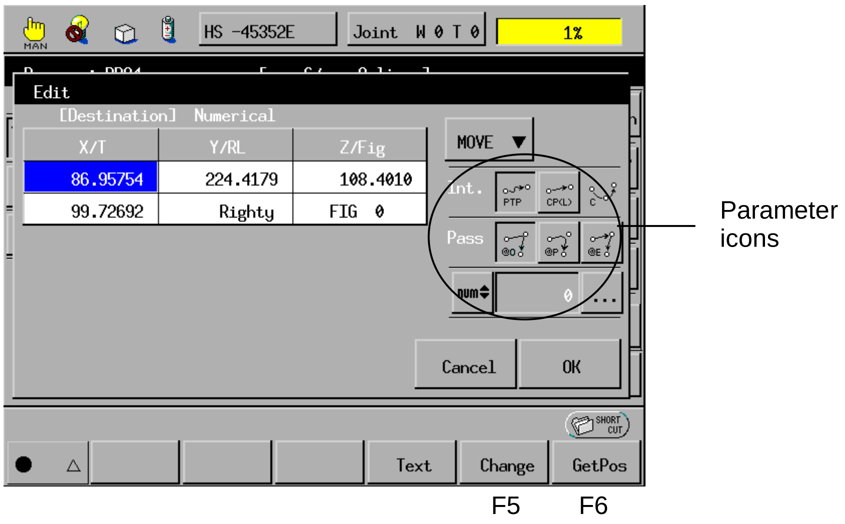
3.2 Editing a Motion Command

What follows is an operation flow for editing an existing motion command by using the easy teaching facility.

Step 1 Carry out Steps 1 through 3 in Section 3.1 to call up the easy teaching window.

Step 2 Move the blue cursor to a program line you want to edit, then press [F5 EditLine].

If the command on the selected line can be edited in the easy teaching window (refer to Section 3.5), then the parameter information will display according to the parameter icons.



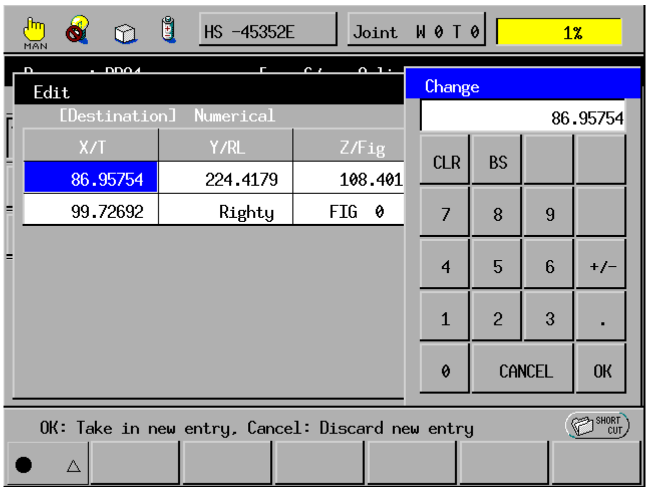
X/T	Y/RL	Z/Fig
86.95754	224.4179	108.4010
99.72692	Righty	FIG 0

When the window is called up, the parameter icons reflect the command information on the selected line.

You may change the parameters by using the parameter icons or change the destination point (or passing points) by pressing [F5 Change] (see Step 3) or [F6 GetPos] (See Step 4). After that, pressing [OK] will make the new settings go into effect.

Step 3 To enter the desired numeric value by using [F5 Change]:

Move the cursor to a field you want to change with the up-, down, left-, and right-arrow keys or jog dial, then press [F5 Change]. The numeric keypad will appear as shown below. Enter the desired numeric value.



X/T	Y/RL	Z/Fig
86.95754	224.4179	108.401
99.72692	Righty	FIG 0

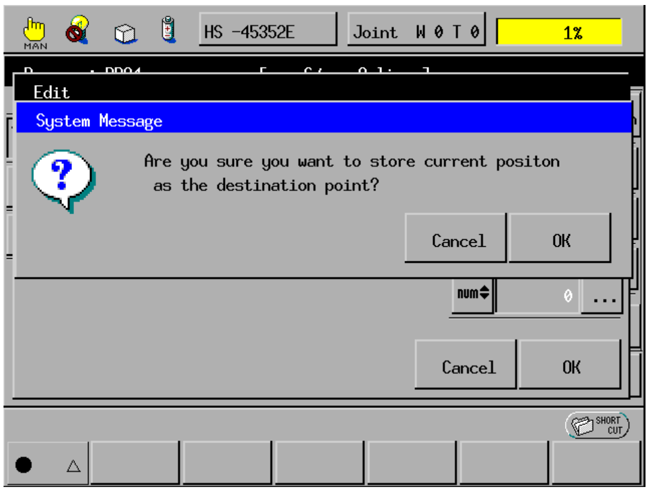
OK: Take in new entry, Cancel: Discard new entry

NOTE: If a motion command you selected defines interpolation along an arc (e.g. MOVE C), you may define the passing points and destination points by using the up- and down-arrow keys or jog dial.

Step 4

To get the current robot position by using [F6 GetPos]:

Press [F6 GetPos], and the controller will get the current robot position into the memory area and overwrite the line you are editing with the gotten data.

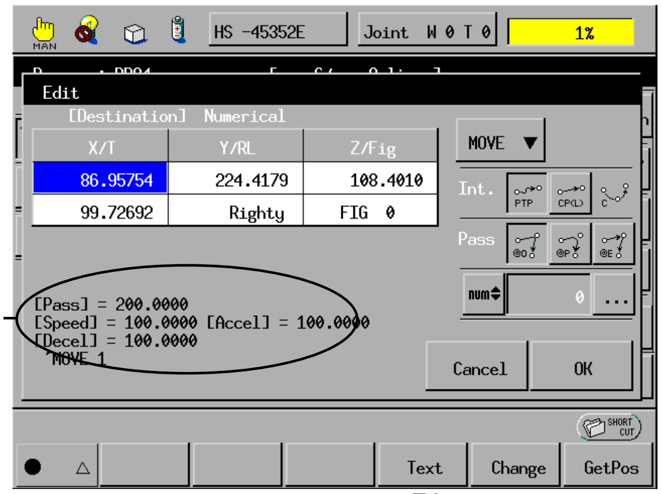


Are you sure you want to store current positon as the destination point?

NOTE: If you are editing a motion command that defines interpolation along an arc (e.g. MOVE C), a window will appear prompting you to choose either a passing point or destination point. Choose either one.

Step 5

Circled below are parameters that cannot be edited in this screen.



[Destination] Numerical

X/T	Y/RL	Z/Fig
86.95754	224.4179	108.4010
99.72692	Righty	FIG 0

```
[Pass] = 200.0000
[Speed] = 100.0000 [Accel] = 100.0000
[Decel] = 100.0000
MOVE 1
```

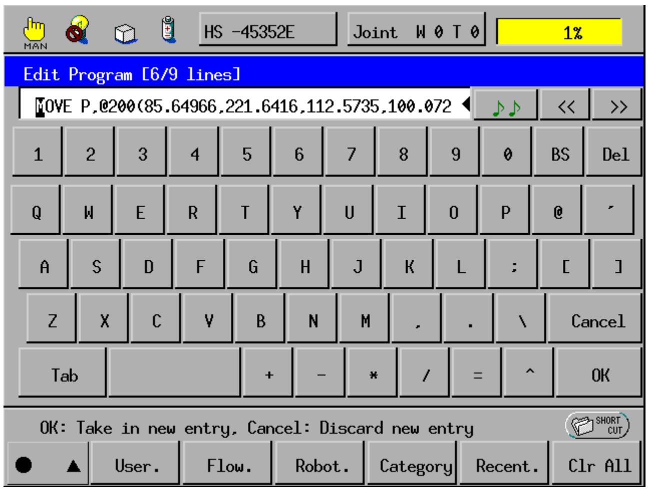
Parameters that cannot be edited here.

F4

To edit them, press [F4 Text]. The alphanumeric key screen will appear as shown in Step 6.

Step 6

Use the alphanumeric key screen shown below to edit parameters in the text mode.

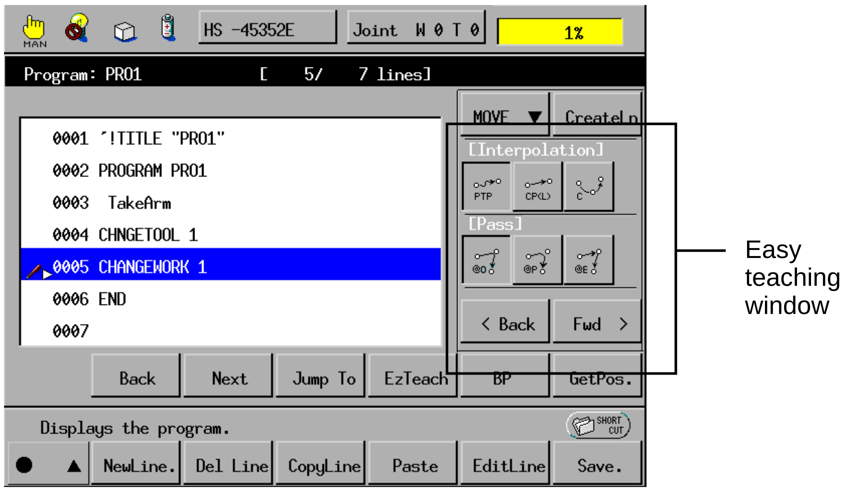

MAN
HS -45352E
Joint W 0 T 0
1%
Edit Program [6/9 lines]
MOVE P,@200(85.64966,221.6416,112.5735,100.072
<<
>>
1 2 3 4 5 6 7 8 9 0 BS Del
Q W E R T Y U I O P @ '
A S D F G H J K L ; [ ]
Z X C V B N M , . \ Cancel
Tab + - * / = ^ OK
OK: Take in new entry, Cancel: Discard new entry
SHORT CUT
User. Flow. Robot. Category Recent. Clr All

NOTE: Pressing [Cancel] in the above screen will cancel all settings made in the easy teaching window.

3.3 Executing Motion Commands

What follows is an operation flow for executing motion commands by using the easy teaching facility.

Step 1 Carry out Steps 1 through 3 in Section 3.1 to call up the easy teaching window.



Program: PR01 [5/ 7 lines]

```
0001 '!TITLE "PR01"
0002 PROGRAM PR01
0003  TakeArm
0004 CHNGETOOL 1
0005 CHANGEWORK 1
0006 END
0007
```

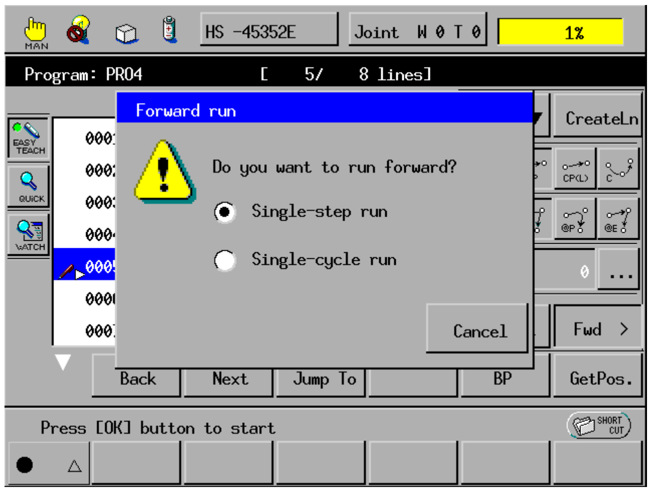
Displays the program.

Move the blue cursor to a program line to be executed, then press [<Back] or [Fwd>].

[Fwd>]: If a command on the selected line can be edited in the easy teaching window (refer to Section 3.5), then the program line will execute [Fwd>]: Pressing this button will execute the command selected with the blue cursor.

[<Back]: Pressing this button will execute the command selected with the blue cursor if the command does not include a destination point. If the command includes a destination point, it will apply a destination point defined in a motion command preceding the selected command.

Step 2 Pressing [<Back] or [Fwd.>] button will call up the execution confirmation window shown below.



Program: PR04 [5/ 8 lines]

Forward run

Do you want to run forward?

[✓ Single-step run]

Single-cycle run

Press [OK] button to start

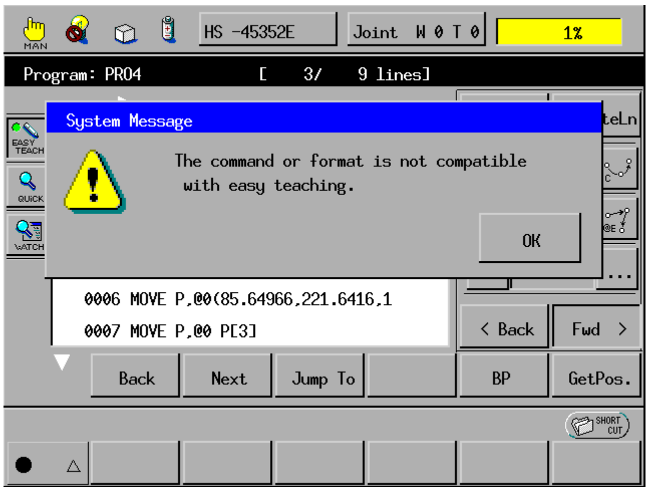
Choose "Single-step run" or "Single-cycle run." Then while holding down the deadman switch, press the OK button. The program will execute. Releasing either one of the deadman switch and OK button will stop the running program.

"Single-step run": Runs a single step of the selected program and stops the execution.

"Single-cycle run": Runs the selected program once from the beginning to the end or to any command not executable by easy teaching.

In either "Single-step run" or "Single-cycle run," you can stop the program run halfway just by performing any of stop operations (e.g., pressing the STOP key) during execution.

Tip If you attempt to run any command that is not executable by easy teaching or contains a syntax error, then the following warning message will appear.



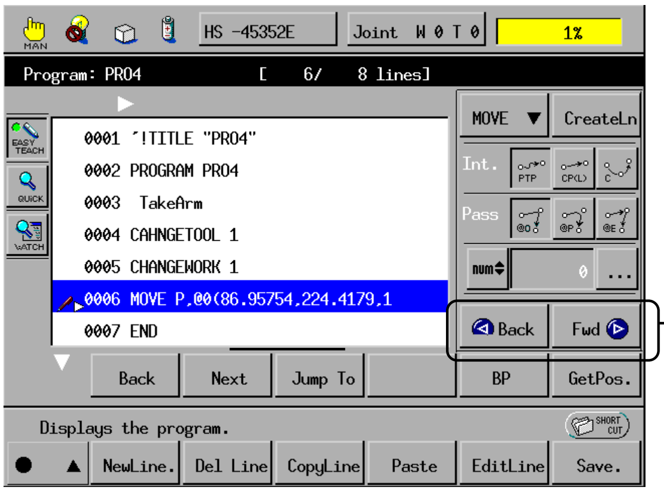
```
Program: PRO4              [    3/    9 lines]
```

The command or format is not compatible with easy teaching.

```
0006  MOVE P,@0(85.64966,221.6416,1
0007  MOVE P,@0 P[3]
```

Tip When the easy teaching window is displayed, pressing the deadman switch will change the functions assigned to the left- and right-arrow cursor keys from the horizontal scrolling on a program line to the same function as the [<Back] and [Fwd>], respectively.

The [<Back] and [Fwd>] icons will be also changed as shown below.



```
Program: PRO4              [    6/    8 lines]

0001  '!TITLE "PRO4"
0002  PROGRAM PRO4
0003   TakeArm
0004  CAHNGETOOL 1
0005  CHANGEWORK 1
0006  MOVE P,@0(86.95754,224.4179,1
0007  END
```

Displays the program.

Changed like this.

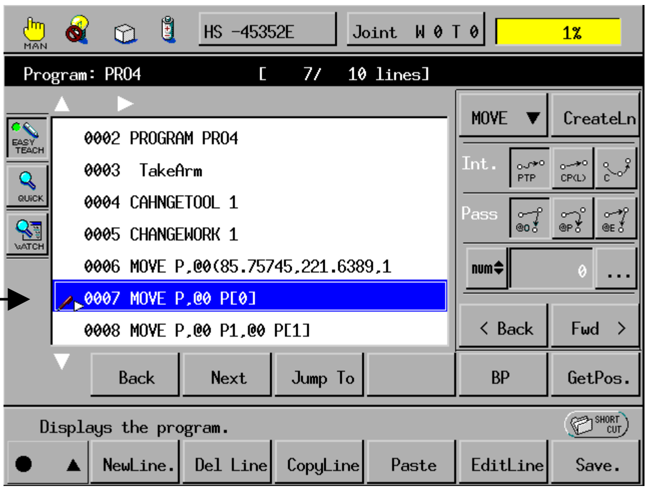
3.4 Additional Explanation about Easy Teaching

[1] Box frame on a program line

If you run a motion command by using the easy teaching window, a box frame may appear on a program line in addition to a blue cursor. The box frame indicates that the boxed line includes a destination point obtained by a command executed most recently.

In the case of [Fwd>]: If you temporarily stop the following program at line 0007 being executed by [Fwd>], for example, then a box frame comes on line 0007 since the destination point obtained by the most recently executed motion command is written in line 0007.

Box frame and blue cursor are overlapped with each other.

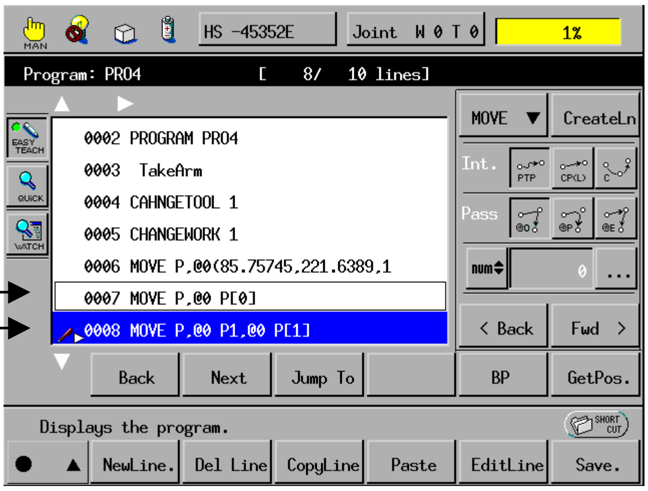


```
Program: PR04            [    7/   10 lines]
0002 PROGRAM PR04
0003  TakeArm
0004 CAHNGETOOL 1
0005 CHANGEWORK 1
0006 MOVE P,@0(85.75745,221.6389,1
0007 MOVE P,@0 P[0]
0008 MOVE P,@0 P1,@0 P[1]
Displays the program.
```

If line 0007 is executed again with [Fwd>] by a single step, the blue cursor moves to line 0008 but the box frame stays on 0007. This is because the destination point obtained by the most recently executed motion command is written in line 0007.

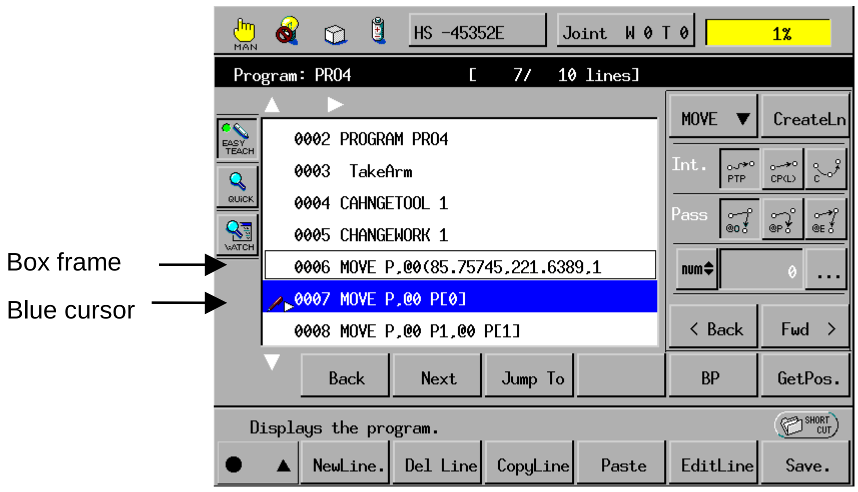
Box frame

Blue cursor



```
Program: PR04            [    8/   10 lines]
0002 PROGRAM PR04
0003  TakeArm
0004 CAHNGETOOL 1
0005 CHANGEWORK 1
0006 MOVE P,@0(85.75745,221.6389,1
0007 MOVE P,@0 P[0]
0008 MOVE P,@0 P1,@0 P[1]
Displays the program.
```

In the case of [<Back]: If line 0007 is executed with [<Back] by a single step (or the execution is stopped temporarily), then the blue cursor stays on line 0007 but the box frame comes on line 0006. This is because the destination point is still in the motion command immediately preceding the line marked with the blue cursor.



```
Program: PR04                [    7/    10 lines]
0002  PROGRAM PR04
0003   TakeArm
0004  CAHNGETOOL 1
0005  CHANGEWORK 1
0006  MOVE P,@0(85.75745,221.6389,1
0007  MOVE P,@0 P[0]
0008  MOVE P,@0 P1,@0 P[1]
Displays the program.
```

[2]   About the GetPos function

You may use the GetPos function when a coding list of the selected program or the easy teaching window is displayed.

The GetPos function may get the current robot position even expressed in immediate operand format provided that the easy teaching window is displayed.

Note: Immediate operand format refers to writing command operands in numeric values as shown below, not in variables.

$$\texttt{MOVE P,@0(}\underbrace{\texttt{1025.721,-354.7859,1026.708,-179.9987,65.01270,160.9215,5}}_{\text{Immediate operands}}\texttt{)}$$

3.5 Applicability of Parameters in Easy Teaching

The following tables show whether parameters of commands are applicable to editing, displaying of the easy teaching window, and execution in easy teaching.

NOTES

(1) If a command contains any parameter not listed in these tables, then no easy teaching window will be displayed. Neither is such a command executable.

(2) When an easy teaching window is displayed:

- Pressing the OK key will eliminate unnecessary space codes in parameters.
- A very small value entered to a destination position or initial passing start position may be automatically expressed in floating point or double-precision floating point format.

Symbols

O:	Operable & editable	G-var:	Global variable
OP:	Operable	L-var:	Local variable
D:	Displayable	Pos:	Position
NO:	Not operable or editable		
CPL:	Operable after compile		
FW:	Forward motion only		

Command	**Arguments**										
	Interpolation		Passing Start Offset		Pose (Destination position)***		Motion Options			Other Options	
MOVE	`P`	O	`@0`	O	Immediate	O	`SPEED`	Immediate	OP, D	`NEXT`	NO, D
	`L`	O	`@P`	O	G-var: Pos	O		G-var: Value	OP, D		
	`C`	O*	`@E`	O	L-var: Pos	NO		L-var: Value	NO		
			`@Numeral`	OP, D			`ACCEL`	Immediate	OP, D		
								G-var: Value	OP, D		
								L-var: Value	NO		
							`DECEL`	Immediate	OP, D		
								G-var: Value	OP, D		
								L-var: Value	NO		

	Interpolation		Reference Position		Passing Start Offset		Approach Distance		Motion Options			Other Option	
APPROACH	`P`	OP	Immediate	OP	`@0`	OP	Immediate	OP	`SPEED`	Immediate	OP	`NEXT`	NO
	`L`	OP	G-var: Pos	OP	`@P`	OP	G-var Value	OP		G-var Value	OP		
			L-var: Pos	NO	`@E`	OP	L-var: Value	NO		L-var: Value	NO		
					`@Numeral`	OP			`ACCEL`	Immediate	OP		
										G-var Value	OP		
										L-var: Value	NO		
									`DECEL`	Immediate	OP		
										G-var Value	OP		
										L-var: Value	NO		

	Interpolation		Passing Start Offset		Depart Distance		Operation Option			Other Options	
DEPART**	`P`	OP	`@0`	OP	Immediate	OP	`SPEED`	Immediate	OP	`NEXT`	NO
	`L`	OP	`@P`	OP	G-var Value	OP		G-var Value	OP		
			`@E`	OP	L-var: Value	NO		L-var: Value	NO		
			`@Numeral`	OP			`ACCEL`	Immediate	OP		
								G-var Value	OP		
								L-var: Value	NO		
							`DECEL`	Immediate	OP		
								G-var Value	OP		
								L-var: Value	NO		

Symbols

O:	Operable & editable	G-var:	Global variable
OP:	Operable	L-var:	Local variable
D:	Displayable	Pos:	Position
NO:	Not operable or editable		
CPL:	Operable after compile		
FW:	Forward motion only		

Command	Argument			
CHANGETOOL	Tool Coordinates			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		
CHANGEWORK	User Coordinates			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		
SPEED	Robot Speed			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		
JSPEED	Joint Speed			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		
ACCEL	Acceleration		Deceleration	
	Immediate	OP, FW	Immediate	OP, FW
	G-var: Value	OP, FW	G-var: Value	OP, FW
	L-var: Value	NO	L-var: Value	NO
JACCEL	Joint Acceleration		Joint Deceleration	
	Immediate	OP, FW	Immediate	OP, FW
	G-var: Value	OP, FW	G-var: Value	OP, FW
	L-var: Value	NO	L-var: Value	NO
DECEL	Deceleration			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		
JDECEL	Joint Speed			
	Immediate	OP, FW		
	G-var: Value	OP, FW		
	L-var: Value	NO		

Symbols

O:	Operable & editable	G-var:	Global variable
OP:	Operable	L-var:	Local variable
D:	Displayable	Pos:	Position
NO:	Not operable or editable		
CPL:	Operable after compile		
FW:	Forward motion only		

Command	Argument

Indirect reference range of "G-var: Value"

		Array		Array		
Numeric , global variable		Immediate	O			Example: P1,P[1] means that the variable number of position global variable P1 is 1.
		Numeric, global variable		Immediate	OP	Example: P[I1],P[I[1]] means the contents of global integer variable number 1 of global position variable P1.
				G-var	NO	
				L-var	NO	
		Numeric, local variable	NO			
		Array, local variable		Immediate	NO	
				G-var	NO	
				L-var	NO	
Numeric, local variable	NO					
Array, global variable		Immediate	NO			
		Numeric, global variable		Immediate	NO	
				G-var	NO	
				L-var	NO	
		Numeric, local variable	NO			
		Array, local variable		Immediate	NO	
				G-var	NO	
				L-var	NO	

Variable type

G-var: Pos	P, T, J
L-var: Pos	DEFPOS, DEFTRN, DEFJNT, DIM POSITION, DIM JOINT, DIM TRANS

Variable Type Global Variable	I, F, D
Value Variable Type Local Variable	DEFINT, DEFSNG, DEFDBL, %, !, #, DIM INTEGER, DIM SINGLE, DIM DOUBLE
Array Type Value Local Variable	

* If you stop the robot temporarily during an arc motion, you may restart it from the stopped position only in the same direction ([Fwd>] or [<Back] run) as before the temporary stop. Restart in the opposite direction is not allowed.

Once you attempt to restart the robot in the opposite direction, Error 27BE ("Arc motion not allowed from the current position") will occur. After that, restart is no longer possible even in the same direction.

No arc motion can start unless the current robot position is the same as a destination position defined in the immediately preceding motion command. If you attempt to do so, Error 27BE ("Arm motion not allowed from the current position") will occur.

** No DEPART motion can start unless the current robot position is the same as a destination position defined in the immediately preceding motion command. If you attempt to do so, Error 27BE ("Arm motion not allowed from the current position") will occur.

*** In easy teaching, you may edit or operate only a single pose in a same command line. The easy teaching facility cannot process a pose array (more than one pose in a line).

Invalid example: `MOVE P,P3,P4,P5` This command line includes three poses (P3, P4, and P5).

3.6 Notes for Easy Teaching

(1) At the start of a pass motion, the robot motion in easy teaching may be slightly different from that in Teach check, Auto, or External mode.

(2) If a figure flag is set to -1, the final robot figure may be different between [Fwd>] and [<Back] run.

(3) If any motion command is executed in easy teaching, then arm semaphore needed for that motion will be automatically obtained.

The arm semaphore will be released when:

- The robot arm has reached the final destination point after a single-step run or single-cycle run.
- The coding list of the selected program is closed.
- In the Shortcut Menu, the [F8 ProgRst.] has been performed.
- In Teach check, Auto, or External mode, the [F7 ProgRst.] has been performed, regardless of whether "This program only" or "All programs" is selected.
- A Level 3 error has occurred except errors caused when you are using TP/MP/OP.
- A Level 4 error or higher one has occurred.
- *Program reset* signal is received via I/O line in Standard mode (only when negative values are set in the data area).
- *Program reset* & *Operation preparation start* signals are received via I/O line in Compatible mode.

IMPORTANT

Once you temporarily stop the robot halfway through a single-step run or a single-cycle run, then manual operation or operation mode switching (between Joint mode, X-Y mode, and Tool mode) is no longer possible. This is because the currently running program has held an arm semaphore.

To make manual operation or operation mode switching possible, release the arm semaphore by carrying out [F8 ProgRst.] in the Shortcut Menu.

3.7 User Preferences for Using Easy Teaching (Add to the Configuration List)

The next table lists user preferences to be added for using easy teaching.

No.	Items	Factory default	Power-on default	Description	Comments
199	Positioning allowance of arc motion in easy teaching	100	100	Allowable positioning error for arc motion in easy teaching	Do not change this setting usually.

4. Operation Log Display Added

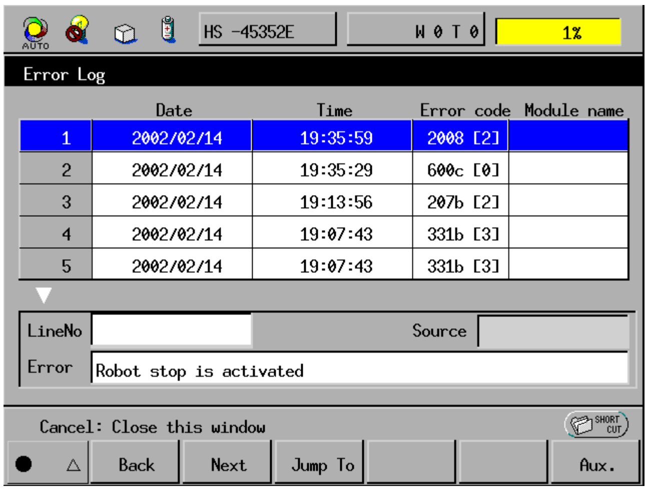
Refer to the SETTING-UP MANUAL, p. 5-133.

The operation log display is newly added to the logging feature of the teach pendant. This changes also the access to the conventional error log.

4.1 Access

```
Top screen—[F6 Set]—[F2 Log.]┬[F1 ErrLog]
                             └[F2 OpeLog]
```

Error Log Display



HS -45352E | W 0 T 0 | 1%

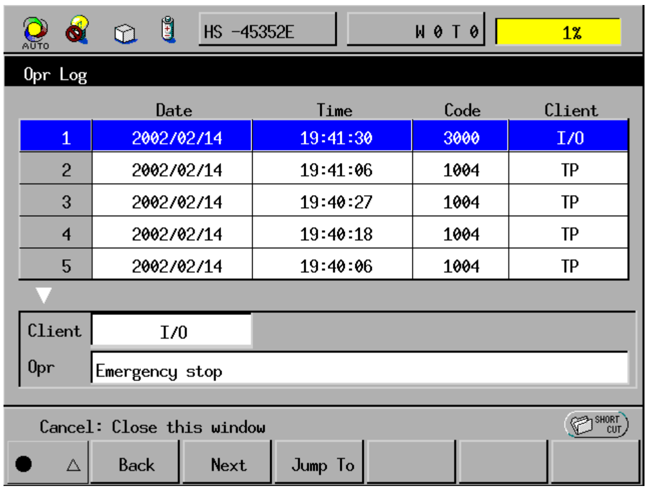
Error Log

	Date	Time	Error code	Module name
1	2002/02/14	19:35:59	2008 [2]	
2	2002/02/14	19:35:29	600c [0]	
3	2002/02/14	19:13:56	207b [2]	
4	2002/02/14	19:07:43	331b [3]	
5	2002/02/14	19:07:43	331b [3]	

LineNo: Source:
Error: Robot stop is activated

Cancel: Close this window

Operation Log Display



HS -45352E | W 0 T 0 | 1%

Opr Log

	Date	Time	Code	Client
1	2002/02/14	19:41:30	3000	I/O
2	2002/02/14	19:41:06	1004	TP
3	2002/02/14	19:40:27	1004	TP
4	2002/02/14	19:40:18	1004	TP
5	2002/02/14	19:40:06	1004	TP

Client: I/O
Opr: Emergency stop

Cancel: Close this window

4.2 Contents of the Operation Log Display

In the bottom line of the log list appears the operation details of the selected item number.

The Client column shows the operation sources that trigger the logged operations. The client may be any of the following:

TP:	Teach pendant
PC:	In WINCAPSII
I/O:	I/O
SYS:	Main system software

NOTE 1: Emergency stop is triggered by operating the teach pendant, but the client is treated as I/O.

NOTE 2: In some processing operations, the log information may be shown in such a format as [cnfPAC 9 val 1]. This means that the operation has changed the environmental settings. In the case of [cnfPAC9 val 1], it means that the 9th item of the PAC parameter table has been changed to value "1."

NOTE 3: The operation logging maintains a record of processing operations required for maintenance and does not maintain all operation records.

5. Direct Teaching Mode in the HS-E Series

Refer to the SETTING-UP MANUAL, page 5-78.

The HS-E series has no air balance cylinder on the Z axis, so the operation procedure for the direct teaching mode differs from that of the conventional 4-axis robots.

Entering the direct teaching mode

Access: [F2 Arm]—[F6 Aux.]—[F3 Direct.]

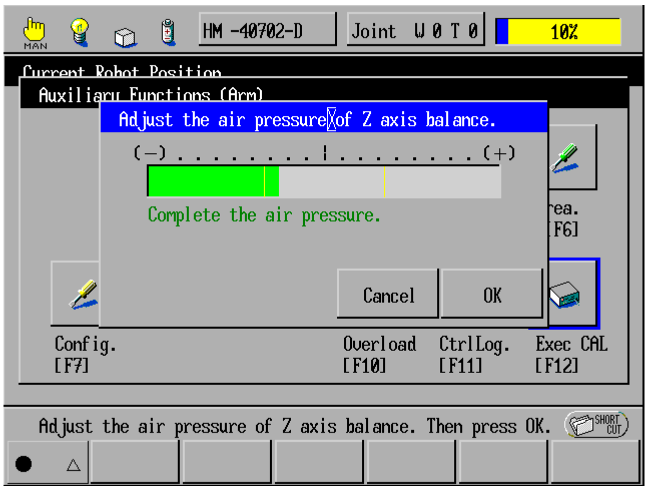
The direct teaching mode allows you to move the robot arm by hand (without using the teach pendant) with the motor OFF and teach the current position to a joint variable, position variable, or homogeneous transform matrix variable. (Usual teaching requires the motor to be turned ON.)

5.1 For Conventional 4-Axis Robots Except the HS-E Series

(1) In the Auxiliary Functions (Arm) window, press [F3 Direct.]. The air balance adjustment window will appear.

NOTE: The air pressure adjustment for Z-axis balance is required only when you make the robot enter the direct teaching mode at the first time after turning the robot controller ON.

(2) According to the instructions given on the screen, adjust the air pressure. If the "Complete the air pressure" is displayed as shown below, press the OK button.



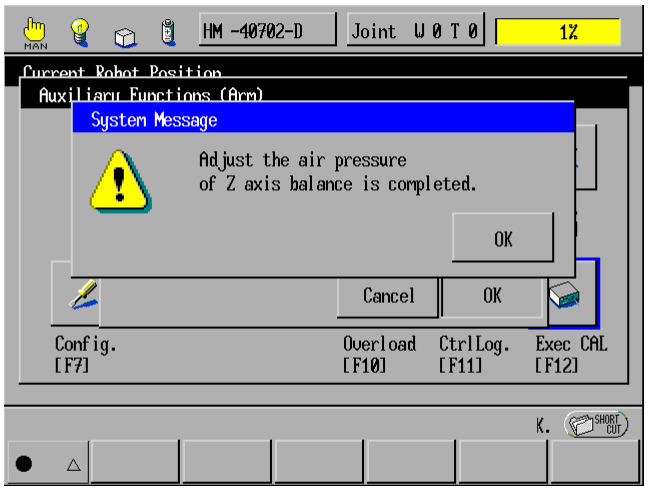
Adjust the air pressure of Z axis balance.

Complete the air pressure.

Adjust the air pressure of Z axis balance. Then press OK.

(3) The following window will appear.

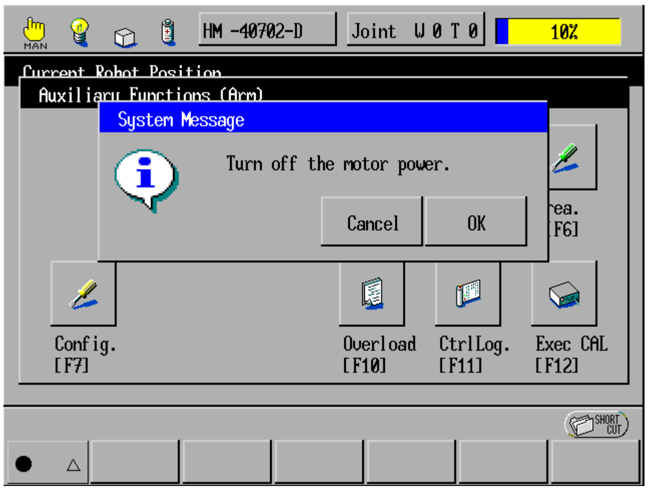
Press the OK button.


System Message

Adjust the air pressure
of Z axis balance is completed.

(4) The following window will appear.

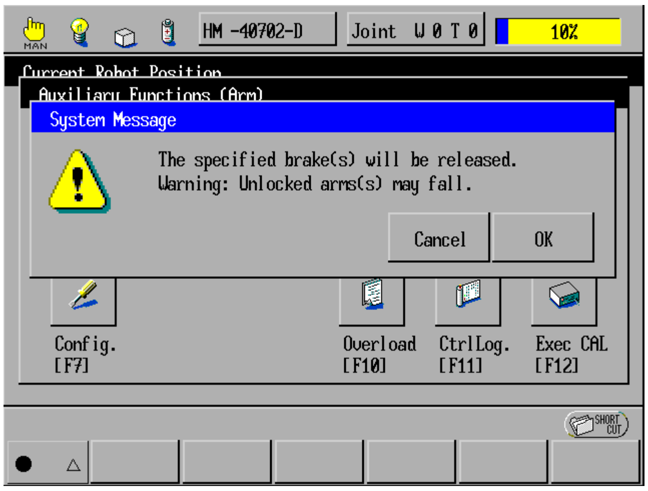
Turn the motor OFF and then press the OK button.


System Message

Turn off the motor power.

(5) The following window will appear.

Check the message and press the OK button.


System Message

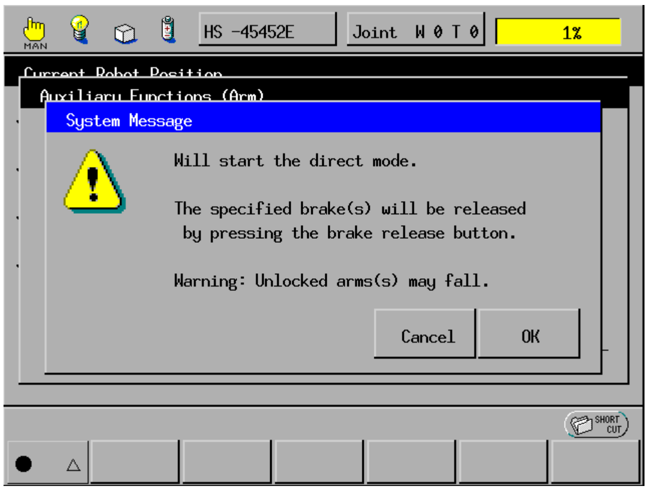
The specified brake(s) will be released.
Warning: Unlocked arms(s) may fall.

5.2 For the HS-E Series

(1) In the Auxiliary Functions (Arm) window, press [F3 Direct.].

The following message will appear.

(2) Hold down the brake release button provided on the top of the 2nd arm, and then press the OK button in this window. (In other robot series, only press the OK button.)



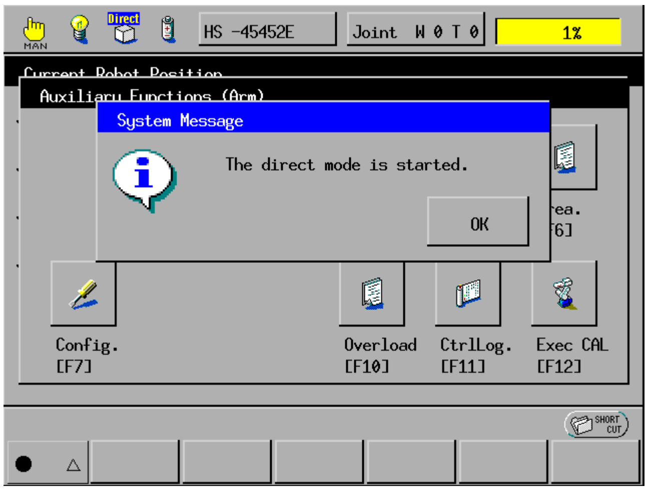
System Message

Will start the direct mode.

The specified brake(s) will be released by pressing the brake release button.

Warning: Unlocked arms(s) may fall.

(3) In the following start window, press the OK button.



System Message

The direct mode is started.

6. Serial Communication -- Ethernet Client and Server Functions Added --

Refer to the PROGRAMMER'S MANUAL, p. 2-8.

Version 1.9 allows you to add Ethernet Server and Client ports to the serial communication ports of the RC5 robot controller. What follows is the additional description to the serial communication.

6.1 Circuit Number

The table below lists the relationship between the circuit numbers and channel numbers assigned in the robot controller.

Circuit and Channel Number Assignment

Circuit number	Channel number	Used as:
0	ch1	Teach pendant port
1	ch2	Universal port
2	ch3	(Reserved)
3	ch4	(Reserved)
4 to 7	ch5 to ch8	Ethernet server ports
8 to 15	ch9 to ch16	Ethernet client ports (See **NOTE** below.)

NOTE: When using the Ethernet client port, you need to open or close a port using com_encom or com_discom statement explained in Section 7.2, "Serial Binary Transmission."

6.2 Communication Commands

In PAC language, the following four commands are used for communication through the RS232C or Ethernet interface. The command format is the same for both the RS232C and Ethernet.

PRINT statement (Output)
WRITE statement (Output)
INPUT statement (Input)
LINEINPUT statement (Line input)

The PRINT and WRITE statements differ in their output form. Character-strings outputted in the form of WRITE statement can be correctly inputted using the INPUT statement.

6.3 Clearing the Communication Buffer

The FLUSH statement clears the RS232C and Ethernet communication buffers.

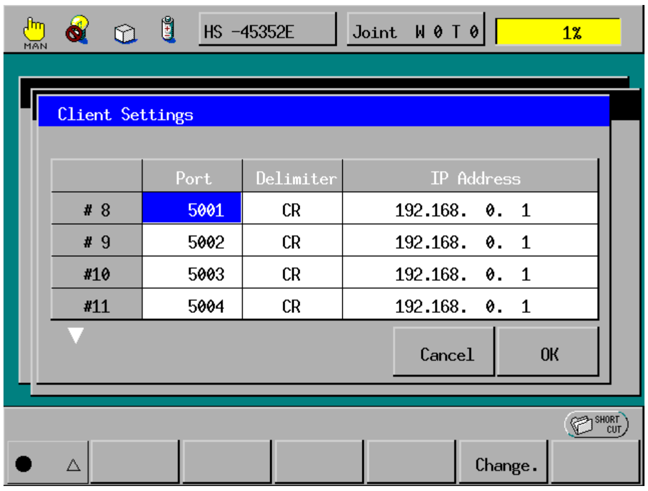
Be sure to clear those buffers before starting communication.

6.4 Setting the Client of the Robot Controller

Access: [F6 Set]—[F5 Set Com.]—[F10 Client]

Configures the client port of the robot controller.

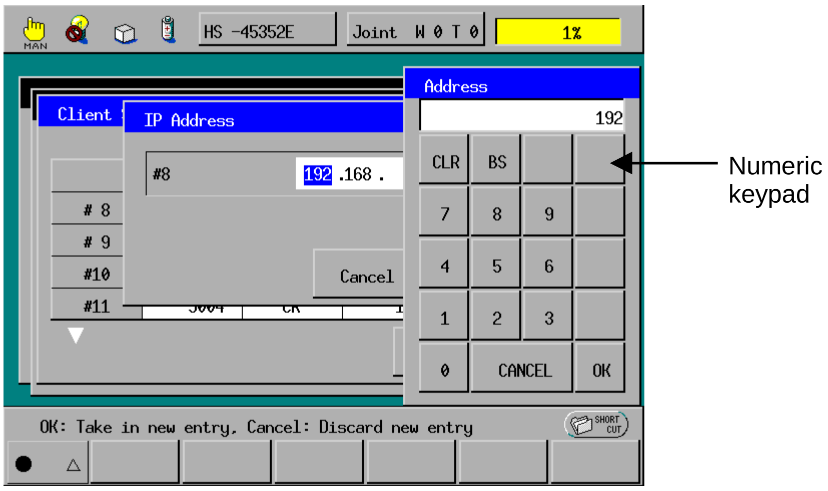
(1) In the Communications Setting Menu, press [F10 Client], and the Client Settings window will appear as shown below.



	Port	Delimiter	IP Address
# 8	5001	CR	192.168. 0. 1
# 9	5002	CR	192.168. 0. 1
#10	5003	CR	192.168. 0. 1
#11	5004	CR	192.168. 0. 1

(2) To set the IP address of the port, select the IP Address field to be set and press [F5 Change.] and then [F5 Edit].

The numeric keypad will appear as shown below.

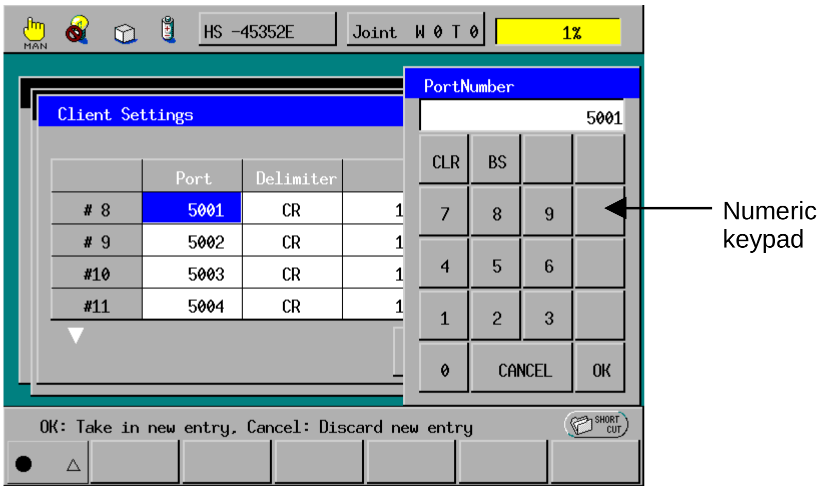


OK: Take in new entry, Cancel: Discard new entry

(3) Enter the desired value with the numerical buttons in the above window, and then press the OK button. The newly entered value will appear on the Client Settings window.

(4) To set the port number, select the Port field to be set and press [F5 Change.].

The numeric keypad will appear as shown below.



5001

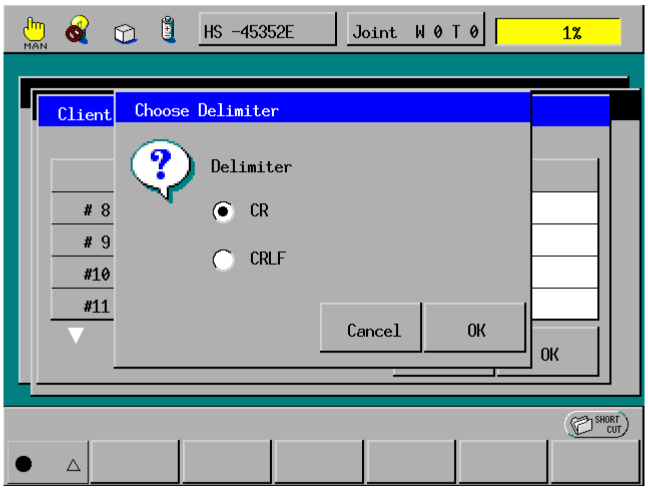
	Port	Delimiter	
# 8	5001	CR	1
# 9	5002	CR	1
#10	5003	CR	1
#11	5004	CR	1

OK: Take in new entry, Cancel: Discard new entry

(5) Enter the desired value to the target port with the numerical buttons in the above window, and then press the OK button. The newly entered value will appear on the Client Settings window.

(6) To set the delimiter, select the Delimiter field to be set and press [F5 Change.].

The Choose Delimiter window will appear as shown below.



[✓ CR] CRLF

(7) Select the desired delimiter code in the above window, and then press the OK button. The new delimiter code will appear on the Client Settings window.

(8) Check the new entry, then press the OK button to make the new entry go into effect.

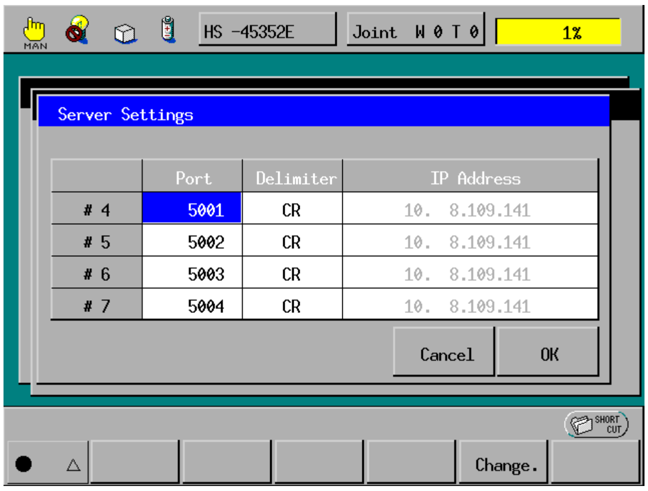
If you press the Cancel button instead of the OK button, the new entry will be cancelled. To close the Client Settings window, press the OK or Cancel button.

6.5 Setting the Server of the Robot Controller

Access: [F6 Set]—[F5 Set Com.]—[F11 Server]

Configures the server port of the robot controller.

(1) In the Communications Setting Menu, press [F11 Client], and the Server Settings window will appear as shown below.

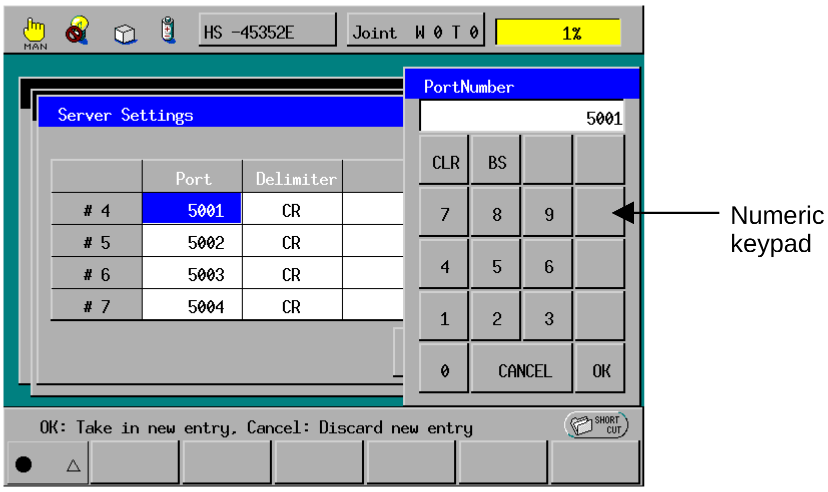


Server Settings

	Port	Delimiter	IP Address
# 4	5001	CR	10. 8.109.141
# 5	5002	CR	10. 8.109.141
# 6	5003	CR	10. 8.109.141
# 7	5004	CR	10. 8.109.141

(2) To set the port number, select the Port field to be set and press [F5 Change.].

The numeric keypad will appear as shown below.



Server Settings

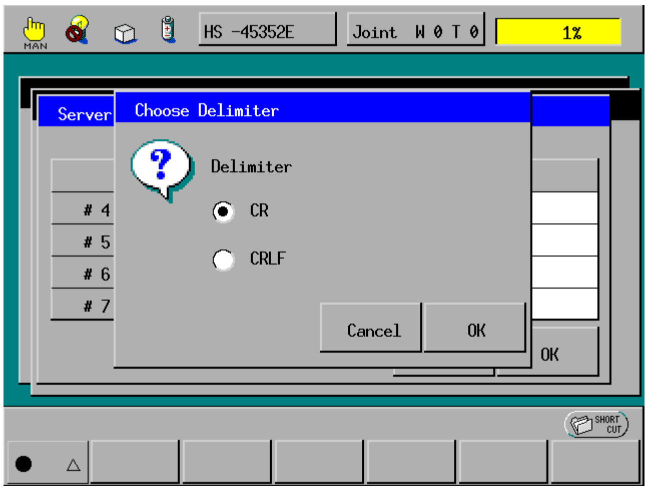
	Port	Delimiter
# 4	5001	CR
# 5	5002	CR
# 6	5003	CR
# 7	5004	CR

OK: Take in new entry, Cancel: Discard new entry

(3) Enter the desired value with the numerical buttons in the above window, and then press the OK button. The newly entered value will appear on the Server Settings window.

(4) To set the delimiter, select the Delimiter field to be set and press [F5 Change.].

The Choose Delimiter window will appear as shown below.



(5) Select the desired delimiter code in the above window, and then press the OK button. The new delimiter code will appear on the Server Settings window.

(6) Check the new entry, then press the OK button to make the new entry go into effect.

If you press the Cancel button instead of the OK button, the new entry will be cancelled. To close the Server Settings window, press the OK or Cancel button.

6.6 Command for RS232C and Ethernet (Server/Client) Port

INPUT (Statement) [Conforms to SLIM]

Function

Obtains data from the RS232C or Ethernet port.

Format

```
INPUT [#<Circuit number>,]<Variable name>[,<Variable name>...]
```

Explanation

This statement stores data received via the RS232C or Ethernet port into the variable specified by `<Variable name>`.

Designate a circuit number to use for <Circuit number>. If <Circuit number> is ignored the default value of ch2 is set. Ch1 cannot be designated because it is used for the pendant. (Refer to Section 6.1, "Circuit Number.")

If plural information is received for the same number of variables, add a comma (,) between each variable.

All information before the delimiter (CR or CR+LF) will be stored into the designated variables. No delimiters will be stored into variables.

Designate the baud rate using a system parameter.

Note (1) Execute a FLUSH command to clear the data remaining in the input buffer of the received data prior to receiving data.

Note (2) If input data exceeds the maximum value of the variable type to which it has been assigned, the result is the maximum value of that type.

Related Terms

FLUSH, PRINT, WRITE

Example

```
DIM li1 As Integer
DEFSTR ls1, ls2, ls3, ls4
INPUT #1, ls1             'Writes data from ch2 to ls1.
INPUT #1, li1, ls2, ls3   'Writes data from ch2 to li1, ls2, and ls3.
INPUT ls4                 'Writes data from ch2 to ls4.
```

LINEINPUT (Statement)

Function

Reads data to a delimiter through the RS232C or Ethernet port and assigns it to a string variable.

Syntax

```
LINEINPUT [#<Circuit number>,] <String variable name>
```

Explanation

All characters, from the data received via the RS232C or Ethernet port to a delimiter (CR or CR + LF), are stored into the variable designated by <Character string type variable name >. Delimiter is not stored into any variable.

For <Circuit number>, designate the circuit number of the RS232C or Ethernet port to be used. If <Circuit number> is ignored, the default value of ch2 is set. Ch1 cannot be designated since it is used for the pendant. (Refer to Section 6.1, "Circuit Number.")

Designate the name of the character string type variable for <Character string variable name>.

Example

```
DEFSTR ls1,ls2,ls3
LINEINPUT #0, ls1           'Receives a character string from port 1 and
                            'assigns it to ls1.
LINEINPUT #1, ls2           'Receives a character string from port 2 and
                            'assigns it to ls2.
LINEINPUT ls3               'Receives a character string from port 2 and
                            'assigns it to ls3.
```

PRINT (Statement) [Conforms to SLIM]

Function

Outputs data from the RS232C or Ethernet port .

Syntax

```
PRINT [#<Circuit number>,] <Message> [<Separator> <Message>...]
[<Separator>]
```

Explanation

This statement outputs the value of <Message> from the RS232C or Ethernet port to an external device.

For <Circuit number>, designate the circuit number of the RS232C or Ethernet port to be used. If <Circuit number> is ignored, the default value of ch2 is set. Ch1 cannot be designated since it is used for the pendant. (Refer to Section 6.1, "Circuit Number.")

A comma (,) or semicolon (;) can be used for <Separator>.

In the case of a comma (,), insert a blank character between the comma and <Message> that follows.

If a comma (,) is designated after <Message> but no <Message> is found, a delimiter (CR or CR+LF) is transferred after the final <Message> once output is finished.

In the case of a semicolon (;), no blank character should be inserted between the semicolon and <Message>.

If a semicolon (;) is designated after <Message> but no <Message> is found, a delimiter (CR or CR+LF) is transferred after the final <Message> and output is finished.

If pose type data is output, each element is separated for output using a blank character.

Related Terms

WRITE

Example

```
DEFINT li1, li2
DEFSTR ls1, ls2
PRINT #1, li1, ls1; ls2     'Outputs the values of li1, ls1, and ls2 from
                            'ch2.
                            '(A blank is made between li1 and ls1, and no
                            'blank is made between ls1 and ls2.)
PRINT li2                   'Outputs the value of li2 from ch2.
```

WRITE (Statement)

Function

Outputs data from the RS232C or Ethernet port .

Syntax

```
WRITE [#<Circuit number>,]<Message>[,<Message>...]
```

Explanation

This statement outputs the value of <Message> from the RS232C or Ethernet port to an external device.

For <Circuit number>, designate the circuit number of the RS232C or Ethernet port to be used. If <Circuit number> is ignored, the default value of ch2 is set. Ch1 cannot be designated since it is used for the pendant. (Refer to Section 6.1, "Circuit Number.")

If plural messages are written, a comma (,) must be used to separate the messages.

The following are the points that differ from the PRINT statement.

Character string data is output in double quotation marks (").

At the end of output information, a delimiter (CR or CR+LF) is added.

Commas (,) separating messages are output as they are.

If pose type data is output, each element is separated for output using a comma (,).

Related Terms

```
PRINT
Example
DEFINT li1, li2
DEFSTR ls1, ls2
WRITE #1, li1, ls1, ls2     'Outputs the values of li1, ls1, ls2 from ch2.
WRITE li2                   'Outputs the value of li2 from ch2.
```

FLUSH (Statement)

Function

Clears the input buffer.

Syntax

```
FLUSH [#<Circuit number>]
```

Explanation

This statement clears the input buffer of the RS232C or Ethernet port.

Designate a circuit number to clear the input buffer for <Circuit number>.

If <Circuit number> is ignored, the default value of ch2 is set. Ch1 cannot be designated since it is used for the pendant. (Refer to Section 6.1, "Circuit Number.")

Execute a FLUSH command to clear the data remaining in the input buffer prior to receiving data.

Related Terms

INPUT

Example

```
FLUSH #1                      'Clears the input buffer of the ch2 circuit.
FLUSH                         'Clears the input buffer of the ch2 circuit.
```

7. Serial Binary Transmission (RS-232C and Ethernet ports available)

In Version 1.4 or earlier, the robot controller can transmit only ASCII codes via the RS-232C ports. In Version 1.5 or later, it may support byte-controlled binary data transmissions, increasing the types of connectable communications devices.

In Version 1.743 or later, Ethernet ports are also available in serial binary transmission as well as RS-232C ports.

7.1 Using Serial Binary Transmission

Connect the robot controller to external equipment with an RS-232C or Ethernet interface cable and use the following commands:

■ Data input/output commands

(1)	`printb`	Output a single byte of data.
(2)	`inputb`	Input a single byte of data.
(3)	`lprintb`	Output multiple bytes of data.
(4)	`linputb`	Input multiple bytes of data.
(5)	`com_encom`	Enable COM port.
(6)	`com_discom`	Disable COM port.
(7)	`com_state`	Get COM or Ethernet port status.

NOTE: The `com_encom`, `com_discom`, and `com`_state are functionally extended for supporting Ethernet, but their conventional functions and syntaxes are not changed.

■ Transmission system

(1)	Transmission method	:	RS-232C
(2)	Transmission port	:	Controller RS-232C port, µVision board RS-232C port
(3)	Pin array	:	Same as conventional
(4)	Transmission status	:	

Controller RS-232C port	Variable, same as the previous controllers
µVision board RS-232C port	Transmission speed : 9600bps Bit length : 8 Parity bit : None Stop bit : 1 Flow control : None

7.2 Serial Binary Transmission Commands (RS232C and Ethernet ports)

printb

Function

Outputs a single byte of data to the RS-232C or Ethernet port.

Syntax

```
printb #<portnumber>,<integervarnumber>
```

`<portnumber>`	Output port number (1: Controller RS-232C port, -1: µVision RS-232C port, 4 to 7: Ethernet server ports, 8 to 15: Ethernet client ports)
`<integervarnumber>`	Integer variable number where output data is stored

Explanation

This command outputs the lower byte of data assigned to `<integervarnumber>` to the port specified by `<portnumber>`.

Example

```
'!TITLE "<Title>"
PROGRAM sample
          .
          .
          .
          printb #1,I10    'Output the lower byte of data stored in I10
                           'to RS-232C port
          .
          .
          .
end
```

inputb

Function

Inputs a single byte of data from the RS-232C or Ethernet port.

Syntax

```
inputb #<portnumber>,<integervarnumber>
```

`<portnumber>`	Input port number (1: Controller RS-232C port, -1: µVision RS-232C port, 4 to 7: Ethernet server ports, 8 to 15: Ethernet client ports)
`<integervarnumber>`	Integer variable number where input data is to be stored

Explanation

This command stores a single byte of data inputted from the specified port, into `<integervarnumber>`

Note: If no data exists in the specified port, executing this command will result in an error. Before execution of this command, transfer data from external equipment. To check whether any data exists or not, use com_state command.

Example

```
'!TITLE "<Title>"
PROGRAM sample
        .
        .
        .
        inputb #1,I10     'Store data inputted from RS-232C port into
                          'I10
        .
        .
        .
end
```

lprintb

Function

Outputs multiple bytes of data to the RS-232C or Ethernet port.

Syntax

```
lprintb #<portnumber>,<arrayheadelement>,<outputbytes>
```

`<portnumber>`	Output port number (1: Controller RS-232C port, -1: µVision RS-232C port, 4 to 7: Ethernet server ports, 8 to 15: Ethernet client ports)
`<arrayheadelement>`	Head element of an array where output data is stored
`<outputbytes>`	Number of bytes to be outputted

Explanation

This command outputs the specified number of bytes of data from the specified element of an array where the output data is stored, to the specified port.

Example

```
'!TITLE "<Title>"
PROGRAM sample
      .
      .
      .
      lprintb #1,I64,30    'Output the lower byte of data from I64 to I93
                           'in succession to RS-232C port.
      .
      .
      .
end
```

linputb

Function

Inputs multiple bytes of data from the RS-232C or Ethernet port.

Syntax

```
linputb #<portnumber>,<arrayheadelement>,<inputbytes>
```

`<portnumber>`	Input port number (1: Controller RS-232C port, -1: µVision RS-232C port, 4 to 7: Ethernet server ports, 8 to 15: Ethernet client ports)
`<arrayheadelement>`	Head element of an array where input data is to be stored
`<inputbytes>`	Number of bytes to be inputted

Explanation

This command inputs the specified number of bytes of data from the specified element of an array, to the specified port.

Note: If no data exists in the specified port, executing this command will result in an error. Before execution of this command, transfer data from external equipment. To check whether any data exists or not, use com_state command.

Example

```
'!TITLE "<Title>"
PROGRAM sample
       .
       .
       .
       linputb #1,I64,30    'Input data from I64 to I93 in succession from
                            'RS-232C port.
       .
       .
       .
end
```

com_encom

Function

Enables the RS-232C port only for binary transmission. (Occupies the COM port.)

If the Ethernet client port has been occupied, this command enables it to establish Ethernet connection.

Syntax

```
com_encom #<portnumber>
```

Explanation

This command discriminates binary data from ASCII data in binary transmission between the PC and robot controller e.g., in WINCAPSII.

After binary transmission is completed, you need to release the communication port by using `com_discom` command.

NOTE: If data is transferred from the PC (e.g., in WINCAPSII) to the robot controller after execution of this command, then the controller will treat it as binary data and will not close the RS-232C port occupied by binary transmission.

NOTE: After executing this command in Ethernet connection, it will not influence data communications with WINCAPSII.

Example

```
'!TITLE "<Title>"
PROGRAM sample
       .
       .
       .
       com_encom #1        'Make port exclusive for binary transmission
       .
       .
       .
end
```

com_discom

Function

Releases the RS-232C port from binary transmission. (Releases the COM port.)

If the Ethernet client port has been used, this command disables the Ethernet client port to disconnect the Ethernet connection.

Syntax

```
com_discom #<portnumber>
```

Explanation

This command disables `com_encom` command to release the RS-232C dedicated to binary transmission for other uses.

NOTE: Executing this command clears the RS-232C port once for preventing data confusion between ASCII and binary data.

NOTE: If this command is executed during Ethernet communication, the connection will be disconnected without waiting for completion of transmission. The receiver may keep waiting for receiving non-transmitted data.

Example

```
'!TITLE "<Title>"
PROGRAM sample
          .
          .
          .
          com_discom #1     'Release port from binary transmission
          .
          .
          .
end
```

com_state

Function

Gets the status of RS-232C or Ethernet port.

Syntax

```
com_state #<portnumber>,<integervar>
```

Explanation

This command gets bytes of data remaining in the transmission buffer, into the integer variable specified by `<integervar>`.

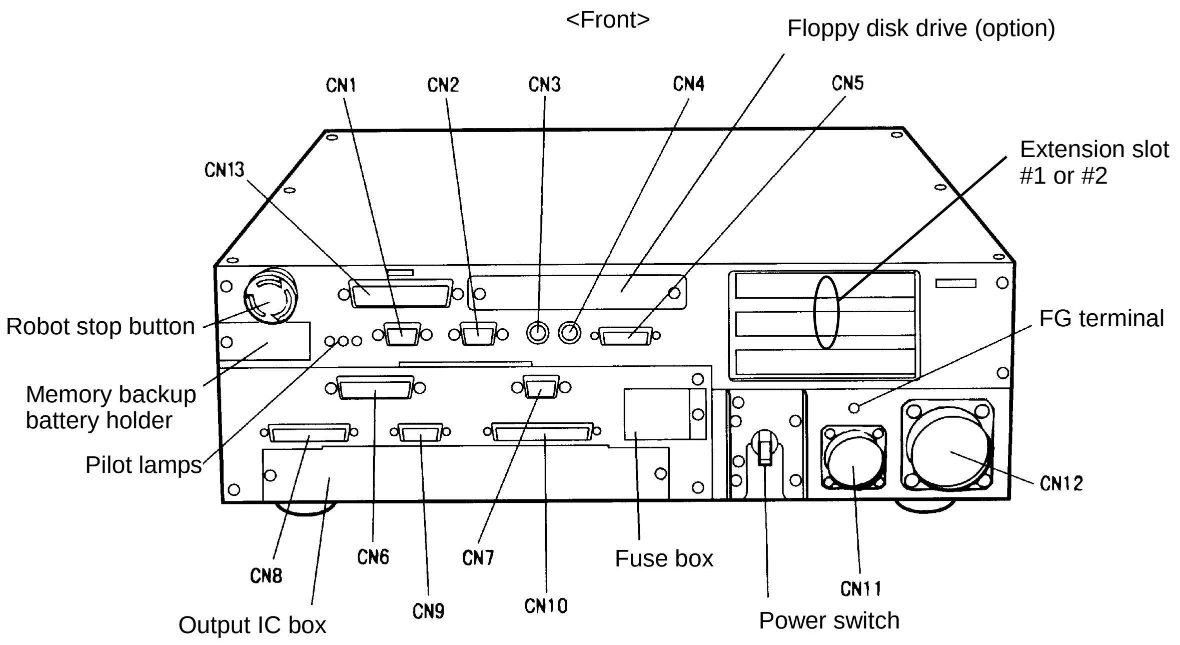
Note that -1 will be returned if a transmission port error occurs. At the time of Ethernet use, -1 will be also returned if network connection of the port is not established.

Example

```
'!TITLE "<Title>"
PROGRAM sample
       .
       .
       .
       com_state #1,I280    'Gets data remaining in transmission buffer
                            'into I280
       .
       .
       .
end
```

8. Configuring the RS-232C Extension Board (Recommended Option)

If you install an RS-232C extension board to the robot controller, the controller may support three RS232C serial data transmission lines (One standard line plus two add-on lines). The RS-232C should be set into extension slot #1 or #2.



8.1 Recommended RS-232C Extension Board

Set up an RS-232C extension board specified below in your charge.

Model	COM-2(PC)F
Manufactured by	CONTEC

NOTE: To support an RS-232C extension board, the robot controller requires some special features to be built in at the factory. When placing an order for the robot controller, specify the RS-232C extension board support.

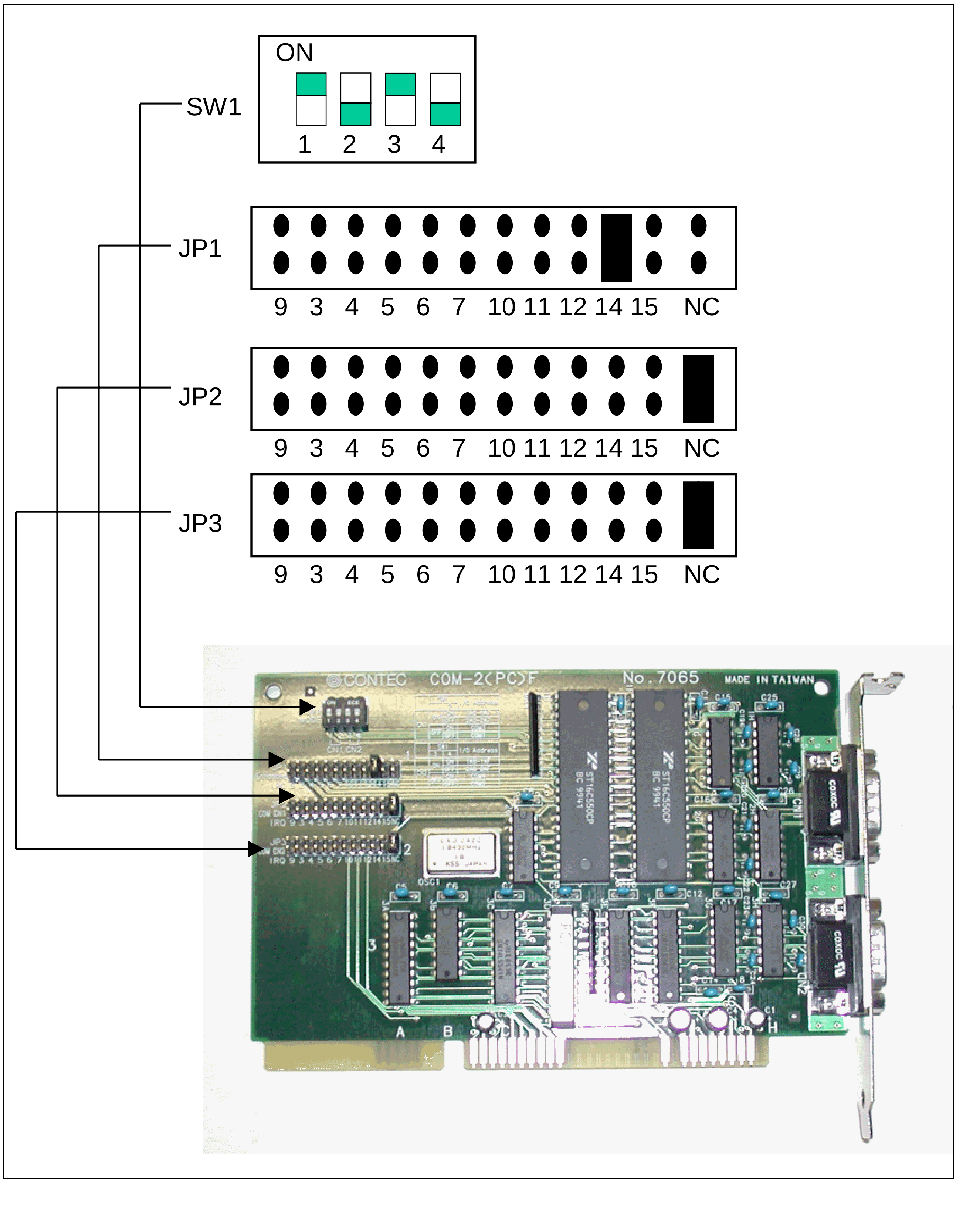
8.2 Installing the Extension Board

For the installation procedure, refer to the OPTIONS MANUAL, Chapter 11, "Mounting Extension Boards."

8.3 Setting the Jumpers and DIP Switch on the RS-232C Extension Board

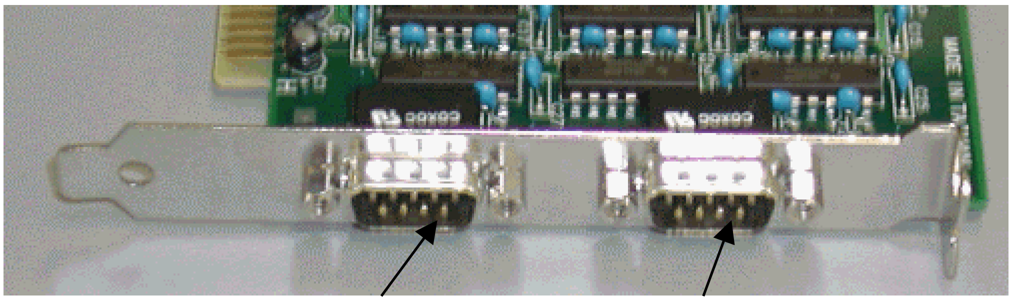
Set the jumpers and DIP switch on the RS-232C extension board as shown below.

Jumper/DIP SW	SW1	JP1	JP2	JP3
Settings	Set selectors 1 and 3 to ON.	Set a jumper cap onto pin 14.	Set a jumper cap onto NC.	Seta a jumper cap onto NC.


ON
1 2 3 4
SW1
JP1
9 3 4 5 6 7 10 11 12 14 15 NC
JP2
9 3 4 5 6 7 10 11 12 14 15 NC
JP3
9 3 4 5 6 7 10 11 12 14 15 NC
©CONTEC COM-2(PC)F
No.7065
MADE IN TAIWAN

8.4 RS-232C Extended Serial Ports and Line Number Assignment

The RS-232C extension board features two COM ports--COM3 and COM4. Two serial data transmission lines #2 and #3 are assigned to COM3 and COM4, respectively.



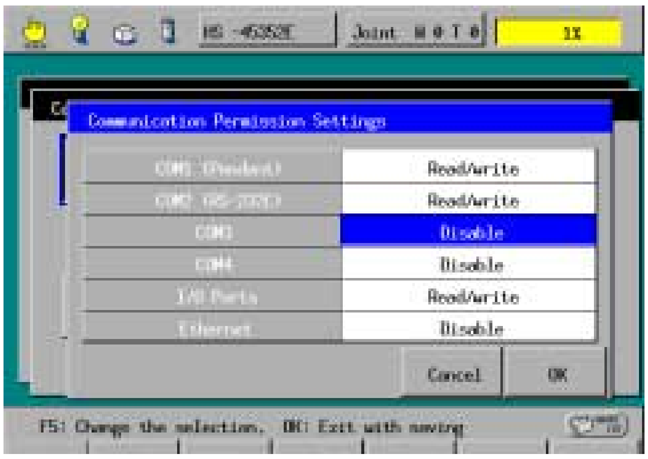
8.5 Communications Configuration of RS-232C Extension Board

Follow the procedure described below to configure communications feature of COM3 and COM4 on the RS-232C extension board.

■ Setting the communication permission

Access: [F6: Set]—[F5: Set Com.]—[F1 Permit.]

CAUTION: COM3 and COM4 do not support data transmission with WINCAPSII. Keep both of those ports "Disable" (Default).



Communication Permission Settings

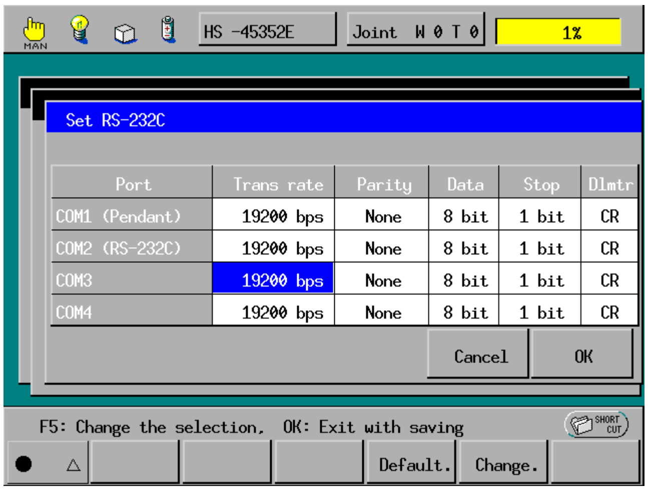
COM1 (Pendant)	Read/write
COM2 (RS-232C)	Read/write
COM3	Disable
COM4	Disable
I/O Ports	Read/write
Ethernet	Disable

F5: Change the selection. OK: Exit with saving

■ Setting the transmission rate for RS-232C serial interface ports

Access: [F6: Set]—[F5: Set Com.]—[F2 Serial IF]

Select each of the COM3 and COM4 and then press [F5 Change.] to the transmission rate, parity (None, Odd or Even) and other values.



Set RS-232C

Port	Trans rate	Parity	Data	Stop	Dlmtr
COM1 (Pendant)	19200 bps	None	8 bit	1 bit	CR
COM2 (RS-232C)	19200 bps	None	8 bit	1 bit	CR
COM3	19200 bps	None	8 bit	1 bit	CR
COM4	19200 bps	None	8 bit	1 bit	CR

F5: Change the selection, OK: Exit with saving

NOTE: The default transmission speed for the RS-232C extension board is 19,200 bps. The maximum transmission speed is 38,400 bps.

If the transmission speed is set to 38,400 bps, however, a communications failure may occur frequently. Even at 19,200 bps, a communications failure may also occur due to electric noises or other interference.

In programming, therefore, you may need to use the `com_state` command for setting retry capabilities as shown in the coding sample below.

8.6 Coding Sample for Transmission Error Recovery

```
'!TITLE "<Title>"
PROGRAM sample
            .
            .
            .
            DEFPOS lp1(10)       'Local position variable.
            DEFINT li1           'Local integer variable.
            ,
            .
            .
            li1 = 0              'Initialize li1.
            .
            .
            .
            WHILE li1 < 10       'Repeat pre-decision.
            .
            .
            .
            INPUT #2,lp1(li1)    'Get data on line #2 into
                                 'li1(li1).
            com_state #2,I280    'Get communication status into I280.
            IF I280 < 0 THEN     'If an error occurs, the value is –1.
            PRINT #2,"R"         'Output retry instruction.
            ELSE
            PRINT #2,"A"         'Output "normal receive".
            li1 = li1 + 1
            END IF
            .
            .
            .
            .
            WEND                 'Repeat 10 times.
            End
```

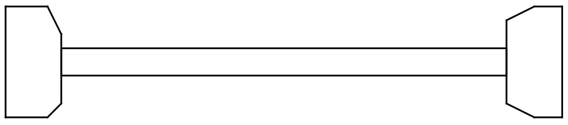
In the coding sample above,
It is assumed that "R" is a retry command that requires the external equipment to make retry operation and "A" is an acknowledge command for normal data reception.

8.7 Limited Warranty

DENSO WAVE provides the user with the communications function built in the controller for using the RS-232C extension board. It does not give you any warranty or technical support for the extension board itself.

9. Pendant Extension Cable (Option)

An optional pendant extension cable has been prepared for the teach pendant, mini-pendant, and operating panel.

Cable Name (Length)	Parts #	Remarks
Pendant extension cable (4m)	410141-2390	
Pendant extension cable (4 m)	410141-2400	

Notes for using the extension cable

(1) The total cable length including an expansion cable should be 12 m or less.

(2) More than one extension cable should not be connected to the teach pendant, mini-pendant, or operating panel.

(3) The relay connector of an extension cable should be protected from any kind of stress or impact.

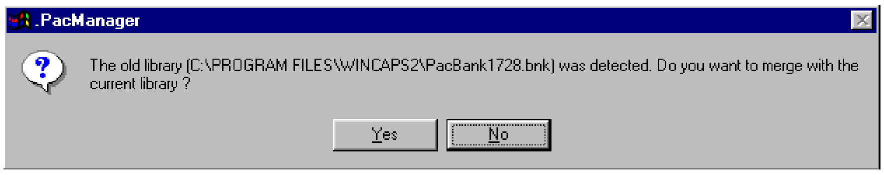
10. What's New in WINCAPSII

WINCAPSII Version 1.9 newly supports the following features.

10.1 Importing the Program Bank Related Contents from the Old Version into the Updated Version

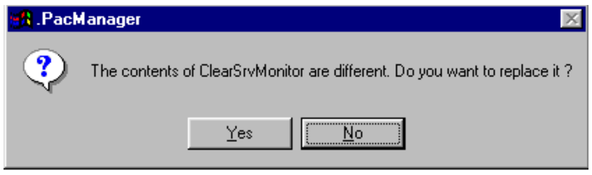
When you update WINCAPSII to a newer version, you need to import program bank related contents customized in the old version into the newer one. WINCAPSII Version 1.9 or later allows you to do it easily by providing the Auto and Manual importing facilities.

If you use either of those facilities, WINCAPSII will unconditionally import class libraries added or customized in the old version into the current program bank.



The old library (C:\PROGRAM FILES\WINCAPS2\PacBank1728.bnk) was detected. Do you want to merge with the current library ?

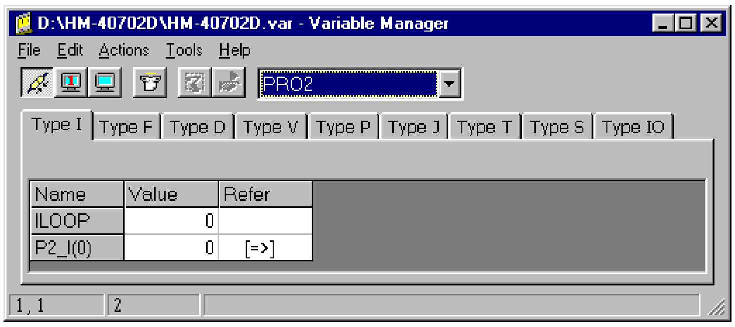
If the importing facility finds a class library that has the same name as one already stored in the current program bank but has different contents, then the following message appears, prompting you to confirm overwriting.



The contents of ClearSrvMonitor are different. Do you want to replace it ?

10.2 Monitoring and Modifying Local Variables

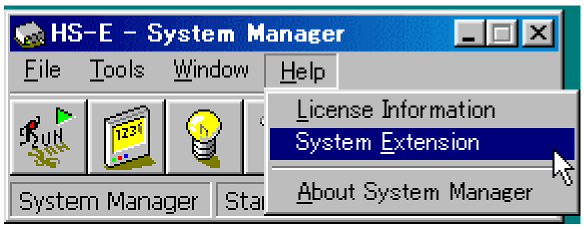
If WINCAPSII is online with the robot controller, you may monitor and modify local variables in each program by switching the scope of variables.



Name	Value	Refer
ILOOP	0	
P2_I(0)	0	[=>]

10.3 Accessing the System Extension in the Robot Controller

Even in WINCAPSII Version 1.9, you may register or delete optional features that the robot controller will use.



11. Commands Added

This section describes newly added commands (statements) that have been used as servo-related PAC libraries. Using these commands will improve the efficiency of program development.

ST_aspACLD (Statement)

Function

Changes the internal load condition values. There are the mass of payload, noted in grams (g), and the payload center of gravity, noted in millimeters (mm), for the load condition values. Designate both of them.

Syntax

```
ST_aspACLD <Mass of payload>, <Payload center of gravity coordinate X >,
<Payload center of gravity coordinate Y>, <Payload center of gravity
coordinate Z>
```

Explanation

The mass of payload is the mass of load (tool and workpiece) mounted on the 6th axis of the robot. This unit is designated as (g).

For the load center of gravity position, designate the payload center of gravity using the TOOL0 coordinates. The unit is millimeters (mm).

The reference position of the TOOL0 coordinates is in the center of the 6th axis flange. For component Y, the direction is from the flange center to the pinhole of ϕ5H7 (orientation vector direction). For component Z, the direction is vertical to the flange surface through the flange center (approach vector direction). For component X, the direction of the X axis (normal vector direction) is the right-hand coordinate system, when the orientation vector is set to the Y axis and the approach vector is set to the Z axis. Refer to "4.7 Control Set of Motion Optimization in User Preferences."

Even if you change only one of the four values of the mass of payload, the payload center of gravity X6, the payload center of gravity Y6, and payload center of gravity Z6 ,describe all of the 4 values again.

It takes about 0.1 sec. to switch the load condition values. Frequently switching the load condition may cause operational delays. Do not change the mode during pass motion while near an obstacle because the path locus may shift. This may delay switching if you change the load condition values.

Related Terms

Refer to the Programmer's Manual, " 4.7 Control Set of Motion Optimization in User Preferences."

Example

```
ST_aspACLD 8500,-50,100,80    'Sets the internal payload conditions.
                              'Mass of payload:8500(g), Payload center of
                              'gravity component X: -50(mm), component Y:
                              '100(mm),component Z: 80(mm)
```

Notes

(1) For the mass of payload, designate it with a numerical value of the specified range for each robot type. If you designate a value out of this range, the error message "60d2 Mass of payload out of setting range" will be displayed.

(2) For the payload center of gravity, enter it so that it satisfies the specified range for each robot type. If it is out of this range, the error message "60d2 Mass of payload out of setting range" will be displayed.

(3) When setting the internal payload condition, observe the following rule relative to the external payload condition. If not, the error message "60d2 Mass of payload out of setting range" will be displayed.

0.5 x External payload condition ≤ Internal payload condition ≤ External payload condition

ST_aspChange (Statement)

Function

Selects the internal mode for proper control setting of motion optimization.

Syntax

```
ST_aspChange <Mode>
```

Explanation

This statement switches the mode for control setting of motion optimization.

<Setting value>

0 → Invalid
1 → Valid only for PTP
2 → Valid only for CP
3 → Valid for both PTP and CP.

It takes about 0.1 sec. to switch the load condition values. Frequently switching the load condition may cause operational delays. Do not change the mode during pass motion, near an obstacle because the path locus may shift.

This may delay switching if you change the load condition values.

Related Terms

Refer to the Programmer's Manual, " 4.7 Control Set of Motion Optimization in User Preferences."

Example

```
ST_aspChange 1            'Sets the internal mode in the control sets of
                          'motion optimization to 1.
```

Notes

For <Mode>, designate it with a numerical value between 0 and 3. If it is out of this range, the error message "6003 Excess in effective value range" will be displayed.

ST_SetGravity (Statement)

Function

Compensates for the static load (gravity torque) applied to each joint and attains balance with gravity torque.

Syntax

```
ST_SetGravity
```

Explanation

Each joint of the robot undergoes downward static load (gravity torque) due to earth gravity. The effects of the gravity torque will vary depending upon the mass of payload (tool and workpiece), the payload center of gravity, and robot figures.

If you limit the motor output torque by setting its drive current limit so that the limited torque becomes lower than the gravity torque, then the robot will move down towards the earth. To prevent it, this statement compensates the limited torque for the gravity torque, keeping the balance of torque.

Related Terms

ST_SetCurLmt, ST_ResetGravity, ST_SetGrvOffset

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm-semaphore. If this command is executed without arm-semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) Set the mass of payload and the payload center of gravity accurately. Otherwise, the robot may move down due to gravity if you set a low current limit value (e.g., less than 30). For the entry procedure of the mass of payload and the payload center of gravity, refer to the PROGRAMMER'S MANUAL "4.7 Setting the Master Control Parameters in User Preferences."

(3) f you do not know the accurate mass of payload or its center of gravity or if the robot moves down in spite of accurate settings, then use the gravity offset function (ST_SetGrvOffset) that compensates for the gravity compensation value.

(4) If you set the gravity offset setting to "1" on the teach pendant, the gravity offset function becomes enabled. The setting made on the teach pendant will take effect immediately following the completion of calibration after the robot controller is turned on.

Example

```
ST_SetGravity            'Enable the gravity compensation function.
Delay 100                'Wait for gravity compensation to go into
                         'effect.
ST_SetCurLmt 2,30        'Set the current limit value of the 2nd axis
                         'to 30%.
```

ST_ResetGravity (Statement)

Function

Disables the balance setting between the limited motor torque and gravity torque, which is made with `ST_SetGravity`.

Syntax

```
ST_ResetGravity
```

Explanation

This command disables the balance setting between the motor torque limited by the current limit function and gravity torque.

Related Terms

ST_ResetCurLmt, ST_SetEralw

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm-semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) If this command is executed when the current limit function is enabled, Error "665B Cannot disable the gravity compensation" will result. Disable the current limit function and then try it again.

(3) If you set the gravity offset setting to "0" on the teach pendant, the gravity offset function becomes disabled. If you do it when the current limit function is enabled, Error 665b will result, as in step (2).

Example

```
ST_ResetGravity
```

ST_SetGrvOffset (Statement)

Function

Compensates the torque of each joint programmed with `ST_SetGravity` for gravity torque.

Syntax

```
ST_SetGrvOffset
```

Explanation

Each joint of the robot undergoes downward static load (gravity torque) due to earth gravity. Although gravity compensation command `ST_SetGravity` allows you to adjust the balance between the limited torque and gravity torque, the balance may be off-balance due to the difference between the mass of payload you set and the actual one.

This offset function presumes the gravity torque when the robot is on halt and calculates the gravity offset value.

Related Terms

ST_SetCurLmt, ST_SetGravity, ST_ResetGrvOffset

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This command should be executed when the motor power is on and the robot is on halt. If it is executed when the motor power is off, Error "6006 Motor power is off" will result. If it is executed when the robot is in motion, Error "600B Robot is running" will result.

(3) If the robot attitude is greatly changed after execution of this command, execute this command again.

(4) If the current limit reset value in User Preferences is set to any value other than "1", "3", "5", or "7", the compensation value will be reset to "0" when you turn on the motor power.

Example

```
ST_SetGrvOffset
```

ST_ResetGrvOffset (Statement)

Function

Disables the gravity offset function.

Syntax

```
ST_ResetGrvOffset
```

Explanation

Disables the gravity offset function which has been enabled with `ST_SetGrvOffset`.

Related Terms

ST_SetGrvOffset

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore.

If this command is executed without arm semaphore obtained, Error 21F7 "Arm semaphore cannot be fetched." will result.

(2) This command should be executed when the robot is on halt. If it is executed when the robot is in motion, Error "600B Robot is running" will result. This command is executable even when the motor power is off.

Example

```
ST_ResetGrvOffset
```

ST_SetCurLmt (Statement)

Function

Sets the limit of motor current to be applied to the specified axis.

Syntax

```
ST_SetCurLmt <AxisNumber>, <Value>
```

Explanation

Limits the value of motor current (torque) to be applied to the axis specified by `<AxisNumber>` to the value specified by `<Value>`. This command is useful when you want to limit torque that a workpiece will undergo during insertion or butting jobs.

The maximum value of `<Value>` is 100 which refers to the motor rating current.

If any value exceeding the allowable limit for each axis is specified, the value will be automatically limited to that allowable limit.

Set a value of 1 or above. If 0 or a negative number is set, Error "6003 Excess in effective value range" will result.

Related Terms

ST_ResetCurLmt, ST_SetGravity, ST_SetGrvOffset, ST_SetEralw

Notes

(1) When the motor current is limited with ST_SetCurLmt, the robot cannot move at the maximum speed or acceleration. Use ST_SetCurLmt only at steps that need the current limit. When using ST_SetCurLmt, decrease the acceleration.

(2) If a workpiece bumps against something at high speed even if the driving force is controlled by limiting the motor current, the impact is considerable due to the inertia of the workpiece, end-effector and axis. Set the current limit just before the workpiece comes into contact with the object and reduce the speed.

(3) Set the current limit when the robot is on halt. If it is set during a pass motion, an error is likely to occur.

(4) Execute this command in a TAKEARMed task that has obtained arm-semaphore. If this command is executed without arm-semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(5) If the current limit reset value in User Preferences is set to any value other than "1", "3", "5", or "7", the compensation value will be reset to "0" when you turn on the motor power.

To make the current limit function effective immediately after switching on the motor power, set the current limit reset value to "1."

6-axis

(6) When setting the current limit, be sure to enable the gravity compensation function. If the current limit is set when the gravity compensation function is disabled, Error "665A Cannot set current limit" will result. For the gravity compensation function, refer to `ST_SetGravity`.

6-axis (7) Set the mass of payload and the payload center of gravity accurately.

Otherwise, the robot may move down due to gravity if you set a low current limit value (e.g., less than 30). For the entry procedure of the mass of payload and the payload center of gravity, refer to the Programmer's Manual, " 4.7 Control Set of Motion Optimization in User Preferences."

6-axis (8) If the current limit reset value is set to "1," the robot might move down due to gravity the moment you turn on the motor power. Reset the current limit by executing `ST_ResetCurLmt` when the motor power is off and then switch on the motor power.

6-axis (9) If you do not know the accurate mass of payload or its center of gravity or if the robot moves down in spite of accurate settings, then use the gravity offset function (`ST_SetGrvOffset`) that compensates for the gravity compensation value.

4-axis HS-E (10) When setting the current limit to the Z-axis or the T-axis of the 4-axis robot without an air balance mechanism, the Z-axis may move down or the T-axis may rotate if you set a low current limit value. Set current limit after execution the gravity compensation function (`ST_SetZBalance`) for the Z and the T-axis.

Example

6-axis

```
ST_SetGravity       'Enable the gravity offset.
ST_SetGrvOffset     'Compensate the gravity offset value
ST_SetEralw 2, 20   'Set the allowable deviation of the 2nd axis to 20
                    'degrees
ST_SetCurLmt 2, 30  'Set the current limit of the 2nd axis to 30%
```

4-axis

```
ST_SetEralw 2, 20   'Set the allowable deviation of the 2nd axis to 20
                    'degrees
ST_SetCurLmt 2, 30  'Set the current limit of the 2nd axis to 30%
```

4-axis HS-E

```
ST_SetZBalance      'Set the gravity compensation value of the Z-axis
ST_SetEralw 3, 100  'Set the allowable deviation of the Z-axis to 100 mm
ST_SetEralw 4, 30   'Set the allowable deviation of the T-axis to 30
                    'degrees
ST_SetCurLmt 3, 10  'Set the current limit of the Z-axis to 10%
ST_SetCurLmt 4, 10  'Set the current limit of the T-axis to 10%
```

ST_ResetCurLmt (Statement)

Function

Resets the motor current limit of the specified axis.

Syntax

```
ST_ResetCurLmt <AxisNumber>
```

Explanation

`ResetCurLmt` releases the drive current limit set for the motor of a joint specified by `<AxisNumber>`. The motor drive current limit and positioning error allowances will revert to the defaults.

[For Ver. 1.4 or earlier]	If you set "0" to `<AxisNumber>`, the drive current limit set for all joints will revert to the default.
[For Ver. 1.5 or later]	If you set "0" to `<AxisNumber>`, the drive current limit set for all joints involved in an arm group semaphore held by the current task running `ST_ResetCurLmt`, will revert to the default.

Related Terms

ST_SetCurLmt, ST_ResetEralw

Notes

(1) When resetting the current limit, this command carries out deviation elimination process. If there is angle deviation due to external force, the time required for deviation elimination process will vary depending upon the set speed and acceleration. To shorten the time, set higher speed and acceleration.

(2) The command can be executed even when the motor power is off. For resetting the current limit when motor power is off, run the following program after finishing the task that is obtaining arm semaphore.

```
PRO999
TAKEARM
ST_ResetCurLmt 0
END
```

(3)

[For Ver. 1.4 or earlier]	Execute this command in a TAKEARMed task that has got robot arm semaphore. If you specify any joints not in the arm semaphore to `<AxisNumber>`, then error "21F7 Cannot take arm semaphore" will result.
[For Ver. 1.5 or later]	Execute this command in a TAKEARMed task that has got an arm group. If you specify any joints not included in the arm group to `<AxisNumber>`, then error "27D* Cannot take J* semaphore" will result.

Example

```
ST_ResetCurLmt 0          'Reset the current limit of all the axes.
ST_ResetGrvOffset 0       'Disable the gravity offset function.
```

ST_SetEralw (Statement)

Function

Modifies the allowable deviation of the specified axis.

Syntax

```
ST_SetEralw <AxisNumber>, <Value>
```

Explanation

Sets the allowable deviation of the axis specified by `<AxisNumber>`. Use this command if angle deviation occurs due to external force when the current limit function is enabled.

The `<Value>` is the arm joint angle and specified in degrees.

"Allowable deviation value" refers to the allowable range of the "Error 611* J* excess error" which will occur for safety if the servo deviation exceeds the specified value.

During assembling operation with the current limit enabled, if servo deviation occurs due to external force, the above error may occur. To avoid this, you may use this command temporarily to increase the allowable deviation value.

This command can also be used to reduce the allowable deviation value for helping quick detection of the downward movement of the robot due to gravity when the current limit function is enabled.

Related Terms

ST_SetCurLmt, ST_ResetEralw

Notes

(1) Run this command in a TAKEARMed task which has obtained arm semaphore.

If the command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_SetEralw 2,20          'Set the permissible deviation of the 2nd
                          'axis to 20 degrees.
```

ST_ResetEralw (Statement)

Function

Resets the allowable deviation value of the specified axis to the initial value.

Syntax

```
ResetEralw <AxisNumber>
```

Explanation

Resets the allowable deviation value of the axis specified by `<AxisNumber>` to the default value.

If `<AxisNumber>` is specified to "0," the allowable deviation values of all the axes will be reset.

[For Ver. 1.4 or earlier]	If you set "0" to `<AxisNumber>`, the positioning error allowance set for all joints will revert to the default.
[For Ver.1.5 or later]	If you set "0" to `<AxisNumber>`, the positioning error allowance set for all joints involved in an arm group semaphore held by the current task running `ST_ResetEralw`, will revert to the default.

Related Terms

ST_ResetCurLmt, ST_SetEralw

Notes

(1) [For Ver. 1.4 or earlier] Execute this command in a TAKEARMed task that has got robot arm semaphore. If you specify any joints not in the arm semaphore to `<AxisNumber>`, then error "21F7 Cannot take arm semaphore" will result.

[For Ver. 1.5 or later] Execute this command in a TAKEARMed task that has got an arm group. If you specify any joints not included in the arm group to `<AxisNumber>`, then error "27D* Cannot take J* semaphore" will result.

(2) Like this command, execution of `ST_ ResetCurLmt` will also reset the allowable deviation values to the initial values.

Example

```
ST_ResetEralw 0           'Return the allowable deviation values of all
                          'axes to initial values.
```

ST_OnSrvLock (Statement)

Function

Servo-locks a specified axis (exclusively designed for four-axis robots).

Syntax

```
ST_OnSrvLock <specified axis>
```

Explanation

Provides a function similar to that of the `ON SVLOCK` instruction in the conventional language.

Servo lock means that robot arms are controlled and their positions are held.

Related Term

ST_OffSrvLock

Notes

(1) Set servo lock as the robot stops. If it is set during path operation, an error may occur.

(2) Execute this command in a TAKEARMed task which has obtained arm semaphore.

If the command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_OnSrvLock 1              'Servo lock for the 1st axis.
ST_OnSrvLock 0              'Servo lock for all axes.
```

ST_OffSrvLock (Statement)

Function

Releases servo lock for the specified axis. (Exclusively designed for four-axis robots)

Syntax

```
ST_OffSrvLock <specified axis>
```

Explanation

Provides a function similar to the `OFF SVLOCK` instruction in the conventional language.

Servo lock refers to the state where the robot arm is controlled to keep its position. When it is released, the robot arm is not kept in its position but moved by an external force applied to it.

Related Term

ST_OnSrvLock

Notes

(1) No operation command can be executed for an axis for which servo lock is released.

(2) Set release of servo lock while the robot is in stopped state. If set during path operation, an error may result.

(3) Execute this command in a TAKEARMed task that has obtained arm semaphore. If the command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(4) If bit 2 of the value set to "25: Current limit reset" in the operating conditions is "0" (initial value), servo lock release is reset (to cause servo lock) upon motor power on. To validate servo lock release immediately after motor power on, set "+2" as the current limit reset value.

Note: Setting example of operating condition "25: Current limit reset"

	PWM *	SVLock *	Curlmt *	
If only SVLock is effective	0	1	0	= 2
If all is effective	1	1	1	= 7

Example

```
ST_OffSrvLock 1           'Release servo lock for the 1st axis.
```

ST_SetCompControl (Statement)

Function

Enables the compliance function (exclusively designed for 6-axis robots)

Syntax

```
ST_SetCompControl
```

Explanation

Enables the compliance function. Enables the compliance conditions set by `ST_SetFrcLimit`, `ST_SetCompRate`, and `ST_SetFrcCoord`.

Related Terms

ST_SetFrcLimit, ST_SetCompRate, ST_SetFrcCoord, ST_ResetCompControl, ST_SetCompFControl

Notes

(1) You will receive an error "60F5 Compliance control is not executable", when this statement is executed while the gravity offset is disabled and the current limiting is enabled. Execute again after you enable the gravity offset and disable the current limiting. See `ST_ResetCurLmt` and `ST_SetGravity` for disabling the current limiting and enabling the gravity offset respectively.

(2) The compliance control will not be enabled while motors are off. The compliance control will be disabled when you turned off motors under the compliance control.

(3) Execute this command in a TAKEARMed task has obtained arm semaphore. If this command is executed without arm semaphore obtained, an error "21F7 Cannot take arm semaphore" will result.

(4) Execute this command when your robot is on halt. Executing this command in a pass motion will cause an end motion. If executing this command in a pass motion causes an error "600B Robot is running," then stop the robot with a Delay command and then execute this command.

(5) If the robot is moved by any external force under compliance control, an error "611* J* excess error" may occur. In such a case, change the allowable deviation by using the `ST_SetEralw`.

(6) This command should not be executed when the robot undergoes any force, e.g., in contact with any surrounding facility. To enable the compliance control in such conditions, use the `ST_SetCompFControl`.

(7) If the robot posture greatly changes after execution of the ST_SetCompControl, then an error may be generated in the gravity offset compensation value and the robot may move in the direction of gravity. If the posture changes greatly under compliance control, use the `ST_ResetCompControl` to disable the compliance control and then execute the `ST_SetCompControl` again to enable the compliance control.

Example

```
ST_SetFrcCoord 1             'Set the compliance control coordinate system.
ST_SetFrcLimit 100, 0, 100, 100, 100, 100
                             'Set the compliance rate
ST_SetCompControl            'Enable the compliance control
ST_SetEralw 1, 90            'Set the allowable deviation
```

ST_SetCompFControl (Statement)

Function

Enables the compliance control function (exclusively designed for 6-axis robots).

Syntax

```
ST_SetCompFControl
```

Explanation

Enables the compliance control function, just like the `ST_SetCompControl`. Note that this command will not execute the gravity offset compensation.

Related Terms

ST_SetCompControl

Notes

(1) If this command is executed when the gravity offset is disabled and the current limiter is disabled, then an error "60F5 Cannot execute compliance control" will occur.

(2) Executing this command when the motors are off will not enable the compliance control. Under the compliance control, turning off the will disable the compliance control.

(3) Execute this command in a TAKEARMed task has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(4) Execute this command when the robot is on halt. Executing this command in a pass motion will cause an end motion. If executing this command in a pass motion causes Error "600B Robot is running," then stop the robot with a Delay command and then execute this command.

(5) Set the payload exactly. If the setting and actual payload differ, the robot arm may fall down in the direction of gravity. To prevent such a fall, execute the `ST_SetGrvOffset`.

Example

```
ST_SetGrvOffset            'Calculate the gravity offset compensation
                           'value.
ST_SetFrcCoord 1           'Set the compliance control coordinate system.
ST_SetFrcLimit 100, 0, 100, 100, 100, 100
                           'Set the compliance rate
ST_SetCompFControl         'Enable the compliance control
```

ST_ResetCompControl (Statement)

Function

Disables the compliance control function (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetCompControl
```

Explanation

Disables the compliance control function.

Related Terms

ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) Execute this command when the robot is on halt. Executing this command in a pass motion will cause an end motion. If executing this command when the robot is in motion immediately stops the robot or causes Error "612* J* overcurrent," then set the value to "1" or stop the robot by using the Delay command.

(3) If the robot comes to a momentary stop during execution of this command so that you perform a Step back or Program reset, then Error "60F9 Improper compliance set/reset operation" will occur.

(4) If you use ST_SetEralw to change the allowable deviation values while the compliance control is enabled, the allowable deviation values will return to their initial values. The allowable deviation values may not be reset to the initial values when an error occurs while executing ST_ResetCompControl. If this is the case, use ST_ResetEralw to initialize the allowable deviation values after disabling the compliance control.

(5) If Error " 608* J* command speed limit over" occurs, use the ST_aspChange to change the optimal load capacity mode to 2 or 3 before execution of the ST_ResetCompControl. After execution of this command, return the optimal load capacity mode to the previous value.

Example

```
ST_ResetCompControl         'Disable the compliance control.
ST_ResetEralw               'Initialize the allowable deviation values
```

ST_SetFrcCoord (Statement)

Function

Selects a force limiting coordinate system (exclusively designed for 6-axis robots).

Syntax

```
ST_SetFrcCoord <Set value>
```

Explanation

Selects a coordinate system for force limiting values specified by ST_SetFrcLimit and ST_SetCompRate. You can use a set value 0 for the base coordinate system, a set value 1 for the tool coordinate system, and a set value 2 for the work coordinate system of your robot.

Related Terms

ST_SetFrcLimit, ST_SetCompRate, ST_SetFrcCoord, ST_ResetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60FA Not Executable in compliance control" will result.

(3) When you specify 1 for <Set value> to select the tool coordinate system, the tool coordinate will be the tool coordinate for enabling the compliance control (executing ST_SetCompControl). When you use the changetool command to change the tool coordinate while the compliance control is enabled, the force limiting coordinate will not be changed.

(4) When you specify 2 for <Set value> to select the work coordinate system, the work coordinate will be the work coordinate for enabling the compliance control (executing ST_SetCompControl). When you use the changework command to change the work coordinate while the compliance control is enabled, the force limiting coordinate will not be changed.

(5) The set value will be initialized to 0 (the base coordinate system) after the controller is turned on.

Example

```
ST_SetFrcCoord 1            'Set the compliance coordinate system to
                            'the tool coordinate
Changetool 2                'Set the tool coordinate to tool2
ST_SetFrcLimit 100, 0, 100, 100, 100, 100
                            'Set the compliance rate
ST_SetCompControl           'Set the compliance rate in Y direction
                            'of the tool 2 coordinate system to 0% and
                            'enable the compliance control
```

ST_SetFrcLimit (Statement)

Function

Sets the force limiting rates (exclusively designed for 6-axis robots).

Syntax

```
ST_SetFrcLimit <Limiting rate along X>, <Limiting rate along Y>, <Limiting
rate along Z>, <Limiting rate about X>, <Limiting rate about Y>, <Limiting
rate about Z>
```

Explanation

Sets the force limiting rates along and about X, Y, and Z axes of a coordinate system specified by ST_SetFrcCoord.

Setting ranges from 0 to 100. Up to two decimal places are valid.

Related Terms

ST_ResetFrcLimit, ST_SetFrcCoord, ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60FA Not Executable in compliance control" will result.

(3) All the set values for along and around the X, Y and Z-axes will be initialized to 100 after the controller is turned on.

Example

```
ST_SetFrcCoord 1           'Set the compliance coordinate system to
                           'the tool coordinate
ST_SetFrcLimit 100, 0, 100, 100, 100, 100
                           'Set the compliance rates
ST_SetCompControl          'Set the compliance rate in Y direction
                           'of the tool coordinate system to 0% and
                           'enable the compliance control
```

ST_ResetFrcLimit (Statement)

Function

Initializes the force limiting rates (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetFrcLimit
```

Explanation

Initializes the force limiting rates. All rates along and about X, Y, and Z axes are set to 100%.

Related Terms

ST_SetFrcLimit

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60FA Not Executable in compliance control" will result.

Example

```
ST_ResetCompControl        'Disable compliance control
ST_ResetFrcLimit           'Initialize the force limiting rates
```

ST_SetCompRate (Statement)

Function

Sets the compliance rates under the compliance control (exclusively designed for 6-axis robots).

Syntax

```
ST_SetCompRate <Compliance along X>, <Compliance along Y>, <Compliance
along Z>, <Compliance about X>, <Compliance about Y>, <Compliance about Z>
```

Explanation

Sets the compliance rates along and about X, Y, and Z axes of a coordinate system specified by SetFrcCoord.

Setting ranges from 0 to 100 and 0 gives the maximum compliance. Up to two decimal places are valid.

Related Terms

ST_ResetCompRate, ST_SetFrcCoord, ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60FA Not Executable in compliance control" will result.

(3) All the set values for along and around the X, Y and Z-axes will be initialized to 100 after the controller is turned on.

Example

```
ST_SetFrcCoord 1           'Set the force limiting coordinate system to
                           'the tool coordinate
ST_SetCompRate 100, 0, 100, 100, 100, 100
                           'Set the compliance rate
ST_SetCompControl          'Set the compliance rate in Y direction of
                           'the tool coordinate system to 0% and enables
                           'the compliance control
```

ST_ResetCompRate (Statement)

Function

Initializes the compliance rates (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetCompRate
```

Explanation

Initializes the compliance rates along and about X, Y, and Z axes to 100%.

Related Terms

ST_SetCompRate

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60F9 Improper compliance set/reset operation" will result.

Example

```
ST_ResetCompControl        'Disable the compliance control
ST_ResetCompRate           'Initialize the compliance rates
```

ST_SetFrcAssist (Statement)

Function

Sets the force assistance under the compliance control (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
SetFrcAssist <Force assistance along X>, <Force assistance along Y>,
<Force assistance along Z>, <Moment assistance about X>, <Moment
assistance about Y>, <Moment assistance about Z>
```

Explanation

Sets the force assistance along and the moment assistance about X, Y, and Z-axes of a coordinate system specified by ST_SetFrcCoord. The maximum set value is 10% of the maximum force limiting value.

The unit for the force setting is [N]. The unit for the moment setting is [Nm]. Up to one decimal place is valid.

Related Terms

ST_ResetFrcAssist, ST_SetFrcCoord, ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) Your robot may move toward the direction to which the force assistance and the moment assistance are applied. If this is the case, reduce the set values.

(3) All the set values for along and around the X, Y and Z-axes will be initialized to 0 after the controller is turned on.

Example

```
ST_SetFrcCoord 1            'Set the force limiting coordinate system to
                            'the tool coordinate
ST_SetFrcAssist -30, 0, 0, 0, 0, 0
                            'Set the force assistance to 30 [N] toward -X
                            'direction
ST_SetFrcLimit 0, 100, 100, 100, 100, 100
                            'Set the force limiting rates
ST_SetCompControl           'Enable the compliance control function.
                            'Force limiting in X direction is 0% and a
                            'force of 30 [N] is applied toward -X
                            'direction
```

ST_ResetFrcAssist (Statement)

Function

Initializes the force assistance (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetFrcAssisit
```

Explanation

Initializes the force assistance along and the moment assistance about X, Y, and Z-axes of a coordinate system specified by ST_SetFrcCoord are set to 100%.

Related Terms

ST_SetFrcAssist

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
STResetCompControl          'Disable the compliance control
ST_ResetFrcAssist           'Initialize the force assistance and the
                            'moment assistance
```

ST_SetCompJLimit (Statement)

Function

Sets the current limit under the compliance control (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
ST_SetCompJLimit <J1 current limit>, <J2 current limit>, <J3 current
limit>, <J4 current limit>, <J5 current limit>, <J6 current limit>
```

Explanation

Sets the current limit under the compliance control. The rated current of a motor corresponds to 100. When you use SetFrcLimit to set 0 to all the directions, the motor currents are limited to values less than the setting.

Setting ranges from 0 to 100.

Related Terms

ST_ResetCompJLimit, ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) The set values will be initialized and all the current limits will be 0 after the controller is turned on.

(3) When you use ST_SetCompJLimit to set relatively large values, your robot may present an oscillation resulting in an error. If this is the case, use ST_SetCompRate and ST_SetDumpRate to adjust the compliance.

Example

```
ST_SetFrcCoord 1            'Set the force limiting coordinate system to
                            'the tool coordinate
ST_SetCompJLimit 30, 0, 0, 0, 0, 0
                            'Set the current limit for J1 to 30%
ST_SetFrcLimit 0, 100, 100, 100, 100, 100
                            'Set the force limiting rates
ST_SetCompControl           'Enable the compliance control function.
```

ST_ResetCompJLimit (Statement)

Function

Initializes the current limit under the compliance control (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetCompJLimit
```

Explanation

Initializes the current limit under the compliance control and set 0 to all axes.

Related Terms

ST_SetCompJLimit, ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_ResetCompControl        'Disable the compliance control
ST_ResetCompJLimit         'Initialize the current limits
```

ST_SetCompVMode (Statement)

Function

Sets the velocity control mode under the compliance control (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
ST_SetCompVMode
```

Explanation

Enables the compliance velocity control mode when ST_SetCompControl is executed.

Related Terms

ST_ResetCompVMode

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_SetCompVMode            'Enable the compliance velocity control mode
ST_SetCompControl          'Enable the compliance control
```

ST_ResetCompVMode (Statement)

Function

Disables the velocity control mode under the compliance control (special compliance control function statement) (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetCompVMode
```

Explanation

Disables the compliance velocity control mode when ST_SetCompControl is executed.

Related Terms

ST_SetCompControl

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_ResetCompVMode          'Disable the compliance velocity control mode
ST_SetCompControl          'Enable the compliance control
```

ST_SetCompEralw (Statement)

Function

Sets the allowable deviation values of the position and the posture of the tool tip under the compliance control (exclusively designed for 6-axis robots).

Syntax

```
ST_SetCompEralw <Allowable deviation along X>, <Allowable deviation Y>,
<Allowable deviation Z>, <Allowable deviation X>, <Allowable deviation Y>,
<Allowable deviation Z>
```

Explanation

Sets the allowable deviation values of the position and the posture of the tool tip under the compliance control. The unit for the allowable deviation along X, Y, and Z is (mm), and the unit for the allowable deviation about X, Y, and Z is (degree). Up to one decimal place is valid.

Related Terms

ST_ResetCompEralw, ST_SetFrcCoord

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) The initial values of allowable deviation are 100 (mm) along X, Y, and Z and 30 (degree) about X, Y, and Z. The maximum value about X, Y, and Z-axes is 175 (degree). If you set larger values than the maximum value, you will receive an error " 6003 Excess in effective value range ".

(3) If the position or the posture deviation of the tool tip exceeds the allowable value, you will receive an error " 60F8 Compliance deviation excess error ".

(4) The coordinate system used for setting the deviation is the one set by ST_SetFrcCoord. When you specify as STSetFrcCoord 1, the setting coordinate system for ST_SetCompEralw is the tool coordinate system.

Example

```
ST_SetFrcCoord 2           'Set the force limiting coordinate system to
                           'the work coordinate system
ST_SetFrcLimit 100, 0, 100, 100, 100, 100
                           'Set the force limiting rate for the Y
                           'direction of the work coordinate system to 0%
ST_SetCompEralw 10, 150, 10, 5, 5, 5
                           'Set the allowable deviation values along X
                           'and Z of the work coordinate to 10 (mm),
                           'along Y to 150 (mm), and about X, Y, and Z to
                           '5 (degree).
```

ST_ResetCompEralw (Statement)

Function

Initializes the allowable deviation values of the position and the posture of the tool end under the compliance control (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetCompEralw
```

Explanation

Initializes the allowable deviation values of the position and the posture of the tool tip under the compliance control. The initial values of allowable deviation are 100 (mm) along X, Y, and Z and 30 (degree) about X, Y, and Z.

Related Terms

ST_SetCompEralw

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

Example

```
ST_ResetCompEralw
```

ST_SetDampRate (Statement)

Function

Sets the damping rates under the compliance control (exclusively designed for 6-axis robots).

Syntax

```
ST_SetDampRate <DampRate along X>, <DampRate along Y>, <DampRate along Z>,
<DampRate about X>, <DampRate about Y>, <DampRate about Z>
```

Explanation

Sets the damping rates along and about X, Y, and Z axes on the coordinate system specified by `ST_SetFrcCoord`.

The entry range is from 0 to 100. Zero gives the maximum damping. Up to two decimal places are valid.

Related Terms

ST_ResetDampRate, ST_SetFrcCoord, ST_SetCompRate

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore " will result.

(2) This statement is not executable under the compliance control. If this command is executed under the compliance control, Error " 60FA Not Executable in compliance control" will result.

(3) When the controller is turned on, all the damping rates will revert to the initial values (100).

(4) Do not set the damping rates to less than the compliance rates specified by the `ST_SetCompRate`. Doing so will cause the robot to vibrate; in some cases, the robot will stop due to an error.

(5) If the `ST_SetCompRate` is executed after execution of the `ST_SetDampRate`, then the damping rates will change to the setting made by `ST_SetCompRate`.

Example

```
ST_SetFrcCoord 1            'Set the compliance coordinate system to
                            'the tool coordinate system
ST_SetCompRate 100, 0, 100, 100, 100, 100
                            'Set the compliance rate
ST_SetDampRate 100, 20, 100, 100, 100, 100
                            'Set the damping rate
ST_SetCompControl           'Set the compliance rate and the damping rate
                            'in Y direction of the tool coordinate system
                            'to 0% and 20%, respectively, and enable the
                            'compliance control
```

ST_ResetDampRate (Statement)

Function

Initializes the damping rates under the compliance control (exclusively designed for 6-axis robots).

Syntax

```
ST_ResetDampRate
```

Explanation

Initializes all damping rates along and about X, Y, and Z axes to 100.

Related Terms

ST_SetDampRate

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore " will result.

(2) This command is not executable under the compliance control. If this command is executed under the compliance control, then Error " 60FA Not Executable in compliance control" will result.

Example

```
ST_ResetDampRate            'Initialize the damping rates
```

ST_SetZBalance (Statement) [Version 1.9 or later]

Function

Sets the gravity compensation value of the Z and T axes (exclusively designed for 4-axis robots).

Syntax

```
ST_SetZBalance
```

Explanation

Having no air balance mechanism, the Z and T axes of a 4-axis robot undergo a downward static load (gravity torque). If you set the current limit value less than the gravity torque in the current limit function, then the Z axis will move down and the T axis will rotate.

This command estimates the gravity torque when the robot is on halt and sets the gravity compensation value, preventing unexpected Z-axis downward movement and T-axis rotation.

Related Terms

ST_ResetZBalance, ST_SetCurLmt

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This command should be executed when the motor power is on and the robot is on halt. If it is executed when the motor power is off, Error "6006 Motor power is off" will result. If it is executed when the robot is in motion, Error "600B Robot is running" will result.

(3) If the work weight changes after execution of this command, the preset gravity compensation value will deviate. You need to execute this command again.

Example

```
ST_SetZBalance
```

ST_ResetZBalance (Statement) [Version 1.9 or later]

Function

Disables the gravity compensation function (exclusively designed for 4-axis robots).

Syntax

```
ST_ResetZBalance
```

Explanation

Disables the gravity compensation function set by the `ST_SetZBalance` for Z and T axes of 4-axis robots.

Related Terms

ST_SetZBalance

Notes

(1) Execute this command in a TAKEARMed task that has obtained arm semaphore. If this command is executed without arm semaphore obtained, Error "21F7 Cannot take arm semaphore" will result.

(2) This command should be executed when the robot is on halt. If it is executed when the robot is in motion, Error "600B Robot is running" will result. This command is executable even when the motor power is off.

Example

```
ST_ResetZBalance
```

12. Error Codes Added or Modified

Refer to the ERROR CODE TABLES.

The table below lists error codes newly added or modified in Main System Software Version 1.9.

Code	Message	Level	Description	Remedy
120A	Slave exclusive flag failure	4	Any exclusive flag of the DeviceNet slave station or the communications processor in the CC-Link remote device is not set normally.	Check that the DeviceNet slave board or CC-Link remote device board is correctly installed. If the error persists, the board may be damaged.
120C	The network error cleared	3	An network error on the DeviceNet, PROFIBUS or CC-Link has been removed.	Carry out the error removal process with the teach pendant, operating panel, or external equipment.
1230	Robot access failure in DPRAM (slave)	4	The robot controller cannot access the DPRAM in the DeviceNet slave module or on the CC-Link remote device board.	Restart the robot controller and then try the operation again.
1238	Slave board access failure in DPRAM	4	The communication software in the DeviceNet slave station or CC-Link remote device cannot access the DPRAM.	Restart the robot controller and then try the operation again.
128A	CC-Link board error at power on	4	When turned on, the robot controller has detected a hardware error on the CC-Link remote device board.	Restart the robot controller and then try the operation again.
128B	CC-Link <- controller handshake error	4	An error has occurred in the process of exchanging the status information between the controller and CC-Link remote device board.	Restart the robot controller and then try the operation again.
128C	CC-Link DPRAM data error	4	A data error has occurred in the DPRAM. on the CC-Link remote device board.	Restart the robot controller and then try the operation again.

Code	Message	Level	Description	Remedy
128D	CC-Link communication error	4	CC-Link does not operate.	1. Check that the CC-Link settings made in the controller are not discrepant from those in the master station. 2. Check that the CC-Link cable is not broken or the connector is plugged in. 3. Check that there is no welding machines or machinery emitting electric noise near the robot unit or its controller. In addition to the above check points, check the status of the LEDs on the front of the CC-Link remote device board.
128E	CC-Link station number error	2	"Remote station address + (Number of local stations - 1)" is out of the range from 1 to 63.	Set it within the specified range.
128F	CC-Link communication speed error	2	The transmission rate switch is set to any position other than 0 to 4.	Set the transmission rate switch to any of the 0 to 4 positions.
129A	CC-Link occupied stations error	4	The number of local (occupied) stations is out of the range from 2 to 4.	Set it within the specified range.
21B9	Local variable editing	2	You attempted to carry out any operation not allowed during editing of local variables.	After completion of local variable editing, retry the operation.
2498	The arm end enters inhibit area 0 (DIRECT)	2	In direct teaching mode, the tool end enters the prohibited area.	Move the tool end outside prohibited area 0.
2499	The arm end enters inhibit area 1 (DIRECT)	2	↑	Move the tool end outside prohibited area 1.
249A	The arm end enters inhibit area 2 (DIRECT)	2	↑	Move the tool end outside prohibited area 2.
249B	The arm end enters inhibit area 3 (DIRECT)	2	↑	Move the tool end outside prohibited area 3.
249C	The arm end enters inhibit area 4 (DIRECT)	2	↑	Move the tool end outside prohibited area 4.
249D	The arm end enters inhibit area 5 (DIRECT)	2	↑	Move the tool end outside prohibited area 5.

Code	Message	Level	Description	Remedy
249E	The arm end enters inhibit area 6 (DIRECT)	2	↑	Move the tool end outside prohibited area 6.
249F	The arm end enters inhibit area 7 (DIRECT)	2	↑	Move the tool end outside prohibited area 7.
27BC	Not setting argument	2	You attempted to run a program involving arguments when the argument setting window could not be opened.	Restart the controller, check that the argument setting window can be opened, and then retry the operation.
27BD	Controller initialization is unusual	5	In the powering-on sequence of the controller, initialization has failed.	Restart the controller.
27BE	Can't start motion from here.	2	In easy teaching, you attempted to run a command not allowed in the current robot status.	Before starting robot motion under arc interpolation control and executing DEPART command, move the robot arm to the destination position defined in the immediately preceding motion command.
2AF4	Cooling fan error	2	The controller detects the built-in cooling fan being out of order.	First remove the error factor, move the robot arm to a safe area, and then replace the cooling fan.
60DA	Can't operate to the Z-axis position	1	With the current settings of arch start and end positions, the robot arm cannot reach the commanded Z-coordinate.	Correct the current arch settings.
64FA	Cutting of a belt was detected	3	The controller detects the U-joint drive belt being broken.	Remove the robot cover and check or replace the belt.
6819	Unexpected command (Host 1)	5	A controller internal error (software error) has occurred.	Restart the controller.
681A	Unexpected command (Host 2)	5	↑	↑
681B	Unexpected command (Host 3)	5	↑	↑
681C	Unexpected command (Servo 1)	5	↑	↑
681D	Unexpected command (Servo 2)	5	↑	↑

Code	Message	Level	Description	Remedy
681E	Unexpected command (Servo 3)	5	↑	↑
6829	Precise control processing delay	4	The precision trajectory processing has not finished in time.	When the precision trajectory processing is in progress, this error may occur if the transmission is frequent via the RS-232C or Ethernet or if the controller is accessed by the keyboard. Decrease the frequency of transmission.
682B	Improper precise control operation	4	During setting for enabling or disabling the precision trajectory processing, Program Reset or Step Back operation has been performed. The occurrence of this error has reset the program(s) currently running.	If you want to execute Program Reset or Step Back when the robot is on halt for disabling/enabling the precision trajectory processing, then be sure to turn the motors off once.
736F	Can't start program with array arguments	2	In stand alone mode, the current system does not support starting of programs containing array argument(s).	Only the CALL statement can call such a program that contains array argument(s).
75B0	Client port open error	3	The specified client port is already occupied or invalid.	Change the port or correct the client port settings.

SUPPLEMENT

Main System Software Version 1.9

First Edition February 2002

DENSO WAVE INCORPORATED
Factory Automation Division

2D20C