YAMAHA
RCX Provider

Version 1.0.1

User's Guide

October 6, 2020

Remarks:

[Revision History]

Version	Date	Content
1.0.0	2012-10-30	First edition.
	2013-04-26	Added the note for “echo-back” function of the YAMAHA controller.
1.0.1	2019-09-27	Fixed license check algorithm.
	2020-10-06	Addition of license addition items.

[Hardware]

Model	Version	Notes

【Attention】

Additional license for " YAMAHA Multi-axis Robots Controller Provider " is required to use this provider.

Contents

1. Introduction

This document is a user's guide for the CAO provider for the YAMAHA robot RCX series. The CAO provider described in this manual (CaoProvRCX.dll) is called the RCX provider.
The next chapter provides the outline of the RCX provider. Chapter 3 provides the command reference.

1.1. Emergency stop device position

A robot emergency stop switch should be prepared and set up near a robot operator before operating the robot, so that the switch can immediately stop the robot motion in an emergency situation.

(1) The emergency stop switch should be red-colored.
(2) After the emergency stop switch is activated, the switch should not return to normal (robot operating) position automatically or by other operator’s careless action.
(3) A robot emergency stop switch should be set up separately from the power switch.

1.2. Installing license

To use OpenCV Provider, you need to install ORiN2 SDK, and also need to input “YAMAHA Multi-axis Robots Controller Provider” license information. If you would like to install it for evaluation, please use the following license.

SME8-C1US-1545-PNLP (valid for 3 months)

How to add the license is as follows.

1. Run the CaoConfig tool from the [Start] menu, and select the [Cao Provider] tab.
2. Select the [YAMAHA RCX CAO Provider] item on the provider list.
3. Click the [...] button of the license input box.
4. Click the [Add] button in the "ORiN2 License Manager" window.
5. Input a license key, and click the [OK] button.
6. Click the [Close] button to exit.

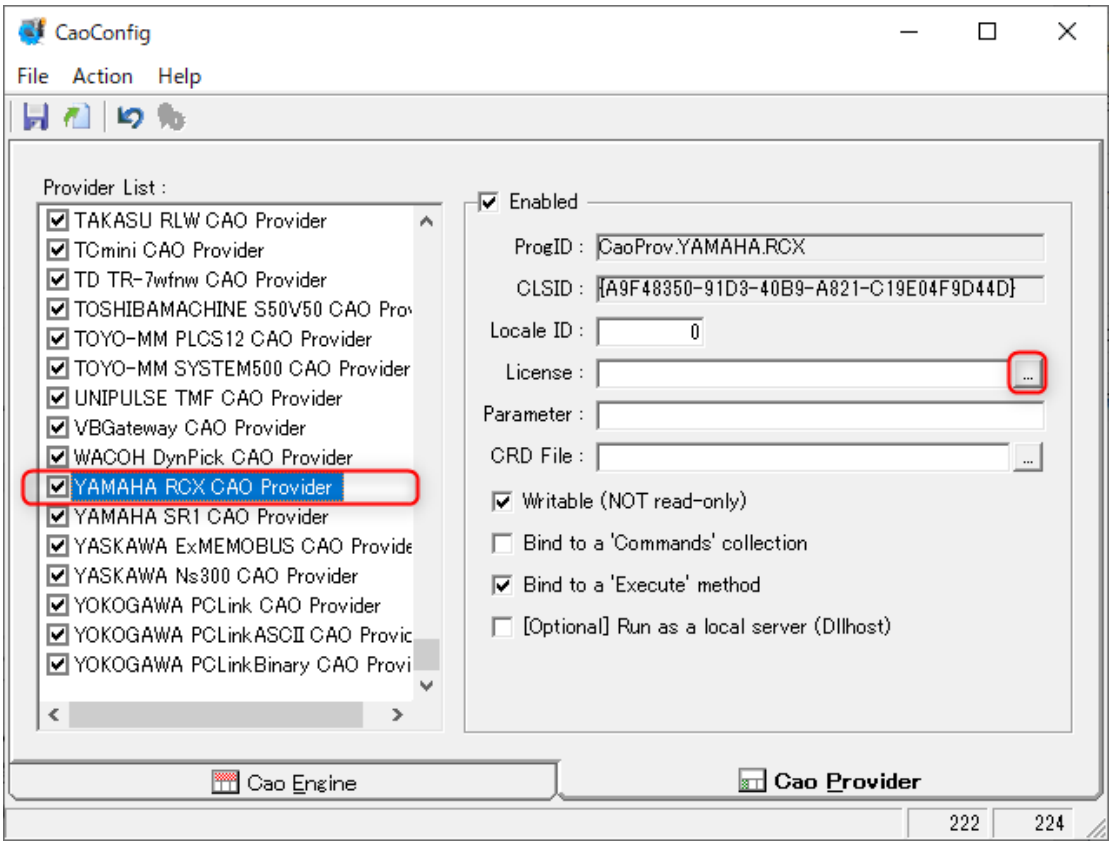

Figure 1-1 Installing ' YAMAHA Multi-axis Robots Controller Provider ' license

2. Outline of Provider

2.1. Outline

The RCX provider is a CAO provider that absorbs the parts dependent on the YAMAHA robot controller and offers the functions defined in the CAO provider interface specifications. The file format is DLL (Dynamic Link Library) and the file is dynamically loaded by CAO engine when it is used. To use the RCX provider, ORiN2SDK needs to be installed or registry needs to be manually registered according to the table below.

Table 2-1 RCX provider

File name	CaoProvRCX.dll
ProgID	CaoProv.YAMAHA.RCX
Registry registration	regsvr32 CaoProvRCX.dll
Remove registry registration	regsvr32 /u CaoProvRCX.dll

2.2. Methods and properties

2.2.1. CaoWorkspace::AddController method

RCX provider refers to the communication connection parameters passed when the AddController method is executed and connects communication.

The options specify the communication method and timeout period and, for Ethernet communications, the Telnet user name and password.

When using Ethernet communication, disable “echo-back” function of the YAMAHA controller. Enabling the function will decrease the communication performance, and the echo-back data may be misrecognized by the YAMAHA provider.

Syntax AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR>,<bstrOption:BSTR>)

<bstrCtrlName>	:	[in] Robot name
<bstrProvName>	:	[in] Option character string Fixed to "CaoProv.YAMAHA.RCX"
<bstrPcName>		[in] Provider execution machine name
<bstrOption>		[in] Option character string

Following is a list of option string items.

Table 2-2 Option character string of CaoWorkspace::AddController

Option	Meaning
Conn=<Connection parameter>	Mandatory. Sets the communication method and connection parameters.
[User=<User name>]	Specify the user name to log in to the RCX controller via Ethernet. (Default: admin)
[Password=<Password>]	Specify the password to log in to the RCX controller via Ethernet. (Default: None)
[Timeout=<Timeout period>]	Specify the timeout period (msec) for transmission. (Default: 500)

2.2.1.1. Conn option

Following is communication parameter string for Conn option. Parameters inside square brackets (“[]”) can be omitted. Underlined parts represent the default values used when the option is not specified.

- **Ethernet device**

  “Conn=eth:<IP Address>[:<PortNo>]”

  <IP Address> : Mandatory. Destination IP address.

  Example: ”127.0.0.1”, ”192.168.0.1”

  <PortNo> : Destination port number.

  Example: “127.0.0.1:23”, ”192.168.0.1:5010”

- **RS232C device**

  “Conn=com:[<ComPort>”[:<BaudRate>[:<Parity>:<DataBits>:<StopBits>]]]”

  <ComPort> : COM port number. ’1’-COM1, ’2’-COM2,...

  <BaudRate> : Transmission rate.4800, 9600, 19200, 38400, 57600, 115200

  <ByteSize> : Parity. ’N’-NONE, ‘E’-EVEN, ‘O’-ODD

  <DataBits> : Number of data bits. ’7’-7bit, ‘8’-8bit

  <StopBits> : Number of stop bits. ‘1’-1bit, ‘2’-2bit

2.2.2. CaoController::AddRobot method

Create a CaoProvRobot object.

The robot name specified here is an arbitrary string. A CaoRobot object is retrieved by calling the AddRobot method.

Syntax AddRobot(<bstrName:BSTR>[,<bstrOption:BSTR>])

< bstrName > : [in] Robot name

< bstrOption > : [in] Option character string

Table 2-3 Option character string of CaoWorkspace::AddController

Option	Meaning
[SubRobot=<TRUE/FALSE>]	Set whether to use the RCX command for sub-robots. (Default: FALSE) This setting can be changed using the CaoRobot::ID property.

2.2.3. CaoController::AddVariable method

Create a variable object.

Refer to 2.3.1 about the variables implemented in the RCX provider.

Syntax AddVariable(<bstrVariableName:VT_BSTR>[,<vntOption:VT_BSTR>])

< bstrVariableName > : [in] Variable name

<bstrOption> : [in] Option character string

Example

```
Dim aaa As Object
Dim bbb As Double

Set aaa = caoCtrl.AddVariable("@POS")
bbb = aaa.Value
```

2.2.4. CaoController::Execute method

Execute the command.

The arguments of the Execute method specify a command as a BSTR and a parameter as a VARIANT array.

For details about commands, refer to 3.1.

Syntax [<vntRet:VT_VARIANT>=]Execute(<bstrCmd:VT_BSTR>[,<vntParam:VT_VARIANT>])

< vntRet > : [out] Return value of command

< bstrCmd > : [in] Command

< vntParam > : [in] Parameter

2.2.5. CaoRobot::AddVariable method

Create a variable object.

Refer to 2.3.2 about the system variables implemented in the RCX provider.

Syntax AddVariable(<bstrVariableName:VT_BSTR>[,<vntOption:VT_BSTR>])

< bstrVariableName > : [in] Variable name

<bstrOption> : [in] Option character string

2.2.6. CaoRobot::Accelerate method

Set the acceleration and deceleration.

Syntax Accelerate <lAxis:VT_I4>, <fAccel:VT_R4>, <fDecel:VT_R4>

< lAxis >	:	[in] Axis number Set the acceleration and deceleration of the specified axis. Specify 0 for the axis number to set the acceleration and deceleration for all the axes.
< fAccel >	:	[in] Acceleration Set the acceleration. Specify 0 for the acceleration to disable the acceleration setting. To set only the deceleration, therefore, specify 0 in this item.
< fDecel>	:	[in] Deceleration Set the deceleration. Specify 0 for the deceleration to disable the deceleration setting. To set only the acceleration, therefore, specify 0 in this item.

2.2.7. CaoRobot::Change method

Change the end-effector.

Syntax Change <bstrName:VT_BSTR>

< bstrName >	:	[in] End-effector number

2.2.8. CaoRobot::Halt method

When a robot motion command of the CaoRobot class, such as the Move and Execute methods, is executed, the robot motion can be stopped using the Halt method.

Syntax Halt <bstrOption:VT_BSTR>

< bstrOption >	:	[in] bstrOption (not used)

2.2.9. CaoRobot::Move method

Move the robot to the specified position. The first argument specifies the Move command type, and the second argument specifies the destination information and speed in a VARIANT array. The following shows the argument specifications of Move.

Syntax Move <lComp:VT_I4>,<vntParam : VT_VARIANT>, <bstrOpt:VT_BSTR>

< lComp > : [in] Interpolation (VT_I4) Refer to Table 2-3.
This parameter determines the execution command and the interpolation.

1 : MOVE P
2 : MOVE L
3 : MOVE C
4 : MOVEI P

< vntParam > : [in] Destination specification
Specify the destination with a character string.
For details about the specification method, refer to "RCX Series Programming Manual."

< bstrOpt> : [in] Option
For the available options, refer to "RCX Series Programming Manual."

Example

・ Example of using Move for single-axis controller

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot(“MF20”)

caoRob.Move 1, "P13", "SPEED=50"    ' 1=PTP
caoRob.Move 2, "P11"                ' 2 = Linear interpolation
caoRob.Move 3, "P10"                ' 3 = Arc interpolation
caoRob.Move 4, "P11", "S=50"        ' Move from current position to P11=170mm/deg
```

2.2.10. CaoRobot::Speed method

Change the program speed.

Syntax Speed <lAxis:VT_I4>, <fSpeed:VT_R4>

< lAxis > : [in] Axis number (not used)

< fSpeed > : [in] Speed

2.2.11. CaoRobot::Execute method

Execute the command.

The arguments of the Execute method specify a command as a BSTR and a parameter as a VARIANT array.

For details about commands, refer to 3.2.

Syntax [<vntRet:VT_VARIANT>=]Execute(<bstrCmd:VT_BSTR>[,<vntParam:VT_VARIANT>])

< vntRet > : [out] Return value of command

< bstrCmd > : [in] Command

< vntParam > : [in] Parameter

2.2.12. CaoRobot::get_ID method

Get whether the target of the robot object is the main robot or the sub-robot.

For the description of values of IDs to be acquired, refer to Table .

Table 2-4 ID values and robot types

ID	Robot type
1	Main robot
2	Sub-robot

2.2.13. CaoRobot::put_ID method

Switch between the main robot and the sub-robot.

For the description of values of IDs to be set, refer to Table .

2.2.14. CaoVariable::get_Value method

Get the value of a variable.

For details about values to be acquired, refer to 2.3.

2.2.15. CaoVariable::put_Value method

Set the value of a variable.

For details about values to be set, refer to 2.3.

2.3. Variable list

2.3.1. Controller class

Table 2-5 Controller class system variable list

Variable identifier	Data type	Explanation	Attribute	
			get	put
@Timeout	VT_I4	Communication timeout period The value can be specified in the Timeout option of AddController().	√	√
@CONFIG	VT_BSTR | VT_ARRAY	Get the controller configuration. The acquired information is stored in an array with the following order: <Setting name> <Axis setting> <Standard interface> <Optional device> <Other settings>	√	-
@EXELVL	VT_BSTR	Execution level status	√	-
@MOD	VT_BSTR	Mode status	√	-
@MSG	VT_BSTR	Message	√	-
@UNIT	VT_BSTR	Point unit coordinate system	√	-
@VER	VT_BSTR	Version	√	-
@MEM	VT_I4 | VT_ARRAY	Available memory capacity The acquired information is stored in an array with the following order: <Source area available capacity> <Object area available capacity>	√	-
@EMG	VT_I4	Emergency stop status 0 Normal status 1 Emergency stop status	√	-
@SELFCHK	VT_BSTR | VT_ARRAY	Error status found in self check Return VT_EMPTY if there is no error.	√	-

@OPSLOT	VT_BSTR | VT_ARRAY	Return the option slot status.	√	-

Table 2-6 Controller class user variable list

Variable identifier	Data type	Explanation	Attribute	
			get	put
P	VT_I4 | VT_ARRAY or VT_R4 | VT_ARRAY	Point data Specify the point number after the variable name. The data type has six elements. A different system of unit is set depending on the data type. VT_I4 | VT_ARRAY Unit: Pulse VT_R4 | VT_ARRAY Unit: Millimeter	√	√
S	VT_I4 | VT_ARRAY or VT_R4 | VT_ARRAY	Shift data Specify the shift number after the variable name. The data type has four elements. A different system of unit is set depending on the data type. VT_I4 | VT_ARRAY Unit: Pulse VT_R4 | VT_ARRAY Unit: Millimeter	√	√

2.3.2. Robot class

Table 2-7 Robot class system variable list

Variable identifier	Data type	Explanation	Attribute	
			get	put
@ARM	VT_BSTR | VT_ARRAY	Arm status The acquired information is stored in an array with the following order: <Current arm setting status> <Arm setting status upon program reset>	√	-
@ORIGIN	VT_BSTR	Origin return status	√	-

@ABSRST	VT_BSTR	Absolute reset status	√	-
@SERVO	VT_BSTR	Servo status	√	-
@SPEED	VT_BSTR | VT_ARRAY	Speed status The acquired information is stored in an array with the following order: <Auto movement speed setting> <Manual movement speed setting>	√	-
@WHERE	VT_I4 | VT_ARRAY	Current position in the pulse coordinate system of the main robot	√	-
@WHERE2	VT_I4 | VT_ARRAY	Current position in the pulse coordinate system of the sub-robot	√	-
@WHRXY	VT_R4 | VT_ARRAY	Current position in the millimeter coordinate system of the main robot	√	-
@WHRXY2	VT_R4 | VT_ARRAY	Current position in the millimeter coordinate system of the sub-robot	√	-
@SHIFT	VT_BSTR	Shift status	√	-
@HAND	VT_BSTR	End-effector status	√	-

2.4. Error code

In the RCX provider, the following unique error codes are defined. For common ORiN2 errors, refer to the error code section in "ORiN2 Programming Guide".

Table 2-8 List of original error codes

Error name	Error code	Explanation
E_CAOP_NO_LICENSE	0x80100000	There is no license. Purchase an additional license.
RCX command error	0x8011xxyy	If an error occurs while an RCX command is executed, an error code with an error group number in xx and an error classification number in yy is returned. For description of error codes, refer to the RCX manual.

3. Command Reference

This chapter explains the commands of the CaoController::Execute and CaoRobot::Execute methods. For detailed operation of the commands, refer to the YAMAHA Robot Controller User's Manual.

3.1. Controller class

Table 3-1 CaoController::Execute command list

Command	Function	
EMGRST	Clear emergency stop	P.17
ArithmeticExpression	Arithmetic expression operation	P.18
StringExpression	String expression operation	P.18
PointExpression	Point expression operation	P.19
ShiftExpression	Shift expression operation	P.19
SendControlCode	Send control command	P.20
NativeSend	Send data	P.20
NativeReceive	Receive data	P.21

3.1.1. CaoController::Execute("EMGRST") command

Clear the internal emergency stop flag of the RCX controller.

Syntax EMGRST()

Return value : None

Example

```
caoCtrl.Execute(“EMGRST”) ‘ Clear emergency stop
```

3.1.2. CaoController::Execute("ArithmeticExpression") command

Solve the specified arithmetic expression and get the result.

For details about specifying an arithmetic expression, refer to "RCX Series Programming Manual."

Syntax ArithmeticExpression(<bstrExpression:VT_BSTR>)

< bstrExpression > : [in] Arithmetic expression (VT_VARIANT)

Return value : [out] Operation result (VT_R8)

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("arithmeticExpression","SQR(100 * 5)")    ' 2.23606E01
```

3.1.3. CaoController::Execute("StringExpression") command

Solve the specified string expression and get the result.

For details about specifying a string expression, refer to "RCX Series Programming Manual."

Syntax StringExpression(<bstrExpression:VT_BSTR>)

< bstrExpression > : [in] String expression (VT_BSTR)

Return value : [out] Operation result (VT_BSTR)

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("StringExpression", """ABC"" + ""DEF""")    'ABCDEF
```

3.1.4. CaoController::Execute("PointExpression") command

Solve the specified point expression and get the result.

For details about specifying a point expression, refer to "RCX Series Programming Manual."

Syntax PointExpression(<bstrExpression:VT_BSTR>)

< bstrExpression > : [in] Point expression (VT_BSTR)

Return value : [out] Operation result

A different data type is acquired depending on the unit system of the operation result.

VT_I4 | VT_ARRAY Unit: Pulse

VT_R4 | VT_ARRAY Unit: Millimeter

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("PointExpression", "P1+WHRXY")
```

3.1.5. CaoController::Execute("ShiftExpression") command

Solve the specified shift expression and get the result.

For details about specifying a shift expression, refer to "RCX Series Programming Manual."

Syntax ShiftExpression(<bstrExpression:VT_BSTR>)

< bstrExpression > : [in] Shift expression (VT_BSTR)

Return value : [out] Operation result

A different data type is acquired depending on the unit system of the operation result.

VT_I4 | VT_ARRAY Unit: Pulse

VT_R4 | VT_ARRAY Unit: Millimeter

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("ShiftExpression", "S1")
```

3.1.6. CaoController::Execute("SendControlCode") command

Send a one-byte control code to the controller. For details about control codes, refer to the YAMAHA Robot Controller User's Manual.

Syntax SendControlCode (<bytCode:VT_UI1>)

< bytCode > : [in] Control code (VT_UI1)

0x03	^C: Abort ORG, XINC, XDEC, etc.

Return value : None

Example

```
caoCtrl.Execute(“SendControlCode”, &H03)        ‘ Send ^C 0x03
```

3.1.7. CaoController::Execute("NativeSend") command

Send data to the controller via Telnet communications. Specify the send command and parameters with a character string.

Syntax NativeSend (<bstrNativeText:VT_BSTR>)

< bstrNativeText > : [in] Data to send (VT_BSTR)

Return value : None

Example

```
caoCtrl.Execute(“NativeSend”, “@?ALM 0,2”)      ' Send parameter "0,2" with @?ALM command
```

3.1.8. CaoController::Execute("NativeReceive") command

Receive data from the controller via Telnet communications. Get data sent from the controller as a character string. Character strings such as "OK c/r l/f" and "NG c/r l/f" are not included.

Syntax NativeReceive ()

Return value : [out] Data sent from the controller (VT_BSTR)

Example

```
Dim aaa As String
aaa = caoCtrl.Execute(“NativeReceive”)                    ' Receive data sent from controller
```

3.2. Robot class

Table 3-2 CaoRobot::Execute command list

Command	Function	
ABSRST	Origin return of absolute motor axis	P.22
DRIVE	Absolute movement command for axis	P.22
DRIVEI	Relative movement command for axis	P.23
ORIGIN	Origin return of axis of incremental specification	P.23
PMOVE	Pallet movement command	P.24
SERVO	Servo status setting	P.24
HAND	End-effector definition	P.25
RIGHTY	Set to right-hand system	P.25
LEFTY	Set to left-hand system	P.26
SHIFT	Set shift coordinates	P.26
ARCH	Change arch position parameter	P.26
ASPEED	Auto movement speed setting	P.27
AXWGHT	Set axis tip weight	P.27
ORGORD	Set origin return processing order	P.28
OUTPOS	Set out effective position parameter	P.28
PDEF	Palette definition	P.28

3.2.1. CaoRobot::Execute("ABSRST") command

Execute origin return operation of the absolute motor axis of the robot.

Syntax ABSRST()

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute “ABSRST”          ' Origin return
```

3.2.2. CaoRobot::Execute("DRIVE") command

Execute absolute position movement command for an axis.

For details about specifying a position and an option character string, refer to "RCX Series Programming Manual."

Syntax DRIVE(< vntPoints:VT_VARIANT | VT_ARRAY>[, <bstrOption:VT_BSTR>])

< vntPoints > : [in] Movement position specification (VARIANT | VT_ARRAY)

Movement position n	(VARIANT | VT_ARRAY)
Axis number	（VT_I4）
Position specification	（VT_BSTR）

<bstrOption> [in] Option character string (VT_BSTR)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Drive", array(array(array(1, "P11"), array(2, "P10")), "S=100")
```

3.2.3. CaoRobot::Execute("DRIVEI") command

Execute relative position movement command for an axis.

For details about specifying a position and an option character string, refer to "RCX Series Programming Manual."

Syntax DRIVEI(< vntPoints:VT_VARIANT | VT_ARRAY>[, <bstrOption:VT_BSTR>])

< vntPoints > : [in] Movement position specification (VARIANT | VT_ARRAY)

Movement position n	(VARIANT | VT_ARRAY)
Axis number	（VT_I4）
Position specification	（VT_BSTR）

<bstrOption> [in] Option character string (VT_BSTR)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "DriveI", array(array(array(1, "P11"), array(2, "P10")), "S=100")
```

3.2.4. CaoRobot::Execute("ORIGIN") command

Execute origin return of an axis of incremental specification.

Syntax ORIGIN()

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute “ORIGIN”          ' Origin return
```

3.2.5. CaoRobot::Execute("PMOVE") command

Execute pallet movement command.

For details about specifying an option character string, refer to "RCX Series Programming Manual."

Syntax PMOVE(<lPalette:VT_I4>, <lPos:VT_I4>[, <bstrOption:VT_BSTR>])

< lPalette >	:	[in] Pallet number (VT_I4)
< lPos >		[in] Point number (VT_I4)
< bstrOption >		[in] Option character string (VT_BSTR)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "PMove", Array(1, 5, "S=50")
```

3.2.6. CaoRobot::Execute("SERVO") command

Execute pallet movement command.

If the axis number is omitted, all the axes are assumed to be specified.

Syntax SERVO (<bstrState:VT_BSTR>[, <lAxis:VT_I4>])

< bstrState >	:	[in] Servo status (VT_BSTR)
< lAxis >		[in] Axis number (VT_I4)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Servo", array("ON")
```

3.2.7. CaoRobot::Execute("HAND") command

Define the end-effector.

Syntax HAND (<lNo:VT_I4>, <fPara1:VT_R4>, <fPara2:VT_R4>, <fPara3:VT_R4>, <fPara4:VT_R4>, <bROption:VT_BOOL>)

<lNo >	:	[in] End-effector number (VT_I4)
<fPara1>		[in] Parameter 1 (VT_R4)
<fPara2>		[in] Parameter 2 (VT_R4)
<fPara3>		[in] Parameter 3 (VT_R4)
<fPara4>		[in] Parameter 4 (VT_R4)
<bROption>		[in] R option (VT_BOOL)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot("MF20")
rbt.Execute "Hand", array(1, 0, 50.0, 0, 0)          ' Define HAND1 at Y axis 50mm
```

3.2.8. CaoRobot::Execute("RIGHTY") command

Set movement in the right-hand system to the point specified in the rectangular coordinate system.

Syntax RIGHTY ()

Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot("MF20")
rbt.Execute "RIGHTY"          ' Set to right-hand system
```

3.2.9. CaoRobot::Execute("LEFTY") command

Set movement in the left-hand system to the point specified in the rectangular coordinate system.

Syntax LEFTY ()

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute “LEFTY”          ‘ Set to left-hand system
```

3.2.10. CaoRobot::Execute("SHIFT") command

Specify a shift variable and, using the shift data specified in it, set a shift coordinate.

Syntax SHIFT (<bstrShift:VT_BSTR>)

<bstrShift> : [in] Shift specification (VT_BSTR)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "SHIFT", "S1"              ' Use S1 shift
```

3.2.11. CaoRobot::Execute("ARCH") command

Set the arch position parameter.

If the axis number is omitted, all the axes are assumed to be specified.

Syntax ARCH (<lPos:VT_I4>[, <lAxis:VT_I4>])

< lPos > : [in] Arch position parameter (VT_I4)

< lAxis > [in] Axis number (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Arch", array(100000, 2)
```

3.2.12. CaoRobot::Execute("ASPEED") command

Change the auto movement speed.

Syntax ASPEED (<lSpeed:VT_I4>)

< lSpeed >	:	[in] Speed (VT_I4)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "ASPEED", 10
```

3.2.13. CaoRobot::Execute("AXWGHT") command

Change the axis tip weight parameter.

Syntax AXWGHT (<lAxis:VT_I4>, <lVal:VT_I4>)

< lAxis >	:	[in] Axis number (VT_I4)
< lVal >	:	[in] Tip weight (VT_I4)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "AxWght", array(2,5)
```

3.2.14. CaoRobot::Execute("ORGORD") command

Set the axis order parameter for origin return and absolute search operations.

Syntax ORGORD (<lOrder:VT_I4>)

< lOrder > : [in] Axis order parameter (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot("MF20")
rbt.Execute "Orgord", 2
```

3.2.15. CaoRobot::Execute("OUTPOS") command

Change the out effective position parameter.

If the axis number is omitted, all the axes are assumed to be specified.

Syntax OUTPOS (<lPos:VT_I4>[, <lAxis:VT_I4>])

< lPos > : [in] Out effective position (VT_I4)

< lAxis > [in] Axis number (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot("MF20")
rbt.Execute "Outpos", array(100000, 2)
```

3.2.16. CaoRobot::Execute("PDEF") command

Make the pallet definition used to execute the pallet movement command.

Syntax PDEF (<lNo:VT_I4>, <lP12:VT_I4>, <lP13:VT_I4>[, <lP15:VT_I4>])

< lNo > : [in] Pallet number (VT_I4)

< lP12> [in] Number of points between P1-P2 (VT_I4)

< lP13> [in] Number of points between P1-P3 (VT_I4)

< lP15> [in] Number of points between P1-P5 (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "PDef", Array(1, 3, 4, 2)      ' Pallet 3x4x2
```

3.2.17. CaoRobot::Execute("TOLE") command

Change the tolerance parameter.

Syntax TOLE (<lPos:VT_I4>[, <lAxis:VT_I4>])

< lPos > : [in] Out effective position (VT_I4)

< lAxis > [in] Axis number (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Tole", array(1)
```

3.2.18. CaoRobot::Execute("WEIGHT") command

Change the tip weight parameter.

Syntax WEIGHT (<lVal:VT_I4>)

< lOrder > : [in] Tip weight parameter (VT_I4)

Return value : None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Weight", 20       '20kg
```

3.2.19. CaoRobot::Execute("TORQUE") command

Change the maximum torque command value of the specified axis.

Syntax TORQUE (<lAxis:VT_I4>, <lVal:VT_I4>)

< lAxis >	:	[in] Axis number (VT_I4)
< lVal >	:	[in] Torque command value (VT_I4)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Torque", array(2, 100)                ' Maximize two axis torque
```

3.2.20. CaoRobot::Execute("TRQTIME") command

Set the current limit timeout period for the specified axis for using the torque limit specification option of the DRIVE instruction.

Syntax TRQTIME (<lAxis:VT_I4>, <lVal:VT_I4>)

< lAxis >	:	[in] Axis number (VT_I4)
< lVal >	:	[in] Timeout period (VT_I4)
Return value	:	None

Example

```
Dim rbt As Object
Set rbt = caoCtrl.AddRobot(“MF20”)
rbt.Execute "Trqtime", array(2, 2500)    ' Set torque timeout to 2.5 sec
```

Appendix A. RCX Command Correspondence Table

Appendix A.1. Controller class

・ Execute method

Table A-1 CaoController::Execute method – RCX command correspondence table

Command name	RCX command
EMGRST	EMGRST
ArithmeticExpression	?"Arithmetic expression"
StringExpression	?"String expression"
PointExpression	?"Point expression"
ShiftExpression	?"Shift expression"
SendControlCode	-
NativeSend	-
NativeReceive	-

Table A-2 CaoController variable object – RCX command correspondence table

Variable identifier	RCX command
@Timeout	-
@CONFIG	?CONFIG
@EXELVL	?EXELVL
@MOD	?MOD
@MSG	?MSG
@UNIT	?UNIT
@VER	?VER
@MEM	?MEM
@EMG	?EMG
@SELFCHK	?SELFCHK
@OPSLOT	?OPSLOT
P	?"Point expression"
	Pn
S	?"Shift expression"
	Sn

Appendix A.2. Robot class

Table A-3 CaoRobot::Execute method – RCX command correspondence table

Command name	RCX command
ABSRST	ABSRST
DRIVE	DRIVE
	DRIVE2
DRIVEI	DRIVEI
	DRIVEI2
ORIGIN	ORIGIN
PMOVE	PMOVE
	PMOVE2
SERVO	SERVO
	SERVO2
HAND	HAND
	HAND2
RIGHTY	RIGHTY
	RIGHTY2
LEFTY	LEFTY
	LEFTY2
SHIFT	SHIFT
	SHIFT2
ARCH	ARCH
	ARCH2
ASPEED	ASPEED
	ASPEED2
AXWGHT	AXWGHT
	AXWGHT2
ORGORD	ORGORD
	ORGORD2
OUTPOS	OUTPOS
	OUTPOS2
PDEF	PDEF
TOLE	TOLE

	TOLE2
WEIGHT	WEIGHT
	WEIGHT2
TORQUE	TORQUE
	TORQUE2
TRQTIME	TRQTIME
	TRQTIME2

Table A-4 Methods other than CaoRobot::Execute – RCX command correspondence table

Method	RCX command
Accelerrate	ACCEL
	ACCEL2
	DECEL
	DECEL2
Change	CHANGE
	CHANGE2
Halt	^C
Move	MOVE
	MOVE2
	MOVEI
	MOVEI2
Speed	SPEED
	SPEED2

Table A-5 CaoRobot variable object – RCX command correspondence table

Variable identifier	RCX command
@ARM	?ARM
@ORIGIN	?ORIGIN
@ABSRST	?ABSRST
@SERVO	?SERVO
@SPEED	?SPEED
@WHERE	?WHERE
@WHERE2	?WHERE2

@WHRXY	?WHRXY
@WHRXY2	?WHRXY2
@SHIFT	?SHIFT
@HAND	?HAND