# YAMAHA
# SR1 Provider

Version 1.0.1

# User's Guide

September 27, 2019

Remarks:

## [Revision History]

| Version | Date | Content |
|---|---|---|
| 1.0.0 | 2012-10-31 | First edition. |
| 1.0.0 | 2013-04-26 | Added the note for “echo-back” function of the YAMAHA controller. |
| | | |
| 1.0.1 | 2019-09-27 | Fixed license check algorithm. |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

## [Hardware]

| Model | Version | Notes |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

## 【Attention】

**Additional license for "SR1 Provider" is required to use this provider.**

## Contents

# 1. Introduction

This document is a user's guide for the single/double axis CAO provider for the YAMAHA robot SR1/DRCX series. The CAO provider described in this manual (CaoProvSR1.dll) is called the SR1 provider.
The next chapter provides the outline of the SR1 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.

# 2. Outline of Provider

## 2.1. Outline

The SR1 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 SR1 provider, ORiN2SDK needs to be installed or registry needs to be manually registered according to the table below.

**Table 2-1 SR1 provider**

| | |
|---|---|
| File name | CaoProvSR1.dll |
| ProgID | CaoProv.YAMAHA.SR1 |
| Registry registration | regsvr32 CaoProvSR1.dll |
| Remove registry registration | regsvr32 /u CaoProvSR1.dll |

## 2.2. Methods and properties

### 2.2.1. CaoWorkspace::AddController method

SR1 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 user name, password and Telnet port number.

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>)

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 YAMAHA controller. (Default: admin) |
| [Password=<Password>] | Specify the password to log in. (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>[:<Flow>]]]]"

<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

< Flow> : Flow control. '1' -Xon/Xoff.. '2'-hardware control.
It is possible to specify it by taking OR.
(default: Flow control '0'-none)

### 2.2.2. CaoController::AddRobot method

The argument of the AddRobot method of the CaoController class specifies the robot name (BSTR type). 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 (VT_BSTR)

< bstrOption > : [in] Option character string (not used)

### 2.2.3. CaoController::AddVariable method

The AddVariable method of CaoController class is a method for access to variables. In the SR1 provider, specify a system variable for the variable name.

Refer to Table 2-5 about the system variables implemented in the SR1 provider.

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

< bstrVariableName > : [in] Variable name (VT_BSTR)

<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 "3Command Reference".

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

| | | |
|---|---|---|
| < vntRet > | : | [out] Return value of command (VT_VARIANT) |
| < bstrCmd > | : | [in] Command (VT_BSTR) |
| < vntParam > | : | [in] Parameter (VT_VARIANT) |

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("SRVO", 1)      ' Servo ON
```

### 2.2.5. CaoRobot::AddVariable method

The AddVariable method of CaoRobot class is a method for access to variables. In the SR1 provider, specify a system variable for the variable name.

Refer to Table 2-6 about the system variables implemented in the SR1 provider.

### 2.2.6. 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.

### 2.2.7. 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 | VT_ARRAY)>

| | | |
|---|---|---|
| < lComp > | : | [in] Command number (VT_I4) Refer to Table 2-3. |
| < vntParam > | : | [in] Destination parameter array (VT_VARIANT \| VT_ARRAY) Refer to Table 2-4. |

**Table 2-3 Move command first argument – MOV command correspondence table**

| First argument | Command | Explanation |
|---|---|---|
| 1 | MOVD | Move to the specified coordinate position. |
| 2 | MOVA | Move to the data position for the specified point number. |
| 3 | MOVI | Move as much as the data amount for the specified point number from the current position. |
| 4 | MOVF | Not used. |
| 5 | MOVL | DRCX (double axis) controller only. Move to the data position for the specified point number using linear interpolation. |
| 6 | MOVC | DRCX (double axis) controller only. Execute arc interpolation movement through the point specified by the number. |
| 7 | MOVM | Move to the specified pallet work position in the matrix. |

Specify a destination parameter in the second argument as an array.

**Table 2-4 Second argument parameter of Move command**

| MOV command | Second argument parameter | Remarks |
|---|---|---|
| MOVD | <X axis position (mm):VT_R8>,<Speed:VT_I4> | |
| MOVA | <Point number:VT_I4>,<Speed:VT_I4> | Point variable P can also be used in the point number. |
| MOVI | <Point number:VT_I4>,<Speed:VT_I4> | Point variable P can also be used in the point number. |
| MOVF | Not used. | Not used. |
| MOVL | <Point number:VT_I4>,<Maximum speed:VT_I4> | Point variable P can also be used in the point number. |
| MOVC | <Point number:VT_I4>,<Maximum speed:VT_I4>,<Path specification:VT_I4> | Point variable P can also be used in the point number. |
| MOVM | <Pallet work position:VT_I4>,<Speed:VT_I4> | For the pallet work position, specify a value between 1 and 65025 (255×255). |

Example

・ Example of using Move for single-axis controller

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")

aaa.Move 1, Array(150.5, 100)      ' MOVD command: Move to 150.5 mm position at 100% speed
aaa.Move 2, Array(10, 50)          ' MOVA command: Move to point number 10 at 50% speed
aaa.Move 3, Array(11, 30)          ' MOVI command: Move as much as data amount at point
                                     number 11 from current position at 30% speed
aaa.Move 7, Array(1, 100)          ' MOVM command: Move to matrix definition 1 position at 100%
speed
```

・ Example of using Move for double-axis controller

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")

aaa.Move 5, Array(20, 100)         ' MOVL command: Move to point number 20 at 100% speed
aaa.Move 6, Array(30, 50, 0)       ' MOVC command: Move along arc path formed by three points,
                                     current position, point number 30, and point number 31, at 50%
                                     speed
```

### 2.2.8. CaoRobot::Execute method

Execute the robot-class 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 "3Command Reference".

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

| | | |
|---|---|---|
| < vntRet > | : | [out] Return value of command (VT_VARIANT) |
| < bstrCmd > | : | [in] Command (VT_BSTR) |
| < vntParam > | : | [in] Parameter (VT_VARIANT) |

Example

```
Dim vRes As Variant
vRes = caoCtrl.Execute("SRVO", 1)      ' Servo ON
```

### 2.2.9. CaoVariable::get_value property

Get the value of the variable corresponding to the object.

For details about the variable implementation status and data type, refer to Table 2-5, Table 2-6.

### 2.2.10. CaoVariable::put_value property

Set the value of the variable corresponding to the object.

For details about the variable implementation status and data type, refer to Table 2-5, Table 2-6.

## 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 |
| @VER | VT_BSTR | Get the version information of the controller. | √ | - |
| @CLOCK | VT_BSTR | Get the total running time of the controller. | √ | - |
| @EMG | VT_I2 | Get the emergency stop status.<br>0: Emergency stop clear status, 1: Emergency stop status | √ | - |
| @MODE | VT_I2 | Get the status of the robot.<br>0: Stop status<br>1: Program running due to PC communications<br>2: Program running due to I/O command | √ | - |
| @Timeout | VT_I4 | Set/get the timeout period (msec) for communication. | √ | √ |

### 2.3.2. Robot class

**Table 2-6 Robot class system variable list**

| Variable identifier | Data type | Explanation | Attribute | |
|---|---|---|---|---|
| | | | get | put |
| @POS[n] | VT_R8 | Get the current position of the robot.<br>[n]: Axis number, 0: All axes, 1: X axis, 2: Y axis, Default: Single axis)<br>The axis specification is used for DRCX (double axis) controller. | √ | - |
| @NO | VT_BSTR | Get the program number of the currently running program. In multi task mode, this returns information about the program of the selected task.<br>(Example) "10/1": No.1 is the first program and currently No.10 program is running. | √ | - |
| @SNO | VT_BSTR | Get the current step number. In multi task mode, this returns information about the program of the selected task. | √ | - |
| @TNO | VT_BSTR | Get the currently selected task number. | √ | - |
| @PNO | VT_BSTR | Get the currently selected point number. In multi task mode, this returns information about the program of the selected task. | √ | - |
| @MEM | VT_BSTR | Get the number of steps that can be added. | √ | - |
| @ROBOT | VT_BSTR | Get the currently set robot type. | √ | - |
| @PVA | VT_BSTR | Get the point variable P number. In multi task mode, this returns information about the program of the selected task.<br><Note> Range in which value retention is effective<br>Power-off -> Retained<br>Operation including a program reset -> Initialized to 0 | √ | - |
| @MSEL | VT_BSTR | Get the currently specified matrix pallet number. In multi task mode, this returns information about the program of the selected task. | √ | - |

| | | | | |
|---|---|---|---|---|
| @SHFT | VT_BSTR | Get the currently selected shift data. In multi task mode, this returns information about the program of the selected task. | √ | - |
| @SRVO[n] | VT_I2 | Get the servo status.<br>0: Servo OFF<br>1: Servo ON<br>[n]: Axis number, 0: All axes, 1: X axis, 2: Y axis, Default: Single axis)<br>The axis specification is used for DRCX (double axis) controller. | √ | - |
| @ORG[n] | VT_I2 | Get the origin return completion status.<br>0: Origin return not completed<br>1: Origin return completed<br>[n]: Axis number, 0: All axes, 1: X axis, 2: Y axis, Default: Single axis)<br>The axis specification is used for DRCX (double axis) controller. | √ | - |

## 2.4. Error code

In the SR1 provider, the following unique error codes are defined. For common ORiN2 errors, refer to the error code section in "ORiN2 Programming Guide".

**Table 2-7 List of original error codes**

| Error name | Error code | Explanation |
|---|---|---|
| E_CAOP_NO_LICENSE | 0x80100000 | There is no license.<br>Purchase an additional license. |
| SR1 command error | 0x8011xxxx | If an error occurs while an SR1 command is executed, an error code with an error number in xxxx is returned.<br>For description of error codes, refer to the SR1 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 | |
|---|---|---|
| ?ALM | Get the history of alarms that occurred in the past. | P.15 |
| ?ERR | Get the history of errors that occurred in the past. | P.16 |
| ?PRM | Get the specified parameter. | P.16 |
| ?STP | Get the total number of steps of a specified program. | P.17 |
| ALMRST | Reset an alarm. | P.17 |
| SendControlCode | Send a control command. | P.17 |
| NativeSend | Send data via Telnet communications. | P.18 |
| NativeReceive | Receive data via Telnet communications. | P.18 |

### 3.1.1. CaoController::Execute("?ALM") command

Get an alarm that occurred on the controller. Data can be acquired in a string array of <Number>,<Alarm date>,<Alarm description>. If the number of display is not specified, the latest alarm data is acquired.
For details about alarm description, refer to the YAMAHA Robot Controller User's Manual.

Syntax ?ALM ( < vntAlmParam:VT_VARIANT > )

< vntAlmParam > : [in] Acquired alarm parameter (VT_VARIANT)

| | |
|---|---|
| <lNo> | History number: VT_I4 |
| [<lCount>] | Number of display: VT_I4 |

Return value : [out] Alarm string array (VT_BSTR | VT_ARRAY)

Example

```
Dim aaa As Variant
aaa = caoCtrl.Execute(“?ALM”, Array(0, 2))                ‘ Get two latest alarms
```

### 3.1.2. CaoController::Execute("?ERR") command

Get an error that occurred on the controller. Data can be acquired in a string array of <Number>,<Error date>,<Error description>. If the number of display is not specified, the latest error data is acquired.
For details about error description, refer to the YAMAHA Robot Controller User's Manual.

Syntax ?ERR ( <vntErrParam:VT_VARIANT> )

< vntErrParam > : [in] Acquired error parameter (VT_VARIANT)

| | |
|---|---|
| <lNo> | History number: VT_I4 |
| [<lCount>] | Number of display: VT_I4 |

Return value : [out] Error string array (VT_BSTR | VT_ARRAY)

Example

```
Dim aaa As Variant
aaa = caoCtrl.Execute("?ERR", Array(0, 2))                ' Get two latest errors
```

### 3.1.3. CaoController::Execute("?PRM") command

Get the setting parameters of the controller. Data can be acquired in an integer array of <Parameter number>,<Data>. If one array element is specified, data for one parameter is acquired.
For details about parameters, refer to the YAMAHA Robot Controller User's Manual.

Syntax ?PRM ( <vntParamData:VT_VARIANT> )

< vntParamData > : [in] Parameter data (VT_VARIANT)

| | |
|---|---|
| <lBeginPrmNo> | Start parameter number |
| [<lEndPrmNo>] | End parameter number |

Return value : [out] Parameter number and data (VT_I4 | VT_ARRAY)

Example

```
Dim aaa As Variant
aaa = caoCtrl.Execute("?PRM", Array(110, 114))   ' Get parameter numbers 110 to 114
```

### 3.1.4. CaoController::Execute("?STP") command

Get the total number of steps of a specified program. Data can be acquired in integer type.

Syntax ?STP ( <lProgramNo:VT_I4> )

< lProgramNo > : [in] Program number (VT_I4)
Return value : [out] Total number of steps of the program (VT_I4)

Example

```
Dim aaa As Variant
aaa = caoCtrl.Execute("?STP", 0)        ' Get total number of steps of program number 0
```

### 3.1.5. CaoController::Execute("ALMRST") command

Reset an alarm that occurred on the controller. Only a resettable alarm can be reset. For details about alarms that cannot be reset, refer to the YAMAHA Robot Controller User's Manual.

Syntax ALMRST ()

Return value : None

Example

```
caoCtrl.Execute("ALMRST") ' Reset alarm
```

### 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. |
| 0x1A | ^Z: End of data transmission |
| 0x02 | ^B: Stop alarm message output |

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> )

< 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 | |
|---|---|---|
| ORG | Execute origin return. | P.20 |
| SRVO | Turn ON/OFF the servo. | P.20 |
| X+ | Move the X axis in a plus direction for a distance of 1/100 the JOG movement speed. | P.21 |
| X- | Move X axis in a minus direction for a distance of 1/100 the JOG movement speed. | P.21 |
| Y+ | Move Y axis in a plus direction for a distance of 1/100 the JOG movement speed. | P.21 |
| Y- | Move Y axis in a minus direction for a distance of 1/100 the JOG movement speed. | P.22 |
| XINC | Keep moving the X axis in a plus direction at the JOG movement speed. | P.22 |
| XDEC | Keep moving the X axis in a minus direction at the JOG movement speed. | P.23 |
| YINC | Keep moving the Y axis in a plus direction at the JOG movement speed. | P.23 |
| YDEC | Keep moving the Y axis in a minus direction at the JOG movement speed. | P.24 |
| DRVD | Move the specified axis to the specified coordinate position. | P.24 |
| DRVA | Move the specified axis to a point data position. | P.25 |
| DRVI | Move the specified axis as much as a point data amount from current position. | P.25 |
| ACHA | Define the arch motion with a position specified. | P.26 |
| ACHI | Define the arch motion with a distance specified. | P.26 |
| P | Set point variable P. | P.27 |
| P+ | Add 1 to point variable P. | P.27 |
| P- | Subtract 1 from point variable P. | P.28 |
| MAT | Define a matrix. | P.28 |
| MSEL | Set a matrix number to be used for matrix movement of the Move method. | P.29 |
| SHFT | Shift the position data as much as the specified point data amount. | P.29 |
| WRITE | Write point data. | P.30 |
| ?P | Get the specified point data. | P.30 |
| ?MAT | Get the specified matrix data. | P.31 |

### 3.2.1. CaoRobot::Execute("ORG") command

Execute the push origin return operation and output the machine reference amount at normal end.

Syntax ORG ( [<lAxisNo:VT_I2>] )

| | | |
|---|---|---|
| [<lAxisNo>] | : | [in] Axis number 1: X axis, 2: Y axis |
| Return value | : | [out] Machine reference amount after completion of origin return (VT_I4) |

Example

```
Dim aaa As Object
Dim bbb As Long

Set aaa = caoCtrl.AddRobot("AAA")
bbb = aaa.Execute("ORG")              ' Origin return
```

### 3.2.2. CaoRobot::Execute("SRVO") command

Execute servo ON/OFF control.

Syntax SRVO ( <lSrvoState:VT_I2>[,<lAxisNo:VT_I2>] )

| | | |
|---|---|---|
| < lSrvoState > | : | [in] Servo status (VT_I2)<br>0: Servo OFF, 1: Servo ON |
| [<lAxisNo>] | : | [in] Axis number 1: X axis, 2: Y axis |
| Return value | : | None |

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("SRVO",1)                 ' Servo ON
```

### 3.2.3. CaoRobot::Execute("X+") command

Move the X axis in a plus direction for the distance specified by the following expression:

Distance (mm) = 1 x (PRM201/100) PRM201:JOG movement speed (mm/s)

Syntax X+ ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("X+")          ' Move X in plus direction by (PRM201)/100
```

### 3.2.4. CaoRobot::Execute("X-") command

Move the X axis in a minus direction for the distance specified by the following expression:

Distance (mm) = 1 x (PRM201/100) PRM201:JOG movement speed (mm/s)

Syntax X- ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("X-")          ' Move X in plus direction by (PRM201)/100
```

### 3.2.5. CaoRobot::Execute("Y+") command

Command for DRCX only.

Move the Y axis in a plus direction for the distance specified by the following expression:

Distance (mm) = 1 x (PRM12/100) PRM12: Teach movement data (%)

If the robot is the rotation axis, the unit is deg/sec.

Syntax Y+ ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("Y+")                    ' Move Y in plus direction by (PRM12)/100
```

### 3.2.6. CaoRobot::Execute("Y-") command

Command for DRCX only.

Move the Y axis in a minus direction for the distance specified by the following expression:

Distance (mm) = 1 x (PRM12/100)  PRM12: Teach movement data (%)

If the robot is a rotation axis, the unit is deg/sec.

Syntax  Y- ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("Y-")                    ' Move Y in minus direction by (PRM12)/100
```

### 3.2.7. CaoRobot::Execute("XINC") command

Keep moving the X axis in a plus direction at PRM201 (JOG movement speed) until a control code (^C) is entered or the software limit is reached.

**[Attention]**

The software limit is not effective until origin return is completed.

Syntax  XINC ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("XINC")                  ' Keep moving X in plus direction at JOG movement speed
```

### 3.2.8. CaoRobot::Execute("XDEC") command

Keep moving the X axis in a minus direction at PRM201 (JOG movement speed) until a control code (^C) is entered or the software limit is reached.

**[Attention]**

The software limit is not effective until origin return is completed.

Syntax XDEC ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("XDEC")          ' Keep moving X in minus direction at JOG movement speed
```

### 3.2.9. CaoRobot::Execute("YINC") command

Command for DRCX only.

Keep moving the Y axis in a plus direction at PRM201 (JOG movement speed) until a control code (^C) is entered or the software limit is reached. If the robot is a rotation axis, the unit is deg/sec.

**[Attention]**

The software limit is not effective until origin return is completed.

Syntax YINC ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("YINC")          ' Keep moving Y in plus direction at JOG movement speed
```

### 3.2.10. CaoRobot::Execute("YDEC") command

Command for DRCX only.

Keep moving the Y axis in a minus direction at PRM201 (JOG movement speed) until a control code (^C) is entered or the software limit is reached. If the robot is a rotation axis, the unit is deg/sec.

**[Attention]**

The software limit is not effective until origin return is completed.

Syntax YDEC ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("YDEC")          ' Keep moving Y in plus direction at JOG movement speed
```

### 3.2.11. CaoRobot::Execute("DRVD") command

Command for DRCX only.

Move the specified axis to the specified coordinate position.

Syntax DRVD (<vntDRVParam:VT_VARIANT | VT_ARRAY> )

< vntDRVParam > : [in] Drive information (VT_VARIANT | VT_ARRAY)

Array data:

| | |
|---|---|
| < vntAxis > | Axis number 1: X axis, 2: Y axis |
| <vntPosition> | Movement position (mm) when the robot is set to a rotation axis (deg) |
| < vntSpeed > | Speed: 1 to 100 |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("DRVD", Array(1,150.55,100))          ' Move X axis to X=150.55 position at 100% speed
```

### 3.2.12. CaoRobot::Execute("DRVA") command

Command for DRCX only.

Move the specified axis to the specified point data position.

Syntax DRVA (<vntDRVParam:VT_VARIANT | VT_ARRAY> )

< vntDRVParam > : [in] Drive information (VT_VARIANT | VT_ARRAY)

Array data:

| | |
|---|---|
| < vntAxis > | Axis number 1: X axis, 2: Y axis |
| <vntPointNo> | Point number: 0 to 999<br>Point variable P can also be used. |
| < vntSpeed > | Speed: 1 to 100 |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("DRVA", Array(1,123,100))          ' Move X axis to point 123 at 100% speed
```

### 3.2.13. CaoRobot::Execute("DRVI") command

Command for DRCX only.

Move the specified axis by the specified point data position amount from the current position.

Syntax DRVI (<vntDRVParam:VT_VARIANT | VT_ARRAY> )

< vntDRVParam > : [in] Drive information (VT_VARIANT | VT_ARRAY)

Array data:

| | |
|---|---|
| < vntAxis > | Axis number 1: X axis, 2: Y axis |
| <vntPointNo> | Point number: 0 to 999<br>Point variable P can also be used. |
| < vntSpeed > | Speed: 1 to 100 |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("DRVI", Array(2,150.55,100))        Move Y axis as much as point 123 data amount from
                                                current position at 100% speed
```

### 3.2.14. CaoRobot::Execute("ACHA") command

Command for DRCX only.

Define the arch motion with a position specified (absolute position in reference to origin).

Syntax ACHA ( <vntACHParam:VT_VARIANT | VT_ARRAY> )

< vntACHParam > : [in] Arch information (VT_VARIANT | VT_ARRAY)

Array data:

| | |
|---|---|
| < vntAxis > | Axis number 1: X axis, 2: Y axis |
| < vntPosition > | Specifiable position: -9999 to 9999 (mm) |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("ACHA", Array(2,10))                ' Arch motion definition to return to Y=10.00 point
```

### 3.2.15. CaoRobot::Execute("ACHI") command

Command for DRCX only.

Define the arch motion with a position specified (relative position in reference to the current position).

Syntax ACHI (<vntACHParam:VT_VARIANT | VT_ARRAY>)

< vntACHParam > : [in] Arch information (VT_VARIANT | VT_ARRAY)

Array data:

| | |
|---|---|
| < vntAxis > | Axis number 1: X axis, 2: Y axis |
| < vntPosition > | Specifiable distance: -9999 to 9999 (mm) |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("ACHI", Array(2,-100))      ' Arch motion definition to return for distance of Y=-100.00
```

### 3.2.16. CaoRobot::Execute("P") command

Set point variable P. A point number between 0 and 999 can be specified.

**[Attention]**

Although the point variable data is retained even if the controller power is turned OFF, an operation that causes a program reset initializes the point variable to 0.

Syntax P ( <lPNo:VT_I4> )

< lPNo > : [in] Point number (VT_I4): A point number between 0 and 999

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("P", 100)          ' Set point variable P to 100
```

### 3.2.17. CaoRobot::Execute("P+") command

Add 1 to point variable P.

Syntax P+ ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("P", 100)                  ' Set point variable P to 100
aaa.Execute("P+")                      ' Set point variable P to 101 (P <- P+1)
```

### 3.2.18. CaoRobot::Execute("P-") command

Subtract 1 from point variable P.

Syntax P- ()

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("P", 100)                ' Set point variable P to 100
aaa.Execute("P-")                    ' Set point variable P to 99 (P <- P-1)
```

### 3.2.19. CaoRobot::Execute("MAT") command

Define a matrix. The numbers of rows and columns can be set to a value between 1 and 255. The pallet number can be set to a unique value between 0 and 31.

Syntax MAT ( <vntMatParam:VT_I4 | VT_ARRAY> )

< vntMatParam > : [in] Matrix definition information (VT_I4 | VT_ARRAY)

Array data:

| | |
|---|---|
| < lRowCount > | Number of rows: 1 to 255 |
| < lColCount > | Number of columns: 1 to 255 |
| <lPalltNo> | Pallet number: 0 to 31 |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("MAT", Array(5,2,1))          ' Set 5 x 2 matrix for first pallet
```

### 3.2.20. CaoRobot::Execute("MSEL") command

Syntax MSEL ( <lPalletNo:VT_I4> )

< lPalletNo > : [in] Pallet number (VT_I4)
Pallet number between 0 and 31 for matrix identification
Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("MAT", Array(5,2,1))          ' Set 5 x 2 matrix for first pallet
aaa.Execute("MSEL", 1)                    ' Select first pallet
aaa.Move 7, Array(1,100)                  ' Move to first pallet position at 100% speed
```

### 3.2.21. CaoRobot::Execute("SHFT") command

Shift the position data as much as the specified point data amount.

This is valid until the SHFT statement is executed again or a program reset occurs.

Syntax SHFT ( <lPointNo:VT_I4> )

< lPointNo > : [in] Point number (VT_I4): A point number between 0 and 999
Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("SHFT", 1)              ' Data amount shift setting for point number 1
```

### 3.2.22. CaoRobot::Execute("WRITE") command

Write point data. To write point data, specify a string array. Points to which data is written do not need to be sequential numbers.

Syntax WRITE ( <vntPointData:VT_BSTR | VT_ARRAY> )

< vntPointData > : [out] Point setting string (VT_BSTR | VT_ARRAY)

| | |
|---|---|
| Px=<Position data> | x: Point number<br>The position data is VT_R8. |

Return value : None

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("WRITE", Array("P0=0.00","P1=350.00","P254=-0.27")) ' Write to point
```

### 3.2.23. CaoRobot::Execute("?P") command

Get point data at the specified point number. The first argument specifies the point number at which data is to be acquired as an array. To acquire only one point, specify an array with one element. Point data at unregistered point numbers is skipped.

Syntax ?P (<vntPointData:VT_I4 | VT_ARRAY> )

< vntPointData > : [in] Point number (VT_I4| VT_ARRAY)

| | |
|---|---|
| <lBeginPointNo> | Point number: VT_I4 |
| [<lEndPointNo>] | Point number: VT_I4 |

Return value : [out] Point number (VT_BSTR |VT _ARRAY)

P(lBeginPointNo) = Position data,...

, P(lEndPointNo) = Position data array

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("?P", Array(15,24))                    ' Get data for point numbers from 15 to 24
```

### 3.2.24. CaoRobot::Execute("?MAT") command

Get the matrix definition. Specify a unique number between 0 and 31 for matrix identification.

Syntax ?MAT ( <lMatNo:VT_I4> )

< lMatNo > : [in] Matrix number (VT_I4)

Return value : [out] Matrix definition data (VT_I4 | VT_ARRAY)

| | |
|---|---|
| Number of rows | VT_I4 |
| Number of columns | VT_I4 |

Example

```
Dim aaa As Object

Set aaa = caoCtrl.AddRobot("AAA")
aaa.Execute("?MAT", 1)                ' Get data for pallet number 1
```

# Appendix A. SR1 (DRCX) Command Correspondence Table

## Appendix A.1. Controller class

**Table A-1 CaoController::Execute method – SR1 command correspondence table**

| Command name | SR1 command |
|---|---|
| ?ALM | ?ALM |
| ?ERR | ?ERR |
| ?PRM | ?PRM |
| ?STP | ?STP |
| ALMRST | ALMRST |
| SendControlCode | - |
| NativeSend | - |
| NativeReceive | - |

**Table A-2 CaoController variable object – SR1 command correspondence table**

| Variable identifier | SR1 command |
|---|---|
| @Timeout | - |
| @VER | ?VER |
| @CLOCK | ?CLOCK |
| @EMG | ?EMG |
| @MODE | ?MODE |

## Appendix A.2. Robot class

### Table A-3 CaoRobot::Execute method – SR1 command correspondence table

| Command name | SR1 command |
|---|---|
| ORG | ORG |
| SRVO | SRVO |
| X+ | X+ |
| X- | X- |
| Y+ | Y+ (DRCX) |
| Y- | Y- (DRCX) |
| XINC | XINC |
| XDEC | XDEC |
| YINC | YINC (DRCX) |
| YDEC | YDEC (DRCX) |
| P | P |
| ?P | ?P |
| P+ | P- |
| MAT | MAT |
| ?MAT | ?MAT |
| MSEL | MSEL |
| SHFT | SHFT |
| DRVD | DRVD (DRCX) |
| DRVA | DRVA (DRCX) |
| DRVI | DRVI (DRCX) |
| ACHA | ACHA (DRCX) |
| ACHI | ACHI (DRCX) |
| WRITE | WRITE PNT |

**Table A-4 Methods other than CaoRobot::Execute – SR1 command correspondence table**

| Method | SR1 command |
|---|---|
| Move | MOVD |
| | MOVA |
| | MOVI |
| | MOVF (not used) |
| | MOVL (DRCX) |
| | MOVC (DRCX) |
| | MOVM |
| Halt | ^C |

**Table A-5 CaoRobot variable object – SR1 command correspondence table**

| Variable identifier | SR1 command |
|---|---|
| @POS | ?POS |
| @NO | ?NO |
| @SNO | ?SNO |
| @TNO | ?TNO |
| @PNO | ?PNO |
| @MEM | ?MEM |
| @ROBOT | ?ROBOT |
| @PVA | ?PVA |
| @MSEL | ?MSEL |
| @SHIFT | ?SHIFT |
| @SRVO | ?SRVO |
| @ORG | ?ORG |