Stream provider

DENSO general purpose communication

Version 1.0.6

User's guide

July 15, 2022

【 Revision history 】

Version	Date	Content
1.0.0.0	2006-02-23	First edition.
1.0.1.0	2009-09-02	b-CAP mode addition and LockMsg option addition
1.0.1.1	2010-02-11	Addition of error code
1.0.2.0	2011-09-26	Addition of synchronous mode and error code.
1.0.3.0	2012-07-03	Receive command correction and command addition
1.0.3	2012-07-17	Document versioning rules was changed.
1.0.4	2012-08-22	“@ConnectCount" The variable is added.
	2015-07-21	"MyPort" option addition.
	2015-07-30	GetStatus command addition.
1.0.5	2019-11-11	“CancelClearOnReceive” option addition.
	2020-07-10	Corrects description of Conn option for Ethernet devices. Corrects an error in the AddController option string in the sample program.
	2021-09-03	Some notations in the document were corrected. Addition of IP address and port number specification method when server mode is specified. Modify all variable names to uppercase. Removed invalid combinations of PacketOpt option Mode option value and supported mode list.
1.0.6	2022-07-15	Fixing a Failure That Cannot Connect Without Lowercase COM When Specifying CONN Optional COM Adding Multicast Options

【 Hardware 】

Model	Version	Notes

Contents

1. Introduction.......... 4
2. Outline of provider.......... 5
2.1. Outline.......... 5
2.2. Operational mode.......... 5
2.2.1. Asynchronous mode.......... 5
2.2.2. Synchronous mode.......... 5
2.3. Method property.......... 6
2.3.1. CaoWorkspace::AddController method.......... 6
2.3.2. CaoController::AddVariable method.......... 13
2.3.3. CaoController::Execute method.......... 13
2.3.4. CaoController::OnMessage event.......... 13
2.4. Variable list.......... 14
2.4.1. Controller class.......... 14
2.5. Error code.......... 14
3. Sample program.......... 15
4. Command reference.......... 17
4.1. Command list.......... 17
4.2. The command is detailed.......... 17

1. Introduction

This book is a user's guide of the Stream provider that is the CAO provider for the stream communication.

The Stream provider sends and receives data by the stream communication for a remote machine. You can select RS-232C communication and Ethernet's Socket streaming communication (TCP/IP or UDP) as the communication method. Also, during Socket streaming communication, you can select the server or client mode and operate it individually.

This book explains the function of this Stream provider and the mounting method.

2. Outline of provider

2.1. Outline

Stream providers are providers that perform RS-232C communication or Ethernet of Socket streaming communication (TCP/IP or UDP).

The Stream provider merely wraps the communication with the interface of CAO, and doesn't process the data that flows to the stream. As a result, this provider can be used for the device without the protocol (teletype protocol) as it is.

In addition, communication via Socket stream communication can be selected by specifying AddController options to operate in client mode or server mode.

The file format of the Stream provider is DLL(Dynamic Link Library), and the details become like Table 2-1.

Table 2-1 Stream provider

File name	CaoProvStream.dll
ProgID	CaoProv.DNWA.STREAM
Registry registration[1]	regsvr32 CaoProvStream.dll
Blotting out of registry registration	regsvr32 /u CaoProvStream.dll

2.2. Operational mode

There are two (the synchronous mode and the asynchronous mode) in the Stream provider as a communicate mode. This can be switched by specifying the Sync option of AddController.

2.2.1. Asynchronous mode

The OnMessage event is generated in the asynchronous mode when there is a reception from the other party of the communication.

When the device is a client of COM or Ethernet, the transmission of data transmits data by the “Send" command of CaoController::Execute(). The response is transmitted for the server of Ethernet with CaoMessage::Reply() of the CaoMessage object acquired in the OnMessage event when data is received.

2.2.2. Synchronous mode

In the synchronous mode, data is sent and received by using CaoController::Execute().

The transmission of data receives “Send" command and data by the “Receive" command.

UDP server mode cannot be used for the communication device in the synchronous mode. Moreover, please adjust the number of maximum, connected clients to one in the TCP server mode.

[1] It is not necessary registration/to blot out by hand power when installing it with ORiN SDK.

2.3. Method property

2.3.1. CaoWorkspace::AddController method

The communication is connected in the Stream provider referring to connected parameter for the communication at AddController.

At this time, the setting of a communication form, connected parameter, and the time-out is specified by the option.

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

bstrCtrlName	: [in] Controller name
bstrProvName	: [in] Provider name. Fixed value =” CaoProv.DNWA.STREAM”.
bstrPcName	: [in] Execution machine name of provider
bstrOption	: [in] Option character string

The list specified for the option character string is shown as follows.

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

Option	Explanation	Effective device		
		Ethernet		com
		Server	Client	
Conn =<Connected parameter>	Indispensability. Communication form and connected parameter. (Reference 2.3.1.1)	∨	∨	∨
PacketOpt [=<Packet parameter>]	The communication packet is set. (default: "12:0:0") (Reference 2.3.1.2)	∨	∨	∨
EtherOpt [=<Ether parameter>]	It is set to communicate Ethernet. (default: “0:5") (Reference 2.3.1.5)	∨	∨	-
MyIP [=<Local IP address>]	NIC can be selected by specifying Internet Protocol address by this option when two or more NIC is used. It is automatically selected when omitting it. When Internet Protocol address not allocated in a local machine is specified, the error is returned.	∨	∨	-

	This parameter is used as the server's IP address when the server mode is specified in the "EtherOpt" optional Mode.			
MyPort [=<Local port number>]	A port number on the local side to use in communication is appointed. When appointed port has been already used, an error is given back.	-	∨	-
Timeout [=<Timeout>]	Timeout period when sending and receiving it. (default: 500)	∨	∨	∨
LockMsg [=TRUE / FALSE]	The reception is notified with the same message object. The message is notified by the emergency message when receiving it.	∨	∨	∨
Sync [=TRUE / FALSE]	The synchronous mode is set. When the synchronous mode is specified, the UDP server cannot be specified. Moreover, when the TCP server is specified, it is not possible to specify it for a number of maximum clients excluding one.	∨	∨	∨
CancelError [=TRUE / FALSE]	It is specified whether the error is generated when the command is canceled. The error code when canceling is as follows. TRUE : 0x80100002 FALSE : 0x00000001	∨	∨	∨
CancelClearOnReceive [=TRUE / FALSE]	Enable / disable automatic cancellation of cancel status when Receive command is executed. If enabled, the ProviderClear command is executed when the Receive command is executed, and the cancel state is released.	∨	∨	∨
Multicast [=TRUE / FALSE]	Configures the reception of UDP multicast addresses.	*1		

*1 : Valid only in UDP server mode

2.3.1.1. Conn option

Connected parameter character string of the Conn option is shown as follows. A possible omission is shown here in the square bracket (""). Moreover, the underlined part under the explanation of each parameter becomes a default value when the option is not specified.

・ **RS-232C device**

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

<COM Port>	:	COM port number. '1'-COM1,'2'-COM2,…
<BaudRate>	:	Transmission rate. 4800,9600,19200,38400,57600,115200.
<Parity>	:	Parity. 'N'-NONE,'E'-EVEN,'O'-ODD.
<DataBits>	:	Number of data bits. '7'-7bit,'8'-8bit.
<StopBits>	:	Number of stop bits. '1'-1bit,'2'-2bit.
<Flow>	:	Flow control. '1' -Xon/Xoff.. '2'-hardware control. It is possible to specify it by taking OR. (default: Flow control '0'-none)

・ **EtherNet device**

"Conn=eth:<IP Address>[:<Port No>]"

<IP Address>	:	IP address . If the server mode is specified in Mode of the "EtherOpt" option, the value of this parameter is ignored unless you enable Multicast option in UDP server mode. Example:"127.0.0.1","192.168.0.1"
<Port No>	:	TCP/UDP connection port number. 5006,5007… It is possible to specify it voluntarily. This parameter is used as the port number of the server when the server mode is specified in the "EtherOpt" optional Mode.

2.3.1.2. PacketOpt option

The parameter character string of the PacketOpt option is shown as follows. A possible omission is shown here in the square bracket (""). Moreover, the underlined part under the explanation of each parameter shows the default value when the option is not specified.

"PacketOpt =[<Mode>[:<Header>[:<Term>]]]"

<Mode>	:	Communication data conversion. The first bit: ISO conversion The second bit: EIA conversion The third bit: Unicode conversion

		The fourth bit: Text mode
		The fifth bit: RoboTalk mode
		The sixth bit: B-CAP mode
		The value that can be input is specified with the following bit flags.
		(example: For the Unicode conversion and the text mode: ‘12'.)
<Header>	:	Header specification.
		This parameter is used when text mode is specified.
		‘0' - none and ‘1' - ENQ(0x05)
<Term>	:	Terminator specification.
		This parameter is used when text mode is specified.
		‘0'-CR(0x0D),‘1'-LF(0x0A),‘2'-CR+LF(0x0D0A)

2.3.1.3. Communication data conversion

The communication data conversion is converted by the method of specifying the data when sending and receiving.

Because each data conversion is a bit flag, two or more kinds can be specified. However, there is the following restrictions in two or more specification.

- The ISO conversion and the EIA conversion cannot be specified at the same time.
- You must specify the text mode when specifying ISO or EIA conversion.
- When the Unicode conversion is specified, it is necessary to specify the text mode.
- The text mode, the RoboTalk mode, and the b-CAP mode cannot be specified at the same time.

The value of the mode that can be specified and the list of the correspondence mode are indicated in Table 2-3 as follows.

Table 2-3 Mode option value and correspondence mode table

Mode / Value	ISO conversion	EIA conversion	Unicode Conversion	Text Mode	RoboTalk Mode	b-CAP Mode
0	-	-	-	-	-	-
8	-	-	-	∨	-	-
9	∨	-	-	∨	-	-
10	-	∨	-	∨	-	-
12	-	-	∨	∨	-	-
13	∨	-	∨	∨	-	-
14	-	∨	∨	∨	-	-
16	-	-	-	-	∨	-
32	-	-	-	-	-	∨

2.3.1.4. Multicast Option

Specifies whether to receive UDP multicast packets.

Specify TRUE when receiving UDP-based multicast packets.

If Multicast option is not specified, FALSE is specified.

If you did not set Multicast option-value, you would have specified a TRUE.

If Multicast option is set to TRUE, multicast addresses addressed to the address specified in CONN option IP address are received.

The explanation of each mode is brought together as follows.

1. ISO code and EIA code conversion

   It specifies it by the second one bit of < Mode > of connected parameter.

   It converts into the code in which data is specified from ASCII code and it sends and receives it. The ISO code conversion at this time and the EIA code conversion cannot be specified at the same time.

   When using it together with the UNICODE transducer function, data is converted into the code specified from UNICODE and sent and received.

Table 2-4 Data conversion when sending and receiving

	ISO code conversion	EIA code conversion
Send	ISO code -> S-JIS	EIA code -> S-JIS
Receive	S-JIS -> ISO code	EIA code -> S-JIS

2. Unicode conversion

   When the third bit of < Mode > of connected parameter is TRUE(1), following Unicode is converted.

   When the Unicode conversion is specified, it is necessary to specify the text mode.

Table 2-5 Data conversion when sending and receiving

Send	Unicode → S-JIS
Receive	S-JIS → Unicode

3. Text mode

   When the fourth bit of < Mode > of connected parameter is TRUE(1), it operates as a text mode.

   < Header > and < Term > specified for the transmitted and received data by connected parameter are added, and inspected.

Table 2-6 Data conversion when sending and receiving

Send	<Data> → <Header><Data><Term>
Receive	<Header><Data><Term> → <Data>

4. RoboTalk mode

   When the fifth bit of < Mode > of connected parameter is TRUE(1), it operates as RoboTalk mode.

   < ENQ >, < data length >, and < CD > of RoboTalk are added, and inspected for the transmitted and

received data.

Table 2-7 Data conversion when sending and receiving

Send	<Data> → <ENQ><Len><Data><CD>
Receive	<ENQ><Len><Data><CD> → <Data>

As for the above-mentioned < data >, the command, ID, and the parameter of RoboTalk are included. Please generate sending and receiving, and go by the client about the analysis.

5. B-CAP mode

When the sixth bit of < Mode > of connected parameter is TRUE(1), it operates as b-CAP mode.

< header >, < data length >, and < terminator > of b-CAP are added, and inspected for the transmitted and received data.

Table 2-8 Data conversion when sending and receiving

Send	<Data> → <header><data length><terminator>
Receive	<header><data length><terminator> → <Data>

As for the above-mentioned < data >, < argument part > is included from < serial number > of b-CAP. Please generate sending and receiving, and go by the client about the analysis.

2.3.1.5. EtherOpt option

The parameter character string of the EtherOpt option is shown as follows. A possible omission is shown here in the square bracket (""). Moreover, the underlined part under the explanation of each parameter shows the default value when the option is not specified. Moreover, when RS-232C is specified for the device, this option is disregarded.

“EtherOpt =[<Mode>[:<ConnMax>]]"

<Mode>	:	Character string conversion. ‘0' . -TCP client mode ‘1'-TCP server - mode ‘2' . -UDP client mode ‘3'-UDP server - mode When the device specified by the Conn option is "Com", this option is

disregarded.

<ConnMax>	:	Number of maximum clients at TCP server mode. (default: 5) This value is ignored when not in TCP server mode.

2.3.2. CaoController::AddVariable method

The variable object is made. Only the variable of 2.4.1 can be used for the variable identifier.

Format AddVariable(<bstrName:BSTR > [,<bstrOption:BSTR>])

bstrName	: [in] Variable name
bstrOption	: [in] Option character string

2.3.3. CaoController::Execute method

The command of the controller class is executed.

Please refer to 4 for details of each command.

2.3.4. CaoController::OnMessage event

When receiving the data at the asynchronous mode, the OnMessage event of CAO is generated.

When the text mode, the RoboTalk mode, and the b-CAP mode are specified, the OnMessage event is generated in each packet of each mode. Whenever receive data is confirmed, the OnMessage event is generated besides.

Receive data is stored in the Message::Value property.

The response data replies to the transmission origin of receive data by using the Message::Reply() method. This method can be used only at the TCP server and the UDP server.

The data type of the reception and the reply is different depending on the mode of the communication data conversion. VT_UI1, except for VT_BSTR and text mode in case of text mode| Data is acquired with VT_ARRAY.

2.4. Variable list

2.4.1. Controller class

Table 2-9 Controller class system variable list

Variable name	Data type	Explanation	Attribute	
			get	put
@TIMEOUT	VT_I4	Sending and receiving time-out time	∨	∨
@SYNCBUFCOUNT	VT_I4	Data size that exists in receive buffer This variable can be specified only at the synchronous mode.	∨	-
@VERSION	VT_BSTR	Version information	∨	-
@CONNECTCOUNT	VT_I4	Present connected number This variable always returns -1, except for the TCP server mode.	∨	-

2.5. Error code

In the Stream provider, the following and peculiar the error code is defined. Please refer to the chapter of the error code of "ORiN2 programming guide" for the ORiN2 commonness error.

Table 2-10 Original error code list

Error name	Error number	Explanation
E_SYNC_ONLY	0x80100001	Only in the synchronous mode, it is possible to execute it.

3. Sample program

The sample communicated with “Server.exe" that starts in the server mode of the TCP/IP communication between “Client.exe" that starts in the client mode is shown as follows.

Server IP address: 10.7.9.250

List 3-1 **SampleServer.frm**

```
Private eng As CaoEngine
Private WithEvents ctrl As CaoController

Private Sub Form_Load()

    Set eng = New CaoEngine

    ' The communication begins in the server mode.
    Set ctrl = eng.Workspaces(0).AddController("Sample", _
                                                "CaoProv.DNWA.STREAM", _
                                                "", _
                                                "Conn=eth:127.0.0.1:5006,EtherOpt=1")

End Sub

' Receive event
Private Sub ctrl_OnMessage(ppCaoMess As CAOLib.ICaoMessage)

    ' The received content replies as it is.
    ppCaoMess.Reply ppCaoMess.Value

End Sub
```

List 3-2 SampleClient.frm

```
Private eng As CaoEngine
Private WithEvents ctrl As CaoController

Private Sub Form_Load()

    Set eng = New CaoEngine

    ' The communication begins in the client mode.
    Set ctrl = eng.Workspaces(0).AddController("Sample", _
                                               "CaoProv.DNWA.STREAM", _
                                               "", _
                                               "Conn=eth:10.7.9.250:5006")
End Sub

Private Sub Command1_Click()

    ' Send processing
    ctrl.Execute text1.Text

End Sub

' Receive event
Private Sub ctrl_OnMessage(ppCaoMess As CAOLib.ICaoMessage)

    ' Data reception
    text2.Text = ppCaoMess.Value

End Sub
```

4. Command reference

4.1. Command list

Table 4-1 Command list

Category	Command	Function	
	Send	Data transmission	P. 17
	Receive	Data reception	P. 18
	ProviderCancel	It sets it in the state of the cancellation.	P. 19
	ProviderClear	Canceled release	P. 19
	Clear	Clearness in receive buffer	P. 19

4.2. The command is detailed.

Send

Syntax *object*.Send <Data>

Argument <Data> = VT_BSTR (text mode)

VT_UI1 | VT_ARRAY (Excluding the text mode)

: Send data

Return value None

Explanation Data is transmitted.

This command is a combination of the communication device and synchronization and the asynchronization setting and has right or wrong of use. This combination is shown below.

Device		Synchronization	Asynchronization
COM		∨	∨
TCP	Server	∨	-
	Client	∨	∨
UDP	Server	-	-
	Client	-	∨

Receive

Syntax `object.Receive( <Size>, <BufKeep> )`

Argument

<Size> = VT_I4 : Receive data size

This parameter is not used at the text mode, the RoboTalk mode, and the b-CAP mode.

<BufKeep> = VT_BOOL : Buffer information maintenance flag

True	:	The read receive buffer is not deleted.
False	:	The read receive buffer is deleted. (default)

Return value

<Data> = VT_BSTR (text mode)

VT_UI1 | VT_ARRAY (Excluding the text mode)

: Receive data

Explanation

It receives the data.

Processing is not returned until the size specified by < Size > is received.

When the text mode, the RoboTalk mode, and the b-CAP mode are specified, the content of < Size > is disregarded. The reception size is done in each packet of each mode.

This command is a combination of the communication device and synchronization and the asynchronization setting and has right or wrong of use. This combination is shown below.

Device		Synchronization	Asynchronization
COM		∨	-
TCP	Server	∨	-
	Client	∨	-
UDP	Server	-	-
	Client	-	-

ProviderCancel

Syntax `object.ProviderCancel`

Argument None

Return value None

Explanation The provider is set in the state of the cancellation.

The execution of the sending and receiving in canceling is interrupted.

Please execute the “ProviderClear" command when you release the cancellation.

This command can be used only at the synchronous mode.

ProviderClear

Syntax `*object*.ProviderClear`

Argument None

Return value None

Explanation The cancellation of the provider is released.

This command can be used only at the synchronous mode.

Clear

Syntax `*object*.Clear`

Argument None

Return value None

Explanation The receive buffer for synchronization is cleared.

This command can be used only at the synchronous mode.