# DENSO
# RFID Table scanner UR20 provider

**Version 1.1.0**

User’s Guide

**July 11, 2024**

Remark

## [Revision history]

| Version | Dating | Content |
|---|---|---|
| 1.0.0 | 2018-11-27 | First edition |
| 1.1.0 | 2018-12-25 | Add Model UR20,UR21,UR22<br>Add COM communication<br>Add Kill commands |
|  | 2020-2-19 | Modified type description of “StartContinuousRFDetection” Command |
|  | 2022-1-07 | Typographical correction |
|  | 2024-7-11 | Added the fact that the available commands for the carrier output strength are different between the UR series for Japanese specifications and the UR series for overseas specifications. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## [Operation check model]

| Model | Version | Note |
|---|---|---|
| UR20 | OS: 01.01.02.00<br>Main AP: 00.01.03.00<br>RF AP: 00.01.07.00<br>RF Firm: 0F.0F.06.F0<br>OEM: 01.02.06.00 | If there is no command operation to the terminal for a certain period of time (60 seconds), the terminal automatically closes the socket by judging it as a line disconnection. Therefore, reconnect the socket when the socket is closed. Or, send a terminal communication confirmation command (P.24) periodically to maintain the line.27 |
| UR21 | OS: 01.01.02.00<br>Main AP: 00.01.03.00<br>RF AP: 00.01.07.00<br>RF Firm: 0F.0F.06.F0<br>OEM: 01.02.06.00 |  |
| UR22 | OS: 01.02.01.00<br>Main AP: 00.02.01.00<br>RF AP: 00.02.03.00<br>RF Firm: 2D.01.00.00<br>OEM: 01.06.06.00 |  |

## Table of contents

# 1. Introduction

This is a user's guide for provider who obtain RFID tags from UR20 series of RFID Table scanners. Figure 1-1 Block diagram shows the overall configuration of this provider and device. Hereinafter, this provider will be referred to as the UR20 provider, and the RFID Table scanners of the UR20 series will be referred to as scanners. The UR20 provider performs data communication with the scanner via TCP / IP or USB serial.

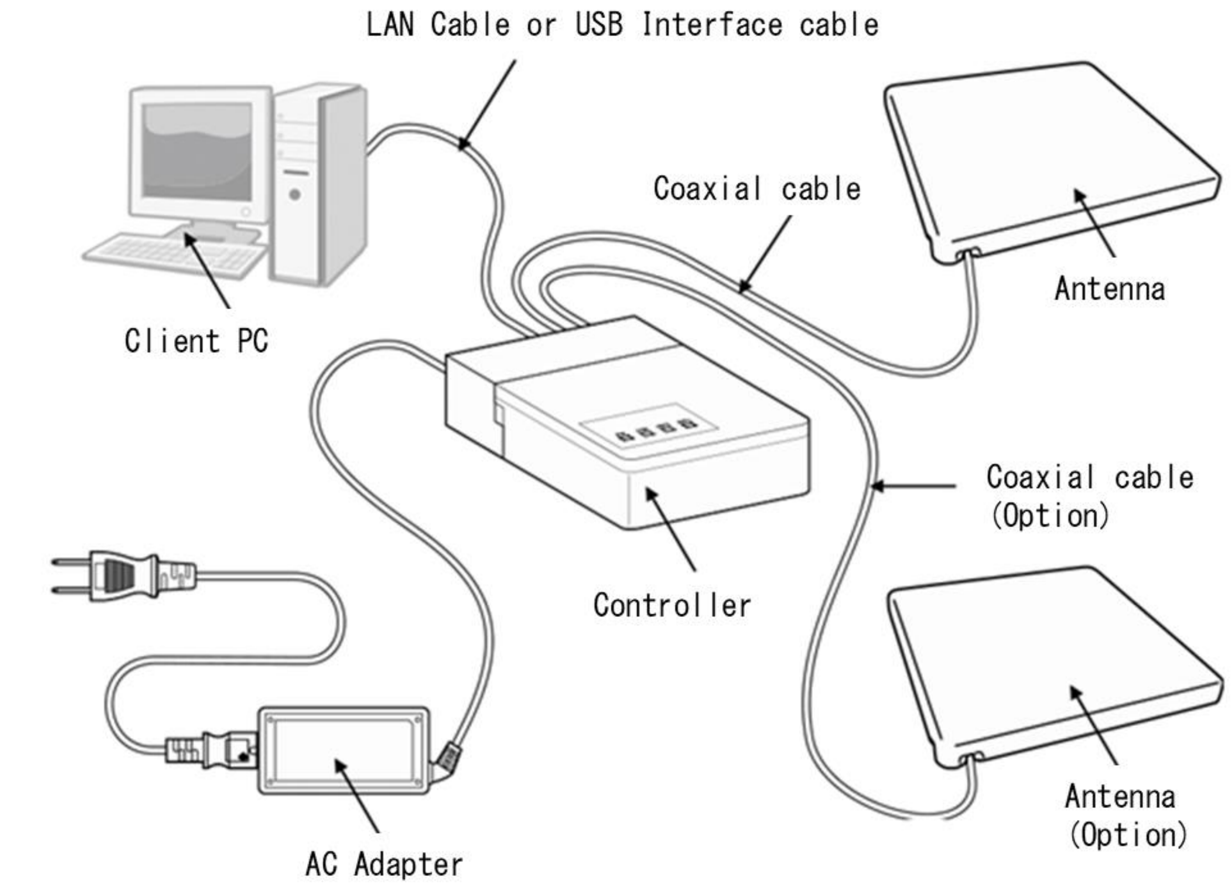

Figure 1-1 Block diagram

## 1.1. Environment and Version Assumed by this Manual

It is assumed that the client PC runs on the Windows and the target CNC is a company-named device type that can be connected to Ethernet. The PC development environment can be developed in a programming environment that supports Component Object Model (COM, Component Object Model).

## 1.2. Reference Sources

All of the programming examples in this book are described in Visual Basic for Applications, but they can be developed in a variety of programming languages, including C++, Java, and.NET. See the ORiN2 Programming Guide for instructions on how to use.

The ORiN 2 Programming Guide corresponds to the following files in the ORiN 2 SDK Installation folder:

- ORiN2¥CAO¥Doc¥ORiN2_ProgrammersGuide_<lang>.pdf

  ※ Replace the <lang> section with the language-specific character strings for each environment.

Examples of ORiN2, COM, and DCOM basic knowledge/technologies required to develop applications using provider is provided.

# 2. Environment setup for application development

## 2.1. Connection between the scanner and the client PC

For information on connecting scanners to client PCs, refer to the instruction manual for your UR20.

## 2.2. Setting up a PC development environment

### 2.2.1. Auto-install UR20 provider

If the ORiN2 SDK Ver is installed, the operating environment (line time) is ready to connect to scanners.

Set up a separate programming environment that supports Component Object Model (COM, Component Object Model) such as Microsoft Visual Studio 6. 0, 2003/2005/2008/2010, and LabVIEW.

### 2.2.2. Manually installing UR20 provider

To use UR20 provider, you must manually register:To register the registry, start the command prompt with the administrator privilege and execute the regsvr32 command.

Also, for CAO engines to operate, one legitimate ORiN2 SDK license must be pre-registered for each PC. See the Add/Remove Licenses section in the ORiN2 SDK User's Guide.

Table 2-1 Manual Installating UR20 provider

| | |
|---|---|
| File name | CaoProvDENSOUR20.dll |
| ProgID | CaoProv.DENSO.UR20 |
| Registry registration | Regsvr32 CaoProvDENSOUR20.dll |
| Deletion of Registry Registration | Regsvr32 /u CaoProvDENSOUR20.dll |

# 3. Programming by UR20 provider

UR20 provider can connect clients and scanners as follows.

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

After you connect to the scanner, you can use the CaoController's Execute method to access the scanner's own information and the information on the RFID tags that the scanner has read.

## 3.1. Sample Programming for Acquiring RFID Tags

This example shows a sample program that captures RFID tags read by scanners. Table 3-1 Sample Program Requirements shows the requirements for the sample program, and Figure 3-1 Sample Program Flow shows the flow of the sample program.

Table 3-1 Sample Program Requirements

| Requirement | Description |
|---|---|
| Destination | Connect by TCP/IP |
| | The destination IP address is 192.168.0.2. |
| | The destination port number is 20000. |
| | UHFTAGRW APs |
| Details of processing | RFID tags are acquired by ReadUII commands. |

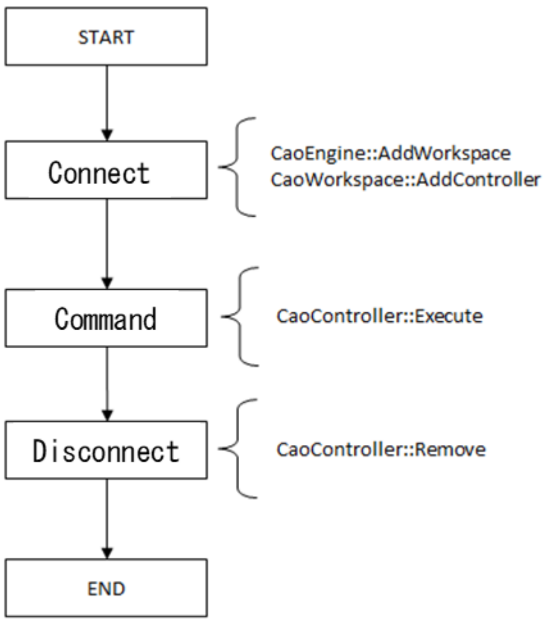

Figure 3-1 Sample Program Flow

Specific codes are shown in the following sections.

### 3.1.1. Sample program

The overall picture of the sample program is shown below.

| Sample | ReadUIISample.vb |
|---|---|

```
Sub Main
    ' Object
    Dim engine As CaoEngine
    Dim workspace As CaoWorkspace
    Dim controller As CaoController

    ' Generating CaoEngine Objects
    Set engine = New CaoEngine
    ' Generating CaoWorkspace Objects
    Set workspace = engine.AddWorkspace("NewWrks", "")
    ' Generating CaoController Objects
    Set controller = workspace.AddController("SampleController", _
                                             "CaoProv.DENSO.UR20", _
                                             "", _
                                             "conn=eth:192.168.0.2")

    ' RFID data acquisition
    Dim uiis As Variant
    Uiis = controller.Execute("ReadUII", False)
    If Not IsEmpty(uiis) Then
        ' Obtain the number of detected items.
        Dim detectedElem As Long
        DetectedElem = UBound(uiis)

        ' Repeat for the number of detected RFID tags
        Dim i As Long
        For i = 0 To detectedElem
            ' Obtain PC data
            Dim pcData As Long
            PcData = uiis(i)(0)

            ' Obtain UII data
            Dim uiiData() As Byte
            UiiData = uiis(i)(1)
            ' Gets UII data size
            Dim uiiDataSize As Long
            UiiDataSize = UBound(uiiData)
            ' Data extraction for UII data size
            Dim j As Long
            For j = 0 To uiiDataSize
                Dim value As Byte
                Value = uiiData(0)
            Next j
        Next i
    End If

    ' Remove CaoController from CaoWorkspace
    Call workspace.Controllers.Remove(controller.Index)
    ' Deleting the CaoController
    Set controller = Nothing

    ' Remove CaoWorkspace from CaoEngine
    Call engine.Workspaces.Remove(workspace.Index)
    ' Deleting the CaoWorkspace
    Set workspace = Nothing

    ' Deleting the CaoEngine
    Set engine = Nothing
End Sub
```

#### 3.1.1.1. Connection

To connect to the scanner:

(1) Prepare variables to hold objects. The objects required for controller-connectivity are the CaoEngine object, the CaoWorkspace object, and the CaoController object. The CaoWorkpace object does not need to have variables to retrieve the CaoController object from the CaoWorkspaces. You also need CaoVariable objects for accessing variables. The following is an example of code for VB6.

```
Dim engine As CaoEngine               ' Variables for CaoEngine objects
Dim workspace As CaoWorkspace         ' Variables for CaoWorkspace objects
Dim controller As CaoController       ' Variables for CaoController objects
```

(2) Create a CaoEngine object. Create a CaoEngine object using the new keyword.

```
' Generating CaoEngine Objects
Set engine = New CaoEngine
```

(3) Obtain or create a CaoWorkspace object. When you create a CaoEngine object, by default, you create a CaoWorkspaces object and a CaoWorkspace object one at a time. The following CaoWorkspace shows examples of code and defaults for creating new CaoWorkspace objects.

```
' Generating CaoWorkspace Objects
Set workspace = engine.AddWorkspace("NewWrks", "")
```

(4) Create a CaoController object. To create a CaoController object, set the names of the provider you want to use and the parameters you want to use. For UR20 provider, optionally specify connectivity. The code example is shown below.

```
' Generating CaoController Objects
Set controller = workspace.AddController("SampleController", _
                                         "CaoProv.DENSO.UR20", _
                                         "", _
                                         "conn=eth:192.168.0.2")
```

### 3.1.1.2. Obtaining RFID tags

Run the ReadUII command, an extended command for the CaoController, to retrieve the RFID tags that the scanners read. For more information, see ReadUII Commands on page 28.

### 3.1.1.3. Disconnection

To disconnect from the controller, delete the created object and delete the deleted object from the collection class that manages the object. The code example is shown below.

```
    ' Remove CaoController from CaoWorkspace
    Call workspace.Controllers.Remove(controller.Index)
    ' Deleting the CaoController
    Set controller = Nothing
    ' Remove CaoWorkspace from CaoEngine
    Call engine.Workspaces.Remove(workspace.Index)
    ' Deleting the CaoWorkspace
    Set workspace = Nothing
    ' Deleting the CaoEngine
    Set engine = Nothing
```

# 4. Command reference

## 4.1. Method/Property List

Table 4-1 Methods/Properties List

| Category | Method/Properties[1] | | Facility | Reference |
|---|---|---|---|---|
| CaoWorkspace | | | | |
| | Addcontroller | M | Connect to controller | P.13 |
| CaoController | | | | |
| | VariableNames | P | Get ConnecTable Variable Name List | P.15 |
| | Variables | P | Obtaining a Collection of Variables Held by a Controller | P.15 |
| | AddVariable | M | Adding Variable Objects | P.16 |
| | Execute | M | Execution of Extended Commands | P.16 |
| CaoVariable | | | | |
| | Value | P | Obtain/Set Values | P.38 |

## 4.2. Method Properties

### 4.2.1. CaoWorkspace classes

#### 4.2.1.1. AddController method

Add controller objects to the CaoWorkspace. The following shows the specifications of the AddController method.

Format

```
CaoController AddController
(
        Controller name          // Controller name (optional)
        "CaoProv.DENSO.UR20",            // Provider name (fixed)
        Machine name                     // Provider Execution Machine Name (unused)
        Option                   // Optional string (optional)
)
```

[1] M: Shows the method, P: property, and E: event, respectively.

Option

The following options are specified in the option string. The option string is a string of the following options connected by commas (,):

| Option | Required | Description | Values range | Default value |
|---|---|---|---|---|
| CONN =<Communication Parameters> | ○ | Specify destination information. | ETH or COM | -- |
| APName =<AP name> | -- | Specifies the AP name of the destination. Specify within 16 characters. If you specify an AP name of 16 characters or more, characters after 17 characters are ignored. | -- | UHFTAGRW |
| Timeout Communications timeout | -- | Specifies the communication timeout in ms. | 0 or more | 2000 |

Examples of uses

```
Dim engine As CaoEngine 'Engine objects
Dim workspace As CaoWorkspace 'WorkSpace objects
Dim controller As CaoController 'Controlle objects

Set engine = New CaoEngine
Set workspace = engine.Workspaces.Item(0)
Set controller = workspace.AddController("SampleController", _
                                        "CaoProv.DENSO.UR20", _
                                        "", _
                                        "conn=eth:192.168.0.2")
```

### 4.2.1.1.1. CONN options

The connection parameter strings for the Conn options are shown below. Here, the box in parentheses ("[]") can be omitted, and the underlined part in the description of each parameter indicates the default value when the option is not specified.

**To connect by TCP/IP**

"Conn = ETH:<destination IP>[:<destination port>[:<local IP>[:<local port>]]]]"

<destination IP> : The IP address of the connection destination is ****. ***. ***. Specify in **** format.

Be sure to specify this item.

Destination port : Specifies the destination port number. 50000

**RS232C**

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

| | | |
|---|---|---|
| <COM Port> | : | COM port number '1' -COM1, '2' - COM2, ... |
| <BaudRate> | : | Communication speed 4800, 9600, 19200, 38400, 57600, 115200 |
| <Parity> | : | Parity 'N'-NONE, 'E'-EVEN, 'O'-ODD |
| <DataBits> | : | Data bit count '7'-7bit, '8'-8bit |
| <StopBits> | : | Stop bit count '1'-1bit, '2'-2bit |
| <Flow> | : | Flow controll '0'-None, '1'-Xon/Xoff, '2'-Hardware Controll. It is able to specified by OR. |

### 4.2.2. CaoController classes

#### 4.2.2.1. VariableNames properties

Gets a list of variable names that can be connected. The names of the variables obtained in this property can be used as the first arguments of the AddVariable method, which will be described later.AddVariable method

Examples of uses

```
Dim engine As CaoEngine      'Engine objects
Dim workspace As CaoWorkspace 'WorkSpace objects
Dim controller As CaoController 'Controlle objects

Set engine = New CaoEngine
Set workspace = engine.Workspaces.Item(0)
Set controller = workspace.AddController("SampleController", _
                                          "CaoProv.DENSO.UR20", _
                                          "", _
                                          "conn=eth:192.168.0.2")

' Get File Name List
Dim variables as Variant
Variables = controller.VariableNames
```

#### 4.2.2.2. Variables properties

Retrieves the variable collection that the controller holds.

Examples of uses

```
Dim engine As CaoEngine 'Engine objects
Dim workspace As CaoWorkspace 'WorkSpace objects
Dim controller As CaoController 'Controlle objects

Set engine = New CaoEngine
Set workspace = engine.Workspaces.Item(0)
Set controller = workspace.AddController("SampleController", _
                                          "CaoProv.DENSO.UR20", _
                                          "", _
                                          "conn=eth:192.168.0.2")

' Variable collection acquisition
Dim variables as CaoVariables
Set variables = controller.Variables

' Variable acquisition
Dim variable as CaoVariable
Set variable = variables.Item(0)
```

#### 4.2.2.3. AddVariable method

Add variable objects to the CaoController. Only the variable name shown in 0 can be used.0
The AddVariable specifications are shown below.

Format

```
CaoVariable AddVariable
(
        Variable name           // Variable name
        Option          // Optional string (optional)
)
```

#### 4.2.2.4. Execute method

Executes the ConController extension commands. The Execute specifications are shown below.

Format

```
Variant Execute
(
        "<Extended Command Name>",              // Extended command name
        "<Option String>"               // Optional string (optional)
)
```

The list of extended commands that can be specified by Execute is shown below. Examples of use are described in the extension command detail.

| Commanded | Description | Reference |
|---|---|---|
| SetCarrierOutputIntensityLevel | Sets the carrier output level. The set value is initialized by disconnection/terminal power off. | P.19 |
| GetCarrierOutputIntensityLevel | Acquires the carrier output intensity levels. | P.19 |

| Commanded | Description | Reference |
|---|---|---|
| SetCarrierOutputIntensitydBm | Set the carrier output strength in dBm × 10 units. The set value is initialized by disconnection/terminal power off. | P.19 |
| GetCarrierOutputIntensitydBm | Acquires the carrier output intensity in units of dBm × 10. | P.20 |
| SetFrequency | Sets the frequency permitted for carrier output with the bit flag. The set value is initialized by disconnection/terminal power off. | P.20 |
| GetFrequency | Acquires the frequency permitted for carrier output using the bit flag. | P.20 |
| SetAntennaPort | Set the antenna port for outputting the carrier for RFID tag communication. The set value is initialized by disconnection/terminal power off. | P.21 |
| GetAntennaPort | Gets the antenna port that outputs the carrier for RFID tag communication. | P.22 |
| SetRereadPrevention | Enables/disables the read prevention function for UII read processing and tag communication processing. The set value is initialized by disconnection/terminal power off. | P.22 |
| GetRereadPrevention | Gets the validity/invalidity of the RFID tag discovery completion determination in UII read processing and tag communication processing. | P.23 |
| SetQValue | Sets the Q value to be used to adjust the probability that responses from tags will collide during UII acquisition. The set value is initialized by disconnection/terminal power off. | P.23 |
| GetQValue | Obtains the Q value that is used to adjust the probability that responses from tags collide during UII acquisition. | P.24 |
| SetSession | Configure the session to use when you acquire UII for the tag. The set value is initialized by disconnection/terminal power off. | P.24 |
| GetSession | Gets the session you want to use when you get the UII for the tag. | P.25 |
| ResetTerminal | Suspends processing. | P.25 |
| Reboot | Restart the RFID Table scanner.<br>After restarting, perform connection processing. | P.26 |
| GetSerialNumber | Gets the serial number of the RFID Table scanner. | P.26 |
| GetVersion | Gets the version information of the RFID Table scanner. | P.27 |
| KeepAlive | Sends terminal communication commands to RFID Table scanners. | P.27 |

| Commanded | Description | Reference |
|---|---|---|
| ReadUII | Detects RFID tags and reads UII data. | P.28 |
| StartContinuousRFDetection | Starts continuous UII reading. | P.29 |
| ReadContinuousUII | Reads UII data of RFID tags detected after StartContinuousRFDetection is executed. | P.30 |
| StopContinuousRFDetection | Stops continuous UII reading. | P.31 |
| ReadMemory | Reads the data in the specified memory of the RFID tag. | P.32 |
| WriteMemory | Writes the data in the specified memory of the RFID tag. | P.34 |
| Lock | Changes the lock status of the tag. | P.35 |
| Kill | Disable of the tag permanently. | P.36 |

#### 4.2.2.4.1. SetCarrierOutputIntensityLevel commands

Sets the carrier output intensity level. This command is only available for UR20 and UR21 for Japan. The set value is initialized with disconnection / terminal power off (Initial value 6).

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specify the carrier output intensity level.<br>Value range: 1 - 8 |
| Returned value | Without | |

Examples of uses

Refer to GetCarrierOutputIntensityLevel.

#### 4.2.2.4.2. GetCarrierOutputIntensityLevel commands

Acquires the carrier output intensity level. This command is only available for UR20 and UR21 for Japan.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Keep the carrier output intensity level. |

Examples of uses

```
    ' Get the carrier output intensity level.
    Dim carrierOutputIntensity As Long
    carrierOutputIntensity = controller.Execute("GetCarrierOutputIntensityLevel")

    ' Set the carrier output intensity level.
    If carrierOutputIntensity <> 8 Then
        Call controller.Execute("SetCarrierOutputIntensityLevel", 8)
    End If
```

#### 4.2.2.4.3. SetCarrierOutputIntensitydBm commands

Set the carrier output strength in dBm × 10 units. This command is available for UR22 for Japan and UR20, UR21, and UR22 for overseas. The set value is initialized by disconnection/terminal power off (initial value 200).

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specify Carrier Output Strength (dBm x 10).<br>Value range: 50-200 |
| Returned value | Without | |

Examples of uses

GetCarrierOutputIntensitydBm commands

#### 4.2.2.4.4. GetCarrierOutputIntensitydBm commands

Acquires the carrier output intensity in units of dBm × 10. This command is available for UR22 for Japan and UR20, UR21, and UR22 for overseas.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Keep the Carrier Output Strength (dBm x 10). |

Examples of uses

```
    ' Acquire the carrier output strength.
    Dim carrierOutputIntensity As Long
    CarrierOutputIntensity = controller.Execute("GetCarrierOutputIntensitydBm")

    ' Set Carrier Output Strength
    If carrierOutputIntensity <> 200 Then
        Call controller.Execute("SetCarrierOutputIntensitydBm", 200)
    End If
```

#### 4.2.2.4.5. SetFrequency commands

Sets the frequency permitted for carrier output with the bit flag. The set value is initialized by disconnection/terminal power off (initial value indicates all corresponding channels are valid, 0x7FF8).

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | The bit flag specifies the frequency at which carrier output is permitted.<br>Value range: 3 bits (0x0008)-14 bits (0x7FF8) |
| Returned value | Without | |

Examples of uses

Refer to GetFrequency commands

#### 4.2.2.4.6. GetFrequency commands

Acquires the frequency permitted for carrier output using the bit flag.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Holds the bit flag of the frequency for which carrier output is permitted. |

Examples of uses

```
    ' Gets the frequency for which carrier output is permitted.
    Dim frequencyFlg As Long
    FrequencyFlg = controller.Execute("GetFrequency")

    ' Set the frequency permitting carrier output.
    If frequencyFlg <> &H7FF8 Then
        Call controller.Execute("SetFrequency", &H7FF8)
    End If
```

#### 4.2.2.4.7. SetAntennaPort commands

Use the bit flag to set the antenna port for outputting the carrier for RFID tag communication. The set value is initialized by disconnection/terminal power off (initial value is 0x01). Only UR21 and UR22 can be used for this command. Note that this command cannot be read if the antenna port 2 is included in the UR20.

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specify the bit flag that indicates the antenna port.<br>0x01: Antenna port 1<br>0x02: Antenna port 2<br>0x03: Antenna ports 1 and 2 |
| Returned value | Without | |

Examples of uses

Refer to GetAntennaPort commands

### 4.2.2.4.8. GetAntennaPort commands

Gets the antenna port that outputs the carrier for RFID tag communication.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Holds the bit flag indicating the antenna port. |

Examples of uses

```
’ Gets the frequency for which carrier output is permitted.
Dim antennaFlg As Long
AntennaFlg = controller.Execute(”GetAntennaPort”)

’ Set the frequency permitting carrier output.
If antennaFlg <> &H3 Then
    Call controller.Execute(”SetAntennaPort”, &H3)
End If
```

### 4.2.2.4.9. SetRereadPrevention commands

Enables/disables the read prevention function for UII read processing and tag communication processing. If the anti-read function is enabled, RF tags that are detected once will not be detected again unless they are ResetTerminal commanded or reconnected. The set value is initialized by disconnection/terminal power off (the initial value is 0).ResetTerminal commands

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specify whether the read protection function is enabled or disabled.<br>0: Invalid<br>1: Valid |
| Returned value | Without | |

Examples of uses

Refer to GetRereadPrevention commands

#### 4.2.2.4.10. GetRereadPrevention commands

Gets the validity/invalidity of the RFID tag discovery completion determination in UII read processing and tag communication processing.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Maintains the read protection function enabled/disabled. |

Examples of uses

```
' Enables/disables the read-twice prevention function.
Dim rereadPrevention As Long
RereadPrevention = controller.Execute("GetRereadPrevention")

' Enable/disable the read-twice prevention function.
If rereadPrevention <> 0 Then
    Call controller.Execute("SetRereadPrevention", 0)
End If
```

#### 4.2.2.4.11. SetQValue commands

Sets the Q value to be used to adjust the probability that responses from tags will collide during UII acquisition. The set value is initialized by disconnection/terminal power off (initial value is 4).

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specifies the Q value.<br>Range: 0-7 |
| Returned value | Without | |

Examples of uses

Refer to GetQValue commands

### 4.2.2.4.12. GetQValue commands

Obtains the Q value that is used to adjust the probability that responses from tags collide during UII acquisition.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Hold the Q value. |

Examples of uses

```
    ' Obtain Q value
    Dim qValue As Long
    qValue = controller.Execute("GetQValue")

    ' Set Q value
    If qValue <> 4 Then
        Call controller.Execute("SetQValue", 0)
    End If
```

### 4.2.2.4.13. SetSession commands

Configure the session to use when you acquire UII for the tag. By setting this, you can change the conditions under which the tag responds to UII acquisition again. For more information, see ISO/ IEC18000 63 (GS1 Gen2). The set value is initialized by disconnection/terminal power off (the initial value is 0).

| Item | Type description | |
|---|---|---|
| Argument | VT_I4 | Specifies the session to use when you acquire the UII for the tag.<br>0: After waiting for RFID tag communication, the UII of the same RF tag can be acquired again.<br>1: The UII data of the same RFID tag can be acquired again after the specified period of time elapses after the UII data of the RFID tag is acquired.<br>2: The UII of the same RFID tag can be acquired again after the specified period of time elapses after the RFID tag communication standby status.<br>Same as 3:2. |
| Returned value | Without | |

Examples of uses

Refer to GetSession commands

#### 4.2.2.4.14. GetSession commands

Gets the session you want to use when you get the UII for the tag.

| Item | Type description | |
|---|---|---|
| Argument | Without | |
| Returned value | VT_I4 | Hold the session. |

Examples of uses

```
’ Get a session
Dim session As Long
Session = controller.Execute(”GetSession”)

’ Set up a session
If session <> 0 Then
    Call controller.Execute(”SetSession”, 0)
End If
```

#### 4.2.2.4.15. ResetTerminal commands

Interrupts running processing (UII reading, tag communication). When this processing is executed, the processing result being executed is discarded. If the anti-read function is enabled, the read data is also cleared.

| Item | Type description |
|---|---|
| Argument | Without |
| Returned value | Without |

Examples of uses

```
Call controller.Execute(”ResetTerminal”)
```

### 4.2.2.4.16. Reboot commands

Restart the RFID Table scanner.

| Item | Type description |
|---|---|
| Argument | Without |
| Returned value | Without |

Examples of uses

```
Call controller.Execute(”Reboot”)
```

### 4.2.2.4.17. GetSerialNumber commands

Gets the serial number of the RFID Table scanner.

| Item | Type description | | |
|---|---|---|---|
| Argument | Without | | |
| Returned value | VT_ARRAY \| VT_BSTR | | |
| | 1 | VT_BSTR | Hold the main board serial number. |
| | 2 | VT_BSTR | Hold the RF board serial number. |

Examples of uses

```
’ Obtaining serial number
Dim serialNumbers As Variant
SerialNumbers = controller.Execute(”GetSerialNumber”)

’ Main board serial number
Dim mainSerial As String
MainSerial = serialNumbers(0)
’ RF board serial number
Dim rfSerial As String
RfSerial = serialNumbers(1)
```

#### 4.2.2.4.18. GetVersion commands

Gets the version information of the RFID Table scanner.

| Item | Type description | | |
|---|---|---|---|
| Argument | Without | | |
| Returned value | VT_ARRAY \| VT_VARIANT | | |
| | 0 | VT_I4 | Hold the OS version. |
| | 1 | VT_I4 | Hold the main application version. |
| | 2 | VT_I4 | Hold the RF chip software version. |
| | 3 | VT_I4 | Hold the OEM software version. |

Examples of uses

```
    ' Get Version
    Dim versions() As Long
    Versions = controller.Execute("GetVersion")

    'OS version
    Dim osVer As Long
    OsVer = versions(0)
    ' Main application version
    Dim mainAppVer As Long
    MainAppVer = versions (1)
    ' RF chip software version
    Dim rfTipVer As Long
    RfTipVer = versions(2)
    'OEM software version
    Dim oemVer As Long
    OemVer = versions(3)
```

#### 4.2.2.4.19. KeepAlive commands

Send the terminal communication command to the RFID Table scanner. If there is no communication with the scanner for 60 seconds, the scanner determines that the line is disconnected and closes the sockets. Send this command within that time.

| Item | Type description |
|---|---|
| Argument | Without |
| Returned value | Without |

Examples of uses

```
' Keep-alive Execute Method Executes KeepAlive commands once every 10 seconds.
Private Sub DoKeepAlive()
    Execute KeepAlive commands.
    Call controller.Execute("KeepAlive")
    Repeat the KeepAlive once every 10 seconds
    Call Application.OnTime(Now + TimeSerial(0, 0, 10), "DoKeepAlive")
```

```
End Sub
```

#### 4.2.2.4.20. ReadUII commands

Detects RFID tags and reads UII data. If more than one RFID tag exists in the detecTable range when executed, all of the tags can be read.

| Item | Type description | | | |
|---|---|---|---|---|
| Argument | VT_BOOL | | | Specifies whether to acquire additional information as well.<br>False: No additional data is retrieved.<br>True: Gets additional data. |
| Returned value | VT_ARRAY \| VT_VARIANT | | | |
| | I | VT_ARRAY \| VT_VARIANT | | Retains data corresponding to one of the RFID tags. |
| | | 0 | VT_I4 | Holds the PC data of the i-th RFID tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | Holds the UII data of the i-th RFID tag in byte array. |
| | | 2 | VT_I4 | This exists only when additional information is acquired.<br>Holds the RSSI values at the time of data reception. |
| | | 3 | VT_I4 | This exists only when additional information is acquired.<br>Holds the antenna number for data reception. |

Examples of uses

```
    Dim uiis As Variant
    Uiis = controller.Execute("ReadUII", False)
    If Not IsEmpty(uiis) Then
        ' Obtain the number of detected items.
        Dim detectedElem As Long
        DetectedElem = UBound(uiis)

        ' Repeat for the number of detected RFID tags
        Dim i As Long
        For i = 0 To detectedElem
            Obtain PC data
            Dim pcData As Long
            PcData = uiis(i)(0)

            ' Obtain  UII data
            Dim uiiData() As Byte
            UiiData = uiis(i)(1)
            ' Gets 'UII data size
            Dim uiiDataSize As Long
            UiiDataSize = UBound(uiiData)
            Data extraction for 'UII data size
            Dim j As Long
            For j = 0 To uiiDataSize
                Dim value As Byte
                Value = uiiData(0)
            Next j
        Next i
    End If
```

#### 4.2.2.4.21. StartContinuousRFDetection commands

Start reading consecutive UII. Execute the ReadContinuousUII and StopContinuousRFDetection commands after executing this command.

| Item | Type description | |
|---|---|---|
| Argument | VT_UI1 | Specify the reaction mode.<br>0: Continuous reaction mode<br>Detects RFID tags that have the same UII in commands many times.<br>1: Two reading prevention mode<br>Detects only one RFID tag with the same UII. |
| Returned value | Without | |

Examples of uses

Refer to ReadContinuousUII commands

#### 4.2.2.4.22. ReadContinuousUII commands

Reads UII data of RFID tags detected after StartContinuousRFDetection is executed.

| Item | Type description | | | |
|---|---|---|---|---|
| Argument | VT_BOOL | | | Specifies whether to acquire additional information as well.<br>False: No additional data is retrieved.<br>True: Gets additional data. |
| Returned value | VT_ARRAY \| VT_VARIANT | | | |
| | I | VT_ARRAY \| VT_VARIANT | | Retains data corresponding to one of the RFID tags. |
| | | 0 | VT_I4 | Holds the PC data of the i-th RFID tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | Holds the UII data of the i-th RFID tag in byte array. |
| | | 2 | VT_I4 | This exists only when additional information is acquired.<br>Holds the RSSI values at the time of data reception. |
| | | 3 | VT_I4 | This exists only when additional information is acquired.<br>Holds the antenna number for data reception. |

Examples of uses

```
<summary> <summary> Whether consecutive reads are started
Private isRunning As Boolean

’ <summary>
’ Continuous Read Start Method
’ </summary>
Public Sub StartContinuousRF()
    Call controller.Execute(”StartContinuousRFDetection”, 0)
    IsRunning = True
    Call DoReadContinuousUII
End Sub

’ <summary>
’ Continuous Read Stop Method
’ </summary>
Public Sub StopContinuousRF()
    IsRunning = False
    Call controller.Execute(”StopContinuousRFDetection”)
End Sub

’ <summary>
’ Continuous read execution
’ </summary>
Private Sub DoReadContinuousUII()
```

```
    If Not isRunning Then
        Exit Sub
    End If

    Dim uiis As Variant
    Uiis = controller.Execute("ReadContinuousUII", False)
    If Not IsEmpty(uiis) Then
        ' Obtain the number of detected items.
        Dim detectedElem As Long
        DetectedElem = UBound(uiis)

        ' Repeat for the number of detected RFID tags
        Dim i As Long
        For i = 0 To detectedElem
            ' Obtain PC data
            Dim pcData As Long
            PcData = uiis(i)(0)

            ' Obtain UII data
            Dim uiiData() As Byte
            UiiData = uiis(i)(1)
            ' Gets UII data size
            Dim uiiDataSize As Long
            UiiDataSize = UBound(uiiData)
            ' Data extraction for UII data size
            Dim j As Long
            For j = 0 To uiiDataSize
                Dim value As Byte
                Value = uiiData(0)
            Next j
        Next i
    End If
    ExecuteTime = Now + TimeSerial(0, 0, 1)
    Call Application.OnTime(executeTime, "DoReadContinuousUII")
End Sub
```

#### 4.2.2.4.23. StopContinuousRFDetection commands

Stops continuous UII reading.

| Item | Type description |
|---|---|
| Argument | Without |
| Returned value | Without |

Examples of uses

Refer to ReadContinuousUII commands

#### 4.2.2.4.24. ReadMemory commands

Reads the data in the specified memory of the RFID tag.

| Item | Type description | | | |
|---|---|---|---|---|
| Argument | VT_VARIANT\| VT_ARRAY | | | |
| | 0 | VT_I4 | | The output antenna port is specified by the bit flag.<br>0x00: Follows the parameter settings.<br>0x01: Use antenna port 1.<br>0x02: Use antenna port 2. |
| | 1 | VT_I4 | | Specifies the memory banks of the RFID tags that you want to read.<br>0x00: Rederved Bank<br>0x01: UII Bank<br>0x02: TID Bank<br>0x03: User Bank |
| | 2 | VT_I4 | | Specifies the number of read bytes in 2-byte units.<br>Range: 2-256 |
| | 3 | VT_I4 | | Specify the read start pointer in 2-byte units.<br>Range depends on the tag. |
| | 4 | VT_I4 | | Specify the access password. Specify 0 except for access to the read locked state area. |
| | 5 | VT_VARIANT \| VT_ARRAY | | You can choose.<br>Specifies the RFID to be read.<br>If not specified, all detected RFID are targeted. |
| | | 0 | VT_I4 | Specifies the PC-data for RFID tags. |
| | | 1 | VT_UI1 \| VT_ARRAY | Specifies the UII-data for RFID tags in byte-array. |
| Returned value | VT_VARIANT \| VT_ARRAY | | | |
| | J | VT_VARIANT \| VT_ARRAY | | Retains data for one RFID tag. |
| | | 0 | VT_I4 | Holds the RFID tag's PC tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | This function holds UII data of RFID tags in a byte array. |
| | | 2 | VT_UI1 \| VT_ARRAY | The read memory data is held in byte array. |

Examples of uses

```
    ' All detected RFID is read.
    ' Antenna Port: Follows the settings
    ' Memory Bank to Read: Reserved Bank
    ' Read bytes: 4 Byte
    ' Read start position: 0
    ' Access password: 0
    Dim readParam() As Variant
    ReadParam = Array(0, 0, 4, 0, 0)

    Dim memory As Variant
    Memory = controller.Execute("ReadMemory", readParam)
    If Not IsEmpty(memory) Then
        ' Obtain the number of detected items.
        Dim detectedElem As Long
        DetectedElem = UBound(memory)

        ' Repeat for the number of detected RFID tags
        Dim i As Long
        For i = 0 To detectedElem
            ' Get memory data
            Dim memoryData() As Byte
            MemoryData = memory(i)(2)

            ' Get memory size
            Dim memorySize As Long
            MemorySize = UBound(memoryData)
            ' Extraction of data for memory size
            Dim j As Long
            For j = 0 To memorySize
              ' Make the current value +1
              MemoryData(j) = memoryData(j) + 1
            Next j

            ' Identify read RFID tags and write values to memories
            Dim rfidData(1) As Variant
            RfidData(0) = memory(i)(0)
            RfidData(1) = memory(i)(1)
            Dim writeParam() As Variant
            WriteParam = Array(0, 0, 4, 0, 0, memoryData, rfidData)
            Call controller.Execute("WriteMemory", writeParam)
        Next i

    End If
```

### 4.2.2.4.25. WriteMemory commands

Writes the data in the specified memory of the RFID tag.

| Item | Type description | | | |
|---|---|---|---|---|
| Argument | VT_VARIANT\| VT_ARRAY | | | |
| | 0 | | VT_I4 | The output antenna port is specified by the bit flag.<br>0x00: Follows the parameter settings.<br>0x01: Use antenna port 1.<br>0x02: Use antenna port 2. |
| | 1 | | VT_I4 | Specifies the memory banks of the RFID tags that you want to write to.<br>0x00: Rederved Bank<br>0x01: UII Bank<br>0x02: TID Bank<br>0x03: User Bank |
| | 2 | | VT_I4 | Specifies the number of bytes to be written in units of 2 bytes.<br>Range: 2-64 |
| | 3 | | VT_I4 | Write start pointer is specified in 2-byte units.<br>Range depends on the tag. |
| | 4 | | VT_I4 | Specify the access password. Specify 0 except for access to the write locked state area. |
| | 5 | | VT_UI1 \| VT_ARRAY | Written data |
| | 6 | | VT_VARIANT \| VT_ARRAY | You can choose.<br>Specifies the RFID to be written.<br>If not specified, all detected RFID are targeted. |
| | | 0 | VT_I4 | Specifies the PC-data for RFID tags. |
| | | 1 | VT_UI1 \| VT_ARRAY | Specifies the UII-data for RFID tags in byte-array. |
| Returned value | VT_VARIANT \| VT_ARRAY | | | |
| | J | | VT_VARIANT \| VT_ARRAY | Retains data for one RFID tag. |
| | | 0 | VT_I4 | Holds the RFID tag's PC tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | This function holds UII data of RFID tags in a byte array. |

Examples of uses

Refer to ReadMemory commands

#### 4.2.2.4.26. Lock commands

Changes the lock status of the tag.

| Item | Type description | | | | |
|---|---|---|---|---|---|
| Argument | VT_VARIANT\| VT_ARRAY | | | | |
| | 0 | VT_I4 | | | The output antenna port is specified by the bit flag.<br>0x00: Follows the parameter settings.<br>0x01: Use antenna port 1.<br>0x02: Use antenna port 2. |
| | 1 | VT_I4 | | | Specify the object. Locks apply only to both Read/Write and Write if the subject is passwords.<br>0x01: Kill password<br>0x02: Access password<br>0x10: UII Bank<br>0x20: TID Bank<br>0x40: User Bank |
| | 2 | VT_I4 | | | Specifies the locked state.<br>0x00: unlock<br>0x01: Lock<br>0x10: Permanent unlock<br>0x11: Permanent lock<br>※ Tags that have been permanently unlocked/locked cannot be changed from then on. |
| | 3 | VT_I4 | | | Set the access password. |
| | 4 | VT_VARIANT \| VT_ARRAY | | | You can choose.<br>Specifies the RFID to be read.<br>If not specified, all detected RFID are targeted. |
| | | 0 | VT_I4 | | Specifies the PC-data for RFID tags. |
| | | 1 | VT_UI1 \| VT_ARRAY | | Specifies the UII-data for RFID tags in byte-array. |
| Returned value | VT_VARIANT \| VT_ARRAY | | | | |
| | J | VT_VARIANT \| VT_ARRAY | | | Retains data for one RFID tag. |
| | | 0 | VT_I4 | | Holds the RFID tag's PC tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | | This function holds UII data of RFID tags in a byte array. |

Examples of uses

```
' All detected RFID is read.
' Antenna Port: Follows the settings
' Locked: Access password
' Lock status: Lock
' Access password: 0
Dim readParam() As Variant
ReadParam = Array(0, 0, 4, 0, 0)

Dim memory As Variant
Memory = controller.Execute("ReadMemory", readParam)
If Not IsEmpty(memory) Then
    ' Obtain the number of detected items.
    Dim detectedElem As Long
    DetectedElem = UBound(memory)

    ' Repeat for the number of detected RFID tags
    Dim i As Long
    For i = 0 To detectedElem
        ' Identify and lock imported RFID tags.
        Dim rfidData(1) As Variant
        RfidData(0) = memory(i)(0)
        RfidData(1) = memory(i)(1)
        Dim lockParam() As Variant
        lockParam = Array(0, 1, 1, 0, rfidData)
        Call controller.Execute("Lock", lockParam)
    Next i

End If
```

### 4.2.2.4.27. Kill commands

Disables tags. Please note that the RFID tags that execute these commands will become unavailable thereafter.

| Item | Type description | | |
|---|---|---|---|
| Argument | VT_VARIANT\| VT_ARRAY | | |
| | 0 | VT_I4 | The output antenna port is specified by the bit flag.<br>0x00: Follows the parameter settings.<br>0x01: Use antenna port 1.<br>0x02: Use antenna port 2. |
| | 1 | VT_I4 | Specifies the kill password. |
| | 2 | VT_VARIANT \| VT_ARRAY | You can choose.<br>Specifies the RFID to be read.<br>If not specified, all detected RFID are targeted. |

| Item | Type description | | | |
|---|---|---|---|---|
| | | 0 | VT_I4 | Specifies the PC-data for RFID tags. |
| | | 1 | VT_UI1 \| VT_ARRAY | Specifies the UII-data for RFID tags in byte-array. |
| Returned value | VT_VARIANT \| VT_ARRAY | | | |
| | j | VT_VARIANT \| VT_ARRAY | | Retains data for one RFID tag. |
| | | 0 | VT_I4 | Holds the RFID tag’s PC tag. |
| | | 1 | VT_UI1 \| VT_ARRAY | holds UII data of RFID tags in a byte array. |

Examples of uses

```
’ Target all detected RFID.
’ Antenna Port: Follows the settings
’ Kill password: 1
’ Specified UII Tag: Detected UII Tag Data

Dim readParam() As Variant
readParam = Array(0, 0, 4, 0, 0)

Dim memory As Variant
memory = controller.Execute(″ReadMemory″, readParam)
If Not IsEmpty(memory) Then
    ’ Get the detected number
    Dim detectedElem As Long
    detectedElem = UBound(memory)

    ’ Repeat for the number of RFID tags detected
    Dim i As Long
    For i = 0 To detectedElem
        ’ Identify the read RFID tag and kill it individually
        Dim rfidData(1) As Variant
        rfidData(0) = memory(i)(0)
        rfidData(1) = memory(i)(1)
        Dim writeParam() As Variant
        KillParam = Array(0, 1, rfidData)
        Call controller.Execute(″Kill″, KillParam)
    Next i

End If
```

### 4.2.3. CaoVariable classes

#### 4.2.3.1. Value properties

Retrieves data from the connected scanner. Variable names act differently. For more information, see 4.3 Variable list.

## 4.3. Variable list

Defines a list of variables that are available for each class. Variables refer to objects in CaoVariable classes.

CaoController class variables

| Variable name | Description | Value | | Reference |
|---|---|---|---|---|
| | | Get | Put | |
| @MAKER_NAME | Gets the name of the manufacturer. | ○ | – | P.38 |
| @VERSION | Gets the provider version. | ○ | – | P.39 |
| @SERIAL | Acquire the serial number | ○ | | P.40 |
| @LASTDEVICEERROR | Gets the previous device error information. If 0x80100001 is not generated, it is Empty. | ○ | – | P.41 |

#### 4.3.1.1. @MAKER_NAME

Obtain the name of the manufacturer.

Data type

| Type description | |
|---|---|
| VT_BSTR | Gets the name of the manufacturer. |

Examples of uses

```
    ' Add Variable
    Dim variable As CaoVariable
    Set variable = controller.AddVariable("@MAKER_NAME")
    ' Value acquisition
    Dim strVal As String
    StrVal = variable.value
```

#### 4.3.1.2. @VERSION

Gets the provider version.

Data type

| Type description | | |
|---|---|---|
| VT_ARRAY \| VT_I4 | | |
| 0 | VT_I4 | OS Version |
| 1 | VT_I4 | Main application version |
| 2 | VT_I4 | RF application version |
| 3 | VT_I4 | RF chip software version |
| 4 | VT_I4 | OEM software version |

Examples of uses

```
’ Add Variable
Dim variable As CaoVariable
Set variable = controller.AddVariable(”@VERSION”)

Dim version As Variant
Version = variable.value

’OS version
If Not IsEmpty(varsion(0)) Then
    Dim os As Integer
    Os = version(0)
End If
’ Main application version
If Not IsEmpty(version(1)) Then
    Dim mainAp As Integer
    MainAp = version(1)
End If
’ RF application version
If Not IsEmpty(version(2)) Then
    Dim rfAp As Integer
    RfAp = version(2)
End If
’ RF chip software
If Not IsEmpty(version(3)) Then
    Dim rfFirm As Integer
    RfFirm = version(3)
End If

’OEM software version
If Not IsEmpty(version(4)) Then
    Dim oem As Integer
    Oem = version(4)
End If
```

### 4.3.1.3. @SERIAL

Gets the provider version.

Data type

| Type description | | |
|---|---|---|
| VT_ARRAY \| VT_BSTR | | |
| 0 | VT_BSTR | Main board serial number |
| 1 | VT_BSTR | RF board serial number |

Examples of uses

```
’ Add Variable
Dim variable As CaoVariable
Set variable = controller.AddVariable(”@SERIAL”)

Dim serial As Variant
Serial = variable.value

’ Main board serial number
If Not IsEmpty(serial(0)) Then
    Dim mainNo As String
    MainNo = serial(0)
End If
’ RF Base Serial Number
If Not IsEmpty(serial(1)) Then
    Dim rfNo As String
    RfNo = serial(1)
End If
```

### 4.3.1.4. @LASTDEVICEERROR

Gets the previous device error information. If 0x80100001 is not generated, it is Empty.

Data type

| Type description | | |
|---|---|---|
| VT_ARRAY \| VT_VARIANT | | |
| 0 | VT_I2 | Holds the value of SW. |
| 1 | VT_I2 | Holds the Result values.<br>※ If the Result does not exist, VT_EMPTY is returned. |
| 2 | VT_I2 | Holds the Error1 values.<br>※ If the Result does not exist, VT_EMPTY is returned. |
| 3 | VT_I2 | Holds the Error2 values.<br>※ If the Result does not exist, VT_EMPTY is returned. |

Examples of uses

```
' Add Variable
Dim variable As CaoVariable
Set variable = controller.AddVariable("@LASTDEVICEERROR")
' Value acquisition
Dim lastError As Variant
LastError = variable.Value
If Not IsEmpty(lastError) Then
    ' Switching section
    Dim sw As Integer
    Sw = lastError(0)

    ' Result data
    If Not IsEmpty(lastError(1)) Then
        Dim result As Integer
        Result = lastError(1)
    End If
    ' Error1
    If Not IsEmpty(lastError(2)) Then
        Dim error1 As Integer
        Error1 = lastError(3)
    End If

    ' Error2
    If Not IsEmpty(lastError(3)) Then
        Dim error2 As Integer
        Error2 = lastError(3)
    End If
End If
```

# 5. UR20 provider error code

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

See the Error Code section in the ORiN2 Programming Guide (Link) for ORiN2 common errors. C:¥ORiN2¥CAO¥Doc¥ORiN2_ProgrammersGuide_ja.pdf

Table 5-1 Unique Error Code Table

| Error Number | Description |
|---|---|
| 0x80110001 | Required options are undefined. Specify the required options. |
| 0x80110002 | An unexpected response was received. Check that the product is compatible with the product. |
| 0x80110003 | Checksum of received data is invalid. Check the communication status. |

# Appendix A.

## Communication Protocol Command Correspondence Table

CaoWorkspace

| Method | Communications command |
|---|---|
| AddController | AP load |
|  | RF Open |
| Controllers::Romove | RF Close |
|  | AP unload |

CaoController::Execute

| Commanded | Communications command |
|---|---|
| SetCarrierOutputIntensityLevel | Parameter setting <Carrier output intensity setting (level designation)> |
| GetCarrierOutputIntensitydLevel | Parameter acquisition <Carrier output intensity setting (level specification)> |
| SetCarrierOutputIntensitydBm | Parameter setting <Carrier output strength setting (dBm setting)> |
| GetCarrierOutputIntensitydBm | Parameter acquisition <Carrier output strength setting (dBm specified)> |
| SetFrequency | Parameter setting <Frequency setting> |
| GetFrequency | Parameter acquisition <Frequency setting> |
| SetAntennaPort | Parameter setting <Antenna port setting> |
| GetAntennaPort | Parameter acquisition <Antenna port setting> |
| SetRereadPrevention | Parameter setting <Prevent reading> |
| GetRereadPrevention | Parameter Retrieve <Set to prevent two readings> |
| SetQValue | Parameter setting <Q value setting> |
| GetQValue | Obtain parameter <Q value set> |
| SetSession | Parameter setting <Session setting> |
| GetSession | Get Parameter <Set Session> |
| ResetTerminal | Processing interruption |
| Reboot | Terminal reset |
| GetSerialNumber | Obtaining serial number |
| GetVersion | Get Version |
| KeepAlive | Confirmation of terminal communication |

| | |
|---|---|
| ReadUII | Read UII (Start) |
| | UII reading (result check) |
| | UII read (buffer fetch) |
| | UII Read (Extract Buffer with Additional Information) |
| StartContinuousRFDetection | Continuous UII Read (Start) |
| ReadContinuousUII | Continuous UII reading (data acquisition) |
| | Continuous UII reading (data acquisition with additional information) |
| StopContinuousRFDetection | Continuous UII Read (End) |
| ReadMemory | Tag Communication (Start) Memory Read |
| | Tag communication (confirmation of result) |
| | Tag Communication (Buffer Remove) Memory Read |
| WriteMemory | Tag Communication (Start) Memory Write |
| | Tag communication (confirmation of result) |
| | Tag communication (result check) Memory write, lock |
| Lock | Tag Communication (Start) Lock |
| | Tag communication (confirmation of result) |
| | Tag communication (result check) Memory write, lock |
| Kill | Tag communication (start) kill |
| | Tag communication (result confirmation) |
| | Tag communication (result confirmation) Memory light, lock, kill |

## Appendix B. Error list from the scanner

UR20 provider return errors from scanners masked by 0x8010****.

| Error Number | Description |
|---|---|
| 0x80100001 | Error from device. Check the values of the @LASTDEVICEERROR variables. |
| 0x801001xx | An error occurs during communication with the RF tag. Review the communication environment. Check the specified password. |
| 0x80100203 | Access outside the memory range. Check the specified parameters. |
| 0x80100204 | Access to lock memory. Perform password authentication. (Permanent locking is not possible. ) |
| 0x8010020B | Memory write power is insufficient. |
| 0x8010020F | Other errors. |