SysmacStudio provider
SysmacStudio of OMRON

Version 1.0.3

User's Guide

April 20, 2023

[Remark]

【Revision history】

Version	Dating	Content
1.0.0.0	2012-06-22	First edition
1.0.0	2012-07-17	Change the document version rules.
1.0.1	2012-12-11	Adding Type options
1.0.2	2018-10-15	Modify command reference. Add how to use SysmacStudio. Fixed the process when connecting.
1.0.3	2023-04-20	Delete unnecessary file output processing. Fixed typos.

Table of contents

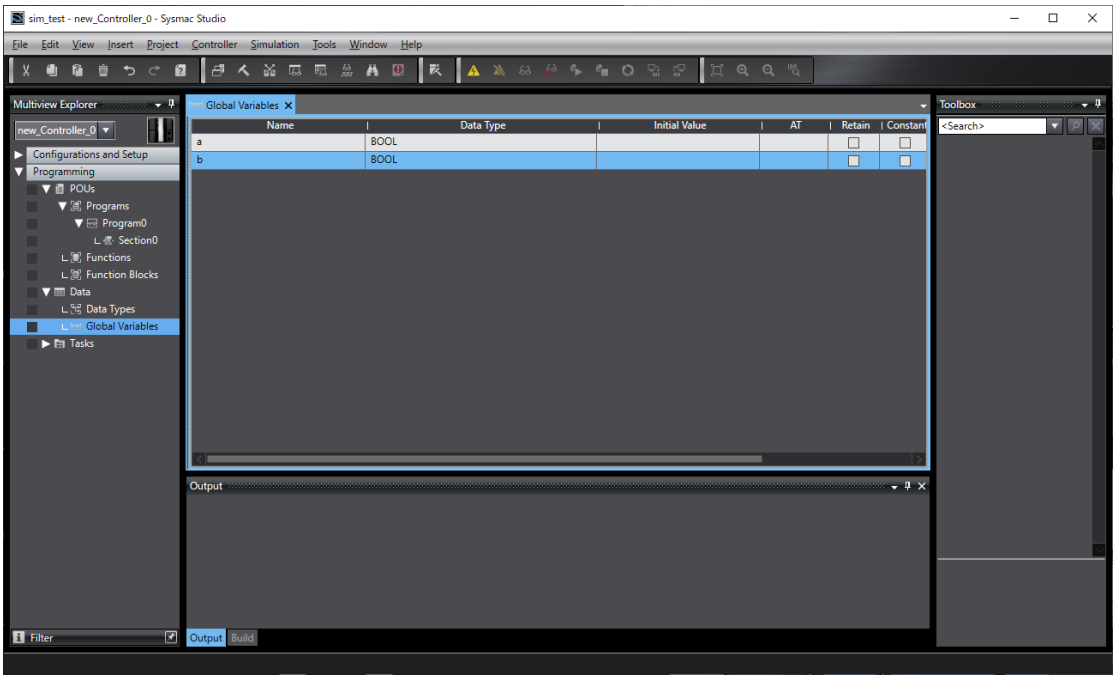
1. Introduction

This document is a User's Guide for SysmacStudio providers, CAOs providers that communicate with OMRON's SysmacStudio.

2. How to set the Sysmac Studio

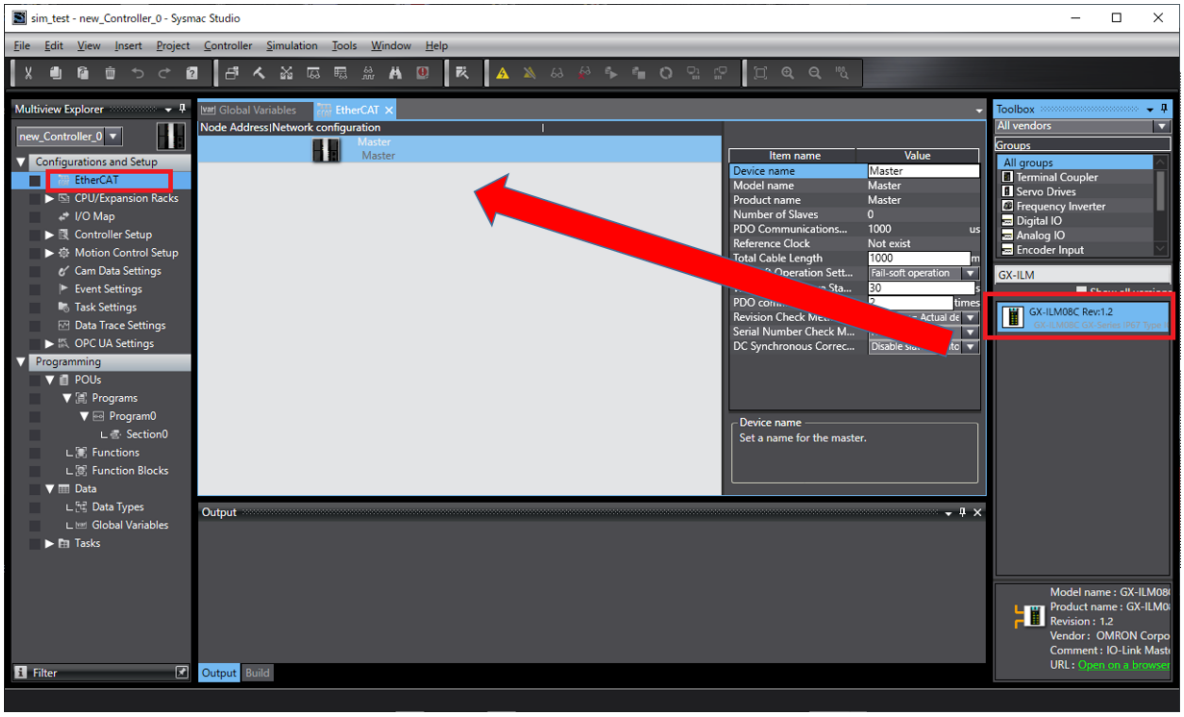
<Setting VAR Variables>

Define the global variables as follows:The path for global variable a is VAR://a.

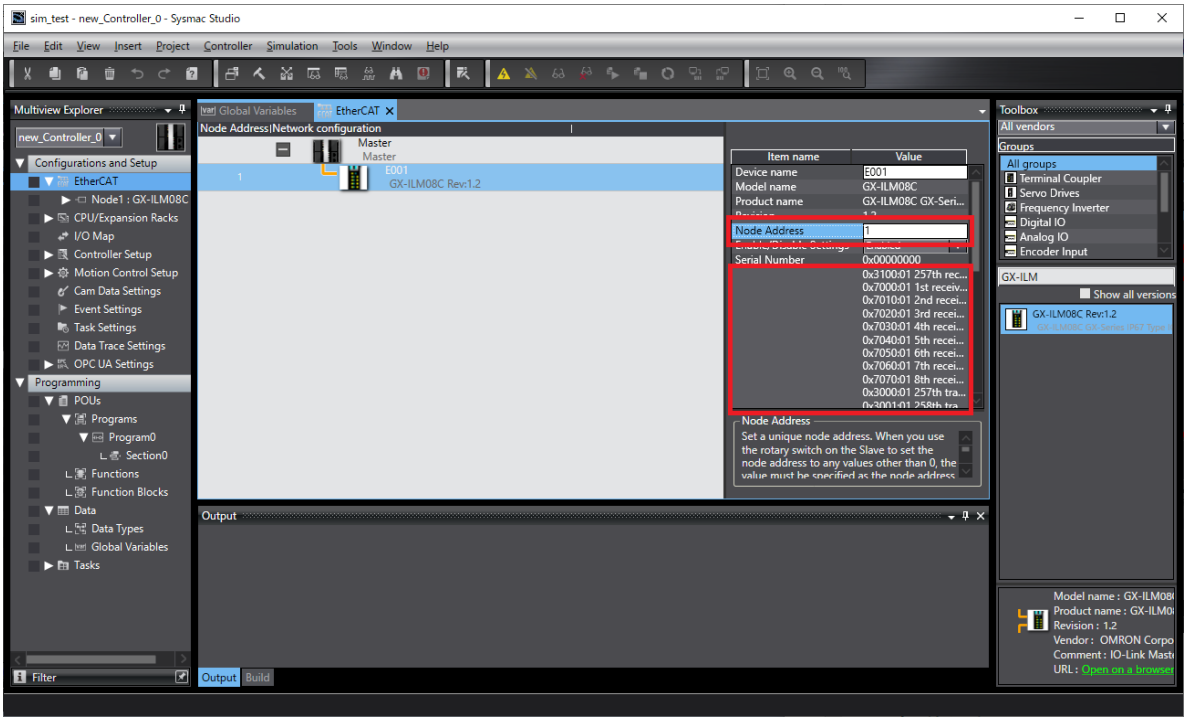


<How to Set the PDO Variable>

Displays "Configurations and Setup" > "EtherCAT". Select the device you want to add to the configuration from the toolbox and drag and drop it to the Master device.

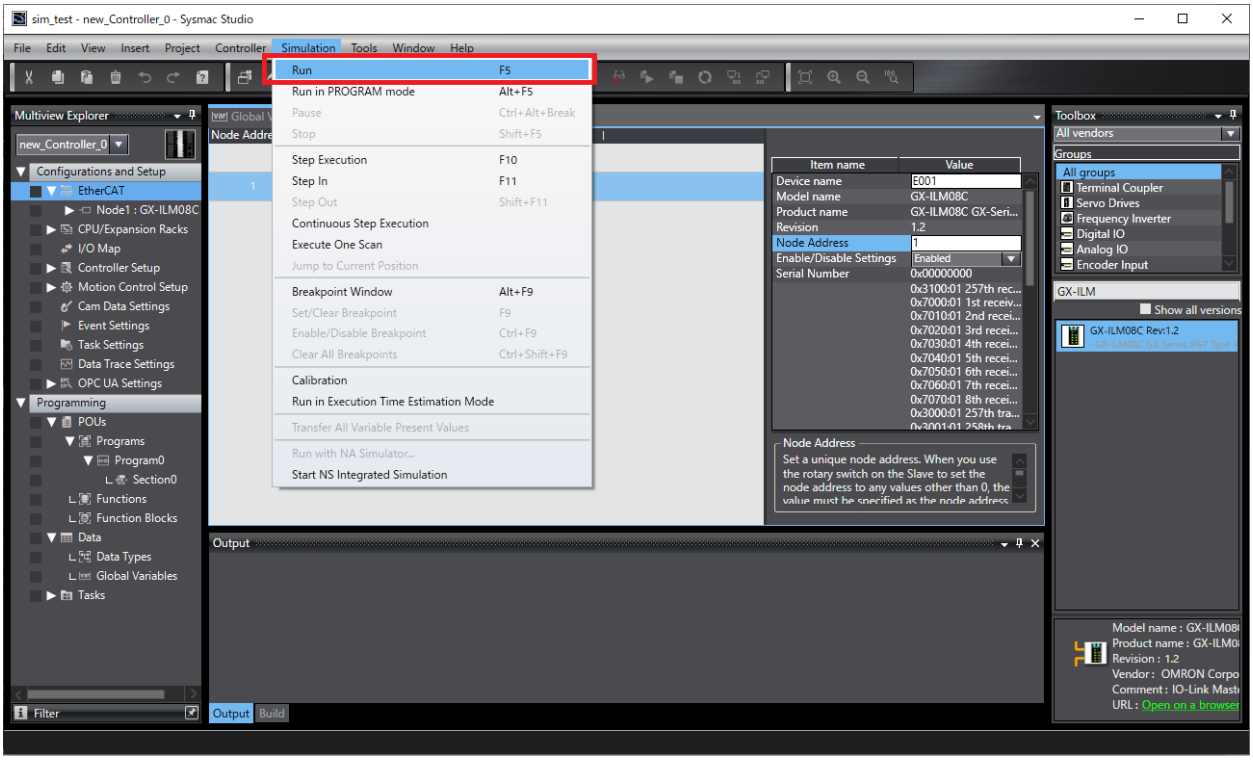


Specify the node address from the properties of the added device. The item displayed in the PDO mapping setting is the available PDO variable.



<Execution of the simulation>

To use SysmacStudio providers, you must run simulations. After completion of the build, press Simulation > Run on the toolbar to start the simulation.



3. Provider overview

3.1. Overview

SysmacStudio providers have a DLL (Dynamic Link Library) file format, which is detailed in Table 3-1.

Table 3-1 SysmacStudio providers

File name	CaoProvSysStd.dll
ProgID	CaoProv.OMRON.Sysmac.Studio
Registry registration[1]	Regsvr32 CaoProvSysStd.dll
Deletion of Registry Registration	Regsvr32 /u CaoProvSysStd.dll

3.2. Method and Properties

3.2.1. CaoWorkspace::AddController method

Connect to the SysmacStudio.

Syntax AddController(<bstrCtrlName:BSTR, <bstrProvName:BSTR>,
<bstrPcName:BSTR> [, <bstrOption:BSTR>])

bstrCtrlName : [in] Controller name

bstrProvName : [in] Provider name. Fixed value="CaoProv.OMRON.Sysmac.Studio".

bstrPcName : [in] the name of the provider's executing machine

bstrOption : [in] option string

Table 3-2 Optional character strings of CaoWorkspace::AddController

Option	Meaning
Server =<IP Address>	Specify the IP address of the PC with the SysmacStudio to which the PC is connected. (Default: 127.0.0.1)
Port [=<port number>]	Specifies the port number of the PC with the SysmacStudio to which the PC is connected. (Default: 7000)

3.2.2. CaoController::AddVariable method

Gets the CaoVariable objects from which you want to get the input-device information. Please refer to cahpter 3.3.1 for input device names that can be used.

Syntax AddVariable(<bstrName:BSTR >, [<bstrOption:BSTR>]

[1] You do not need to manually register/delete ORiN SDK installations.

bstrCtrlName : [in] variable name
bstrOption : [in] option string

Table 3-3 Optional character strings of CaoController::AddVariable

<table>
<tr><th>Option</th><th>Meaning</th></tr>
<tr><td>Path =
<Path of Variables></td><td>The path of the memory that is the object of the variable.
If you specify this option, the generated CaoVariable behaves as <VAR variables>, and if you specify it at the same time, this option is ignored.
E.g. when accessing the global variable ‘a’
"Path=VAR://a"</td></tr>
<tr><td>PDO=
<node address></td><td>The node address of the PDO that is the object of the variable.
This option is specified according to the following grammar:
<NodeNumber>:0x<index>:<subIndex>
<NodeNumber> Node addresses of the ECAT slaves(1～192)
<Index> The index number defined in the PDO map.Specifies with four hexadecimal digits.
<SubIndex> Subindex number defined in the PDO map.Specifies a hexadecimal two-digit number.
If you specify this option, the generated CaoVariable behaves as <PDO Variables>, and if specified concurrently with the Path option, this option takes precedence.

E.g. node address "1" and index number "0x7000",
subindex "0"
"PDO = 1:0x7000:00"</td></tr>
<tr><td>Type = <data type></td><td><VAR variable> specifies the data type that you want to use to retrieve and configure the value of <VAR variable>. (Default: VT_UI1)
This option is valid only when the Path option is specified.
This option specifies the following string:
<table><tr><th>Data type</th><th>Meaning</th></tr></table></td></tr>
</table>

VT_BOOL	BOOL
VT_I1	1-byte signed integer
VT_UI1	1-byte unsigned integer
VT_I2	2-byte signed integer
VT_UI2	2-byte unsigned integer
VT_I4	4-byte signed integer
VT_UI4	4-byte unsigned integer
VT_R4	4-byte floating point number.
VT_R8	8-byte floating-point number.

E.g. for WORD type (2-byte unsigned integers)
"Type=VT_UI2"

3.2.3. CaoController::Execute method

Runs provider-specific extension commands that belong to CaoController classes.

The arguments for the Execute method specify commands in BSTR and parameters in the VARIANT array.

Syntax [<vntRet:VARIANT> =]Execute(<bstrCmd:BSTR >[, <vntParam:VARIANT>]]

bstrCmd	:	[in]	Command name
vntParam	:	[in]	Parameter
vntRet	:	[out]	Return value

If you invoke a method that is not defined in this class using the runtime binding feature, the Execute method is automatically called in the following specifications:

vntRet = Obj.CommandName(Param1, Param2, ...)

↓

vntRet = Obj.Execute("CommandName", Array(Param1, Param2, ...))

1. Command names are passed in BSTR strings to the first arguments.
2. The second arguments contain all the parameters in the VARIANT array.

If you are using the Binding feature, check the Bind to a'Execute'method of the specified providers from the CaoConfig.exe >CaoProvider tabs.

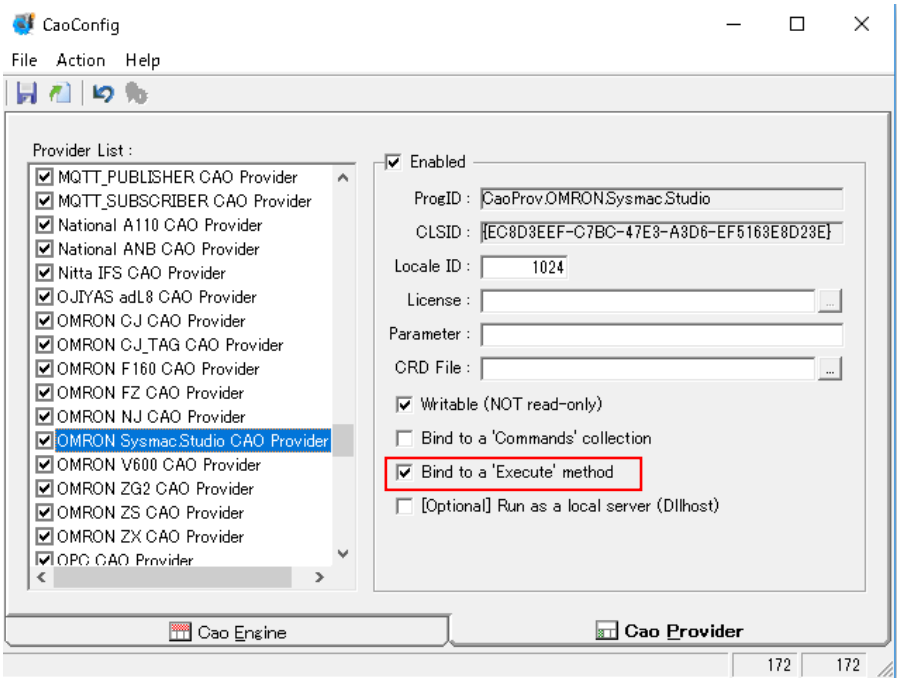


Figure 3-1 Setting for Binding feature

3.2.4. CaoController::Execute("GetVarAddr") Commands

Gets an address list of tag revisions and variables. Details of the return values are shown in the table below.

Syntax GetVarAddr (<Path1>[, <Path2>, <Path3>,...]]

< Path(n) > : [in] Path of the variable (VT_BSTR)
(n ≦ 100) You can specify up to 100 paths.
Returned value : [out] VarAddr (Tag Revision & Address info)
(VT_ARRAY | VT_VARIANT: 2 elements)

Table 3-4 GetVarAddr Return Value Type

Data type		Description	Values range
VT_ARRAY |VT_VARIANT		VarAddr	--
0	VT_ARRAY |VT_UI1:8 elements	Tag Revision	0x0000 0000 0000 0000 ～ 0xffff ffff ffff fffe
1	VT_ARRAY |VT_VARIANT: n elements	AddrList Variable's address list	--

Data type			Description	Values range
0 ～ n	VT_ARRAY | VT_UI4		Address information	--
	0	VT_UI4	Address-type Numeric value representing the meaning of the address	1 <Path(n)> represents variables (other than VAR_IN_OUT). The address obtained is the address of the managed object of the specified variable. 2 <Path(n)> represents memories. The acquired address is a physical address. 3 <Path(n)> represents special variables. The address obtained is the address of the management object of the specified POU. 4 <Path(n)> indicates variables (VAR_IN_OUT). The address obtained is the address of the managed object of the specified variable.
	1	VT_UI4	Address	When address type="2", the start address of one word containing the start bits represented by <Path(n)>.
	2	VT_UI4	Bit offset	When the address type is "1", the bit offset from the head variable (0 or more). If address type="2", the bit offset from the address (0-15).
	3	VT_UI4	Bit-size	

Example

```
Dim vntAddr As Variant
vntAddr = caoCtrl.Execute("GetVarAddr",ARRAY("VAR://a","VAR://b"))
vntAddr = caoCtrl.GetVarAddr("VAR://a","VAR://b")
```

3.2.5. CaoController::Execute("ASyncReadMem") Commands

Retrieves the value stored at the specified address in asynchronous mode. If indeirect is specified in <access> of the parameter and there are no synchronization buffers, direct accesses are performed. Let the <AddrList> and <Access> elements in <VarAddr> be the same number.

Syntax ASyncReadMem (<VarAddr>, <Access>, <SegmentCheck>)

< VarAddr > : [in] Tag Revision & Address (VT_ARRAY |VT_VARIANT)
*See GetVarAddr for more information.

< Access > : [in] How to access variables (VT_ARRAY |VT_UI4)

0	Not specified
1	Indirect Access the variable synchronization buffer. The latest value is always stored even in the middle of the control thread.
2	Direct You access the variable's entity. Only after the end of the control thread is the latest value stored.

< SegmentCheck > : [in] Segment check (VT_UI4)
Number indicating the segment check to be performed

0	Do not check.
Other than 0	Refer to the RM Component I/F Specifications for the details of the checks to be performed.

Returned value : [out] List of out values
(VT_ARRAY |VT_VARIANT [VT_ARRAY |VT_UI1]: n elements)
Return values for n elements specified by VarAddr are returned. If the variable type is 2 bytes or more, it is returned as a 1-byte array. In addition, even if the start position or end position of the specified area is in the middle of a byte, the byte data including the specified area is returned without performing processing such as bit shift.

Example

```
Dim vntAddr As Variant
Dim vntVal As Variant
Dim vntA,vntB As Variant
vntRetAddr = caoCtrl.GetVarAddr("VAR://a","VAR://b")
vntVal = caoCtrl.Execute("AsyncReadMem", Array(vntRetAddr, Array(0, 0), 0))
vntVal = caoCtrl.ASyncReadMem(vntRetAddr, Array(0, 0), 0)
```

```
vntA = vntVal(0)(0)
vntB = vntVal(1)(0)
```

3.2.6. CaoController::Execute("ASyncWriteMem") Commands

Update the value stored at the specified address in asynchronous mode. If indirect is specified in <access> and there are no buffers for synchronization, direct accesses are performed. Use the same number of elements for <Value>, <AddrList>, and <Access>.

Syntax ASyncWriteMem (<ValueList>, <VarAddr>, <Access>, <SegmentCheck>)

<ValueList> : [in] value list
(VT_ARRAY |VT_VARIANT[VT_ARRAY |VT_UI1]: n elements)

<VarAddr> : [in] Tag Revision & Address (VT_ARRAY |VT_VARIANT)
*See GetVarAddr for more information.

<Access> : [in] How to access variables (VT_ARRAY |VT_UI4)

0	Not specified
1	Indirect Access the variable synchronization buffer. The latest value is always stored even in the middle of the control thread.
2	Direct You access the variable's entity. Only after the end of the control thread is the latest value stored.

<SegmentCheck> : [in] Segment check (VT_UI4)
Number indicating the segment check to be performed

0	Do not check.
Other than 0	Refer to the RM Component I/F Specifications for the details of the checks to be performed.

Returned value : [out] Not available

Example

```
Dim vntAddr As Variant
Dim vntVal As Variant
Dim vntA,vntB As Variant
vntRetAddr = caoCtrl.GetVarAddr("VAR://a","VAR://b")
vntVal = caoCtrl.Execute("ASyncWriteMem", _
Array(Array(Array(1), Array(1)), vntRetAddr, Array(0,0), 0))
vntVal = caoCtrl.ASyncWriteMem(Array(Array(1), Array(1)), vntRetAddr, Array(0,0), 0)
```

3.2.7. CaoController::Execute("ReadPDO") Commands

Reads an axis variable.

Syntax ReadPDO (<NodeAddrList>)

<NodeAddrList> : [in] node address list (VT_ARRAY |VT_VARIANT)
Returned value : [out] List of values (VT_ARRAY |VT_BSTR)

Table 3-5 ReadPDO arguments

Data type			Description	Values range
VT_ARRAY |VT_VARIANT: n elements			NodeAddList	--
0 ~ N	VT_ARRAY |VT_VARIANT: 3 elements		Node address	--
	0	VT_UI1	Node addresses of the ECAT slaves	1~192
	1	VT_UI2	Index number defined in the PDO map	0x0000 ～ 0xFFFF
	2	VT_UI1	Subindex number defined in the PDO map	0x00 ～ 0xFF

Example

```
Dim vntVal As Variant
vntVal = caoCtrl.Execute("ReadPDO", Array(Array(1, 28672, 0), Array(1, 32768,0)))
vntVal = caoCtrl.ReadPDO(Array(1, 28672, 0), Array(1, 32768,0))
```

3.2.8. CaoController::Execute("WritePDO") Commands

Writes data to the axial variable area.

Syntax WritePDO (<NodeAddrList>)

<NodeAddrList> : [in] Node address list (VT_ARRAY |VT_VARIANT)
Returned value : [out] Not available

Table 3-6 WritePDO arguments

Data type			Description	Values range
VT_ARRAY |VT_VARIANT: n elements			NodeAddList	--
0 ~ N	VT_ARRAY |VT_VARIANT: 3 elements		Node address	--
	0	VT_UI1	Node addresses of the ECAT slaves	1~192
	1	VT_UI2	Index number defined in the PDO map	0x0000 ～ 0xFFFF
	2	VT_UI1	Subindex number defined in the PDO map	0x00 ～ 0xFF
	3	VT_BSTR	Value to be written	

Example

```
Dim vntVal As Variant
vntVal = caoCtrl.Excute("WritePDO",Array(1, 28672, 0, 1), Array(1, 32768, 0, 1))
vntVal = caoCtrl.WritePDO(Array(1, 28672, 0, 1), Array(1, 32768, 0, 1))
```

3.2.9. CaoController::Execute("GetMode") Commands

Gets the operation mode of the CPU unit.

Syntax GetMode (<Section>)

<Section> : [in] Items to be retrieved (VT_BSTR)

""	Gets all current states.
"mode"	Retrieves only WorkMode values.
"state"	Retrieves only WorkState values.
"error"	Retrieves only ErrorState values.

Returned value : [out] The status of the retrieved items (VT_BSTR)

The following status is acquired depending on the item to be acquired.

ErrorState NoError, HaltError, ContinuousError

WorkMode RUN, PROGRAM

WorkState InitializingWaitPLC, InitializingWaitIO, InitializingWaitControl, Initializing, ReadyWaitPLC, ReadyWaitIO, ReadyWawitControl, Ready, ProgramWaitPLC, ProgramWawitSubsystems,

Program, RunWaitPLC, RunWaitSubsystems, Run, DownloadingWaitControl, DownloadinigWaitIO, DownloadingwaitPLC, Downloading, SubsystemsTerminate, Backup, Terminate

Example

```
Dim vntRet As Variant
vntRet = caoCtrl.Execute("GetMode", "mode")
vntRet = caoCtrl.GetMode("mode")
```

3.2.10. CaoController::Execute("SetMode") Commands

Changes the operation mode of the CPU unit.

Syntax SetMode (<bstrSection >,<bstrState>)

< bstrSection > : [in] Change target (VT-BSTR)

"mode"	Operation mode
"error"	Error status

< bstrState > : [in] Status (VT_BSTR)

The conditions that can be specified are as follows in the <bstrSection> specification.

"mode"	Run,program
"error"	Alert,error,warning,notice,clear

Returned value : [out] Not available

Example

```
Dim vntRet As Variant
vntRet = caoCtrl.Execute("SetMode", Array("mode", "program"))
vntRet = caoCtrl.SetMode("mode","program")
```

3.2.11. CaoController::Execute("SetScanExec") Commands

Changes the running status of the Runtime simulator. When the Runtime simulator is in operation mode, it executes one cycle and waits when the SetScanExec ('1') command is received. When the SetScanExec('0') command is received or a predetermined period of time elapses in the standby state, the command enters the running state continuously.

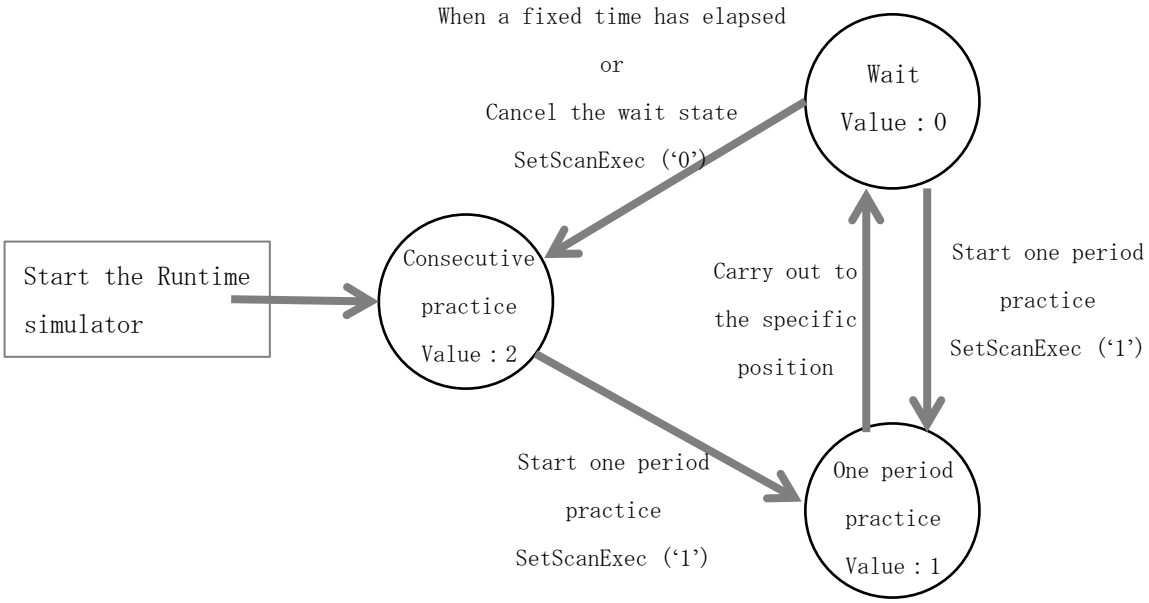


Figure 3-2 State transition diagram of the simulator.

Syntax SetScanExec (<bstrValue>)

< bstrValue > : [in] Setting (VT-BSTR)

0	Cancels the standby state.
1	Start execution for one cycle and wait for a certain period of time.

Returned value : [out] Not available

Example

```
Dim vntRet As Variant
vntRet = caoCtrl.Execute("SetScanExec", 0)
vntRet = caoCtrl.SetScanExec(0)
```

3.2.12. CaoController::Execute("CheckScanState") Commands

Gets the running status of the Runtime simulator.

Syntax CheckScanState ()

Argument : [in] None

Returned value : [out] Run status (VT-BSTR)

0	Waiting
1	One cycle is running.
2	Continuous execution

Example

```
Dim vntRet As Variant
vntRet = caoCtrl.Execute("CheckScanState")
vntRet = caoCtrl.CheckScanState()
```

3.2.13. CaoVariable::get_Value properties

Retrieves the value of the variable that you created in chapter 3.2.2. Please refer to chapter 3.3.1 for details.

3.2.14. CaoVariable::put_Value properties

Sets the value of the variable that you created in chapter 3.2.2. Please refer to chapter 3.3.1 for details.

3.3. Variable list

3.3.1. Controller class

Table 3-7 Controller Class User Variables List

Variable name	Data type	Description	Attribute	
			Get	Put
<VAR variable>	<Type> options | VT_ARRAY	Sets the value to the memory. If you specify the "Path" option at the time of AddVariable, this option is the behavior of this variable. The array data type uses the type specified by the <Type> options during AddVariable.	✔	✔

<PDO variable>	VT_BSTR	Obtain and set values for PDO. If you specify the "PDO" option at the time of AddVariable, the operation of this variable will occur.	✔	✔

Table 3-8 Controller Class System Variables

Variable name	Data type	Description	Attribute	
			Get	Put
@MODE	VT_BSTR	Gets the WorkMode values. The following values can be specified: "RUN", "PROGRAM"	✔	✔
@ERROR	VT_BSTR	Gets the ErrorState values. "alert", "error", "warning", "notice", "clear"	✔	✔
@STATE	VT_BSTR	Gets the WorkState values.	✔	-

4. Command reference

Indicates the list of commands.

Table 4-1 List of CaoExtension::Execute commands4-1

Commanded	Facility	Page
etVarAddr	Gets the address of the variable.	P. 10
ASyncReadMem	Gets the value stored at the specified address.	P. 12
ASyncWriteMem	Update the value stored at the specified address.	P. 13
ReadPDO	Read the axis variable.	P. 14
WritePDO	Write the value in the axis variable area.	P. 14
GetMode	Acquire the operation mode of the CPU unit	P. 15
SetMode	Change the operation mode of the CPU unit.	P. 16
SetScanExec	The start of execution of one cycle is instructed, or the instruction is released.	P. 16
CheckScanState	Used to acquire the running status of the Runtime simulator.	P. 18

5. Sample program

List 5-15 SampleText.frm

```
Dim m_eng As CaoEngine
Dim m_ctrl As CaoController

Private Sub Form_Load()
    Set m_eng = New CaoEngine
    Set m_ctrl = m_eng.Workspaces(0).AddController("SYSMAC", "CaoProv.OMRON.Sysmac.Studio", "",
"server=127.0.0.1,port=7000")

    Dim vntVal As Variant
    Dim vntRetAddr1 As Variant
    Dim vntRetAddr2 As Variant
    vntRetAddr1 = m_ctrl.Execute("GetVarAddr", Array("VAR://TEST1"))
    vntRetAddr2 = m_ctrl.Execute("GetVarAddr", Array("VAR://TEST3"))

    vntVal = m_ctrl.Execute("ASyncWriteMem", Array(Array(Array(0)), vntRetAddr1))
    vntVal = m_ctrl.Execute("ASyncReadMem", Array(vntRetAddr1))

    vntVal = m_ctrl.Execute("ASyncWriteMem", Array(Array(Array(1)), vntRetAddr1))
    vntVal = m_ctrl.Execute("ASyncReadMem", Array(vntRetAddr2))

    vntVal = m_ctrl.Execute("ReadPDO", Array(Array(1, 24698, 0)))

    vntVal = m_ctrl.Execute("SetMode", Array("mode", "run"))
    vntVal = m_ctrl.Execute("GetMode", Array("Mode"))

    vntVal = m_ctrl.Execute("SetMode", Array("mode", "program"))
    vntVal = m_ctrl.Execute("GetMode", Array("Mode"))

    m_ctrl.Execute "SetScanExec", 1

    vntVal = m_ctrl.Execute("CheckScanState")
    vntVal = m_ctrl.Execute("SetScanExec", 0)
    vntVal = m_ctrl.Execute("CheckScanState")

End Sub
```