Digital
Memory link provider

Version 1.1.2

User's guide

December 18, 2019

Remarks:
This document is translated into English by machine translation.

【 revision history 】

Version	Date	Content
1.0.0	2019-04-26	First edition.
1.1.0	2019-06-20	Supported serial communication.
1.1.1	2019-08-21	Fixed memory leak when OnMessage event occurred
1.1.2	2019-12-18	Fixed a bug that the reception process frequently times out

【 Operation check model 】

Model	Version	Notes
Pro-face (GP4100 series)	PFXGP4114T2D	
Pro-face (LT4000M series)	PFXLM4B01DAK	

Contents

1. Introduction

This book is a user's guide of the provider (provider at the following and names) that acquires memory information on the device (indicator and name at the following) with built-in protocol of the link of the memories of the digital company. Moreover, only the Ethernet communication of the memory link protocols corresponds in this provider.

Figure 1-1 shows the overall configuration of the provider and display unit in Ethernet communication, and Figure 1-2 shows the overall configuration in serial communication. This provider communicates with the display unit by Ethernet communication or serial communication.

■ Ethernet communication

・ 1:1Connection

・ 1:n Connection

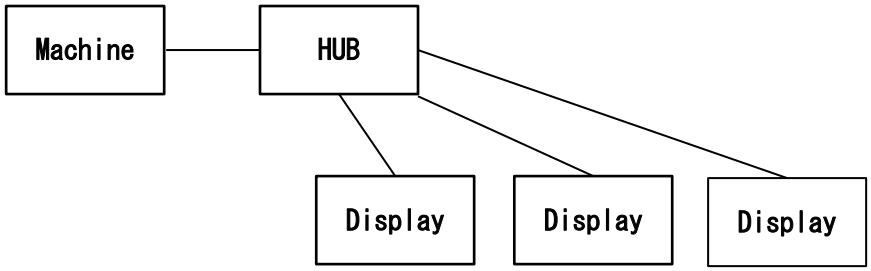
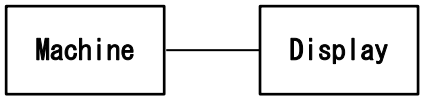


Figure1-1 Ethernet communication configuration diagram

■ Serial communication

・ 1:1 Connection



・ 1:n Connection

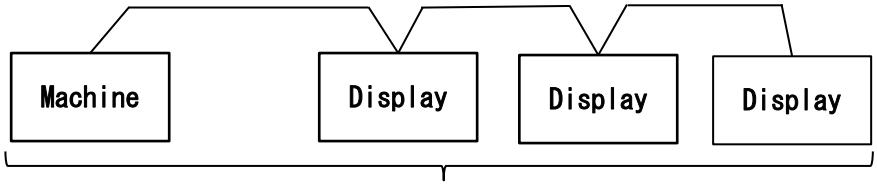


Figure 1-2 Serial communication configuration diagram

Moreover, Figure 1-3,4 shows the provider and the example of the device's corresponding.

For Ethernet and serial (1: 1 connection), use CaoController (see 3.2.2) to acquire data from the display unit. For serial (1: n connection), use CaoExtension (see 3.2.4) to get the data from the display of the specified unit No.

Figure1-3 Composition of provider and correspondence chart with device information

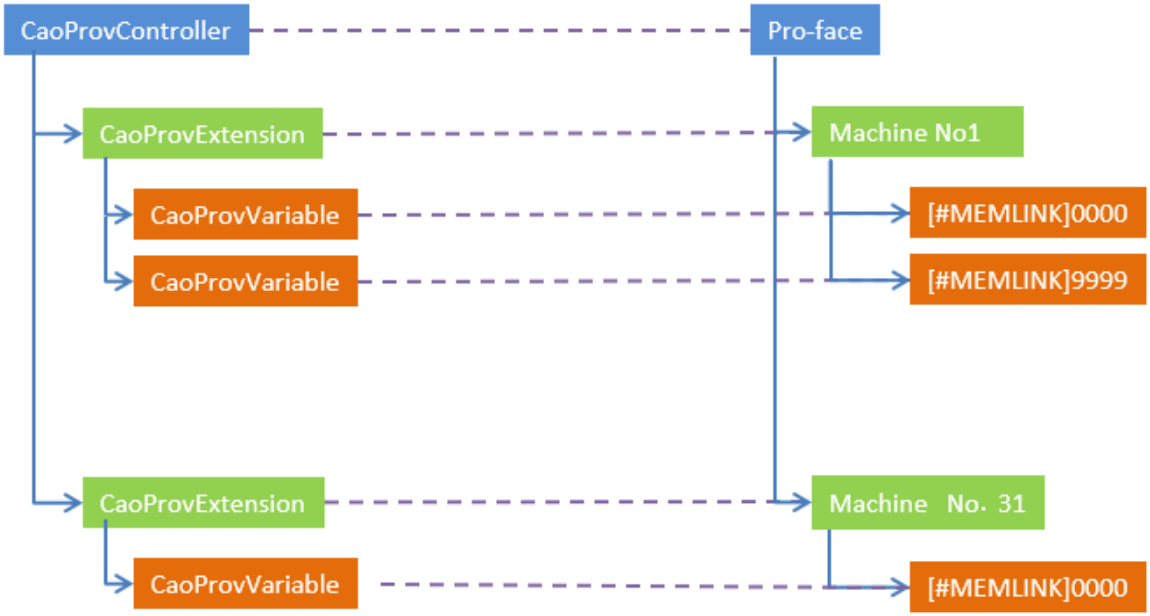


Figure 1-4 Correspondence diagram of provider configuration and device information (Unit No. designation)

1.1. Communication specification of provider

The thread for the reception starts to always observe the demand polling and the error notification sent from the indicator and this provider always confirms receive data from the indicator.

1.1.1. Ethernet communication specifications

1.1.1.1. Processing procedure of reception thread

Figure 1-3 shows the processing procedure of the reception thread.

Receive the data transmitted from the indicator by the confirmation processing of receive data. In the receive data confirmation processing, it stands by during the fixed time even if it is time when it in there is no receive data. Please specify the standby time by RecieveTimeout option of AddController.

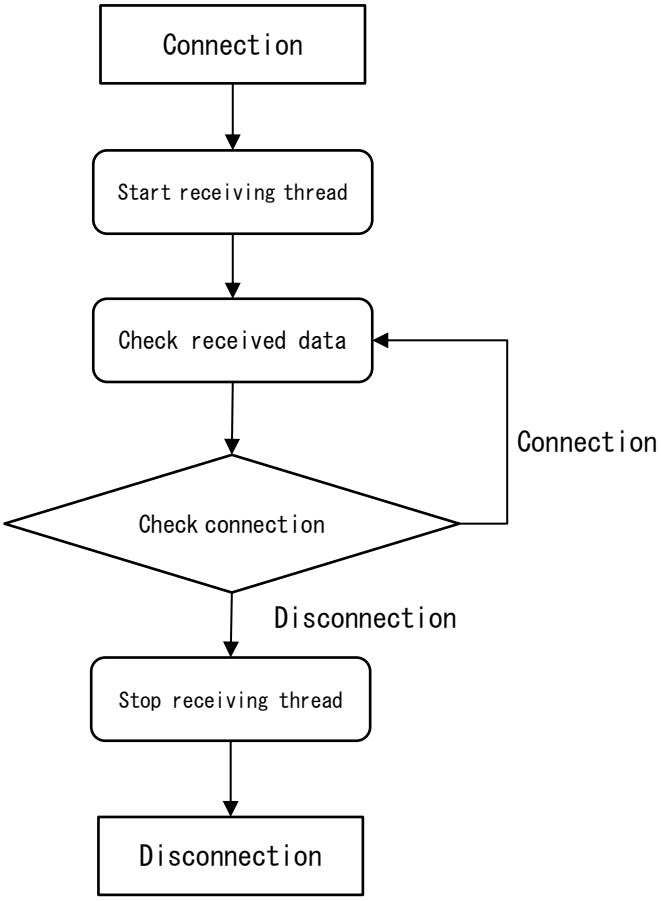


Figure1-5 Reception thread processing procedure flow chart

1.1.1.2. Disconnection detection

This provider does the disconnection detection by setting the option when connecting it. Judge the disconnection, and do the stop of the reception thread and the disconnection process of the socket when you do neither the indicator nor the data communication within a certain fixed time. Refer to optional DisconnectDetectTime of AddController for the specification of effective invalidity of the disconnection detection and time until the disconnection detection is done. The data communication indicates the following operation. Note no inclusion about Keepalive of TCP.

- ·Read command of memory link
- ·Write command of memory link
- ·Demand polling reception from indicator

Moreover, note that it becomes impossible to communicate after that when the disconnection detection is done. Delete CaoController once, and add it again when you want to connect

it again.

1.1.1.3. All processes from the command transmission to the reception

Figure1-6 shows the flow from sending the acquisition command to receiving data from the provider during Ethernet communication. The provider stands by until the data notification from the reception thread comes after transmitting the command. Please specify the standby time by CommandTimeout option of AddController.

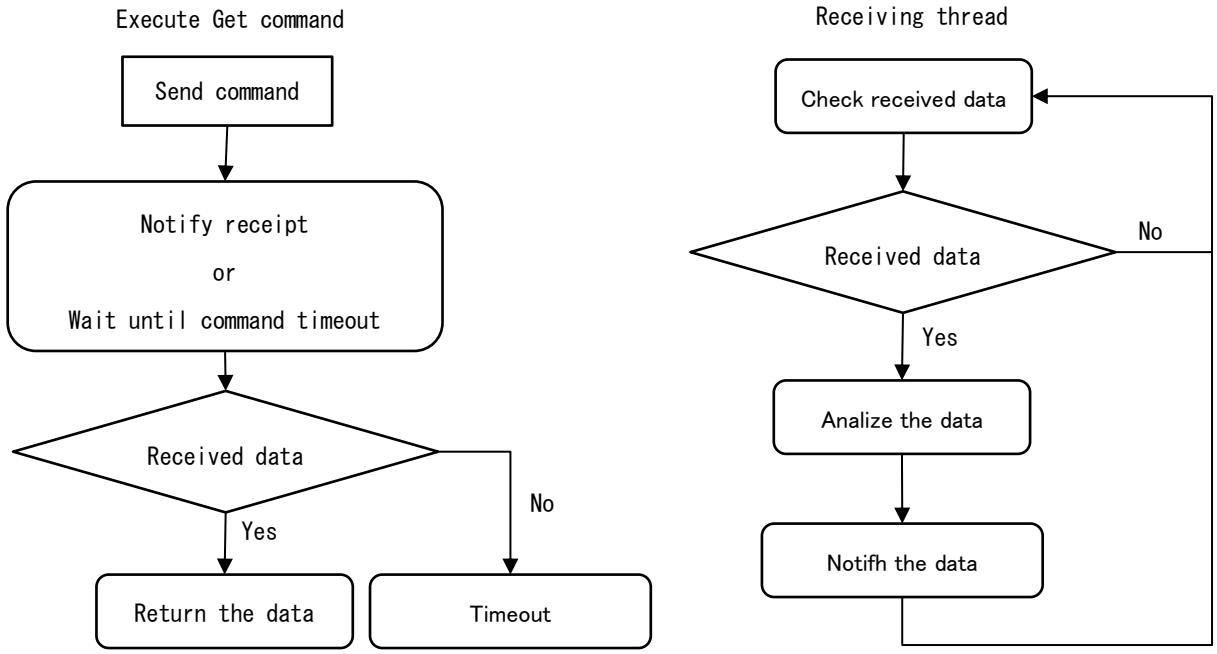


Figure1-6 Flow chart from command transmission to reception

1.1.2. Serial communication specifications

In serial communication, there is no demand polling or error notification function like Ethernet communication, so the reception thread is not used and the received data is confirmed and analyzed after the command is sent.

1.1.2.1. From command transmission to reception in serial communication

Figure1-6 shows the flow from sending an acquisition command from a provider to receiving data during serial communication. After sending the command, the provider checks the received data from the connection destination, and if there is received data, returns it.

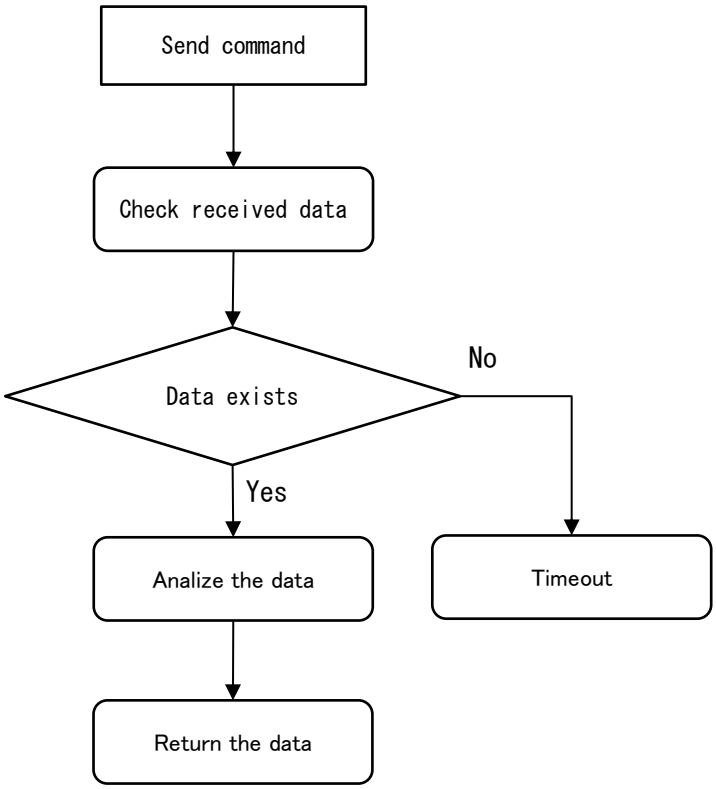


Figure 1-7 Flow chart from command transmission to reception in serial communication

1.1.2.2. Notes on display unit settings

When communicating with a serial connection, set the display unit as follows.

Setting contents	Value
Communication protocol	Extended mode
Communication format	1:1 binary or 1:n binary
ETX, sum check	enabled
ACK	enabled
NAK	enabled

2. Environmental setup for application development

2.1. Connection of indicator and client PC

Refer to the manual of the Pro-face series used for the connection of indicator and client PC.

2.2. Setup of PC development setting

2.2.1. Manual installation of memory link provider

When the memory link provider is installed by hand power, it is necessary to register the following registry. Start the command prompt by the manager authority, and execute the regsvr32 command when you register the registry.

Table2-1 Memory link provider

File name	CaoProvDigitalMemoryLink.dll
ProgID	CaoProv.Digital.MemoryLink
Registry registration	regsvr32 CaoProvDigitalMemoryLink.dll
Blotting out of registry registration	regsvr32 /u CaoProvDigitalMemoryLink.dll

3. Command reference

3.1. Method/property list

Table3-1Method/property list

Category	[1]Method/property		Function	Reference
CaoWorkspace				
	AddController	M	Connect it with the controller.	P.12
CaoController				
	VariableNames	P	Acquisition of variable identifier list that can be connected	P.14
	AddVariable	M	Addition of variable object	P.15
	AddExtension	M	Addition of extension object	P.15
	Execute	M	Execution of enhancing command	P.15
	OnMessage	E	Message reception event	P.16
CaoVariable				
	Value	P	Acquisition/setting of value	P.16

3.2. Method property

3.2.1. CaoWorkspace class

3.2.1.1. AddController method

Add the controller object to CaoWorkspace. In the memory link provider, connect it with the corresponding display machine referring to the parameter passed when the AddController method is executed. The specification of the AddController method is shown as follows.

Format

AddController

```
(
        "< controller name >"          // Controller name (arbitrariness)
        "CaoProv.Digital.MemoryLink",  // Provider name (fixation)
        "< machine name >"             // Provider execution machine name (unused)
        "< option >"                   // Optional character string (It is possible to
omit it).
)
```

[1]M:Method and P: The property is E: The event is shown respectively..

Option

The option specified for an optional character string is shown as follows. An optional character string becomes a character string to which each option shown in the following ties by comma (,).

Ethernet communication			
Option	Indispensability	Explanation	Range of value
CONN=< parameter in communication >	✓	Specify connection destination information.	ETH
CommandTimeout =< command timeout >	--	Specify timeout period (ms) to the data reception after executing the command. Enlarge and set the value when the time-out error occurs frequently.	1 - ULONG_MAX Default : 1000
DisconnectDetectTime=< Disconnection detection time >	--	Specify disconnection detection time (ms). The disconnection detection becomes invalid because it sets 0. Set the numerical value of one or more when you keep effective. A value big as for 1000ms or more is recommended to be specified at the demand polling time of the cycle when the indicator side has the demand polling function. Please refer to chapter 1.1.1.2 for the specification of the disconnection detection.	0 - LONG_MAX Default : 0

Serial communication			
option	Required	Description	Value range
CONN=<communication paramerter>	○	Specify connection destination information.	COM
Timeout=<timeout time>	--	Specify the waiting time (ms) to timeout in one communication with the connection destination.	0 - ULONG_MAX default:1000

Usage example

```
CaoEngine caoEngine;              // Engine object
CaoWorkspace caoWorkSpace;        // Workspace object
CaoController caoController;      // Controller object
```

```
caoEngine = new CaoEngine();
caoWorkSpace = caoEngine.AddWorkspace("NewWrks","");
caoController = caoWorkSpace.Controllers.Add("MemoryLink",
"CaoProv.Digital.MemoryLink", "", "Conn=ETH:192.168.0.150,CommandTimeout=1000");
```

3.2.1.1.1. CONN is optional.

Connected parameter character string of optional Conn is shown as follows. It is shown to omit it here in the square bracket) and the underlined part under the explanation of each parameter shows the default value when the option is not specified respectively.

When connecting it by TCP/IP

"Conn=ETH:< IP >[:< Port > [:< Local IP >[:< Local Port >]]]"

< IP > : Specify IP address of the connection destination.
Specify this item.

< Port > : Specify the connection destination port number.
Default : 1024

< Local IP > : Specify IP address of the local.
Default: Unspecification

< Local port > : Default: Unspecification

When connection it by Serial

"Conn=COM:<COM Port>[:BaudRate>[:<Parity>:<DataBits>:<StopBits>[:Flow]]]"

<COM Port> : COM Port number. '1' -COM1, '2' - COM2, ...

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

<Parity> : Parity. N'-NONE, 'E'-EVEN, 'O'-ODD

<DataBits> : DataBits.'7'-7bit, '8'-8bit

<StopBits> : StopBits. '1'-1bit, '2'-2bit

<Flow> : Flow. 'O'-None, '1'-Xon/Xoff, '2'-Hardware
You can specify by OR.

3.2.2. CaoController class

3.2.2.1. VariableNames property

Acquire the variable identifier list that can be connected. Describe the variable identifier acquired in this property later. The variable identifier acquired in this property can be used for the first argument of the AddVariable method of the following description.

Usage example

```
    // File name list acquisition
    object variables;
    variables = controller.VariableNames;
```

3.2.2.2. AddVariable method

Add the variable object to CaoController. Only the one shown in 3.4.1 can be used for the variable identifier.

The specification of AddVariable is shown as follows.

Format

AddVariable

```
(
        "< variable identifier >"       // Variable identifier
        "< option >"            // Optional character string (It is possible to omit it).
)
```

3.2.2.3. AddExtension method

Add the extension object to CaoController.

Thne specification of AddExtension is shown as follows.

Fomat

AddExtension

```
(
        "<extension name>",     // name (Any string)
        "<option>"              // option string (required)
)
```

Option

The options specified in the option string are shown below. The option string is a string in which the following options are connected by a comma (,).

option	required	commnet	value range
MachineNo=<number>	✔	Number of Display.	0 - 31

3.2.2.4. Execute method

Execute the enhancing command of ConController. The specification of Execute is shown as follows.

Format

Execute

```
(
        "< enhancing command name >"          // Enhancing command name
        "< optional character string >"       // Optional character string (It is
possible to omit it).
)
```

Please refer to 3.3 for the enhancing command list that can be specified with Execute.

3.2.2.5. OnMessage event

Controller's error notification and the change in the state can be received as OnMessage event. Please refer to Chapter 3.5 for details of the event.

3.2.3. CaoVariable class

3.2.3.1. Value property

Data acquisition/is set from the connected indicator. Operation is different depending on the variable identifier. Details are 3.4. Please refer to the variable list.

3.2.4. CaoExtension class

This class cannot be used when connected via Ethernet.

It communicates with the display of the specified unit number by specifying the unit number at serial (1: n connection).

3.2.4.1. VariableNames property

Get a list of connectable variable names. The variable name obtained with this property can be used as the first argument of the AddVariable method described later.

example

```
    // get fine name list.
    object variables;
    variables = extension.VariableNames;
```

3.2.4.2. AddVariable method

Add a variable object to CaoExtension. Only the variables shown in 3.4.2 can be used. The specification of AddVariable is shown below.

Format

AddVariable

```
(
        "<name>",               // variable name
        "<option>"              // option string
)
```

3.2.4.3. Execute method

Execute the extension command of ConController. The specification of Execute is shown below.

Format

Execute

```
(
        "<command name>",               // command name
        "<option>"                      // option string
)
```

See 3.3 for a list of extended commands that can be specified with Execute. Examples of use are described in detail of extended commands.

3.3. Command list

In the command, set memory information in the indicator acquiring.

Command	Explanation	Reference
GetMemory	Acquire data for a few minutes of the element specified from the specified address beginning position.	P.18
SetMemory	Set data to the address area of the memory link.	P.19
GetMemoryRaw	Acquire the address area data of the memory link in the life value.	P.20
SetMemoryRaw	Set data to the address area of the memory link.	P.21
SendRawCommand	Transmit a set value as it is as command data.	P.22

About unlike the acquisition setting command

In the memory link protocol, the value is exchanged by the big endian in two byte address. In Raw system (GetMemoryRaw/SetMemoryRaw) command, treat like two byte data of a

communication within specification and the big endian. On the other hand, exchange it in the (GetMemory/SetMemory) command by data that converts the type to become a specified data type (Include the endian conversion).

About the number of addresses that can be exchanged by one communication

The maximum number of addresses where acquisition can be set by one communication in the communication specification of the memory link protocol becomes 512. It takes time to the amount communication to communicate two or more times when the number of addresses exceeds 512 in this provider.

About the data acquisition setting specification of VT_I1 and VT_UI1 in the type specification command

Do the acquisition setting to one subordinate position byte of the address when you do the acquisition setting of data with VT_I1 VT_UI1. Set it after masking doing one high rank byte of the address data that acquires information on the address set once and acquires it when setting it. Therefore, note that the wire traffic increases to the extent that it acquires compared with other type specification and it sets it in VT_I1 and VT_UI1.

3.3.1. GetMemory

Acquire it in the data type that specifies data from the address of the link of the memories of the indicator.

Specify the acquired number of elements of the address beginning position and data and types of data. Acquire it in the type that specified data from the specified address beginning position for a few minutes of the element.

Item	Type explanation		
Argument	VT_ARRAY | VT_VARIANT		
	1	VT_UI2	Specify the acquired address beginning position. Range of value: 0-9999
	2	VT_UI2	Specify the acquired number of elements. Please refer to 3.4.1.4.1 for details.
	3	VT_BSTR	Specify the acquired data type. Please refer to 3.4.1.4.1 for details.
	4	VT_BOOL	It is possible to omit it. Specify whether for number 1 of reading data to arrange the return value. TRUE: Return it by the array.

Item	Type explanation		
Argument	VT_ARRAY | VT_VARIANT		
Return value	VT_ARRAY | VT_VARIANT (specified data type)		
	n	VT_VARIANT	Acquire data for a few minutes of the element in the specified data type.

Usage example

```
// Making of argument parameter (Treat 50 elements from 100 addresses as VT_I2).
object[] objParam = { 100, 50, "I2"};

// Acquisition of data
object returnValue;
returnValue = this.m_caoController.Execute("GetMemory", objParam);
// Convert the acquired object data into the array of the unsigned short type.
short[] retVal = (short[])returnValue;
```

3.3.2. SetMemory

Set it by the data type that specifies data for the address of the link of the memories of the indicator.

Specify the number of elements set as the address beginning position, the type of data, and the set data. Set it according to the data type specified from the specified address beginning position.

Item	Type explanation		
Argument	VT_ARRAY | VT_VARIANT		
	1	VT_UI2	Specify the set address beginning position. Range of value: 0-9999
	2	VT_UI2	Specify the set number of elements. Please refer to 3.4.1.4.1 for details.
	2	VT_BSTR	Specify the set data type. Please refer to 3.4.1.4.1 for details.
	3	VT_ARRAY | VT_VARIANT (specified data type)	
		n	Specify the data written from the memory address beginning position in an nth area.

Item	Type explanation
Argument	VT_ARRAY | VT_VARIANT
Return value	None

Usage example

```
// Making of argument parameter
object[] objParam = new object[4];
objParam[0] = 100; // Specification of set address beginning position
objParam[1] = 2; // Set number of elements
objParam[2] = "R4"; // Data type VT_R4
objParam[3] = new float[2]{1.2f, 2.3f}; // The setting data

// Setting of data
this.m_caoController.Execute("SetMemory", objParam);
```

3.3.3. GetMemoryRaw

Acquire data from the address of the link of the memories of the indicator in the life value. Specify the number of elements of the address beginning position and data.

Acquire data from the specified address beginning position for a few minutes of the data element. The acquired data acquires the life value that has been sent from the indicator with VT_UI2.

Item	Type explanation		
Argument	VT_ARRAY | VT_VARIANT		
	1	VT_UI2	Specify the acquired address beginning position. Range of value: 0-9999
	2	VT_UI2	Specify the number of elements of acquired data. Range of value: 1-10000
	3	VT_BOOL	It is possible to omit it. Specify whether to arrange the return value when the acquired number of elements is one. TRUE: Return it by the array.
Return value	VT_ARRAY | VT_UI2 Or, VT_UI2.		
	n	VT_UI2	Acquire the data of the specified address area.

Usage example

```
// Making of argument parameter (Treat 50 elements from 100 addresses as VT_UI2).
object[] objParam = {100, 50};

// Acquisition of data
object returnValue;
returnValue = this.m_caoController.Execute("GetMemoryRaw", objParam);
// Convert the acquired object data into the array of the unsigned short type.
ushort[] retVal = (ushort[])returnValue;
```

3.3.4. SetMemoryRaw

Set data from the address of the link of the memories of the indicator by the life value. Specify the address beginning position and the setting data, and set the data specified from the starting position by the life value.

Item	Type explanation		
Argument	VT_ARRAY | VT_VARIANT		
	1	VT_UI2	Specify the set address beginning position. Range of value: 0-9999
	2	VT_UI2	Number of set elements Range of value: 1-10000
	3	VT_ARRAY | VT_UI2 Or, VT_UI2.	
		n	Specify the set data of a few minutes of the element from the memory address beginning position.
Return value	None		

Usage example

```
// Making of argument parameter
object[] objParam = new object[3];
objParam[0] = 110; // Specification of set address beginning position
objParam[1] = 5; // Number of set elements
objParam[2] = new ushort[5]{1, 2, 3, 4, 5}; // Write data
// Setting of data
this.m_caoController.Execute("SetMemoryRaw", objParam);
```

3.3.5. SendRawCommand

Transmit the raw data of the command.

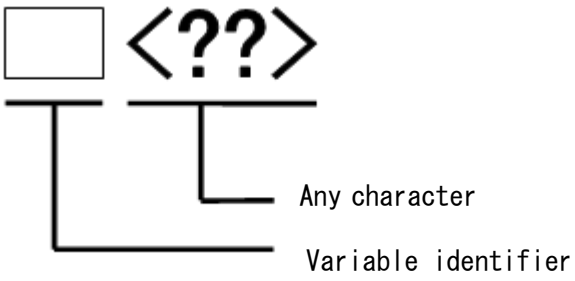
Item	Type explanation		
Argument	VT_ARRAY | VT_UI1		
	n	VT_UI1	Command byte array
Return value	VT_ARRAY | VT_UI1		
	n	VT_UI1	Byte array of return value

3.4. Variable list

The variable list that can be used in each class is shown in 3.4.1. The variable indicates the object of the CaoVariable class. To register two or more variables (usefulness to change only the option), an arbitrary character string can be given.

The format to give the variable identifier an arbitrary character string is shown below.

Two or more variable commonness and specification formats



3.4.1. CaoController class variable

Variable identifier	Explanation	Value		Reference
		get	put	
@MAKER_NAME	Acquire the manufacturer name.	✔	-	P.23
@VERSION	Acquire the DLL version.	✔	-	P.23
@LAST_DEVICE_ERROR	Acquire the error code corresponding to the error notification message notified at the end.	✔	-	P.23
MEMLINK<??>	Acquisition/set memory information in the indicator.	✔	✔	P.24

MEMLINKRAW<??>	Acquisition/set memory information in the indicator by the life value.	✔	✔	P.26

3.4.1.1. @MAKER_NAME

Acquire the manufacturer name.

Data type

Type explanation	
VT_BSTR	Acquire the manufacturer name.

Usage example

```
// Variable addition
CaoVariable caoVariable;
caoVariable = this.m_caoController.AddVariable ("@MAKER_NAME");
// Value acquisition
String strMakerName;
strMakerName = caoVariable.Value;
```

3.4.1.2. @VERSION

Acquire the version of DLL.

Data type

Type explanation	
VT_BSTR	Acquire the version of DLL. *.*.*

Usage example

```
// Variable addition
CaoVariable caoVariable;
caoVariable = this.m_caoController.AddVariable("@VERSION");
// Value acquisition
String strVersion;
strVersion = caoVariable.Value;
```

3.4.1.3. @LAST_DEVICE_ERROR

The error code corresponding to the message content notified besides the error notification message (3.5.2.1-3.5.2.3 references) on the device side at the end is acquired. Please refer to error notification message (3.5.2.1-3.5.2.3) on the device side for the corresponding error number.

Type explanation	
VT_BSTR	Acquire the error code corresponding to the error message.

Usage example

```
// Variable addition
CaoVariable caoVariable;
caoVariable = this.m_caoController.AddVariable("@LAST_DEVICE_ERROR");
// Error code acquisition
object LastErrorCode = caoVariable.Value;
```

3.4.1.4. MEMLINK<??>

Set memory information in the indicator acquiring.

Type explanation	
VTARRAY | VT_UI2	Set the value of the specified memory address area acquiring.

Option	Indispensability	Explanation	Range	Default
StartPos=[<Memory address beginning position >]	--	Specify the starting position of the memory address that does the acquisition setting.	0-10000	0
Elem [=< the number of of element >]	--	Specify the number of elements of data that does Put/Get. The range of the value is different according to a specified type. Please refer to 3.4.1.4.1 for the range of the value of a specified each type.	1-20000	2: when data type is BSTR 1:Other data type
VT[=<Variable type >]	--	Specify the data type that does Put/Get.	--	UI2
Array[=True or False]	--	Specify whether to treat as an array when the number of elements in which Put/Get is done is one. Treat as True = array.	--	False

3.4.1.4.1. VT is optional.

Specify the data type that does Put/Get.

A list of the data type that can be specified and an optional range of Elem of each data type are shown below.

VT	Data type	Explanation	Ranges of element number
I1	VT_I1	There is a sign of 8bit and it read/writes it as an integer type. One subordinate position byte of one address (two byte data) is assumed to be one element.	1-10000
I2	VT_I2	There is a sign of 16bit and it read/writes it as an integer type.	1-10000
I4	VT_I4	There is a sign of 32bit and it read/writes it as an integer type. Two the addresses (two byte data) worth of data is assumed to be one element.	1-5000
UI1	VT_UI1	Read/write it as a data of sign none type of 8bit. One subordinate position byte of one address (two byte data) is assumed to be one element.	1-10000
UI2	VT_UI2	Read/write it as an integer of sign none type of 16bit.	1-10000
UI4	VT_UI4	Read/write it as an integer of sign none type of 32bit. Two the addresses (two byte data) worth of data is assumed to be one element.	1-5000
R4	VT_R4	Read/write it as a floating type of 32bit. Two the addresses (two byte data) worth of data is assumed to be one element.	1-5000
BSTR	VT_BSTR	Read/write it as a character-code for one element byte. Specify the range of the element by the even number so that the data of one address may become two bytes' worth of a character-code.	2-20000

Usage example

```
    // Variable addition
    CaoVariable caoVariable;
    // The memory address beginning position is made, and make 110 acquisition setting numbers of addresses and make the variable by two.
    caoVariable = this.m_caoController.AddVariable("MEMLINK_1", "StartPos=110, Elem=2, VT=I2");
    // Error code acquisition
    object memoryData = caoVariable.Value;
    ushort[] ushMemData = (ushort[])memoryData;
    // One addition to the acquired data.
    for(int i= 0; i < ushMemData.Length ; i++ )
    {
```

```
        ushMemData[i] += 1;// One addition.
    }
    caoVariable.Value = ushMemData; // Reflect the updated data.
```

3.4.1.5. MEMLINKRAW<??>

The acquisition setting is done by the life value memory information in the indicator.

Type explanation	
VTARRAY | VT_UI2	The life value ..value.. acquisition setting of the specified memory address area is done.

Option	Indispensability	Explanation	Range of value	Default value
Memory address beginning =< position StartPos >	--	Specify the starting position of the memory address that does the acquisition setting.	0-10000	0
Elem =< the number of of element >	--	Specify the number of elements of data that does Put/Get.	1-10000	1
Array[=True or False]	--	Specify whether to treat as an array when the number of elements in which Put/Get is done is one. Treat as True = array.	--	False

Usage example

```
    // Variable addition
    CaoVariable caoVariable;
    // The memory address beginning position is made, and make 110 acquisition setting numbers of addresses and make the variable by two.
    caoVariable = this.m_caoController.AddVariable("MEMLINKRAW_1", "StartPos=110,Elem=2");
    // Error code acquisition
    object memoryData = caoVariable.Value;
    ushort[] ushMemData = (ushort[])memoryData;
    // One addition to the acquired data.
    for(int i= 0; i < ushMemData.Length ; i++ )
    {
        ushMemData[i] += 1;// One addition.
    }
    caoVariable.Value = ushMemData; // Reflect the updated data.
```

3.4.2. CaoExtension class variable

The following is a list of variables that can be used in CaoExtension.

variable name	commnet	Value		refer
		get	put	
MEMLINK<??>	Get / set memory information in the display unit.	✔	✔	P.24
MEMLINK_RAW<??>	Get / set memory information in the display unit with raw values.	✔	✔	P.26

3.5. Event list

Notify the cutting message when the communication doesn't come even if the error notification message from the device side and the specified demand polling receipt time are exceeded.

Message number	Explanation	Reference
Provider error notification		
0	Device cutting notification error	P.28
99	Receive data error	P.28
Device error notification		
10	Living confirmation time-out error	P.28
11	Reception time-out error	P.28
12	Response time-out error	P.29
13	Reception of unconfirmed error	P.29

Usage example

```
public Main()
{
      m_caoEngine = new CaoEngine();
      m_caoWorkSpace = m_caoEngine.AddWorkspace("NewWrks", "");
      m_caoController = m_caoWorkSpace.Controllers.Add("MemoryLink",
      "CaoProv.Digital.MemoryLink", "", "Conn=ETH:192.168.0.150,CommandTimeout=1000");

      // Registration of OnMessage event handler
      m_caoController.OnMessage += new
_ICaoControllerEvents_OnMessageEventHandler(OnMessage);
}
```

```
// Message reception processing
private void OnMessage(CaoMessage pICaoMsg)
{
      try
      {
            // Display ID of the reception message.
            MessageBox.Show(pICaoMsg.Number.ToString());
      }
      catch (Exception ex)
      {
            MessageBox.Show(ex.Message);
      }
}
```

3.5.1. Provider side message notification

It becomes a message notified from the provider.

3.5.1.1. Informative message of device cutting

The specified communication at time is notified optional DisconnectDetectTimeout it from the time communicated at the end when not done. It enters the state that cannot be communicated with the indicator and connect it again again after it notifies.

3.5.1.2. Reception error message of data outside assumption

The content of receive data was received and the one outside assumption was received. Confirm the version of Dll and the version of the memory link.

3.5.2. Device error notification

Notify the error notification message that the device of the connection destination (indicator) has notified.

3.5.2.1. Other party living watch time-out error message

It is an error message notified from the device side.

The content of the error message becomes "Error Alive time out".

Update the value of variable @LAST_DEVICE_ERROR in "0x80100001" when this error is notified.

3.5.2.2. Time-out error message between protocols

It is an error message notified from the device side.

The content of the error message「Error Response time out」

Update the value of variable @LAST_DEVICE_ERROR in "0x80100002" when this error is notified.

3.5.2.3. Time-out error frame message between characters

It is an error message notified from the device side.

The content of the error message becomes "Error Receive time out".

Update the value of variable @LAST_DEVICE_ERROR in "0x80100003" when this error is notified.

3.5.2.4. Reception of message of exception error

It becomes a reception from the indicator the message not assumed.

Refer to the specification of the notification of the error message of the memory link protocol for the content of this error message.

Update the value of variable @LAST_DEVICE_ERROR in "0x80100004" when you receive the exception error message.

4. Sample programming

In the memory link provider, client PC and the indicator can be connected according to the following procedures.

- Making of CaoEngine
- Making of CaoWorkspace
- Making of CaoController

After it connects it with the indicator, it can access information on the indicator by using the Execute method of CaoController or generating the CaoVariabe object.

4.1. Outline

The sample program that reads and writes the value of the data register of global OM of PLC as an example is shown here. Table 4-1 shows the requirement for the sample program.

Table4-1Requirement for sample program

Requirement	Explanation
Connection destination	Connect it by TCP/IP.
	Connection destination Internet Protocol address is 192.168.0.150.
	The connection destination port number is 1024.
Content of processing	Read the value from Proface # MEMLINK110.
	Write value +1 acquired from Proface # MEMLINK110.

A concrete code is shown from the following paragraphs.

4.1.1. Sample program

The whole image of the sample program is shown as follows.

Sample **MemoryLinkGetSet.cs**

```
// object
private CaoEngine m_caoEngine;
private CaoWorkspace m_caoWorkSpace;
private CaoController m_caoController;

private List<short> m_memoryListData = new List<short>();

public MemoryLinkGetSet()
{
    // Specify the memory beginning position and the number.
    int iStartPos = 110;
    int iElem = 2;
```

```
    // Connection
    Connect();
    GetMemoryData(iStartPos,iElem);
    SetMemoryData(iStartPos,iElem);
    // Cutting
    Disconnect();
}

// Connection method
private void Connect()
{
    this.m_caoEngine = new CaoEngine();
    this.m_caoWorkSpace = this.m_caoEngine.AddWorkspace("NewWrks","");
    this.m_caoController = m_caoWorkSpace.Controllers.Add("MemoryLink",
                           "CaoProv.Digital.MemoryLink",
                            "",
                           "Conn=ETH:192.168.0.150");
}

// Cutting method
private void Disconnect()
{
    if(this.m_caoController != null)
    {
        // Delete CaoController from CaoWorkspace.
     this.m_caoWorkSpace.Controllers.Remove(this.m_caoController.Name);
    // Deletion of CaoController
     System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoController);
     this.m_caoController = null;
    }
    // Delete CaoWorkspace from CaoEngine.
    this.m_caoEngine.Workspaces.Remove(this.m_caoWorkSpace.Name);
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoWorkSpace);
    // Delete CaoWorkspace.
    this.m_caoWorkSpace = null;
    // Delete CaoEngine.
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoEngine);
    this.m_caoEngine = null;

}
// Method of memory data acquisition
private void GetMemoryData(int iStartPos,int iElem)
{
    // Setting of parameter
    object[] objParam = { iStartPos, iElem , "I2"};
    // Acquisition of data
```

```
    object returnValue;
    returnValue = this.m_caoController.Execute("GetMemory", objParam);
    // Convert the acquisition data into the array of ushort.
    short[] retVals = (short[])returnValue;

    this.m_memoryListData.Clear();
    // Store the acquired data in the list.
    foreach(short memData in retVals)
    {
        this.m_memoryListData.Add(memData);
    }
}
// Method of setting memory data
private void SetMemoryData(int iStartPos,int iElem)
{
    // Setting of parameter for writing
    short[] writeData = new short[iElem];
    object[] objParam = new object[4];

    for (int i = 0; i < iElem; i++)
    {
    // One addition to data and store it in the array for the setting data.
        this.m_memoryListData[i]++;
        writeData[i] = this.m_memoryListData[i];
    }
    objParam[0] = iStartPos;
    objParam[1] = iElem;
    objParam[2] = "I2";
    objParam[3] = writeData;
    // Execute writing.
    this.m_caoController.Execute("SetMemory", objParam);
}
```

4.1.1.1. Connection

Take the following procedures to connect it with the indicator.

(1) Prepare the variable to maintain the object. The object necessary for the controller

connection is CaoEngine object, CaoWorkspace object, and CaoController object. When the CaoController object is acquired from CaoWorkspaces, the CaoWorkpace object need not prepare the variable. Moreover, the CaoVariable object to access the variable is needed. The example of the code in VB6 is shown as follows.

```
private CaoEngine m_caoEngine;                // Variable for CaoEngine object
private CaoWorkspace m_workSpace;    // Variable for CaoWorkspace object
private CaoController m_caoController;       // Variable for CaoController object
```

(2) Generate the CaoEngine object. The CaoEngine object uses and generates the New key word.

```
 //  generation of CaoEngine object
this.m_caoEngine = new CaoEngine();
```

(3) Acquire the CaoWorkspace object or generate it. When the CaoEngine object is generated, the CaoWorkspaces object and the CaoWorkspace object are generated with default one by one. The example of the code of newly generating the CaoWorkspace object and CaoWorkspace of default are shown as follows.

```
 //  generation of CaoWorkspace object
this.m_caoWorkSpace = this.m_caoEngine.AddWorkspace("NewWrks","");
```

(4) Generate the CaoController object. Set the CaoController object and set the provider name used and the parameter to use it to generate it. In the memory link provider, specify Internet Protocol address in the option. The example of the code is shown as follows.

```
// Generation of CaoController object
this.m_caoController = m_caoWorkSpace.Controllers.Add("MemoryLink",
                        "CaoProv.Digital.MemoryLink",
                         "",
                         "Conn=ETH:192.168.0.150");
```

4.1.1.2. Acquisition of memory data

(1) Make the parameter of the argument passed to the Execute command.

```
// Setting of parameter
object[] objParam = { iStartpos, iElem, "I2"};
```

(2) Acquire data.

```
// Acquisition of data
object returnValue;
returnValue = this.m_caoController.Execute("GetMemory", objParam);
```

(3) Convert the acquired data into the array of the short type and set it to the list.

```
// Convert the acquisition data into the array of short.
short[] retVals = (short[])returnValue;

this.m_memoryListData.Clear();
// Store the acquired data in the list.
foreach(short memData in retVals)
{
      this.m_memoryListData.Add(memData);
}
```

4.1.1.3. Setting of memory data

(1) Setting of parameter for writing. Set it after one addition when you set the write data.

```
// Setting of parameter for writing
short[] writeData = new short[iElem];
object[] objParam = new object[3];
// Set it after one addition to the acquired data.
for (int i = 0; i < iElem; i++)
{
      // One addition to data and store it in the array for the setting data.
       writeData[i] = this.m_memoryList[i] + 1;
}
objParam[0] = iStartpos;  // Address beginning position
objParam[1] = "I2";
objParam[2] = writeData; // Writing data
```

(2) Execute writing by the content of the set parameter.

```
// Execute writing.
this.m_caoController.Execute("SetMemory", objParam);
```

4.1.1.4. Cutting

Delete the generated object, and delete the object deleted from the collection class that manages the object when cutting it with the controller. The example of the code is shown as follows.

```
//  delete CaoController from CaoWorkspace.
```

```
    Call workspace.Controllers.Remove(controller.Index)
     //  deletion of CaoController
    Set controller = Nothing
     //  delete CaoWorkspace from CaoEngine.
    Call engine.Workspaces.Remove(workspace.Index)
     //  deletion of CaoWorkspace
    Set workspace = Nothing
     //  deletion of CaoEngine
    Set engine = Nothing
```

5. Memory link provider error code

In this provider, the following original error codes in which the mask is done with 0x8011**** exist. (Table5-1Reference)

Please refer to the chapter of the error code of "ORiN2SDK user's guide" for common error of ORiN2.

Table5-1Original error code table

Error number	Explanation
0x80110003	Reception error of data outside assumption Review the communication environment.
0x80110004	Type specification error Review a specified type of data. Please refer to Chapter 3.4.1.4.1 for details.
0x80110005	Set element number error It is not corresponding to the number of elements that the number of elements of setting data specifies. Please refer to Chapter 3.4.1.4.1 for details.
0x80110006	When the type specification was VT_BSTR, the odd number was set as a number of specified elements. The number of elements must become an even number when the type specification is VT_BSTR.

The error code corresponding to the error message from the device side by the error notification is followed. Table5-2Refer to [wo].

Table5-2Error code table

Error number	Explanation
0x80100001	Other party living watch time-out error message Please refer to chapter 3.5.2.1 for details.
0x80100002	Time-out error message Please refer to chapter 3.5.2.2 for details.
0x80100003	Error notification out error message. Please refer to chapter 3.5.2.3 for details.
0x80100004	Reception message of other error. Please refer to chapter 3.5.2.4 for details.

Moreover, this provider does the mask by "0x801010**" and returns the error code from the

device.

Please refer to the memory link command error code list for a detailed error code from the device side..

Appendix_A Memory link command error code list

Error number	Explanation
06	The checksum code is not corresponding.
10	An undefined command was received.
12	The number of specified data is not corresponding to the number of receive data.
15	The specified display attribute is data outside the format.
16	The specified font size is data outside the format.
17	The specified coordinate data is data outside the format.
18	Line kind specified code is data outside the format.
19	The specified tiling pattern is data outside the format.
1A	The specified radius exceeds the display region.
1B	The specified beginning angle/end angle is data outside the format.
1C	Character kind specified code is data outside the format.
1D	The specified rotation code is data outside the format.
1E	The specified direction code is data outside the format.
1F	The specified emphasis code is data outside the format.
20	The specified arrow pattern is data outside the format.
21	The specified arrow direction code is data outside the format.
22	The specified method of the chamfering is data outside the format.
23	The specified centering code is data outside the format.
24	The specified attribute code is data outside the format.
25	The contrast adjustment command was transmitted to the model that was not able to adjust the contrast.
26	The specified contrast setting value is outside the range.
27	The brightness adjustment command was transmitted to the model that was not able to do the brilliance control.
28	The specified brightness setting value is outside the range.
29	The flow message is not set.
2A	The specified font code is outside the format.

Error number	Explanation
2B	The specified priority code is outside the format.
FA	The address in the specified system area is outside the range.
FB	It wrote/read it exceeding the range in the specified system area.
FC	Abnormality is found in the received data format.
FF	The state that the indicator was not able to transmit data continued for ten seconds or more.

Appendix_B Table for communication protocol command

CaoController::Execute

Command	Communication command
SetMemory	Write
GetMemory	Read
SetMemoryRaw	Write
GetMemoryRaw	Read