

# KEYENCE LJ-V7000 Provider

Version 1.0.1

## User's Guide

June 29, 2020

Remarks:

Do not change the settings with other applications while the device is connected by the provider.

This document is translated from Japanese into English by the machine translation.

**[Revision History]**

Version	Date	Contents
1.0.0	2018-08-30	First Edition
	2020-03-31	Clerical corrections.
1.0.1	2020-06-29	Fixed a bug that an application error occurs when 0 is entered in the argument of GetError command.

**[Hardware]**

Model	Version	Notes

## Contents

1. Introduction .....	5
2. Outline of provider .....	6
2.1. Outline.....	6
2.2. Method and property .....	7
2.2.1. CaoWorkspace::AddController method .....	7
2.2.2. CaoController::Execute method .....	8
2.3. Event .....	8
2.3.1. CaoController::OnMessage event .....	8
2.4. Error code.....	9
3. Command reference .....	10
3.1. Systems control .....	11
3.1.1. CaoController::Execute("GetError") command .....	11
3.1.2. CaoController::Execute("ClearError") command .....	11
3.2. Measurement controlling.....	12
3.2.1. CaoController::Execute("Trigger") command.....	12
3.2.2. CaoController::Execute("StartMeasure") command .....	12
3.2.3. CaoController::Execute("StopMeasure") command .....	12
3.2.4. CaoController::Execute("AutoZero") command.....	13
3.2.5. CaoController::Execute("Timing") command.....	13
3.2.6. CaoController::Execute("Reset") command.....	14
3.2.7. CaoController::Execute("ClearMemory") command .....	14
3.3. Setting change/read related .....	14
3.3.1. CaoController::Execute("GetTime") command .....	14
3.3.2. CaoController::Execute("ChangeActiveProgram") command.....	15
3.3.3. CaoController::Execute("GetActiveProgram") command .....	15
3.4. Acquisition of measurement results .....	15
3.4.1. CaoController::Execute("GetMeasurementValue") command.....	15
3.4.2. CaoController::Execute("GetProfile") command.....	16
3.4.3. CaoController::Execute("GetBatchProfile") command.....	17
3.4.4. CaoController::Execute("GetProfileAdvance") command.....	19
3.4.5. CaoController::Execute("GetBatchProfileAdvance") command .....	20
3.5. Storage function related .....	21
3.5.1. CaoController::Execute("StartStorage") command.....	21
3.5.2. CaoController::Execute("StopStorage") command .....	21
3.5.3. CaoController::Execute("GetStorageStatus") command.....	22
3.5.4. CaoController::Execute("GetStorageData") command.....	22

---

3.5.5. CaoController::Execute("GetStorageProfile") command.....	23
3.5.6. CaoController::Execute("GetStorageBatchProfile") command .....	25
3.6. High-speed data communications .....	26
3.6.1. CaoController::Execute("StartHighSpeedDataCommunication") command .....	26
3.6.2. CaoController::Execute("StopHighSpeedDataCommunication") command .....	26

## 1. Introduction

This document is the user's guide for the LJ-V7000 provider, a CAO provider for the LJ-V7000 series of two-dimensional laser displacement monitors manufactured by KEYENCE.

LJ-V7000 providers communicate with Ethernet connected LJ-V7000 series controllers to control measurements and acquire measurement results.

This document describes the functionality of this LJ-V7000 provider and the methods it implements.

## 2. Outline of provider

### 2.1. Outline

LJ-V7000 provider provides communication functions by executing commands in the KEYENCE Communication Library via CaoController::Execute.

**Table 2-1 LJ-V7000 provider**

File name	CaoProvKEYENCELJV7000.dll
ProgID	CaoProv.KEYENCE.LJ-V7000
Registration	Regsvr32 CaoProvKEYENCELJV7000.dll
Deregistration	Regsvr32 /u CaoProvKEYENCELJV7000.dll

**Table 2-2 Dependent Modules**

No.	File name	Description
1	LJV7_IF.dll	KEYENCE's Ethernet, USB-communication library
2	vc redistrib_x86.exe	Microsoft Visual C++2008 redistributable package (x86) (You must install before you register a provider)



number for high-speed data communication>:<Number of profiles to be transmitted at one time>]	OFF (the first parameter is 0), the second and subsequent parameters can be omitted. (Default: 0) <High-Speed Data Communication ON/OFF>: 0: OFF 1: ON Example : "HSDComm=0" "HSDComm=1:24692:10"
---	---

#### Example

---

```
Dim caoCtrl As Object
caoCtrl = caoWs.AddController ("LJV7",
                              "CaoProv. KEYENCE. LJ-V7000",
                              "",
                              "conn=192.168.0.1:24691, DeviceID=1,
                              Head=1, HSDComm=1:24692:10")
```

---

## 2.2.2. CaoController::Execute method

Sends and receives commands. Specify the command name as the first argument and the command parameter as the second argument. Details of each command are given in 3. See 3 Command reference.

**Syntax** Execute(<bstrCommandName:BSTR>[, <vntParam:VT\_VARIANT>])

bstrCommandName : [in] Command name  
 vntParam : [in] Command argument

## 2.3. Event

### 2.3.1. CaoController::OnMessage event

Data output during high-speed data communication is received.

Data reception may delay when sampling frequency of device is high. In that case, receive the multiple profiles collectively by one of the following means.

1. Increase the 2nd argument of StartHighSpeedDataCommunication command (dwProfileCnt).
2. Increase the 3rd argument of HSDComm option of AddController(Number of profiles to be transmitted at one time).



**Table 2-4 Event type list**

Type	Description	Value content
0x (x: DeviceID)	Profile information	The profile measured by the controllers with the IDs specified by the DeviceID options is stored in the order shown in Information Table 2-5. Profile data is received at the same time as profile data is received from the same DeviceID of controllers. The value does not change while high-speed data communication continues.
1x (x: DeviceID)	Profile data	Gets the profile data measured by the controller with the IDs specified by the DeviceID options. You receive data when you accumulate data in the StartHighSpeedDataCommunication arguments or the number of profiles specified by the HSDComm options. Refer to Section 9.2.9.6 of the LJ-V7000 Series Communication Library Reference Manual for the number of profile data.

**Table 2-5 Stored profile data (VT\_ARRAY|VT\_VARIANT)**

Index	Content	Data type
0	Number of profiles stored in one unit of data	VT_UI1
1	ON/OFF of profile compression (time axis)	VT_UI1
2	Number of data points in one profile	VT_UI2
3	First X coordinate	VT_I4
4	Spacing of the data points in the X direction	VT_I4

## 2.4. Error code

The LJ-V7000 provider defines the following unique error codes:

For ORiN2 common errors, see the Error Codes section in the ORiN2 programming guide. [C:\ORiN2\CAO\Doc\ORiN2\\_ProgrammersGuide\\_en.pdf](C:\ORiN2\CAO\Doc\ORiN2_ProgrammersGuide_en.pdf)

**Table 2-6 Unique error code list**

Error name	Error number	Description
Library error	0x8010xxxx	A communication library error. See "LJ-V7000 Series Communication Library" for details of the error code.

### 3. Command reference

This chapter explains the commands of the CaoController::Execute method. Refer to KEYENCE's "LJ-V7000 Series Communication Library Reference Manual" for detailed operation of the commands.

**Table 3-1 CaoController::Execute command list**

Command	LJ-V7000 Command	Function	Page
Systems control			
GetError	GetError	Gets system error information for the controller.	11
ClearError	ClearError	Release the system error of the controller.	11
Measurement controlling			
Trigger	Trigger	Issue a trigger	12
StartMeasure	StartMeasure	Start batch measurement	12
StopMeasure	StopMeasure	Batch measurement is completed.	12
AutoZero	AutoZero	Enter ZERO (for high-function modes only)	13
Timing	Timing	Enter TIMING (for high-function modes only)	13
Reset	Reset	Enter RESET (for high-function modes only)	14
ClearMemory	ClearMemory	Clear the memory inside the controller of the displacement meter.	14
Setting change/read related			
GetTime	GetTime	Gets the date and time of the controller.	14
GetActiveProgram	GetActiveProgram	Get the active program	15
ChangeActiveProgram	ChangeActiveProgram	Switch the active program	15
Acquisition of measurement results			
GetMeasurementValue	GetMeasurementValue	Gets the current value of the OUT measurement.	15
GetProfile	GetProfile	Gets the specified profile	16
GetBatchProfile	GetBatchProfile	Gets the profile of the specified batch measurement	17
GetProfileAdvance	GetProfileAdvance	Gets the specified profile	19
GetBatchProfileAdvance	GetBatchProfileAdvance	Gets the profile of the specified batch measurement	20
Storage function related			
StartStorage	StartStorage	Start storage	21
StopStorate	StopStorate	Exit the storage	21
GetStorageStatus	GetStorageStatus	Retrieve the storage status	22
GetStorageData	GetStorageData	Gets the OUT measurement in storage	22
GetStorageProfile	GetStorageProfile	Gets the profile data in storage	23
GetStorageBatchProfile	GetStorageBatchProfile	Gets batch profile data in storage	25
High-speed data communications			
StartHighSpeed DataCommunication	HighSpeedDataEthernet CommunicationInitialize PreStartHighSpeedData Communication StartHighSpeedData Communication	Start high-speed data communication	26

StopHighSpeed DataCommunication	StopHighSpeedData Communication HighSpeedData CommunicationFinalize	Stop high-speed data communication	26
------------------------------------	--	------------------------------------	----

### 3.1. Systems control

#### 3.1.1. CaoController::Execute("GetError") command

Retrieves the specified number of system error codes that have occurred in the controller.

**Syntax** GetError (<byRcvMax>)

byRcvMax : Maximum number of system errors to be acquired (VT\_UI1)

Return value : The number of system error codes generated by the controllers and the acquired error codes (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-2.

**Table 3-2 GetError data**

Index	Content	Data type
0	Number of system errors occurring in the controller	VT_UI1
1~<byRcvMax>	Acquired error code	VT_UI2

**Example**

---

```
Dim retVal As Variant
retVal = caoCtrl.Execute("GetError", 5)
```

---

#### 3.1.2. CaoController::Execute("ClearError") command

Releases the specified system error code. See Section 9.2.3 of the "LJ-V7000 Series Communication Library Reference Manual" for information on cancelable error codes.

**Syntax** ClearError (<wErrCode>)

wErrCode : System error code (VT\_UI2) to be released

Return value : None

**Example**

---

```
caoCtrl.Execute "ClearError", &H0085
```

---

## 3.2. Measurement controlling

### 3.2.1. CaoController::Execute("Trigger") command

Issue a trigger.

**Syntax** Trigger()

Argument : None

Return value : None

**Example**

---

```
caoCtrl.Execute "Trigger"
```

---

### 3.2.2. CaoController::Execute("StartMeasure") command

Start batch measurement.

**Syntax** StartMeasure()

Argument : None

Return value : None

**Example**

---

```
caoCtrl.Execute "StartMeasure"
```

---

### 3.2.3. CaoController::Execute("StopMeasure") command

End the batch measurement.

**Syntax** StopMeasure()

Argument : None

Return value : None

**Example**


---

```
caoCtrl.Execute "StopMeasure"
```

---

**3.2.4. CaoController::Execute("AutoZero") command**

Issues an autozero request for the specified OUT measurement.

**Syntax** AutoZero (<byOnOff>, <dwOut>)

byOnOff	:	ON/OFF specify (VT_UI1)
dwOut	:	OUT measurement (bit specify) to be processed (VT_UI4) E.g., dwOut = 0b0000000000010010 (0x0012) if OUT2,5 is the subject
Return value	:	None

**Example**


---

```
caoCtrl.Execute "AutoZero", Array(1, &H0012)
```

---

**3.2.5. CaoController::Execute("Timing") command**

Issues a timing request for the specified OUT measurement.

**Syntax** Timing (<byOnOff>, <dwOut>)

byOnOff	:	ON/OFF specify (VT_UI1)
dwOut	:	OUT measurement (bit specify) to be processed (VT_UI4) E.g., dwOut = 0b0000000000010010 (0x0012) if OUT2,5 is the subject
Return value	:	None

**Example**


---

```
caoCtrl.Execute "Timing", Array(1, &H0012)
```

---

### 3.2.6. CaoController::Execute("Reset") command

Issues a reset request for the specified OUT measurement.

**Syntax** Reset (<dwOut>)

dwOut : OUT measurement (bit specify) to be processed (VT\_UI4)  
 E.g., dwOut = 0b0000000000010010 (0x0012) if OUT2,5 is the subject

Return value : None

**Example**

---

```
caoCtrl.Execute "AutoZero", &H0012
```

---

### 3.2.7. CaoController::Execute("ClearMemory") command

Clears the data stored in the controller.

**Syntax** ClearMemory ()

Argument : None

Return value : None

**Example**

---

```
caoCtrl.Execute "ClearMemory"
```

---

## 3.3. Setting change/read related

### 3.3.1. CaoController::Execute("GetTime") command

Gets the date and time of the controller.

**Syntax** GetTime()

Argument : None

Return value : Controller date and time (VT\_DATE)

**Example**

---

```
Dim retVal As Variant
retVal = caoCtrl.Execute("GetTime")
```

---

**3.3.2. CaoController::Execute("ChangeActiveProgram") command**

Switches the active program number.

**Syntax** ChangeActiveProgram(<byProgNo>)

ByProgNo : Program No. after switching(VT\_UI1)

Return value : None

**Example**

---

```
caoCtrl.Execute "ChangeActiveProgram", 2
```

---

**3.3.3. CaoController::Execute("GetActiveProgram") command**

Gets the current active program number.

**Syntax** GetActiveProgram()

Argument : None

Return value : Active Program No. (VT\_UI1)

**Example**

---

```
Dim retVal As Variant
retVal = caoCtrl.Execute("GetActiveProgram")
```

---

**3.4. Acquisition of measurement results****3.4.1. CaoController::Execute("GetMeasurementValue") command**

Obtain the latest measurement value.

**Syntax** GetMeasurementValue ()

Argument : None

Return value : All OUT measurements (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-3.

**Table 3-3 GetMeasurementValue data**

Index	Content	Data type
0	Whether the OUT1 measurements are valid or invalid	VT_UI1
1	Results of OUT1 intersection decisions	VT_UI1
2	OUT1 measure	VT_R4
...	...	...
45	Whether the OUT16 measurements are valid or invalid	VT_UI1
46	Results of OUT16 intersection decisions	VT_UI1
47	OUT16 measure	VT_R4

**Example**


---

```
Dim retVal As Variant
Dim outVal As Single
retVal = caoCtrl.Execute("GetMeasurementValue")
outVal = retVal(47) 'Get measurement value of OUT16
```

---

**3.4.2. CaoController::Execute("GetProfile") command**

Gets the profile of the specified high-speed mode.

**Syntax** GetProfile (<byTargetBank>,<byPosMode>,<dwGetProfNo>,<byGetProfCnt>,<byErase>)

byTargetBank : Which of the active or deactivating planes you obtain from?  
(VT\_UI1)

byPosMode : How to specify the profile acquisition position  
(VT\_UI1)

dwGetProfNo : Profile No. to be acquired when 2 is specified by byPosMode.  
(VT\_UI4)

byGetProfCnt : Number of profiles to read (VT\_UI1)

byErase : Is the read profile deleted from the previous profile?  
(VT\_UI1)

Return value : Profile data (VT\_ARRAY|VT\_VARIANT)



The data is stored as shown in Table 3-4.

**Table 3-4 GetProfile data**

Index1	Index2	Content	Data type
0	0	Number of profiles stored in one unit of data	VT_UI1
	1	ON/OFF of profile compression (time axis)	VT_UI1
	2	Number of data points in one profile	VT_UI2
	3	First X coordinate	VT_I4
	4	Spacing of the data points in the X direction	VT_I4
	5	Latest profile number at the time of acquisition	VT_UI4
	6	Profile No. of the oldest profile held by the controller	VT_UI4
	7	Profile No. of the oldest profile read	VT_UI4
	8	Number of profiles read this time	VT_UI1
1	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_I4

**Example**

```

Dim retVal As Variant
Dim infoVal As Variant
Dim profVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetProfile", Array(0, 0, 0, 10, 0))
infoVal = retVal(0)
profVal = retVal(1)
data = profVal(infoVal(0) * infoVal(2) * 9) 'Get 1st data of 10th profile

```

### 3.4.3. CaoController::Execute("GetBatchProfile") command

Gets the profile of the batch measurement for the specified high-speed mode.

**Syntax** GetBatchProfile (<byTargetBank>,<byPosMode>,<byGetBatchNo>,  
<dwGetProfNo>,<byGetProfCnt>,<byErase>)

byTargetBank : Which of the active or deactivating planes you obtain from?  
(VT\_UI1)

byPosMode : How to specify the batch position

(VT\_UI1)

byGetBatchNo : Profile No. to be acquired when 2 is specified by byPosMode.  
(VT\_UI4)

dwGetProfNo : Acquisition start profile No. in the batch(VT\_UI4)

byGetProfCnt : Number of profiles to read (VT\_UI1)

byErase : Is the read profile deleted from the previous profile?  
(VT\_UI1)

Return value : Profile data (VT\_ARRAY|VT\_VARIANT)  
The data is stored as shown in Table 3-5.

**Table 3-5 GetBatchProfile data**

Index1	Index2	Content	Data type
0	0	Number of profiles stored in one unit of data	VT_UI1
	1	ON/OFF of profile compression (time axis)	VT_UI1
	2	Number of data points in one profile	VT_UI2
	3	First X coordinate	VT_I4
	4	Spacing of the data points in the X direction	VT_I4
	5	Latest batch number at the time of acquisition	VT_UI4
	6	Number of profiles in the latest batch	VT_UI4
	7	The oldest batch number held by the controller	VT_UI4
	8	Number of profiles in the oldest batch held by the controller	VT_UI4
	9	Batch number read this time	VT_UI4
	10	Number of profiles in the batch read this time	VT_UI4
	11	What is the oldest profile in the batch you read?	VT_UI4
	12	Number of profiles read this time	VT_UI1
	13	Is batch measurement of the latest batch number completed?	VT_UI1
1	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_I4

**Example**


---

```
Dim retVal As Variant
Dim infoVal As Variant
```

```

Dim profVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetBatchProfile", Array(0, 0, 0, 0, 10, 0))
infoVal = retVal(0)
profVal = retVal(1)
data = profVal(infoVal(0) * infoVal(2) * 9) 'Get 1st data of 10th profile

```

---

### 3.4.4. CaoController::Execute("GetProfileAdvance") command

Gets the profile of the specified high-function mode.

**Syntax** GetProfileAdvance ()

Argument : None

Return value : Profile data (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-6.

**Table 3-6 GetProfileAdvance data**

Index1	Index2	Content	Data type
0	0	Number of profiles stored in one unit of data	VT_UI1
	1	ON/OFF of profile compression (time axis)	VT_UI1
	2	Number of data points in one profile	VT_UI2
	3	First X coordinate	VT_I4
	4	Spacing of the data points in the X direction	VT_I4
1	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_I4
2	0~47	OUT measurement data It is stored in the same data order as Table 3-3.	VT_UI1 or VT_I4

**Example**

```

Dim retVal As Variant
Dim infoVal As Variant
Dim outVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetProfileAdvance")
infoVal = retVal(0)
outVal = retVal(2)
data = outVal(3 * 4 + 2) 'Get measurement value of OUT5

```

---

### 3.4.5. CaoController::Execute("GetBatchProfileAdvance") command

Gets the profile of the batch measurement for the specified high-function mode.

**Syntax** GetBatchProfileAdvance (<byPosMode>,<byGetBatchNo>,<dwGetProfNo>,<byGetProfCnt>)

byPosMode : How to specify the batch position  
(VT\_UI1)

byGetBatchNo : Profile No. to be acquired when 2 is specified by byPosMode.  
(VT\_UI4)

dwGetProfNo : Acquisition start profile No. in the batch(VT\_UI4)

byGetProfCnt : Number of profiles to read (VT\_UI1)

Return value : Profile data (VT\_ARRAY|VT\_VARIANT)  
The data is stored as shown in Table 3-7.

**Table 3-7 GetBatchProfileAdvance data**

Index1	Index2	Content	Data type
0	0	Number of profiles stored in one unit of data	VT_UI1
	1	ON/OFF of profile compression (time axis)	VT_UI1
	2	Number of data points in one profile	VT_UI2
	3	First X coordinate	VT_I4
	4	Spacing of the data points in the X direction	VT_I4
	5	Batch number read this time	VT_UI4
	6	Number of profiles in the batch read this time	VT_UI4
	7	What is the oldest profile in the batch you read?	VT_UI4
	8	Number of profiles read this time	VT_UI1
1	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_I4
2	0~	OUT measurement data for each profile The data of the following number of elements is stored <16 OUT's worth of data (in the same data order as in Table 3-3)><number of profiles read>	VT_UI1 or VT_R4
3	0~47	OUT measurement data for the batch to be acquired It is stored in the same data order as Table 3-3.	VT_UI1 or VT_R4

4	0~47	Latest OUT measurement data at the time of command execution It is stored in the same data order as Table 3-3.	VT_UI1 or VT_R4
---	------	---	-----------------

**Example**


---

```

Dim retVal As Variant
Dim infoVal As Variant
Dim outVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetBatchProfileAdvance", Array(0, 0, 0, 10))
infoVal = retVal(0)
outVal = retVal(2)
data = outVal(48 * 9 + 3 * 4 + 2) 'Get measurement value of OUT5 of 10th profile

```

---

**3.5. Storage function related****3.5.1. CaoController::Execute("StartStorage") command**

Start storage.

**Syntax** StartStorage()

Argument : None  
Return value : None

**Example**


---

```
caoCtrl.Execute "StartStorage"
```

---

**3.5.2. CaoController::Execute("StopStorage") command**

Storage ends.

**Syntax** StopStorage()

Argument : None  
Return value : None

**Example**


---

```
caoCtrl.Execute "StopStorage"
```

---

### 3.5.3. CaoController::Execute("GetStorageStatus") command

Gets the status of the storage.

**Syntax** GetStorageStatus (<dwReadArea>)

dwReadArea : Read storage plane  
(VT\_UI4)

Return value : Storage information (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-8.

**Table 3-8 GetStorageStatus data**

Index	Content	Data type
0	Free storage	VT_UI1
1	Program No. of the target storage plane	VT_UI1
2	Storage object	VT_UI1
3	Number of storage points	VT_UI4
4	Number of storage planes	VT_UI4
5	No. of the active storage surface	VT_UI4

**Example**

```
Dim retVal As Variant
Dim statusVal As Byte
retVal = caoCtrl.Execute("GetStorage", 0)
statusVal = retVal(2) 'Get storage target
```

### 3.5.4. CaoController::Execute("GetStorageData") command

Obtain the OUT measurement data of the storage.

**Syntax** GetStorageData (<dwSurface >,< dwStartNo >,< dwDataCnt >)

dwSurface : Read storage plane (VT\_UI4)

dwStartNo : Data No. to start reading(VT\_UI4)

dwDataCnt : Number of data points to be read (VT\_UI4)

Return value : Storage-OUT Measurement data (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-9.

**Table 3-9 GetStorageData data**

Index1	Index2	Content	Data type
0	0	Free storage	VT_UI1
	1	Program No. of the target storage plane	VT_UI1
	2	Storage object	VT_UI1
	3	Number of storage points	VT_UI4
	4	Data number from which reading started	VT_UI4
	5	Number of data read this time	VT_UI4
	6	Standard time	VT_DATE
1	0~	Value counted in units of 10 ms from the reference time of each data The number of read data points and the counter value are stored.	VT_UI4
2	0~	Storage OUT measurement data The data of the following number of elements is stored <16 OUT's worth of data (in the same data order as in Table 3-3)>×<the number of data points read>	VT_UI1 or VT_R4

**Example**

```
Dim retVal As Variant
Dim outVal As Variant
retVal = caoCtrl.Execute("GetStorageData", Array(0, 0, 10))
outVal = retVal(48 * 6 + 3 * 1 + 1) 'Get judge result of OUT2 of 7th profile
```

### 3.5.5. CaoController::Execute("GetStorageProfile") command

Retrieves the storage profile data.

**Syntax** GetStorageProfile (<dwSurface >,< dwStartNo >,< dwDataCnt >)

dwSurface : Read storage plane (VT\_UI4)  
dwStartNo : Data No. to start reading(VT\_UI4)  
dwDataCnt : Number of data points to be read (VT\_UI4)  
Return value : Storage profile data (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-10.

**Table 3-10 GetStorageProfile data**

Index1	Index2	Content	Data type
0	0	Free storage	VT_UI1
	1	Program No. of the target storage plane	VT_UI1
	2	Storage object	VT_UI1
	3	Number of storage points	VT_UI4
	4	Data number from which reading started	VT_UI4
	5	Number of data read this time	VT_UI4
	6	Standard time	VT_DATE
	7	Number of profiles stored in one unit of data	VT_UI1
	8	ON/OFF of profile compression (time axis)	VT_UI1
	9	Number of data points in one profile	VT_UI2
	10	First X coordinate	VT_I4
	11	Spacing of the data points in the X direction	VT_I4
1	0~	Value counted in units of 10 ms from the reference time of each data The number of read data points and the counter value are stored.	VT_UI4
2	0~	Latest OUT measurement results at the time of profile acquisition The data of the following number of elements is stored <16 OUT's worth of data (in the same data order as in Table 3-3)>×<the number of data points read>	VT_UI1 or VT_R4
3	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_I4
4	0~	OUT measurement results for each profile The data of the following number of elements is stored <16 OUT's worth of data (in the same data order as in Table 3-3)>×<the number of data points read>	VT_UI1 or VT_R4

**Example**

```

Dim retVal As Variant
Dim infoVal As Variant
Dim profVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetStorageProfile", Array(0, 0, 10))
infoVal = retVal(0)
profVal = retVal(1)
data = profVal(infoVal(7) * infoVal(9) * 9 + 1) 'Get 2nd data of 10th profile

```



### 3.5.6. CaoController::Execute("GetStorageBatchProfile") command

Obtain the OUT measurement data of the storage.

**Syntax** GetStorageBatchProfile (<dwSurface >,< dwGetBatchNo >,< dwGetBatchTopProfNo >,  
< byGetProfCnt >)

dwSurface : Read storage plane (VT\_UI4)  
 dwGetBatchNo : Batch number to be read(VT\_UI4)  
 dwGetBatchTopProfNo : Acquisition start profile No. in the batch(VT\_UI4)  
 byGetProfCnt : Number of profiles to read (VT\_UI1)  
 Return value : OUT measurement data (VT\_ARRAY|VT\_VARIANT)

The data is stored as shown in Table 3-11.

**Table 3-11 GetStorageBatchProfile data**

Index1	Index2	Content	Data type
0	0	Free storage	VT_UI1
	1	Program No. of the target storage plane	VT_UI1
	2	Storage object	VT_UI1
	3	Number of storage points	VT_UI4
	4	Batch number read this time	VT_UI4
	5	Number of profiles in the batch read this time	VT_UI4
	6	What is the oldest profile in the batch you read?	VT_UI4
	7	Number of profiles read this time	VT_UI1
	8	Standard time	VT_DATE
	9	Number of profiles stored in one unit of data	VT_UI1
	10	ON/OFF of profile compression (time axis)	VT_UI1
	11	Number of data points in one profile	VT_UI2
	12	First X coordinate	VT_I4
	13	Spacing of the data points in the X direction	VT_I4
1	0~	Profile data The acquired profile data is stored in order. For information on the number of data units and the order of data store, refer to Section 9.2.9.6 and 9.2.9.7 of the LJ-V7000 Series Communication Library Reference Manual.	VT_UI4
2	0~	OUT measurement results for each profile The data of the following number of elements is stored <16 OUT's worth of data (in the same data order as in Table 3-3)><the number of data points read>	VT_UI1 or VT_R4

3	-	Value counted in units of 10 ms from the reference time of each data	VT_UI4
4	0~47	OUT measurement results for batches It is stored in the same data order as Table 3-3.	VT_UI1 or VT_R4

**Example**


---

```

Dim retVal As Variant
Dim infoVal As Variant
Dim profVal As Variant
Dim data As Integer
retVal = caoCtrl.Execute("GetStorageProfile", Array(0, 0, 10))
infoVal = retVal(0)
profVal = retVal(1)
data = profVal(infoVal(9) * infoVal(11) * 9 + 1) 'Get 2nd data of 10th profile

```

---

**3.6. High-speed data communications****3.6.1. CaoController::Execute("StartHighSpeedDataCommunication") command**

Start high-speed data communication. This command executes the following three commands in the LJ-V7000 Communication Library.

1. HighSpeedDataEthernetCommunicationInitialize
2. PreStartHighSpeedDataCommunication
3. StartHighSpeedDataCommunication

**Syntax** StartHighSpeedDataCommunication (<wHighSpeedPortNo >,< dwProfileCnt >)

wHighSpeedPortNo : Port number (VT\_UI2) used for high-speed data communication

dwProfileCnt : Number of Profiles to be sent collectively (VT\_UI4)

If you receive the specified number of profiles, OnMessage events occur.

Return value : None

**Example**


---

```

caoCtrl.Execute "StartHighSpeedDataCommunication", Array(24692, 10)

```

---

**3.6.2. CaoController::Execute("StopHighSpeedDataCommunication") command**

Start high-speed data communication. This command executes the following two commands in the LJ-V7000 Communication Library.

1. StopHighSpeedDataCommunication
2. HighSpeedDataCommunicationFinalize

**Syntax** StopHighSpeedDataCommunication ()

Argument : None

Return value : None

**Example**

---

```
caoCtrl.Execute "StopHighSpeedDataCommunication"
```

---