OMRON programmable controller
CJ series provider

Version 1.2.8

User’s guide

February 6, 2023

[Remarks]

[Revision history]

Version	Date	Description
1.0.0	2016-03-02	First edition.
1.0.1	2016-05-27	Modified error codes. According to the specification of CX-Programmer, modified the endian conversion which is done at the 4-byte or larger data obtainment
1.02	2016-06-22	Changed the default value of VT option when there is no entry. “BIT” for bit address entry. “I2” for channel address entry.
	2016-08-10	Added description that communication may fail if communication packet size which arises from variable reading/writing increases.
	2016-10-31	Changed so that ETH is specified by Conn option.
1.1.0	2018-06-08	Added Retry option.
1.2.0	2018-07-05	TCP compatible.
1.2.1	2019-02-07	TCP packet division support.
1.2.2	2019-08-19	Bug fixed, Expansion data memory (EM bank10 – 18).
1.2.3	2020-01-29	Added BitOpt option for compatibility (Appendix C).
1.2.4	2020-07-10	Exclusive control process correction. Added precautions when connecting NSJ controllers. Src IP Address,Src Port No elements are specified in Conn options. Specifies information error procedures from relay error responses and error responses. Appendix E added.
	2020-10-01	Added operation confirmed models
1.2.5	2021-02-09	Improved disconnection processing.
1.2.6	2021-06-18	Added error code when getting unexpected data
	2022-05-30	Added the Performance measurements section.
1.2.7	2022-08-02	Fixed the value writing process when specifying a channel. Corrected description of STATUS in Mode option.
	2022-11-03	Chart number correction.
1.2.8	2023-02-06	Fixed a failure that could cause memory corruption if packet fragmentation occurred.

[Operation confirmed models]

Model	Version	Note
CJ2H	Ver1.2	
CS1H-CPU64H		
CS1W-EIP21		EtherNet/IP unit

Contents

1. Introduction

This is a user's guide of CAO provider that enables data reading and writing from/to CJ series PLC manufactured by OMRON. Hereafter, this provider (CaoProvOmronCJ.dll) is called CJ provider.

Chapter 2 describes the outline of CJ provider and detailed information about variables.

The support status of communication command implemented in CJ provider and data streams depend on the communication destination PLC (CJ series). For details about communication, please refer to OMRON's document "sbca-304r-30_cs1_cj1_cp1_com_cmd" and "sbca-350l-22_cj2h-cpu6-eip_cj2m-cpu" (both are pdf –format).

2. Outline of provider

2.1. Outline

CJ provider is CAO provider that enables data reading/writing from/to CJ series (OMRON's PLC) via Ethernet (UDP/TCP) connection by using FINS command. The file format is DLL (Dynamic Link Library) and it will be loaded from CAO engine automatically when it is used. Before using CJ provider, you need to install ORiN2SDK or to execute registration manually based on the following information.

Table 2-1 CJ provider

File name	CaoProvOmronCJ.dll
ProgID	CaoProv.OMRON.CJ
Registration	regsvr32 CaoProvOmronCJ.dll
Deregistration	regsvr32 /u CaoProvOmronCJ.dll

2.2. Method and Properties

2.2.1. CaoWorkspace::AddController method

CJ provider establishes communication by referring to the communication connection parameter at the timing of AddController. (It runs as a TCP client.)

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

bstrCtrlName : [in] Controller name

bstrProvName : [in] Provider name. Fixed to " CaoProv.OMRON.CJ"

bstrPcName : [in] Computer name where provider runs.

bstrOption : [in] Option character strings

The following table shows a list of option character strings

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

Option	Description
Conn=<Communication parameter>	Required. Communication configuration and connection parameter. (See 2.2.1.1)
ConnTimeout[=<Timeout time>]	Timeout time for TCP communication. (msec) (default:500)
Timeout[=<Timeout period>]	Timeout period at the data receiving and sending (millisecond) (Default :3000)
FinsParam=<DNA>:<DA1>:<DA2> [:<GCT>[:<SNA>[:<SA1>[:<SA2>:[<SID>]]]]]	Required (partly omittable). FINS header structure parameter. (See 2.2.1.2)
Retry[=<Retry count>]	Specify the number of communication retry at data sending/receiving. (default:0)
KeepAlive[=<Keepalive time>]	keep-alive time for TCP communication (msec) (default:60000)
PutOpt:[=< Write operation >]	Specify the data structure when writing to the channel address with BIT, BOOL, UI1, I1. (See Appendix D)

2.2.1.1. Conn option

The following shows the connection parameter strings of Conn option. Parameters surrounded by the square brackets ([]) can be omitted. Underlined part shows the default value when the option is not specified.

(CJ provider only supports UDP/TCP connection of Ethernet device.)

Ethernet device

"Conn=ETH:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

"Conn=UDP:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

"Conn=TCP: <Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

< Dest IP Address >	:	Dest IP address Example : "127.0.0.1", "192.168.0.1"
<Dest Port No>	:	Dest connection port number. 9600, 5006, 5007, … Enter any number.
<Src IP Address>	:	Src IP address (For several NICs) To automatically determine the IP address, specify "255.255.255.255". Example : "127.0.0.1", "192.168.0.1" , "255.255.255.255"
<Src Port No>	:	Src port number. (For several NICs) [1][2] Example : 0, 9600, 5006, 5007, … Enter any number.

[1] When connecting to the NSJ controller, the PC and the port must be the same. Set Dest Port No and Src Port No together. When using the NSJ expansion unit (NSJW-ETN21), there is no need to set Src Port No.

[2] Set Dest Port No and Src Port No when connecting to a PLC on the network destination. Refer to Appendix E when connecting to the PLC of the network destination.

2.2.1.2. FinsParam option

The following shows the connection parameter strings of FinsParam option. Parameters surrounded by the square brackets ([]) can be omitted. Underlined part shows the default value when the option is not specified.

"FinsParam=<DNA>:<DA1>:<DA2>[:<GCT>[:<SNA>[:<SA1>[:<SA2>:[<SID>]]]]]"

<DNA> : Destination network address
- 0 : Own network
- 1 to 127 : Destination network address

<DA1> : Destination node address (Destination node number)
- 0 : Communication within own PLC
- 1 to 32 : For Controller Link network
- 1 to 254 : For Ethernet
- 255 : Broadcasting

<DA2> : Destination unit address
- 0 : CPU unit
- 254 : Applicable controller link unit / Ethernet unit
- 16 to 31 : CPU Bus unit
- 225 : INNER board

<GCT> : Maximum bridge count (Gateway Count)
- 2, 7

<SNA> : Source network address
- 0 : Own network
- 1 to 127 : Destination network address

<SA1> : Source node address (Source node number)
- 0 : Communication within own PLC
- 1 to 32 : For Controller Link network
- 1 to 254 : For Ethernet

<SA2> : Source unit address
- 0 : CPU unit
- 16 to 31 : CPU Bus unit

<SID> : Service ID
- 0 to 255 : Any number

2.2.2. CaoController::AddVariable method

AddVariable method of CaoController class creates a variable object for data reading/writing from/to PLC (CJ series).

This method determines I/O memory's area type and area address within PLC (CJ series) to be accessed by specifying a variable name which is based on the format described in 2.2.2.1.

As an option, you can convert the data type and specify the number of data to read/write. [3]

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

<bstrVariableName> : [in] Variable name

<bstrOption> : [in] Option character string

The following table lists the option character strings.

Table 2-3 Option character string of CaoController::AddVariable

Option	Description
Mode[=<Access mode>]	Specify the access mode to access Timer, Counter and Task flag. (See 2.2.2.2)
VT[=<Variable type>]	Specify the data type to Put/Get. (See 2.2.2.3)
Elem[=<Number of element>]	Specify the number of elements to Put/Get. When data type is VT_BSTR, specify with number of bytes of character strings. Number of elements can be specified with decimal or hexadecimal (0x0A, &h0A, 0AH). (Default :1)
Array[=< True or False >]	When the data to Put/Get is one element, specify if the data is handled as an array or not. (Default :False)
ID[=<ID initial value>]	Specify the ID initial value of variable.(Decimal) (Default :0)

[3] If communication packet size that arises from data reading/writing increases, communication may fail due to the FINS command regulation. To avoid this, reduce the numbers of data element to be read/written at the same time.

2.2.2.1. Specifying area type and address of the access target data with a variable name

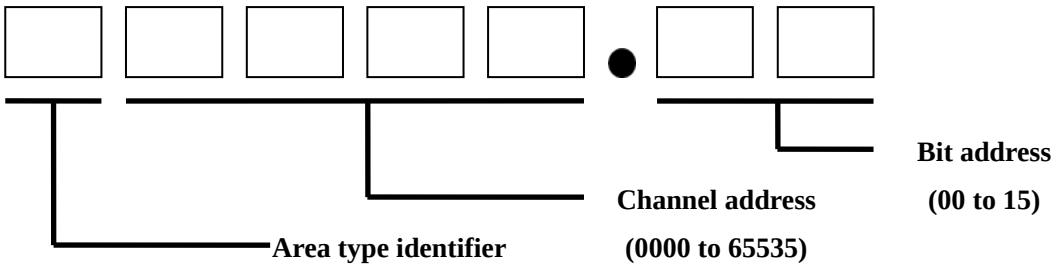
A variable name that specifies an access target data consists of an area type identifier (alphabets of zero to 4 characters), channel address (decimal number of one to 4 digits), and bit address (decimal number of one to two digits).

If the data mode of the access target IO memory is “bit”, enter a dot (“.”) between a channel address and a bit address. If the access target IO memory is “channel (word)”, do not enter a bit address.

If the access target data is Timer, Counter, or Task flag, only the format for Channel (Word) address entry is available. To specify the data mode, use Mode option (See 2.2.2.2).

The following shows the format to specify the memory type and memory address.

Format for Bit address entry



Format for Channel (Word) address entry

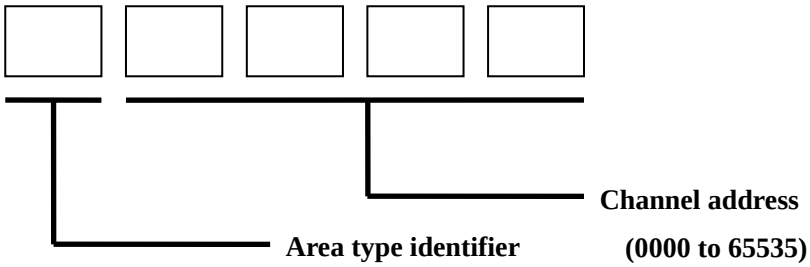


Table 2-4 shows the list of identifier for each IO memory area type.

Table 2-4 Area type identifier list

Area type		Identifier
Channel I/O	CIO	(none)
Internal auxiliary relay	WR	W
Holding relay	HR	H
Auxiliary relay	AR	A
Timer	TIM	T
Counter	CNT	C
Data memory	DM	D
Expansion data memory	EM bank 0	E0_
	EM bank 1	E1_
	EM bank 2	E2_
	EM bank 3	E3_
	EM bank 4	E4_
	EM bank 5	E5_
	EM bank 6	E6_
	EM bank 7	E7_
	EM bank 8	E8_
	EM bank 9	E9_
	EM bank A	EA_
	EM bank B	EB_
	EM bank C	EC_
	EM bank D	ED_
	EM bank E	EE_
	EM bank F	EF_
	EM bank 10	E10_
	EM bank 11	E11_
	EM bank 12	E12_
	EM bank 13	E13_
	EM bank 14	E14_
	EM bank 15	E15_
	EM bank 16	E16_
	EM bank 17	E17_

	EM bank 18	E18_
Task flag	TK	TK

2.2.2.2. Mode option

Use this option to specify the data mode when the target area type is Timer, Counter, or Task flag.
(Default :NORMAL)

NORMAL : Treat the data mode as a bit. This setting is ignored if the area type is other than Timer, Counter, or Task flag.

CURRENT : Valid only for Timer or Counter.
Treat the data mode as a current value.

STATUS : Valid only for Task flag. Treat the data mode as a bit.

2.2.2.3. VT option

Specify the data type to Put/Get.

The following data types are available to specify VT option.

(Default data type: “BIT” for bit address entry. “I2” for channel address entry)

Table 2-5 Data types available to VT option

VT	Data type	Available address type	Description
BIT	VT_UI1	bit/channel	Convert data to binary data (0/1) and then read/write it. Val==0:0, Val!=0:1
BOOL	VT_BOOL	bit/channel	Convert data to binary data (0/1) and then read/write it. Val==VARIANT_FALSE:0, Val= VARIANT_TRUE:1
BSTR	VT_BSTR	channel	Write/Read BSTR as ASCII
I1	VT_I1	channel	Write/Read data as 1-byte data
I2	VT_I2	channel	Write/Read data as 2-byte data
I4	VT_I4	channel	Write/Read data as 4-byte data
I8	VT_I8	channel	Write/Read data as 8-byte data
UI1	VT_UI1	channel	Write/Read data as 1-byte data
UI2	VT_UI2	channel	Write/Read data as 2-byte data
UI4	VT_UI4	channel	Write/Read data as 4-byte data
UI8	VT_UI8	channel	Write/Read data as 8-byte data
R4	VT_R4	channel	Write/Read data as 4-byte data
R8	VT_R8	channel	Write/Read data as 8-byte data

2.2.2.4. ELEM option

The maximum value that can be specified for the "ELEM" option is determined by the combination of the data type and the VT option.

The maximums for ELEM options are as follows:

Data type	VT	PutOpt	Maximum read length	Maximum write length
bit	BIT,BOOL	TRUE/FALSE	1998	1994
channel	BIT,BOOL,I1,UI1	FALSE	1998	997
		TRUE	1998	1994
	I2,UI2	TRUE/FALSE	999	997
	I4,UI4,R4	TRUE/FALSE	499	498
	I8,UI8	TRUE/FALSE	249	249

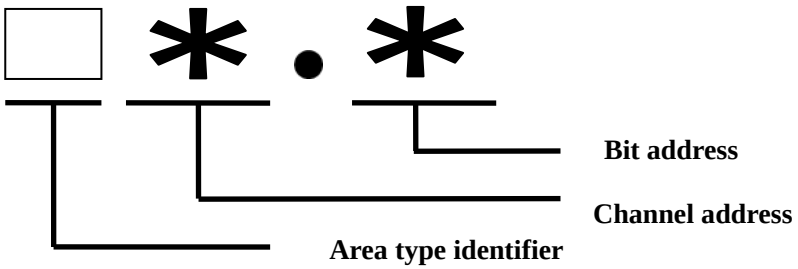
2.2.2.5. Specifying an address with a wildcard

With a wildcard (*) , you can define an address part of a variable ambiguously so that the access destination address can be changed at any time later.

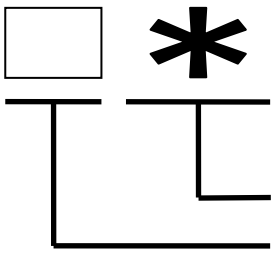
To specify the access destination address, use a variable ID at the timing of Put/Get. (See 2.2.5)

The following formats show how to specify addresses with wildcards.

Format for Bit address entry



Format for Channel (Word) address entry



Channel address

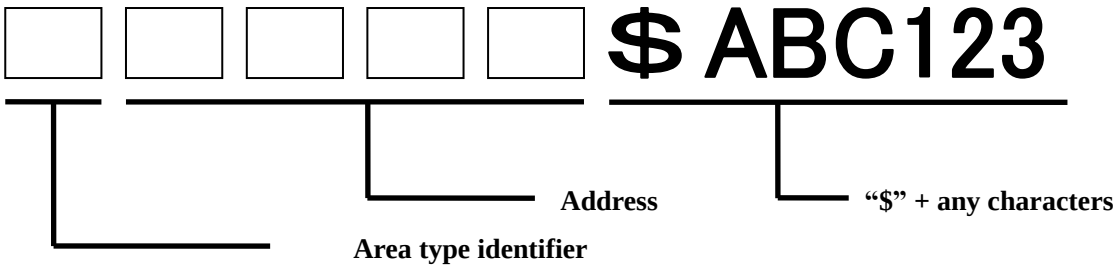
Area type identifier

2.2.2.6. Using characters in a variable name

To register several variables that have the same address and same area type identifier, add any characters with a dollar sign (“$”) after the address part of a variable name. This is practical if you prepare several option patterns for identical address..

The following format shows how to add characters.

Format common to Bit/ Channel (Word) address entry



2.2.3. CaoVariable:put_Value property

This property converts the value handed as an argument based on the specified option, and then sends a FINS command (command code :01 02) to write the converted value in the memory area specified by a variable.

2.2.4. CaoVariable:get_Value property

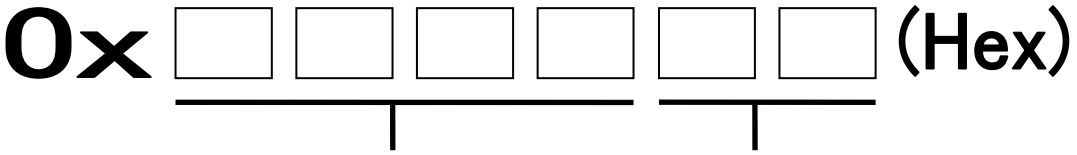
This property sends a FINS command (command code : 01 01) that reads data from the memory area specified by a variable name. The size of the data is determined by the option entry. The read-out data is converted to the data type specified by the option entry and then returned.

2.2.5. CaoVariable:put_ID property

Set a variable ID.

This ID is referred only when an address part of variable name is specified with a wildcard (*). (See 2.2.2.5)

If a variable name is not specified with a wildcard, this ID simply works as an ID. The following format shows the meaning and valid range of each entry when a variable name is specified with a wildcard.



Upper 4 bytes
・used as a channel address.
・Available range:0000 to FFFF

Lower 1 byte
・used as a bit address.
・Available range:00 to 0F

※ To enter channel address only (not enter a bit address), enter 0x00 in the lower 1 byte.

2.2.6. CaoVariable:get_ID property

This property obtains a variable ID that is currently set.

2.3. Variable list

2.3.1. CaoController class

Table 2-6 CaoController class user variable list

Variable name	Data type	Description	Attribute	
			get	put
(any name)	Depending on the variable type	Specify I/O memory's area type and area address within PLC (CJ series) to be accessed. (See 2.2.2)	✓	✓

Table 2-7 CaoController class system variable list

Variable name	Data type	Description	Attribute	
			get	put
@MAKER_NAME	VT_BSTR	Return manufacturer name = "OMRON".	✓	−
@VERSION	VT_BSTR	Version information	✓	−
@LAST_ERROR	VT_VARIANT | VT_ARRAY	Detailed information of the latest communication error report [0] VT_UI2 : Command code where an error occurs [1] VT_BOOL: Relay error [2] VT_BOOL: PLC stopped with error [3] VT_BOOL: Error occurs but PLC is running [4] VT_UI1 : Main response code [5] VT_UI1 : Sub response code [6] VT_UI1 :Network address where an error occurs [7] VT_UI1 : Node address where an error occurs	✓	−

2.4. Error code

CJ provider defines the following original error codes.

When "relay error occurrence" or "error response" is returned, the error information of the previous communication can be acquired by reading the system variable "@LAST_ERROR".

For information about ORiN2 common errors, refer to the chapter of “Error code” in ORiN2 programming guide.

Table 2-8 Original error code

Error name	Error number	Description
Relay error	0x80100000	Due to the error occurrence on the communication path, a relay error was reported.
Error response	0x80100001	An error response was arrived from the connection destination.
Insufficient data received	0x80100002	Analysis failed due to the insufficient data received.
Command response unmatched	0x80100003	Received command response does not match with the transferred FIN command.
Illegal command	0x80100004	When an illegal FINS/TCP command is received, it is returned.
Illegal node	0x80100005	When an illegal node address is specified, it is returned.
Illegal header	0x80100100	When the header of the FINS/TCP header is not 'FINS'(ASCII code), it is returned.
Outside data length range	0x80100101	It is returned when the data length of the FINS/TCP header is outside the range.
Command outside support	0x80100102	It is returned when the command of the FINS/TCP header is outside the support.
All connections are being used.	0x80100103	When the connection is using everything, it is returned.
The node is being connected.	0x80100104	It is returned with the specified node address when having already connected it.
Protected node	0x80100105	When it accesses the protected node address by Internet Protocol address outside specification, it is returned.

Outside node range	0x80100106	It is returned when the node address of the client is outside the range.
The same node	0x80100107	When the same node address is used with the client and the server, it is returned.
Unexpected error	0x801001FF	It is returned when unexpected data is received.

If a relay error response or error response occurs, you can investigate the cause by examining the detailed error using the @LAST_ERROR variable.

The procedure for resolving the problem is as follows.

1. Relay error response or error response occurred
2. Using @LAST_ERROR to Get More Info
3. Check error details from main response code and sub response code [4]
4. Check the corrective action for main unit stop error, main unit continuation error, and network relay error [4]

[4] For details on the error, see "FINS Command List Completion Code List" in the Communication Command Reference Manual ("sbca-304r-30_cs1_cj1_cp1_com_cmd.pdf").

Appendix A. Examples of variable naming

The following shows variable naming examples.

Example 1) To access channel I/O(CIO)-typed channel address 0x0010, bit12.

16.12 or **0016.12** (Zero-filled and zero-suppressed addresses are deemed as the same address.)

Example 2) To access expansion data memory EM bank 4-typed channel address 0x00FF.

E4_255 or **E4_0255**

Example 3) To specify wildcards in order to access an internal auxiliary relay (WR) with bit address.

W*.*

To set an ID in order to access the memory address 0x0010, bit12.

ID=4108 (0x001012)

Example 4) To specify a wildcard in order to access an holding relay (HR) with channel address.

H*

To set an ID in order to access the memory address 0x0010.

ID=4096 (0x001000)

Example 5) Register two variables to detect flag-up of counter (CNT)-typed address 0x00A0 and to obtain the current value.

C0160$flag and **C0160$val** (Letters written after the dollar sign (”$”) are not analyzed and registered as a different variable.)

Appendix B. Memory allocation in PLC (CJ series) at variable writing

The following example shows how data is mapped into PLC (CJ series) when data is written by put_Value property (assuming that all memories in PLC (CJ series) are cleared with zero).

Example 1) Set both Bit5 and Bit7 of a holding relay (HR) channel address 0x0010 to ON(1)

・Register two variables for setting.

AddVariable

<Variable1>

Name : H0016.05

Option : Mode=NORMAL, VT=BOOL, Elem=1, Array=FALSE

<Variable2>

Name : H0016.07

Option : Mode=NORMAL, VT=BOOL, Elem=1, Array=FALSE

・Write values in each variable.

put_Value

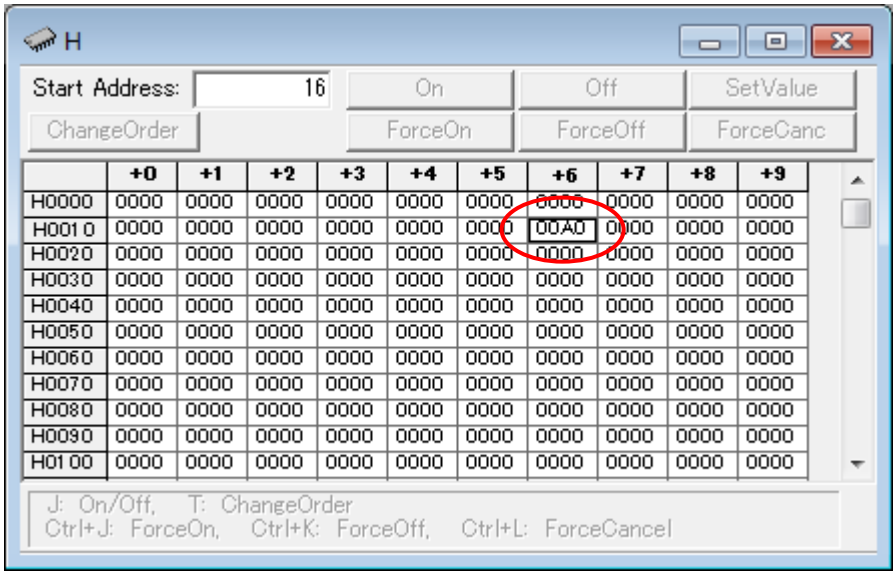
<Variable1>

VT_BOOL : TRUE

<Variable2>

VT_BOOL : TRUE

・Memory status in PLC (CJ series) (Use CX-Programmer)



Values of H0016 00A0(0000000010100000)

・Register a variable to read this channel address for confirmation purpose.

AddVariable

<Variable3>

Name : H0016

Option : Mode=NORMAL, VT=UI2, Elem=1, Array=FALSE

・Read a value.

get_Value

<Variable3>

VT_UI2 : 160 (0x00A0)

Example 2) Write one-byte data in data memory (DM) channel address 0x0100.

・Register one variable for setting.

AddVariable

<Variable4>

Name : D0100

Option : Mode=NORMAL, VT=UI1, Elem=1, Array=FALSE

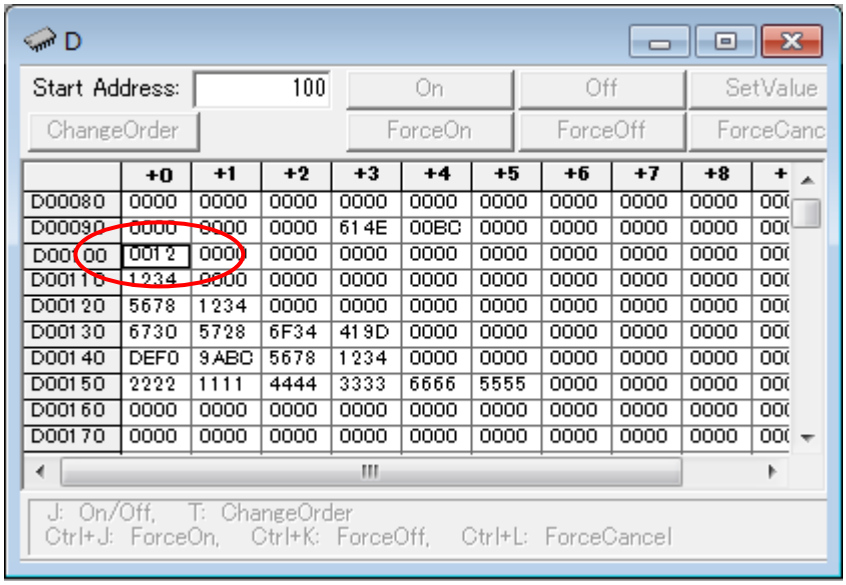
・Write values

put_Value

<Variable4>

VT_UI1 :18 (0x12)

・Memory status in PLC (CJ series) (Use CX-Programmer)



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+
D00080	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00090	0000	0000	0000	614E	00BC	0000	0000	0000	0000	000
D00100	0012	0000	0000	0000	0000	0000	0000	0000	0000	000
D00110	1234	0000	0000	0000	0000	0000	0000	0000	0000	000
D00120	5678	1234	0000	0000	0000	0000	0000	0000	0000	000
D00130	6730	5728	6F34	419D	0000	0000	0000	0000	0000	000
D00140	DEF0	9ABC	5678	1234	0000	0000	0000	0000	0000	000
D00150	2222	1111	4444	3333	6666	5555	0000	0000	0000	000
D00160	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00170	0000	0000	0000	0000	0000	0000	0000	0000	0000	000

Values of D0100 0012(0000000000010010)

・Read this channel address to confirm.

get_Value

<Variable4>

VT_UI1 :18 (0x12)

Example 3) Write two bytes data in data memory (DM) channel address 0x0110.

・Register one variable for setting.

AddVariable

<Variable5>

Name : D0110

Option : Mode=NORMAL, VT=UI2, Elem=1, Array=FALSE

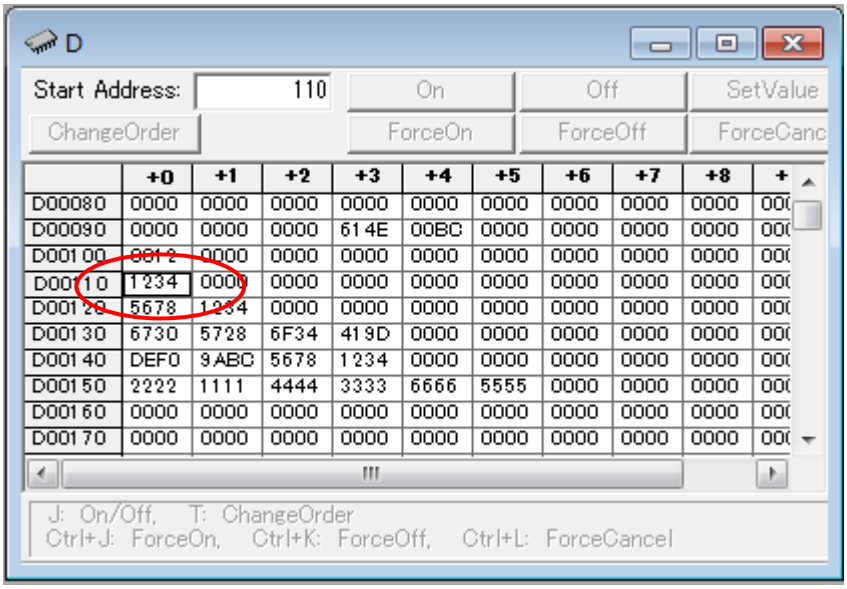
・Write values

put_Value

<Variable5>

VT_UI2 :4660 (0x1234)

・Memory status in PLC (CJ series)(Use CX-Programmer)



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+
D00080	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00090	0000	0000	0000	614E	00BC	0000	0000	0000	0000	000
D00100	3812	0000	0000	0000	0000	0000	0000	0000	0000	000
D00110	1234	0000	0000	0000	0000	0000	0000	0000	0000	000
D00120	5678	1234	0000	0000	0000	0000	0000	0000	0000	000
D00130	6730	5728	6F34	419D	0000	0000	0000	0000	0000	000
D00140	DEF0	9ABC	5678	1234	0000	0000	0000	0000	0000	000
D00150	2222	1111	4444	3333	6666	5555	0000	0000	0000	000
D00160	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00170	0000	0000	0000	0000	0000	0000	0000	0000	0000	000

Values of D0110 1234(0001001000110100)

・Read this channel address to confirm.

get_Value

<Variable5>

VT_UI2 :4660 (0x1234)

Example 4) Write four bytes data in data memory (DM) channel address 0x0120.

・Register one variable for setting.

AddVariable

<Variable6>

Name : D0120

Option : Mode=NORMAL, VT=I4, Elem=1, Array=FALSE

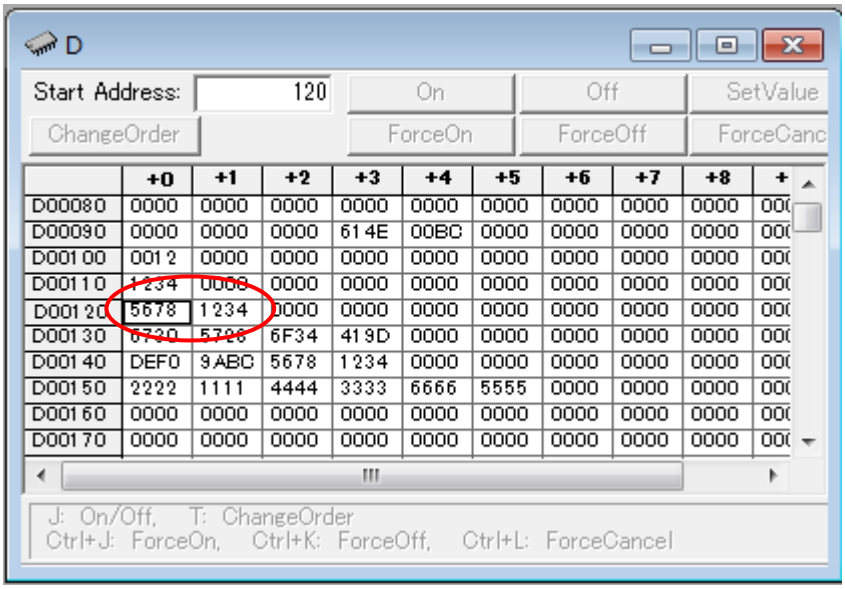
・Write values

put_Value

<Variable6>

VT_I4 :305419896 (0x12345678)

・Memory status in PLC (CJ series)(Use CX-Programmer)



Values of D0120 5678(0101011001111000)

Values of D0121 1234(0001001000110100)

・Read this channel address to confirm.

get_Value

<Variable6>

VT_I4 :305419896 (0x12345678)

Example 5) Write eight bytes data in data memory (DM) channel address 0x0140.

・Register one variable for setting.

AddVariable

<Variable7>

Name : D0140

Option : Mode=NORMAL, VT=I8, Elem=1, Array=FALSE

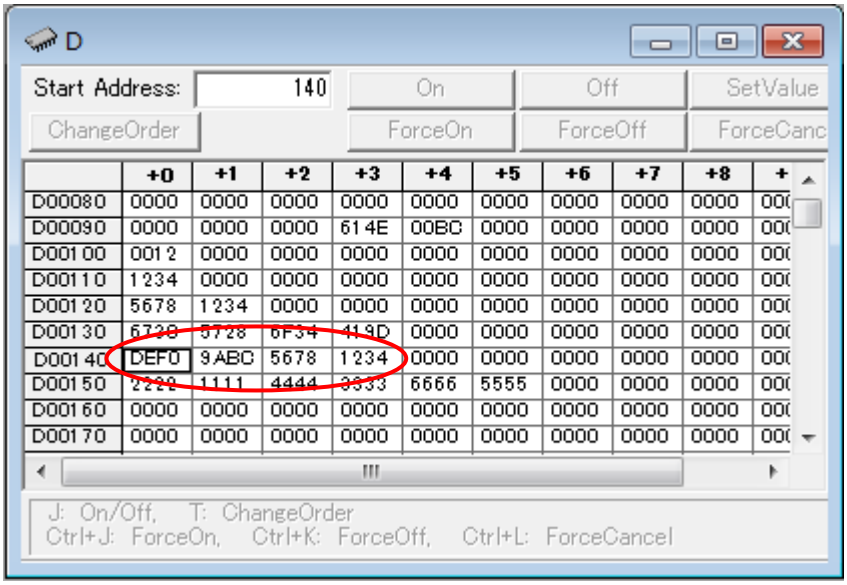
・Write values

put_Value

<Variable7>

VT_I8 :1311768467463790320 (0x123456789ABCDEF0)

・Memory status in PLC (CJ series)(Use CX-Programmer)



Values of D0140 DEF0(1101111011110000)

Values of D0141 9ABC(1001101010111100)

Values of D0142 5678(0101011001111000)

Values of D0143 1234(0001001000110100)

・Read this channel address to confirm.

get_Value

<Variable7>

VT_I8 :1311768467463790320 (0x123456789ABCDEF0)

Example 6) Write three of four bytes data in data memory (DM) channel address 0x0150.

・Register one variable for setting.

AddVariable

<Variable8>

Name : D0150

Option : Mode=NORMAL, VT=I4, Elem=3, Array=TRUE

・Write values

put_Value

<Variable8>

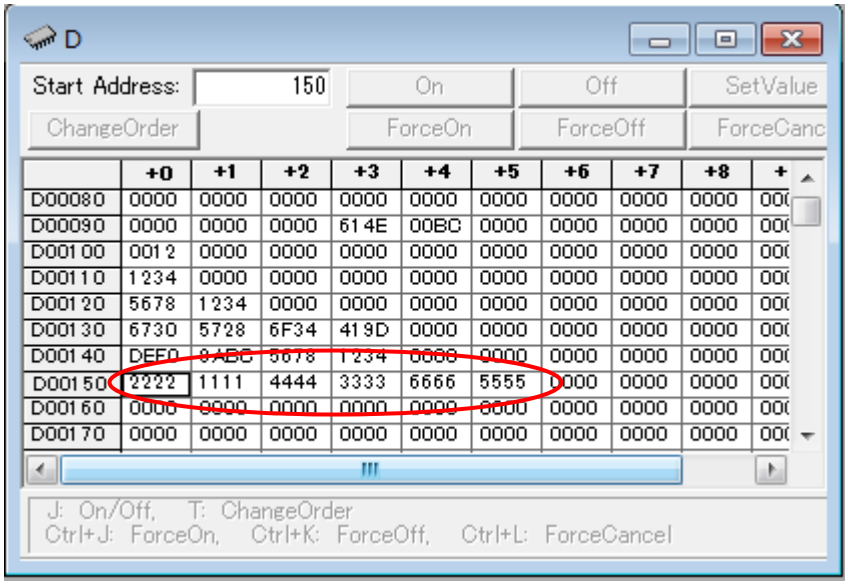
VT_ARRAY | VT_I4 :

286335522 (0x11112222),

858997828 (0x33334444),

1431660134 (0x55556666)

・Memory status in PLC (CJ series) (Use CX-Programmer)



Values of D0150 2222(1101111011110000) } Element 0

Values of D0151 1111(1001101010111100)

Values ofD0152 4444(0101011001111000) } Element 1

Values of D0153 3333(0001001000110100)

Values ofD0154 6666(0001001000110100) } Element 2

Values of D0155 5555(0001001000110100)

・Read this channel address to confirm.

get_Value

<Variable8>

VT_ARRAY | VT_I4 :

286335522 (0x11112222),

858997828 (0x33334444),

1431660134 (0x55556666)

Appendix C. Compatibility processing when VT = BIT is specified for channel address

The behavior when VT = BIT is specified when specifying the channel address has been modified to obtain the value of 1 byte from v1.2.3, and obtain 0 if 0 and 1 if it is not 0. To return to processing below v1.2.3, you need to specify the “BitOpt” option in the AddController option.
See Table 2-9 for details.

Table 2-9 Option character strings of CaoWorkspace::AddController

Option	Description
BitOpt[=< True or False >]	Specify behavior when VT = BIT is specified in acquisition from channel address. ・ FALSE (latest operation) Gets 1 byte of data, returns 0 if it is 0, and returns 1 if it is not 0. ・ TRUE (operation of version less than v1.2.3): Get one byte of data and return the value as it is. (Default: FALSE)

Appendix D. About write processing when VT = BIT, VT = BOOL, VT = UI1, VT = I1 to the channel address is specified

The behavior when VT = BIT, VT = BOOL, VT = UI1, VT = I1 is specified when specifying the channel address can now be selected from v1.2.7. If you create the file after v1.2.7, please specify "True" for this option.

Table 2-10 Option character strings of CaoWorkspace::AddController

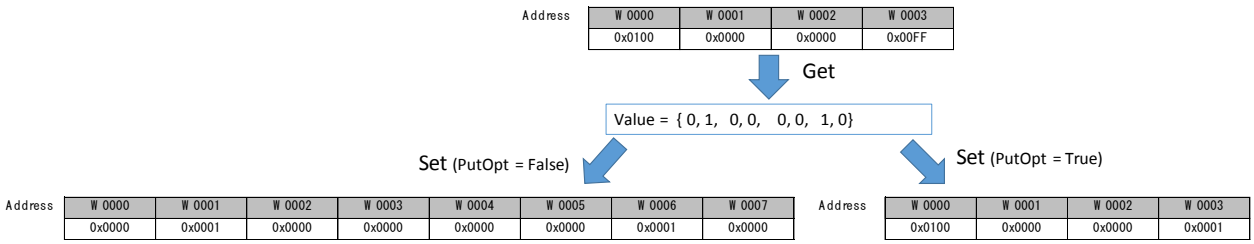
Option	Description
PutOpt[=< True or False >]	・ FALSE (works for versions less than v1.2.7) When writing data, the high-order byte is complemented with 0 and written. ・ TRUE (operation added in v1.2.7 or later) Data is stored and written in order from the lower byte. (Default: FALSE)

The following is an image of each operation.

<Variable>

Name : W0

Option : VT=BIT, Elem=8



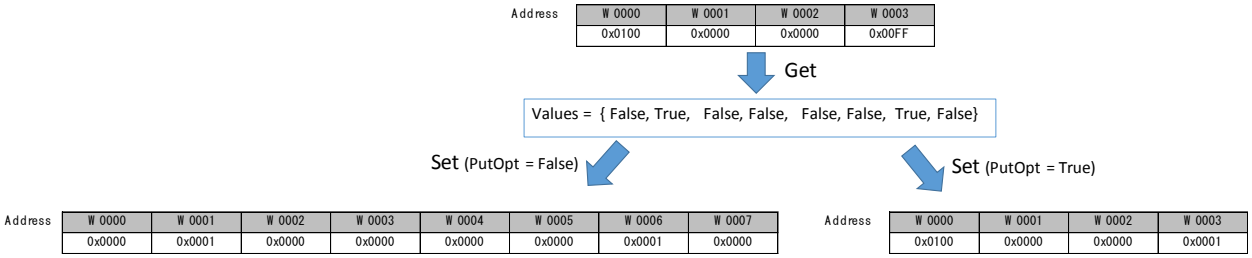
FALSE: When reading, 1-channel data is divided into high-order byte data and low-order byte data, and if each is other than 0, it is set to 1. At the time of writing, 1-bit data is treated as 1-channel data, and the upper byte is complemented with 0 and written to the device.

TRUE: When reading, 1-channel data is divided into high-order byte data and low-order byte data, and if each is other than 0, it is set to 1. At the time of writing, 2-bit data is treated as 1-channel data, and the data is stored in order from the lower byte and written to the device.

<Variable>

Name : W0

Option : VT=BOOL, Elem=8



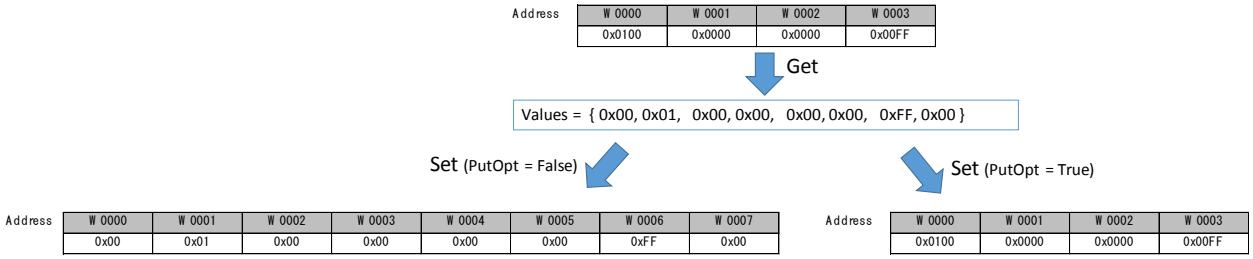
FALSE: When reading, 1-channel data is divided into high-order byte data and low-order byte data, and if each is other than 0, it is set to True. At the time of writing, 1 Boolean data is treated as 1 channel data, and the upper byte is complemented with 0 and written to the device.

TRUE: When reading, 1-channel data is divided into high-order byte data and low-order byte data, and if each is other than 0, it is set to True. At the time of writing, 2-Boolean data is treated as 1-channel data, and the data is stored in order from the lower byte and written to the device.

<Variable>

Name : W0

Option : VT=UI1 or I1, Elem=8



FALSE: At the time of reading, 1 channel data is divided into upper byte data and lower byte data and the value is acquired. At the time of writing, 1-byte data is treated as 1-channel data, and the upper byte is complemented with 0 and written to the device.

TRUE: When reading, 1 channel data is divided into upper byte data and lower byte data and the value is acquired. At the time of writing, 2-byte data is treated as 1-channel data, and the data is stored in order from the lower byte and written to the device.

Appendix E. About Accessing PLCs on the Network

Appendix E.1. PLC settings

To access PLCs on the network, you need to create a routing table.

In addition, it can be accessed only by FinsUDP communication as specified on the protocol. In this case, the connection source port number must match the connection destination port number.

It is to use CX-Integrator to set the routing table.[5]

It is described in the following configuration example.

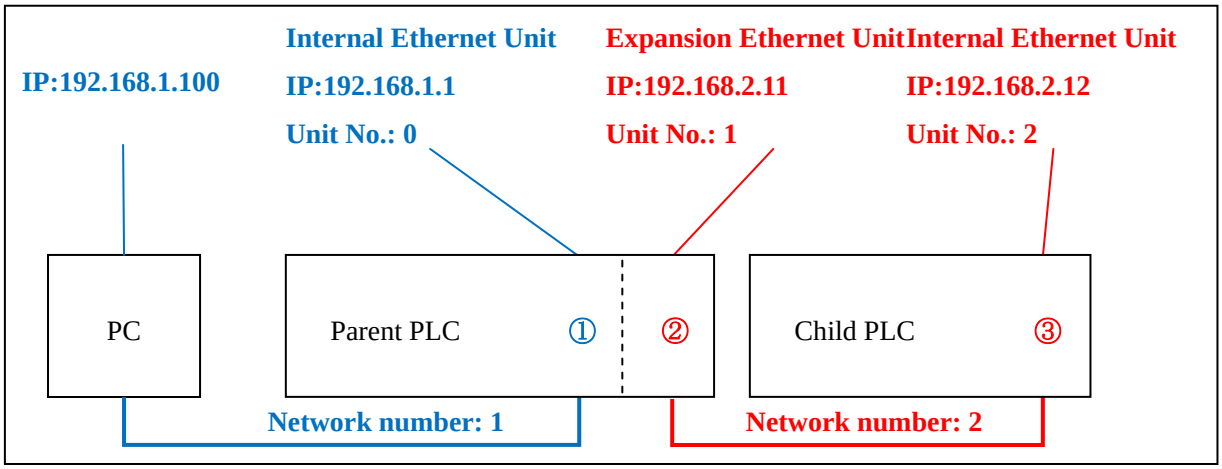


Table 2-11 Description of the target unit

Target unit	Description
①	Built-in Ethernet unit of the main PLC
②	Extended Ethernet unit of the main PLCs
③	Built-in Ethernet unit of the PLCs

[5] Refer to CX-Integrator Help for instructions on how to configure CX Integrator.

Set the routing table of the parent PLC as follows.

Set the routing table of the child PLC as follows.

Appendix E.2. Connection confirmation

This section describes the procedure for checking the connection to the PLC for which the routing table is set.

Table 2-12: Connection check

Step	Description
1	Check the connection to the parent PLC. Set AddController options as follows: Conn=UDP:192.168.1.1:9600:255.255.255.255:9600, FinsParam=1:1:0:2:1:100:0 After connecting, check whether the value can be obtained from the parent PLC. If an error occurs, review the routing table of the parent PLC.
2	Check if the value can be retrieved from the child PLC via the parent PLC. Set AddController options as follows: Conn=UDP:192.168.1.1:9600:255.255.255.255:9600, FinsParam=2:12:0:2:1:100:0 After connecting, check whether the value can be obtained from the child PLC. If an error occurs, review the routing table of the child PLC.

Appendix F. Performance measurements

This section describes the results of performance measurements under the following conditions.

Table 2-1 Performance measurement conditions

Model used	OMRON SYSMAC CJ2H CPU64-EIP
Network	100BASE-TX
CPU	Intel(R) Core(TM) i7-7700 3.60GHz
OS	Windows10 20H2 64bit
Measurement condition	The average time taken to obtain a value of 100 times for a variable that accesses a 2-byte integer tag is used as the execution time.

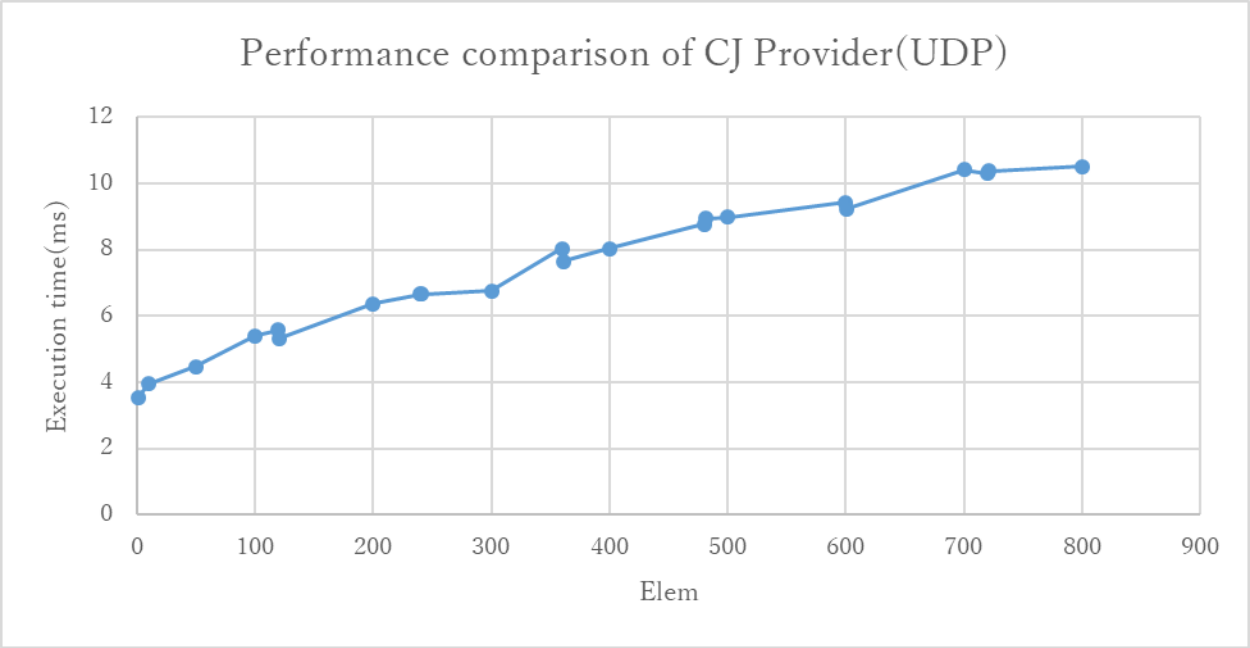


Fig. 2-1 Performance measurements