OMRON
ZN-CTX21 provider
User's Guide

Version 1.0.0

December 17, 2021

NOTE:

[Revision History]

Version	Date	Content
1.0.0	2021-12-17	First edition

[Compatible models]

Model	Version	Notes
ZN-CTX21		

[Operation check model]

Model	Version	Notes
ZN-CTX21		

Contents

1. Introduction

This user's guide is for providers who communicate with OMRON's ZN-CTX21 and collect data. Fig. 1-1 shows the overall configuration of this provider and the device. The providers are referred to as ZN-CTX21 providers.

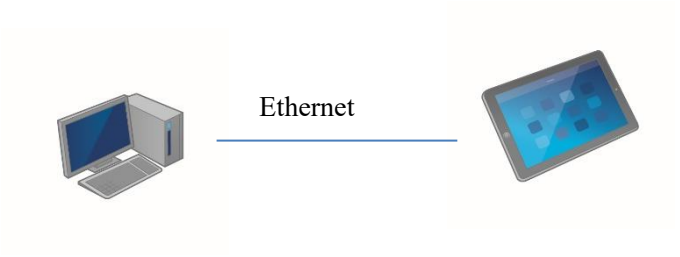

Fig. 1-1: Configuration Diagram

Fig. 1-2 shows the correspondence between this provider and each device.
(* This is an example. It does not represent everything.)

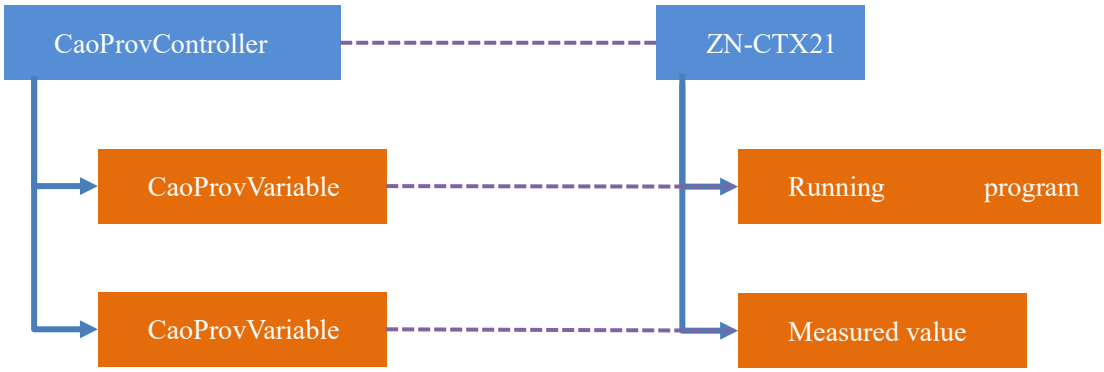

Fig. 1-2 Provider configuration and device information

2. Setting Up Your Environment for Application Development

2.1. Setting up a PC development environment

2.1.1. Installing ZN-CTX21 Providers Manually

If you install ZN-CTX21 providers manually, you must register the registry as shown below. To register the registry, start the command prompt with administrator privileges and execute regsvr32 command. When executing the command, either move to the path where the file is located or specify the file path.

Table 2-1: ZN-CTX21 Provider

File name	CaoProvOMRONZNCTX21.dll
ProgID	CaoProv.OMRON.ZN-CTX21
Registry registration	Regsvr32 CaoProvOMRONZNCTX21.dll
Deletion of Registry Registration	Regsvr32 /u CaoProvOMRONZNCTX21.dll

3. Command Reference

3.1. Method/Property List

Table 3-1: List of methods and properties

Category	Methods/Properties[1]		Function	Reference
CaoWorkspace				
	AddController	M	Connected to controller	P.7
CaoController				
	Index	P	Obtaining the Controller Number	P.9
	Name	P	Get Controller Name	P.9
	GetVariableNames	M	Get a list of variable names that can be connected	P.9
	Variables	P	Retrieving Variable Collections Holded by the Controller	P.9
	AddVariable	M	Adding Variable Objects	P.10
	Execute	M	Execute Extended Commands	P.10
CaoVariable				
	Index	P	Obtaining Variable Numbers	P.12
	Name	P	Retrieving Variable Names	P.12
	Value	P	Get/set value	P.12

3.2. Method properties

3.2.1. CaoWorkspace classes

3.2.1.1. AddController method

In CaoWorkspace, add a controller object. ZN-CTX21 providers connect to the appropriate ZN-CTX21 by referring to the parameters passed when AddController method is executed. The following are the specifics of AddController method:

Format

```
AddController
(
"<controller name>",                    // Controller name (optional)
"CaoProv.OMRON.ZN-CTX21",               // Provider name (fixed)
```

[1] M:Indicates methods, P: properties, and E: events, respectively.

```
"<machine name>",                    // Provider execution machine name (unused)
"<Option>"                           // Option string
)
```

Option

The following options are specified in the option string: The option string is a string consisting of the following options separated by a comma (,).

Option	Required	Description	Value Range	Default value
CONN	✓	Specify the IP address and port number of the connection destination. If the port number is omitted, the port number is 2323.		
TIMEOUT	-	Specify the connection timeout.	0~4294967295	500

Sample usage (C#)

```
// Engine
ORiN2.ManagedCAO.CCaoEngine engine = new ORiN2.ManagedCAO.CCaoEngine();
// Workspace
ORiN2.ManagedCAO.CCaoWorkspace workspace = engine.AddWorkspace("NewWrks", "");
// Controller
ORiN2.ManagedCAO.CCaoController controller= workspace.AddController(
    "ZN-CTX21",
    "CaoProv.OMRON.ZN-CTX21",
    "",
    "CONN=TCP:192.168.0.20");
```

3.2.1.1.1. CONN Optional

The following is a Conn optional connection parameter string: Here, braces ("[]") are optional, and the underlined part in the description of each parameter indicates the default value when no options are specified.

"CONN=TCP:<IP>[:<Port>]"

<IP> : Destination IP address.

<Port> : Destination port. (2323)

3.2.2. CaoController classes

3.2.2.1. Index Properties

Gets the controller number as a Long type (4-byte integer type). This number identifies the corresponding controller in the controller collection held by CaoWorkspace.

Sample usage (C#)

```
// Get Index
long index = controller.Index;
```

3.2.2.2. Name Properties

Retrieves the controller name specified by AddController method of CaoWorkspace class.

Sample usage (C#)

```
System.Diagnostics.Debug.WriteLine(controller.Name);
```

3.2.2.3. GetVariableNames method

Gets a list of variable names that can be connected.

No options are used.

Format

```
GetVariableNames(
"<Option>"                  // Option string (unused)
)
```

Sample usage (C#)

```
// Get variable name list
string[] variableNames = controller.GetVariableNames("");
```

3.2.2.4. Variables Properties

Gets a collection of variables that the controller holds.

Sample usage (C#)

```
// Variable Collection Retrieval
ORiN2.ManagedCAO.CCaoVariables variables = controller.Variables;
// Variable acquisition
ORiN2.ManagedCAO.CCaoVariable variable = variables[0];
```

3.2.2.5. AddVariable method

Adds a variable object to CaoController. Only the variable names shown in 3.3.2 can be used.

AddVariable is specified as follows.

Format

```
AddVariable
(
"<variable name>",      // Variable Name
"<Option>"              // Option string (unused)
)
```

3.2.2.6. Execute method

Execute CaoController extension. Execute is specified as follows.

Format

```
Execute
(
"<extension command name>",     // Extended command name
"<Option string>"               // Option string (optional)
)
```

The following is a list of extended commands that can be specified in Execute. The usage examples are described in detail in the extended commands.

Command	Description	Reference
ProviderCancel	Cancels the provider operation.	P.10
ProviderClear	Clears ProviderCancel operation.	P.11
StartRec	Starts recording the measured value.	P.11
StopRec	Stops recording the measured value and writes the measurement result to the SD card.	P.11
ClearError	Clears the error that has occurred.	P.11
ClearAlarm	Cancels the alarm that has occurred.	P.11
GetVersion	Gets the device version.	P.12

3.2.2.6.1. ProviderCancel Commands

Cancels the provider operation.

If you cancel the operation of the provider, the connection with ZN-CTX21 is closed.

At this time, all commands waiting for a reply will fail and return.

3.2.2.6.2. ProviderCancelClear Commands

Clears ProviderCancel operation.

ZN-CTX21 providers implement the commands, but do nothing in particular when invoked.

3.2.2.6.3. StartRec Commands

ZN-CTX21 starts recording the readings.

There are no arguments or return values.

Sample usage (C#)

```
// Execute StartRec
Controller.Execute("StartRec", null);
```

3.2.2.6.4. StopRec Commands

Stop recording the measured values with ZN-CTX21 and export the measured values to an SD-Card.

There are no arguments or return values.

StopRec command will not return until the write to SD card is complete, so the default timeout (500ms) may cause the command to time out without completing the process.

Oracle recommends that you set a longer timeout when using StopRec commandI will.

Sample usage (C#)

```
// Execute StartRec
Controller.Execute("StopRec", null);
```

3.2.2.6.5. ClearError Commands

Clears the errors that have occurred in ZN-CTX21.

There are no arguments or return values.

Sample usage (C#)

```
// Execute StartRec
Controller.Execute("ClearError", null);
```

3.2.2.6.6. ClearAlarm Commands

Clears the alarms that are occurring on ZN-CTX21.

There are no arguments or return values.

Sample usage (C#)

```
// Execute StartRec
Controller.Execute("ClearAlarm", null);
```

3.2.2.6.7. GetVersion Commands

Gets the version of ZN-CTX21 device.

Values that can be obtained by GetVersion are VT_VARIANT| VT_ARRAY, each element has the following meaning:

Table 3-2: GetVersion Element List

Index	Type	Description
0	VT_UI1	Major version
1	VT_UI1	Minor version
2	VT_UI1	Release number
3	VT_BSTR	Serial number
4	VT_UI1	Model ID

3.2.3. CaoVariable classes

3.2.3.1. Index Properties

Gets the variable number as a Long type (4-byte integer type). This number indicates the number that identifies the variable in the variable collection held by CaoController class.

Sample usage (C#)

```
// Get Index
int index = caoVar.Index;
```

3.2.3.2. Name Properties

Retrieves the variable name specified by AddVariable method of CaoContrller class.

Sample usage (C#)

```
System.Diagnostics.Debug.WriteLine(caoVar.Name);
```

3.2.3.3. Value Properties

Retrieves data from the connected ZN-CTX21. The behavior depends on the variable name. For details, refer to section 3.3.

3.3. Variable list

Defines a list of variables that can be used in each class. Variables refer to objects of CaoVariable classes.

3.3.1. System and User Variables

Only system variables are available for ZN-CTX21 providers.

System Variables

A variable that accesses only the information in the object that holds the variable. System variables are often static data. System variables are preceded by "@".

e.g., provider version, device manufacturer, current time of day

3.3.2. CaoController class-variable

Table 3-3: CaoController Class-Variable List

Variable Name	Description	Value Type
@MAKER_NAME	Manufacturer Name	VT_BSTR
@VERSION	Provider version information	VT_BSTR
@ERROR_CODE	Error code	VT_UI2
@STATUS	Operating status	VT_UI1
@RECORD_DATA_COUNT	Number of recorded data	VT_UI2
@SD_INSERT_STATUS	SD insert condition	VT_UI1
@ALARM_OUTPUT_STATUS	Alarm output status	VT_UI1
@CURRENT_STATUS	Status list (see Table 3-4 @CURRENT_STATUS Element List)	VT_VARIANT| VT_ARRAY
@INSTANTANOUS_POWER	Instantaneous power (kWh)	VT_R8
@ELECTRIC_ENERGY	Amount of power (kW)	VT_R8
@INSTANTANOUS_CURRENT_CH1	Instantaneous carrying current CH1(A)	VT_R8(※1)
@INSTANTANOUS_CURRENT_CH2	Instantaneous carrying current CH2(A)	VT_R8(※1)
@INSTANTANOUS_CURRENT_CH3	Instantaneous carrying current CH3(A)	VT_R8(※1)
@MEASURED_TIME	Measurement date and time	VT_DATE
@CURRENT_DATA	Most recent readings (see Table 3-5 @CURRENT_DATA Element List)	VT_VARIANT| VT_ARRAY
@OPERATION_MODE	Operating mode (set value)	VT_UI4
@RECORD_MODE	Recording mode (set value)	VT_UI4

@RECORD_INTERVAL	Recording interval (set value)	VT_UI4
@INTEGRATED_ELECTRIC_ENERGY_RESET_INTERVAL	Cumulative amount of electric energy reset interval (setting value)	VT_UI4
@APPLICABLE_CIRCUIT	Applicable circuit (set value)	VT_UI4
@DEDICATED_CT_TYPE	Dedicated CT type (set value)	VT_UI4
@NUMBER_OF_CHANNELS_USED	Number of channels used (set value)	VT_UI4
@VOLTAGE_TO_BE_MEASURED	Measurement target voltage (set value)	VT_R8
@POWER_FACTOR	Power factor (set value)	VT_R8
@FREQUENCY	Frequency (set value)	VT_UI4
@RATED_PRIMARY_SIDE_CURRENT_VALUE	Rated primary current value (set value)	VT_UI4
@LOW_CUT_CURRENT	Low-cut current (set value)	VT_R8
@MEASUREMENT_RANGE	Measurement range (set value)	VT_UI4
@PRICE_PER_CO2_CONVERSION_VALUE_SETTING	Cost/CO2 conversion setting (set value)	VT_R8
@CONVERSION_VALUE_UNIT_SETTING	Conversion unit setting (set value)	VT_UI4
@INTEGRATED_ELECTRIC_ENERGY_THRESHOLD	Threshold value (cumulative amount of electric energy) (setting value)	VT_UI4
@START_TRIGGER	Start trigger (set value)	VT_UI4
@START_TIME	Start time (set value)	VT_BSTR
@END_TRIGGER	End trigger (set value)	VT_UI4
@END_TIME	End time (set value)	VT_BSTR
@ELAPSED_TIME	Elapsed time (set value)	VT_UI4
@REC_RETURN_FUNCTION_AT_STARTUP	REC recovery function at startup (set value)	VT_UI4
@PRESENCE_OR_ABSENCE_OF_NETWORK_FUNCTION	Network function (set value)	VT_UI4
@IP_ADDRESS	IP address (set value)	VT_BSTR
@SUBNET_MASK	Subnet mask (set value)	VT_BSTR
@DEFAULT_GATEWAY	Default gateway (set value)	VT_BSTR

@PORT_NUMBER	Port number (set value)	VT_UI4

※1: If the channel does not exist, the value is VT_EMPTY.

3.3.2.1. Configuring @CURRENT_STATUS Values

The @CURRENT_STATUS is used to acquire the status of ZN-CTX21 in a single communication.

The value that can be obtained with @CURRENT_STATUS is VT_VARIANT| VT_ARRAY, each element has the following meaning:

Table 3-4: @CURRENT_STATUS Element List

Index	Type	Description
0	VT_UI2	Error code
1	VT_UI1	Operating status
2	VT_UI2	Number of recorded data
3	VT_UI1	SD insert condition
4	VT_UI1	Alarm output status

3.3.2.2. Configuring @CURRENT_DATA Values

The @CURRENT_STATUS is used to acquire the status of ZN-CTX21 in a single communication.

The value that can be obtained with @CURRENT_STATUS is VT_VARIANT| VT_ARRAY, each element has the following meaning:

Table 3-5: @CURRENT_DATA Element List

Index	Type	Description
0	VT_R8	Instantaneous power (kWh)
1	VT_R8	Amount of power (kW)
2	VT_R8	Instantaneous carrying current CH1(A)(※2)
3	VT_R8	Instantaneous carrying currentCH2(A) (※2)
4	VT_R8	Instantaneous carrying currentCH3(A) (※2)
5	VT_DATE	Measurement date and time

※2: If the channel does not exist, the value is VT_EMPTY.

4. Programming by ZN-CTX21 provider

With ZN-CTX221 providers, you can connect ZN-CTX21 to the client computer as follows:

- Creating a CaoEngine
- Creating a CaoWorkspace
- Creating a CaoController

After connecting to ZN-CTX21, CaoVariabe can be accessed by creating a new.

4.1. Sample Programming to Collect Values for Variables in ZN-CTX21 Provider

This example shows a sample program that collects the values of the variables in ZN-CTX21 providers.

Sample **Program.cs**

```
using System;
using System.Linq;
using ORiN2.ManagedCAO;

namespace ZNCTX21Sample
{
    class Program
    {
        ///<summary>
        /// Converts an object to a display string
        ///</summary>
        ///<param name = "obj">Object to be displayed </ param>
        ///<returns> character string for displaying </>
        static string Dump(object obj)
        {
            if (obj is Array arr)
            {
                return $"[{string.Join(",", arr.Cast<object>().Select(x => Dump(x)))}]";
            }
            else
            {
                return Convert.ToString(obj);
            }
        }
```

```
static void Main(string[] args)
{
    try
    {
        // Create a CaoEngine
        // Releasing engine releases all objects under it
        // Exiting the part enclosed by using clause, engine is automatically released.
        using (CCaoEngine engine = new CCaoEngine())
        {
            // Add a controller
            var ctrl = engine.Workspaces[0].AddController(
                "ZNCTX21",
                "CaoProv.OMRON.ZN-CTX21",
                null,
                "CONN=TCP:192.168.0.20");

            // Gets a list of defined variable names
            string[] varNames = ctrl.GetVariableNames(null);

            foreach (string varName in varNames)
            {
                // Add a variable
                var variable = ctrl.AddVariable(varName, null);
                // Get the value of a variable
                object value = variable.Value;
                // Prints the name and value of a variable on the screen
                Console.WriteLine("[{0}] Value: {1}", variable.Name, Dump(value));
            }
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex);
    }
    // Wait for the program to finish until the key is pressed.
    Console.ReadKey();
```

```
            }
        }
    }
```

5. ZN-CTX21 Provider Error Codes

This provider has the following unique error codes masked with the 0x8011****. (Refer to Table 5-1 Unique Error Codes)

For information about common ORiN2 errors, see the Error Codes section in ORiN2 Programming Guide.

Table 5-1: Unique Error Codes

Error Number	Description
0x80110001	Response format error
0x80111001	Parameter number out of range
0x80111002	Setting value out of range
0x80111003	Setting value read error
0x80111004	Setting value write error
0x80111005	Time out of range
0x80111006	Time acquisition error
0x80111007	Time setting error
0x8011100A	Mode that cannot be executed
0x8011100B	Error release failure
0x80111010	Sensor error
0x80111011	Hardware error
0x80111020	SD card not inserted
0x80111021	SD Card Write Error
0x80111022	SD card read error
0x80111023	SD card access disabled battery voltage drop
0x80111024	SD card write protect
0x80111025	SD card access in progress
0x80111026	SD card recognition error
0x80111083	Input value out of range
0x80112002	Format error
0x80112003	CRC error
0x80112004	Data length error
0x80112002	Undefined command