KEYENCE

376GB

LJ-V7000 Series Communication Library

Reference Manual

Please read this manual before use.
After reading this manual, store it in a safe place where it can be used at any time.

Contents

1 Software License Agreement

NOTICE TO USER: PLEASE READ THIS SOFTWARE LICENSE AGREEMENT (THIS “AGREEMENT”) CAREFULLY. BY USING ALL OR ANY PORTION OF THIS “SOFTWARE”, YOU ARE AGREEING TO BE BOUND BY ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO ANY TERMS OF THIS AGREEMENT, DO NOT USE THIS SOFTWARE.

1. Grant of License.

Conditioned upon compliance with all of the terms and conditions of this Agreement, KEYENCE grants you a nonexclusive and nontransferable license.

2. Restrictions

Except for installation of updates or new functions provided by KEYENCE, you may not modify or add any function to this Software.

a) You may not reverse engineer, decompile or disassemble this Software.

b) You may not create derivative works based on this Software.

c) Other than expressly stated by KEYENCE, you may not resell, retransfer, rent or otherwise redistribute this Software to any third parties. However, you may redistribute this Software with the application that you developed using this Software.

3. Intellectual Property Rights.

Except as expressly stated herein, KEYENCE reserves all right, title and interest in this Software, and all associated copyrights, trademarks, and other intellectual property rights therein.

4. Disclaimer.

Keyence is licensing this Software to you “AS IS” and without any warranty of any kind. In no event will KEYENCE or its suppliers be liable to you for any damages, claims, costs or any lost profits caused by using this Software.

5. Support

KEYNCE shall not provide technical support in accordance with this Software including the use of this Software.

6. Termination.

6.1 Your license under this Agreement will terminate automatically if you destroy this Software and the copy of this Software in your possession or voluntarily return this Software to us.

6.2 Your license under this Agreement will terminate automatically without any notice from KEYENCE if you fail to comply with any of the terms and conditions of this Agreement. Promptly upon termination, you shall cease all use of this Software and destroy all copies, full or partial, of this Software in your possession or control.

6.3 You will compensate KEYENCE for costs or any lost profits caused by your violation or breach of any term of this Agreement.

7. Governing Law.

This Agreement will be governed by and construed in accordance with the substantive laws of Japan without regards to the principles of conflicts of law.

2 Introduction

The LJ-V7000 Series communication library provides a communication interface for controlling the LJ-V7000 Series from a user application (Win32 DLL). For specific ways to use the communication library, refer to the sample program.

3 Operating Environment

OS	Windows 7 (Home Premium/Professional/Ultimate) Windows Vista (Home Basic/Home Premium/Business/Ultimate) Windows XP (SP2 or later) (Home Edition/Professional Edition)
CPU	Core i3 2.3 GHz or faster
Memory	2 GB or more
Secondary cache memory	2 MB or more
Free drive space	10 GB or more
Interface	A PC equipped with either of the interfaces below. USB 2.0/1.1 *1, Ethernet 1000BASE-T/100BASE-TX *2

*1 Operation is not guaranteed with connections via a USB hub

*2 Operation is not guaranteed with connections to a LAN or via a router

3.1 Execution environment

This section describes the necessary environment to execute applications that use the LJ-V7000 Series communication library.

3.1.1 Microsoft C runtime library

The Microsoft C runtime library is required for the DLL to operate.
Run vcredist_x86.exe included on the installation media to install the library.

3.1.2 Microsoft .NET Framework

The Microsoft .NET Framework is required to run the sample application.
Run NetFx20SP2_x86.exe included on the installation media to install the library.

4 USB Driver

Install and use LJ-Navigator2 for the USB driver.

5 File Structure

LJV7_IF.dll	The DLL.
LJV7_IF.lib	The import library for LJV7_IF.dll.
LJV7_ErrorCode.h	The header file that defines the error codes.
LJV7_IF.h	The header file that defines the LJV7_IF.dll interface.
Source	The folder for the sample source code. The source code for the sample program created in C#.

6 Incorporating the Library

6.1 File structure

The files required at execution are listed below.
Place these folders/files in the same folder as the executable file.

- LJV7_IF.dll

6.2 Linking

6.2.1 C++

6.2.1.1 Linking

The library can be linked implicitly or explicitly.
To implicitly link the library, link with "LJV7_IF.lib".
* "LJV7_IF.lib" was built with Visual C++ 2008 SP1.

6.2.1.2 Include files

Include the following header files in the necessary source files.

- LJV7_IF.h
- LJV7_ErrorCode.h

6.2.2 C#/VB.NET

Call each interface using the DllImport attribute.
When passing a structure as an interface argument, specify the StructLayout attribute and pass a structure of the same memory structure as the DLL.
For details, refer to the NativeMethods class (NativeMethods.cs) in the sample.
The processing to call each function has been implemented.

7 Types

In this document, variable types are described according to the following definitions.

CHAR	Signed 8-bit integer
BYTE	Unsigned 8-bit integer
SHORT	Signed 16-bit integer
WORD	Unsigned 16-bit integer
LONG	Signed 32-bit integer
DWORD	Unsigned 32-bit integer
FLOAT	Single precision floating point number (32 bits)
DOUBLE	Double precision floating point number (64 bits)

8 Constant, Structure Definitions

8.1 Constant definitions

Name	Setting value storage level designation

Definition	Typedef enum {
	    LJV7IF_SETTING_DEPTH_WRITE = 0x00, // Write settings area
	    LJV7IF_SETTING_DEPTH_RUNNING = 0x01, // Running settings area
	    LJV7IF_SETTING_DEPTH_SAVE = 0x02 // Save area
	} LJV7IF_SETTING_DEPTH;

Description	This enumeration designates the operation target level in functions that modify or read settings. For details on the setting value storage level, refer to "9.2.9.3 Write processing for settings".
Comment	The controller retains three sets of settings data. Those levels are used in the applications below. **Write settings area** Settings that do not affect operation. In order to not allow an error in controller operations from inconsistencies in settings that occur temporarily when changing multiple settings, the operation of the controller can be changed without causing an error by reflecting the settings from this area to the running settings area after writing the settings to this area. **Running settings area** The settings the controller is using in its operation. When the controller starts, this area is initialized with the settings in the save area. **Save area** The settings that are saved even when the controller's power is turned off.

Name	Initialization target setting item designation

Definition	Typedef enum {
	    LJV7IF_INIT_SETTING_TARGET_PRG0 = 0x00, // Program 0
	    LJV7IF_INIT_SETTING_TARGET_PRG1 = 0x01, // Program 1
	    LJV7IF_INIT_SETTING_TARGET_PRG2 = 0x02, // Program 2
	    LJV7IF_INIT_SETTING_TARGET_PRG3 = 0x03, // Program 3
	    LJV7IF_INIT_SETTING_TARGET_PRG4 = 0x04, // Program 4
	    LJV7IF_INIT_SETTING_TARGET_PRG5 = 0x05, // Program 5
	    LJV7IF_INIT_SETTING_TARGET_PRG6 = 0x06, // Program 6
	    LJV7IF_INIT_SETTING_TARGET_PRG7 = 0x07, // Program 7
	    LJV7IF_INIT_SETTING_TARGET_PRG8 = 0x08, // Program 8
	    LJV7IF_INIT_SETTING_TARGET_PRG9 = 0x09, // Program 9
	    LJV7IF_INIT_SETTING_TARGET_PRG10 = 0x0A, // Program 10
	    LJV7IF_INIT_SETTING_TARGET_PRG11 = 0x0B, // Program 11
	    LJV7IF_INIT_SETTING_TARGET_PRG12 = 0x0C, // Program 12
	    LJV7IF_INIT_SETTING_TARGET_PRG13 = 0x0D, // Program 13
	    LJV7IF_INIT_SETTING_TARGET_PRG14 = 0x0E, // Program 14
	    LJV7IF_INIT_SETTING_TARGET_PRG15 = 0x0F, // Program 15
	} LJV7IF_INIT_SETTING_TARGET;

Description	This enumeration designates which settings to initialize in settings initialization function.
Comment	-

Name	Definition that indicates the validity of a measurement value

Definition	Typedef enum {
	    LJV7IF_MEASURE_DATA_INFO_VALID = 0x00, // Normal measurement data
	    LJV7IF_MEASURE_DATA_INFO_ALARM = 0x01, // Measurement alarm data
	    LJV7IF_MEASURE_DATA_INFO_WAIT = 0x02 // Judgment wait data
	} LJV7IF_MEASURE_DATA_INFO;

Description	This enumeration indicates the validity or invalidity of the measurement value.
Comment	-

Name	Definition that indicates the tolerance judgment result of the measurement value

Definition	Typedef enum {
	    LJV7IF_JUDGE_RESULT _HI = 0x01, // HI
	    LJV7IF_JUDGE_RESULT _GO = 0x02, // GO
	    LJV7IF_JUDGE_RESULT _LO = 0x04 // LO
	} LJV7IF_JUDGE_RESULT;

Description	This enumeration indicates the tolerance judgment result for the measurement value in bit units.
Comment	If the measurement value is measurement alarm data, the judgment result is 0x05 (both HI and LO bits are 1).

Name	Get profile target buffer designation

Definition	Typedef enum {
	    LJV7IF_PROFILE_BANK_ACTIVE = 0x00, // Active surface
	    LJV7IF_PROFILE_BANK_INACTIVE = 0x01 // Inactive surface
	} LJV7IF_PROFILE_BANK;

Description	When the memory allocation is "double buffer" in the get profile command, this enumeration designates which surface to get the profiles from.
Comment	"Active surface" refers to the surface of the buffer that profile data is being written onto. For further details, refer to "9.2.9.2 Internal memory".

Name	Get profile position specification method designation (batch measurement: off)

Definition	Typedef enum {
	    LJV7IF_PROFILE_POS_CURRENT = 0x00, // From current
	    LJV7IF_PROFILE_POS_OLDEST = 0x01, // From oldest
	    LJV7IF_PROFILE_POS_SPEC = 0x02, // Specify position
	} LJV7IF_PROFILE_POS;

Description	In the get profile command, this enumeration indicates the specification method for the profiles to get out of the profile data retained in the controller. In get profile, the profiles are stored from oldest to newest. **From current** Gets the current profiles. The end of the acquired profiles becomes the current profile. **From oldest** Gets the oldest profile. The head of the acquired profiles becomes the oldest profile. **Specify position** Gets the specified number of profiles from the specified profile position. The head of the acquired profiles becomes the profiles at the specified position.
Comment	For the specified number of profiles, refer to the individual structure definitions.

Name	Get profile batch data position specification method designation (batch measurement: on)

Definition	Typedef enum {
	LJV7IF_BATCH_POS_CURRENT = 0x00, // From current
	LJV7IF_BATCH_POS_SPEC = 0x02, // Specify position
	LJV7IF_BATCH_POS_COMMITED = 0x03, // From current after batch commitment
	LJV7IF_BATCH_POS_CURRENT_ONLY = 0x04 // Current only
	} LJV7IF_BATCH_POS;

Description	In the get batch profile command, this enumeration indicates the specification method for the profiles to get in what batch out of the batch data retained in the controller. In get profile, the profiles are stored from oldest to newest. **From current** Gets the profiles in the current batch data. **Specify position** Gets the profiles in the batch data with the specified number. **From current after batch commitment** Gets the profiles in the current batch data after commitment. **Current only** Gets one current profile in the current batch data.
Comment	For the specified number of profiles, refer to the individual structure definitions.

Name	Number of OUT settings
Definition	Const static LONG LJV7IF_OUT_COUNT = 16;
Description	This constant indicates the number of OUT settings.
Comment	-

Name	Number of simultaneously connectable controllers
Definition	Const static LONG LJV7IF_DEVICE_COUNT = 6;
Description	This constant is the upper limit for the number of controllers that can simultaneously communicate.
Comment	-

8.2 Structure definitions

Name	Ethernet settings structure

Definition	Typedef struct {
	BYTE abyIpAddress[4];
	WORD wPortNo;
	BYTE reserve[2];
	} LJV7IF_ETHERNET_CONFIG;

Description	This structure contains the settings passed during an Ethernet communication connection. **abyIpAddress** The IP address of the controller to connect to. For 192.168.0.1: Set abyIpAddress[0]=192, abyIpAddress[1]=168, and so on. **wPortNo(in)** The port number of the controller to connect to.
Comment	-

Name	Date/time structure

Definition	Typedef struct {
	BYTE byYear;
	BYTE byMonth;
	BYTE byDay;
	BYTE byHour;
	BYTE byMinute;
	BYTE bySecond;
	BYTE reserve[2];
	} LJV7IF_TIME;

Description	The date/time for the controller. byYear Year. Set from 0 to 99, which means 2000 to 2099. byMonth Month.1 to 12. byDay Day.1 to 31. byHour Hour.0 to 23. byMinute Minute.0 to 59. bySecond Second.0 to 59.
Comment	-

Name	Setting item designation structure

Definition	Typedef struct {
	BYTE byType;
	BYTE byCategory;
	BYTE byItem;
	BYTE reserve;
	BYTE byTarget1;
	BYTE byTarget2;
	BYTE byTarget3;
	BYTE byTarget4;
	} LJV7IF_TARGET_SETTING;

Description	Information for specifying target setting items. **byType, byCategory, byItem** When modifying or reading a setting, these variables are used to specify the target setting item. **byTarget1, byTarget2, byTarget3, byTarget4** These variables are used when specifying further details for the setting item. For example, when configuring OUT measurement mode, these are used to specify the OUT number.
Comment	For details, see the appendix.

Name	Measurement results structure

Definition	Typedef struct {
	BYTE byDataInfo;
	BYTE byJudge;
	BYTE reserve[2];
	FLOAT fValue;
	} LJV7IF_MEASURE_DATA;

Description	Measurement value and judgment results. **byDatainfo** This variable indicates whether or not the measurement value (fValue) is valid, and if it is not a valid value, what kind of data it is. See LJV7IF_MEASURE_DATA_INFO. **byJudge** Tolerance judgment result. See LJV7IF_JUDGE_RESULT. **fValue** Measurement value. The unit used for measurement values is the minimum display unit set for Target OUT in program settings. When the minimum display unit is 1 mm to 0.001 mm, the measurement value unit is [mm]. When 1 um to 0.1 um, the measurement value unit is [um]. The unit for sectional areas is mm^2, and the unit for angles is deg. When not a valid value, a large negative value is stored (-10^{10}).
Comment	-

Name	Profile information structure

Definition	Typedef struct {
	    BYTE byProfileCnt;
	    BYTE byEnvelope;
	    BYTE reserve[2];
	    WORD wProfDataCnt;
	    BYTE reserve2[2];
	    LONG IXStart;
	    LONG IXPitch;
	} LJV7IF_PROFILE_INFO;

Description	Information related to the profile. **byProfileCnt** Wheter dicates the amount of profile data stored. (When 2 head/combine (wide) is off, 2 profile data units is stored, otherwise 1 profile data unit is stored.) **byEnvelope** Whether profile compression (time axis) is on. 0: off, 1: on. **wProfDataCnt** Profile data count (initial setting: 800). **IXStart** 1st point X coordinate. **IXPitch** Profile data X direction interval.
Comment	IXStart and IXPitch are stored in 0.01 µm units.

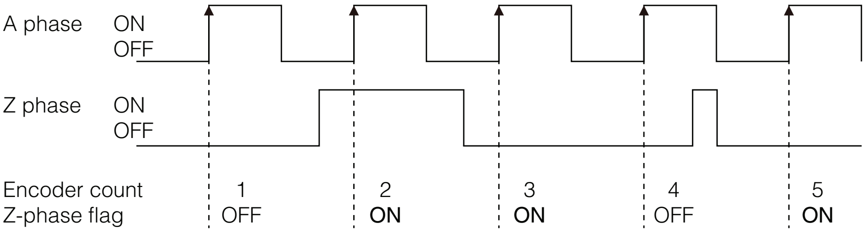
Name	Profile header information structure

Definition	Typedef struct {
	    DWORD reserve;
	    DWORD dwTriggerCnt;
	    DWORD dwEncoderCnt;
	    DWORD reserve2[3];
	} LJV7IF_PROFILE_HEADER;

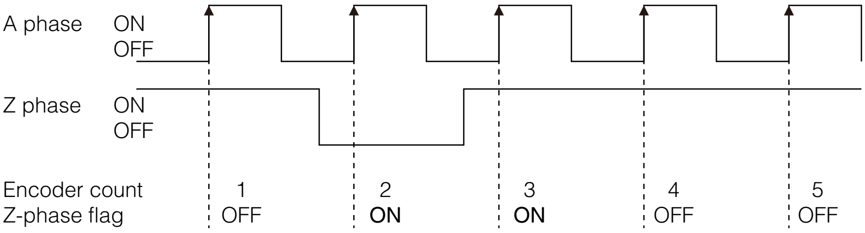
Description	The header information added to the profile. **reserve** 7th bit: Indicates whether the encoder's Z phase has been entered. (*) MSB ↓ LSB 31 ··· 7 6 5 4 3 2 1 0 **dwTriggerCnt** Indicates which number trigger from the start of measurements this profile is. (Trigger counter) **dwEncoderCnt** The encoder count when the trigger was issued. (Encoder counter)
Comment	Other than when settings are modified or the program is switched, the trigger counter and the encoder counter are reset at the following times. • When the memory is cleared in high-speed mode (profile only) • When laser emission stops and is restarted with the LASER_OFF terminal • When laser emission is allowed after it was prohibited with the REMOTE terminal

*: About the Z-phase flag
This flag can be used when the controller is version 3.0 or later.
This flag is turned ON when Z-phase ON input is received during the period between the previous trigger input (or the start of measurement if there was no previous trigger input) and the current trigger input.

Example: Single phase 1x multiplier encoder trigger with no skipping



Note: When the Z-phase input uses a negative logic encoder, set the TRG minimum input time, which is a common measurement setting, to 7 μs. With negative logic, the Z-phase flag turns ON as shown in the following figure.



Name	Profile footer information structure

Definition	Typedef struct {
	DWORD reserve;
	} LJV7IF_PROFILE_FOOTER;

Description	The footer information added to the profile. None (reserved only).
Comment	-

Name	High-speed mode get profile request structure (batch measurement: off)

Definition	Typedef struct {
	BYTE byTargetBank;
	BYTE byPosMode;
	BYTE reserve[2];
	DWORD dwGetProfNo;
	BYTE byGetProfCnt;
	BYTE byErase;
	BYTE reserve[2];
	} LJV7IF_GET_PROFILE_REQ;

Description	The get profile designation information when the operation mode is "high-speed (profile only)" and batch measurements are off in the get profile command. **byTargetBank** Specifies whether to get the profiles from the active surface or whether to get the profiles from the inactive surface. See LJV7IF_PROFILE_BANK. (P.9) **byPosMode** Specifies the get profile position specification method. See LJV7IF_PROFILE_POS. **dwGetProfNo** When byPosMode is LJV7IF_PROFILE_POS_SPEC, specifies the profile number for the profile to get. **byGetProfCnt** The number of profiles to read. **byErase** Specifies whether or not to erase the profile data that was read and the profile data older than that. 0: Do not erase, 1: erase
Comment	If the communication buffer is insufficient, the number of profiles specified by byGetProfCnt may not be acquired. In this situation, the maximum number of profiles that can be acquired is returned.

Name	High-speed mode get profile request structure (batch measurement: on)

Definition	Typedef struct {
	BYTE byTargetBank;
	BYTE byPosMode;
	BYTE reserve[2];
	DWORD dwGetBatchNo;
	DWORD dwGetProfNo;
	BYTE byGetProfCnt;
	BYTE byErase;
	BYTE reserve[2];
	} LJV7IF_GET_BATCH_PROFILE_REQ;

Description	The get profile designation information when the operation mode is "high-speed (profile only)" and batch measurements are on in the get profile command. **byTargetBank** Specifies whether to get the profiles from the active surface or whether to get the profiles from the inactive surface. See LJV7IF_PROFILE_BANK. (P.9) **byPosMode** Specifies the get profile position specification method. See LJV7IF_BATCH_POS. **dwGetBatchNo** When byPosMode is LJV7IF_BATCH_POS_SPEC, specifies the batch number for the profile to get. **dwGetProfNo** Specifies the profile number to start getting profiles from in the specified batch number. **byGetProfCnt** The number of profiles to read. **byErase** Specifies whether or not to erase the batch data that was read and the batch data older than that. 0: Do not erase, 1: erase
Comment	If the communication buffer is insufficient, the number of profiles specified by byGetProfCnt may not be acquired. In this situation, the maximum number of profiles that can be acquired is returned.

Name	Advanced mode get profile request structure (batch measurement: on)

Definition	Typedef struct {
	    BYTE byPosMode;
	    BYTE reserve[3];
	    DWORD dwGetBatchNo;
	    DWORD dwGetProfNo;
	    BYTE byGetProfCnt;
	    BYTE reserve[3];
	} LJV7IF_GET_BATCH_PROFILE_ADVANCE_REQ;

Description	The get profile designation information when the operation mode is "advanced (with OUT measurement)" and batch measurements are on in the get batch profile command. **byPosMode** Specifies the get profile position specification method. See LJV7IF_BATCH_POS. **dwGetBatchNo** When byPosMode is LJV7IF_BATCH_POS_SPEC, specifies the batch number for the profiles to get. **dwGetProfNo** Specifies the profile number for the profiles to get. **byGetProfCnt** The number of profiles to read.
Comment	If the communication buffer is insufficient, the number of profiles specified by byGetProfCnt may not be acquired. In this situation, the maximum number of profiles that can be acquired is returned.

Name	High-speed mode get profile response structure (batch measurement: off)

Definition	Typedef struct {
	    DWORD dwCurrentProfNo;
	    DWORD dwOldestProfNo;
	    DWORD dwGetTopProfNo;
	    BYTE byGetProfCnt;
	    BYTE reserve[3];
	} LJV7IF_GET_PROFILE_RSP;

Description	The profile information returned for the get profiles command when the operation mode is "high-speed (profile only)" and batch measurements are off. **dwCurrentProfNo** The profile number at the current point in time. **dwOldestProfNo** The profile number for the oldest profile held by the controller. **dwGetTopProfNo** The profile number for the oldest profile out of those that were read this time. **byGetProfCnt** The number of profiles that were read this time.
Comment	-

Name	High-speed mode get profile response structure (batch measurement: on)

Definition	Typedef struct {
	    DWORD dwCurrentBatchNo;
	    DWORD dwCurrentBatchProfCnt;
	    DWORD dwOldestBatchNo;
	    DWORD dwOldestBatchProfCnt;
	    DWORD dwGetBatchNo;
	    DWORD dwGetBatchProfCnt;
	    DWORD dwGetBatchTopProfNo;
	    BYTE byGetProfCnt;
	    BYTE byCurrentBatchCommited;
	    BYTE reserve[2];
	} LJV7IF_GET_BATCH_PROFILE_RSP;

Description	The profile information returned for the get profiles command when the operation mode is "high-speed (profile only)" and batch measurements are on. **dwCurrentBatchNo** The batch number at the current point in time. **dwCurrentBatchProfCnt** The number of profiles in the newest batch. **dwOldestBatchNo** The batch number for the oldest batch held by the controller. **dwOldestBatchProfCnt** The number of profiles in the oldest batch held by the controller. **dwGetBatchNo** The batch number that was read this time. **dwGetBatchProfCnt** The number of profiles in the batch that was read this time. **dwGetBatchTopProfNo** Indicates what number profile in the batch is the oldest profile out of the profiles that were read this time. **byGetProfCnt** The number of profiles that were read this time. **byCurrentBatchCommited** Indicates if the batch measurements for the newest batch number has finished. 0: Not finished, 1: finished
Comment	-

Name	Advanced mode get profile response structure (batch measurement: on)

Definition	Typedef struct {
	DWORD dwGetBatchNo;
	DWORD dwGetBatchProfCnt;
	DWORD dwGetBatchTopProfNo;
	BYTE byGetProfCnt;
	BYTE reserve[3];
	} LJV7IF_GET_BATCH_PROFILE_ADVANCE_RSP;

Description	The profile information returned for the get profiles command when the operation mode is "advanced mode (with OUT measurement)" and batch measurements are on. **dwGetBatchNo** The batch number that was read this time. **dwGetBatchProfCnt** The number of profiles in the batch that was read this time. **dwGetBatchTopProfNo** Indicates what number profile in the batch is the oldest profile out of the profiles that were read this time. **byReadProfCnt** The number of profiles that were read this time.
Comment	-

Name	Get storage status request structure

Definition	Typedef struct {
	DWORD dwReadArea;
	}LJV7IF_GET_ STRAGE_STATUS_REQ;

Description	Get target designation information in the get storage status command. **dwReadArea** The storage surface to read. • When the memory allocation setting is "double buffer" 0: Active surface, 1: Surface A, 2: Surface B • When the memory allocation setting is "entire area (overwrite)", fixed as 1 • When the memory allocation setting is "entire area (do not overwrite)" 0: Active surface, surface specification (1 to 999)
Comment	"Active surface" refers to the surface of the buffer that profile data is being written onto. For further details, refer to "9.2.9.2 Internal memory".

Name	Get storage status response structure

Definition	Typedef struct {
	DWORD dwSurfaceCnt;
	DWORD dwActiveSurface;
	} LJV7IF_GET_STRAGE_STATUS_RSP;

Description	The storage status information returned for the get storage status command. **dwSurfaceCnt** Storage surface count **dwActiveSurface** The active storage surface. When the active program has storage off, 0.
Comment	For details about "Storage surface", refer to "9.2.9.2.2 For operation mode: advanced (with OUT measurement)".

Name	Storage information structure

Definition	Typedef struct {
	BYTE byStatus;
	BYTE byProgramNo;
	BYTE byTarget;
	BYTE reserve[5];
	DWORD dwStorageCnt;
	} LJV7IF_ STORAGE_INFO;

Description	Information related to the storage status. **byStatus** Storage status. 0: Empty (Takes on this value when the target surface has not operated even once in a program with storage on) 1: Storing (only the active storage surface can be 1) 2: Storage complete **byProgramNo** The program number for the relevant storage surface. **byTarget** Storage target.0: Data storage, 2: profile storage, 3: batch profile storage. However, when batch measurements are on and profile compression (time axis) is on, 2: profile storage is stored. **dwStorageCnt** Storage count (batch count when batch is on)
Comment	For details about "Storage surface", refer to "9.2.9.2.2 For operation mode: advanced (with OUT measurement)".

Name	Get storage data request structure

Definition	Typedef struct {
	BYTE reserve[4];
	DWORD dwSurface;
	DWORD dwStartNo;
	DWORD dwDataCnt;
	} LJV7IF_GET_STORAGE_REQ;

Description	The get data designation information in the get data storage data command and the get profile storage data command. **dwSurface** Storage surface to read. **dwStartNo** The data number to start reading. **dwDataCnt** The number of items to read.
Comment	For details about "Storage surface", refer to "9.2.9.2.2 For operation mode: advanced (with OUT measurement)".

Name	Get batch profile storage request structure

Definition	Typedef struct {
	BYTE reserve[4];
	DWORD dwSurface;
	DWORD dwGetBatchNo;
	DWORD dwGetBatchTopProfNo;
	BYTE byGetProfCnt;
	BYTE reserved[3];
	} LJV7IF_GET_BATCH_PROFILE_STORAGE_REQ;

Description	Get data designation information in the get batch storage data command. **dwSurface** Storage surface to read. **dwGetBatchNo** Batch number to read. **dwGetBatchTopProfNo** Specifies from what profile number in the batch to get the data. **byGetProfCnt** The number of profiles to read.
Comment	For details about "Storage surface", refer to "9.2.9.2.2 For operation mode: advanced (with OUT measurement)".

Name	Get storage data response structure

Definition	Typedef struct {
	DWORD dwStartNo;
	DWORD dwDataCnt;
	LJV7IF_TIME stBaseTime;
	} LJV7IF_GET_STORAGE_RSP;

Description	The get data information returned for the get storage data command and the get profile storage command. **dwStartNo** The data number to start reading. **dwDataCnt** The number of items to read. **stBaseTime** Base time.
Comment	For details about base time, refer to "9.2.9.10 Time data added to storage data".

Name	Get batch profile storage response structure

Definition	Typedef struct {
	DWORD dwGetBatchNo;
	DWORD dwGetBatchProfCnt;
	DWORD dwGetBatchTopProfNo;
	BYTE byGetProfCnt;
	BYTE reserve[3];
	LJV7IF_TIME stBaseTime;
	} LJV7IF_GET_BATCH_PROFILE_STORAGE_RSP;

Description	The get data information returned for the get batch profile storage command. **dwGetBatchNo** The batch number that was read this time. **dwGetBatchProfCnt** The number of profiles in the batch that was read this time. **dwGetBatchTopProfNo** Indicates what number profile in the batch is the oldest profile out of the profiles that were read this time. **byGetProfCnt** The number of profiles that were read this time. **stBaseTime** Base time.
Comment	-

Name	High-speed communication prep start request structure

Definition	Typedef struct {
	BYTE bySendPos;
	BYTE reserve[3];
	} LJV7IF_HIGH_SPEED_PRE_START_REQ;

Description	High-speed communication start preparation request command **bySendPos** Send start position. 0: From previous send complete position (from oldest data if 1st time), 1: From oldest data (reacquire), 2: From next data
Comment	-

8.3 Callback function interface definition

Format	void (*pCallBack)(BYTE* pBuffer, DWORD dwSize, DWORD dwCount, DWORD dwNotify, DWORD dwUser);
Parameters	**pBuffer(in)** A pointer to the buffer that stores the profile data. The profile data is stored in this buffer with "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER" as a single unit of profile data, and only the number of profiles that could be acquired (dwCount) are returned. **dwSize(in)** The size in BYTEs per single unit of the profile "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER" contained in pBuffer. **dwCount(in)** The number of profiles stored in pBuffer. **dwNotify(in)** Notification of an interruption in high-speed communication or a break in batch measurements. For details, see "8.3.1 Supplement". **dwUser(in)** User information set when high-speed communication was initialized.
Return value	None
Explanation	When using the high-speed communication function, this callback function is called when data is received and when there is a change in the communication state. This callback function is called from a thread other than the main thread. Take care to only implement storing profile data in a thread save buffer in the callback function. As the thread used to call the callback function is the same as the thread used to receive data, the processing time of the callback function affects the speed at which data is received, and may stop communication from being performed properly in some environments. Refer to the sample program for details. Profile data is stored in 0.01 μm units.

8.3.1 Supplement

8.3.1.1 dwNotify parameter

This section describes the dwNotify parameter used in the callback function.

In high-speed communication, the callback function is called when any number of events occur, in addition to when profile data is received. These events can be checked with the dwNotify parameter.

dwNotify = 0: Indicates that profile data is being communicated correctly. Refer to the table below for values other than 0.

		○: May be returned. ×: Will not be returned.	Batch off	Batch on
LSB	0	Continuous send was stopped (stop by command)	○	○
	1	Continuous send was stopped (automatic stop) [*1]	○	○
	2	Continuous send was stopped (automatic stop) [*2]	○	○
	3	Reserved		
	4	Reserved		
	5	Reserved		
	6	Reserved		
	7	Reserved		
	8	Send interrupted by clear memory	○	○
	9	Reserved		
	10	Reserved		
	11	Reserved		
	12	Reserved		
	13	Reserved		
	14	Reserved		
	15	Reserved		
	16	Finished sending the batch measurement amount of data [*3]	×	○
	17	Reserved		
	18	Reserved		
	19	Reserved		
	20	Reserved		
	21	Reserved		
	22	Reserved		
	23	Reserved		
	24	Reserved		
	25	Reserved		
	26	Reserved		
	27	Reserved		
	28	Reserved		
	29	Reserved		
	30	Reserved		
MSB	31	Reserved		

[*1] The setting was modified

[*2] The program was switched

[*3] However, when REMOTE OFF/LASER OFF is turned ON, profiles up to the one currently being sent are forwarded, so some of the batch measurement data may not be forwarded.

Bit 0 to 2 and bit 8 indicate that continuous send was stopped.

To restart continuous send, start the high-speed data communication in the following order: "Finalize high-speed data communication" → "Disconnect ethernet communication" → "Start ethernet communication" → "Initialize ethernet high-speed communication" → "Request preparation for Ethernet high-speed data communication".

Bit 16 is only valid when batch measurements are on.

When batch measurements are on, the batch measurements can be ended even when the configured batch count is not fulfilled. Therefore, the callback function is notified with this bit on in order to determine the break in batch data.

9 Functions

9.1 Function list

9.1.1 Operations for the DLL

These functions are processed normally even when the controller is in the system error state.

Function name	Overview
LJV7IF_Initialize	Initializes the DLL
LJV7IF_Finalize	Performs the termination processing for the DLL
LJV7IF_GetVersion	Gets the DLL version

9.1.2 Establish/disconnect the communication path with the controller

These functions are processed normally even when the controller is in the system error state.

Function name	Overview
LJV7IF_UsbOpen	Establishes a USB connection
LJV7IF_EthernetOpen	Establishes an Ethernet connection
LJV7IF_CommClose	Disconnects the connection (both USB and Ethernet)

9.1.3 System control

Excluding LJV7IF_RetrunToFactorySetting, these functions are processed normally even when the controller is in the system error state. LJV7IF_RetrunToFactorySetting may fail in the system error state (when a head is not connected, etc.).

Function name	Overview
LJV7IF_RebootController	Reboots the controller
LJV7IF_RetrunToFactorySetting	Returns the controller to the factory settings
LJV7IF_GetError	Gets the controller system error information
LJV7IF_ClearError	Clears the controller system error

9.1.4 Measurement control

Processing for these functions fails when the controller is in the system error state.

Function name	Overview
LJV7IF_Trigger	Issues a trigger
LJV7IF_StartMeasure	Starts measurements
LJV7IF_StopMeasure	Stops measurements
LJV7IF_AutoZero	Issues auto zero
LJV7IF_Timing	Issues timing
LJV7IF_Reset	Issues a reset
LJV7IF_ClearMemory	Clears the internal memory

9.1.5 Functions related to modifying or reading settings

Processing for these functions fails when the controller is in the system error state.

Function name	Overview
LJV7IF_SetSetting	Sends a setting to the controller
LJV7IF_GetSetting	Gets a setting from the controller
LJV7IF_InitializeSetting	Initializes a controller setting
LJV7IF_ReflectSetting	Reflects the contents of the write settings area in the running settings area and the save area
LJV7IF_RewriteTemporarySetting	Overwrites the contents of the write settings area with the settings in the running settings area and the save area
LJV7IF_CheckMemoryAccess	Checks whether or not settings are being saved to the save area
LJV7IF_SetTime	Sets the date/time for the controller
LJV7IF_GetTime	Gets the date/time for the controller
LJV7IF_ChangeActiveProgram	Changes the active program number
LJV7IF_GetActiveProgram	Gets the active program number

9.1.6 Acquiring measurement results

Processing for these functions fails when the controller is in the system error state.

Function name	Overview
LJV7IF_GetMeasurementValue	Gets measurement values
LJV7IF_GetProfile	Gets profiles when the operation mode is "high-speed (profile only)"
LJV7IF_GetBatchProfile	Gets profiles when the operation mode is "high-speed (profile only)" * Use LJV7IF_GetProfile when Compression (time axis) is on.
LJV7IF_GetProfileAdvance	Gets profiles when the operation mode is "advanced (with OUT measurement)"
LJV7IF_GetBatchProfileAdvance	Gets profiles when the operation mode is "advanced (with OUT measurement)" * Use LJV7IF_GetProfileAdvance when Compression (time axis) is on.

9.1.7 Storage function related

Processing for these functions fails when the controller is in the system error state.

Function name	Overview
LJV7IF_StartStorage	Starts storage
LJV7IF_StopStorage	Stops storage
LJV7IF_GetStorageStatus	Gets the storage status
LJV7IF_GetStorageData	Gets the stored data when the storage target is "OUT value"
LJV7IF_GetStorageProfile	Gets the stored profiles when the storage target is "Profiles"
LJV7IF_GetStorageBatchProfile	Gets the stored profiles when the storage target is "Profiles" * Use LJV7IF_GetStorageProfile when Compression (time axis) is on.

9.1.8 High-speed data communication related

Processing for these functions fails when the controller is in the system error state.

Function name	Overview
LJV7IF_HighSpeedDataUSBCommunicationInitalize	Performs the initialization required for high-speed data communication (USB)
LJV7IF_HighSpeedDataEthernetCommunicationInitalize	Performs the initialization required for high-speed data communication (Ethernet)
LJV7IF_PreStartHighSpeedDataCommunication	Requests preparation before starting high-speed data communication
LJV7IF_StartHighSpeedDataCommunication	Starts high-speed data communication
LJV7IF_StopHighSpeedDataCommunication	Stops high-speed data communication
LJV7IF_HighSpeedDataCommunicationFinalize	Performs high-speed data communication termination processing

9.2 Function reference

The type of the return value for the functions where there is a possibility of an error occurring is LONG. Normally, 0 (ERR_NONE) is returned, and the return code is expressed in the lower 2 bytes (the upper 2 bytes are reserved).
For the common return codes for functions, see "10 Common Return Codes". For the individual return codes for functions, see the function description in this chapter. The return codes are listed as the lower 2 bytes in hexadecimal (example: 0x0100).

9.2.1 Operations for the DLL

■ Initialize DLL

Format	LONG LJV7IF_Initialize(void);
Parameters	-
Return value	No individual return code
Explanation	This function initializes the DLL. (Always run this function)
Supported version	1.00

■ Finalize DLL

Format	LONG LJV7IF_Finalize(void);
Parameters	-
Return value	No individual return code
Explanation	This function performs the termination processing for the DLL. (Always run this function)
Supported version	1.00

■ Get DLL version

Format	DWORD LJV7IF_GetVersion(void);
Parameters	-
Return value	DLL version
Explanation	This function gets the DLL version. The version is expressed as a hexadecimal number. Viewed as hexadecimal, the 4th digit is the major version, the 3rd digit is the minor version, the 2nd digit is the revision, and the 1st digit is the build. For example, the initial version (1.2.3.4) is expressed as 0x1234. The major version is incremented when the DLL's backward compatibility is lost. The minor revision is incremented when the version is updated with additional functions.
Supported version	1.00

9.2.2 Establish/disconnect the communication path with the controller

For communication devices, see "9.2.9.1 Communication devices".

■ USB communication connection

Format	LONG LJV7IF_UsbOpen(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	This function establishes a connection with the controller so that the library can communicate with a USB-connected controller.
Supported version	1.00

■ Ethernet communication connection

Format	LONG LJV7IF_EthernetOpen (LONG lDeviceId, LJV7IF_ETHERNET_CONFIG* pEthernetConfig);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pEthernetConfig(in)** Ethernet communication settings. For each member, see "8 Constant, Structure Definitions"
Return value	No individual return code
Explanation	This function establishes a connection with the controller so that the library can communicate with an Ethernet-connected controller.
Supported version	1.00

■ Disconnect communication path

Format	LONG LJV7IF_CommClose(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	This function closes the USB or Ethernet connection. Even if this function is called when a connection has not been established, an error does not occur.
Supported version	1.00

9.2.3 System control

For communication devices, see "9.2.9.1 Communication devices".

■ Reboot the controller

Format	LONG LJV7IF_RebootController(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x80A0: Accessing the save area
Explanation	This function reboots the controller and connected devices. An error occurs while accessing the save area.
Supported version	1.00

■ Return to factory state

Format	LONG LJV7IF_RetrunToFactorySetting(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	This function returns all of the controller's settings to the factory state. After processing returns from this interface, write processing is being performed to the save area in the controller. Before turning off the power, ensure that you check the access status to the save area with the LJV7IF_CheckMemoryAccess function (see "9.1.5 Functions related to modifying or reading settings").
Supported version	1.00

■ Get system error information

Format	LONG LJV7IF_GetError (LONG lDeviceId, BYTE byRcvMax, BYTE* pbyErrCnt, WORD* pwErrCode);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	byRcvMax(in)
	Specifies the maximum amount of system error information to receive. (Size of the buffer passed in pwErrCode)
	pbyErrCnt(out)
	The buffer to receive the amount of system error information.
	pwErrCode(out)
	The buffer to receive the system error information. In order from the newest error, *pbyErrCnt items (byRcvMax items max) worth of system error information is stored.

Return value	No individual return code
Explanation	This function gets the controller's system error information. For the details of the meanings of the error codes that are returned, refer to the "LJ-V7000 Series User's Manual".
Supported version	1.00

■ Clear system error

Format	LONG LJV7IF_ClearError(LONG lDeviceId, WORD wErrCode);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	wErrCode(in)
	The error code for the error you wish to clear.

Return value	No individual return code
Explanation	This function clears the system error occurring on the controller. When all of the system errors that are occurring are successfully cleared, the controller will start measurements. Only the errors listed below can be cleared. 0x0084: Two heads were connected when previously started, but only one head could be recognized 0x0085: The connected head type is different than when previously started
Supported version	1.00

9.2.4 Measurement control

For communication devices, see "9.2.9.1 Communication devices".

■ Trigger

Format	LONG LJV7IF_Trigger(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x8080: The trigger mode is not "external trigger"
Explanation	This function issues a trigger.
Supported version	1.00

■ Start batch measurements

Format	LONG LJV7IF_StartMeasure(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x8080: Batch measurements are off 0x80A0: Batch measurement start processing could not be performed because the REMOTE terminal is off or the LASER_OFF terminal is on
Explanation	This function starts batch measurements. When batch measurements have already been started, nothing happens and there is no error.
Supported version	1.00

■ Stop batch measurements

Format	LONG LJV7IF_StopMeasure(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x8080: Batch measurements are off 0x80A0: Batch measurement stop processing could not be performed because the REMOTE terminal is off or the LASER_OFF terminal is on
Explanation	This function stops batch measurements. When batch measurements have not been started, nothing happens and there is no error.
Supported version	1.00

■ Auto zero

Format	LONG LJV7IF_AutoZero(LONG lDeviceId, BYTE byOnOff, DWORD dwOut);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byOnOff(in)** Other than 0: Auto zero on request, 0: off request. **dwOut(in)** Specifies the OUT to target for processing as a bit. From the LSB, OUT1, OUT2, to OUT16 are indicated by bits, and the OUT is the target for processing when the bit is 1 (upper 16 bits are reserved). Example: When you wish to set OUT1 and OUT5 to be the targets for processing Specify dwOut = 0x00000011 (... 0000 0000 0001 0001).
Return value	0x8080: The operation mode is "high-speed (profile only)"
Explanation	This function issues an auto zero request. Even when the OUT targeted for processing is configured to not be measured, an error will not occur.
Supported version	1.00

■ Timing

Format	LONG LJV7IF_Timing(LONG lDeviceId, BYTE byOnOff, DWORD dwOut);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	byOnOff(in)
	Same specification method as auto zero (LJV7IF_AutoZero).
	dwOut(in)
	Same specification method as auto zero (LJV7IF_AutoZero).

Return value	0x8080: The operation mode is "high-speed (profile only)"
Explanation	This function issues a timing request. Even when the OUT targeted for processing is configured to not be measured, an error will not occur.
Supported version	1.00

■ Reset

Format	LONG LJV7IF_Reset(LONG lDeviceId, DWORD dwOut);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	dwOut(in)
	Same specification method as auto zero (LJV7IF_AutoZero).

Return value	0x8080: The operation mode is "high-speed (profile only)"
Explanation	This function issues a reset request. Even when the OUT targeted for processing is configured to not be measured, an error will not occur.
Supported version	1.00

■ Clear memory

Format	LONG LJV7IF_ClearMemory(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	When the operation mode is "high-speed (profile only)", this function clears the profile data accumulated in internal memory. When the operation mode is "advanced (with OUT measurement)", the accumulated storage data is cleared.
Supported version	1.00

9.2.5 Functions related to modifying or reading settings

For communication devices, see "9.2.9.1 Communication devices".

■ Send setting

Format	LONG LJV7IF_SetSetting(LONG lDeviceId, BYTE byDepth, LJV7IF_TARGET_SETTING TargetSetting, void* pData, DWORD dwDataSize, DWORD* pdwError);
Parameters	**lDeviceId(in)** Indicates the communication device to communicate with. **byDepth(in)** Specifies the level to reflect the setting value to that was sent. (LJV7IF_SETTING_DEPTH) **TargetSetting(in)** Identifies the item that is the target to send. **pData(in)** Specifies the buffer that stores the setting data to send. **dwDataSize(in)** The size in BYTEs of the setting data to send. **pdwError(out)** The buffer for receiving detailed setting errors (see "9.2.9.4 Detailed setting errors"). * For details on the parameters, see the appendix.
Return value	No individual return code
Explanation	This function sends the setting for the specified item to the controller. For the procedure to reflect the setting on the controller, see "9.2.9.3 Write processing for settings".
Supported version	1.00

■ Get setting

Format	LONG LJV7IF_GetSetting(LONG lDeviceId, BYTE byDepth, LJV7IF_TARGET_SETTING TargetSetting, void* pData, DWORD dwDataSize);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byDepth(in)** Specifies the level of the setting value to get. (LJV7IF_SETTING_DEPTH) **TargetSetting(in)** Identifies the item that is the target to get. **pData(out)** Specifies the buffer to receive the setting data that was acquired. **dwDataSize(in)** The size of the buffer to receive the acquired data in BYTEs. * For details on the parameters, see the appendix.
Return value	No individual return code
Explanation	This function gets the setting for the specified item from the controller.
Supported version	1.00

■ Initialize setting

Format	LONG LJV7IF_InitializeSetting(LONG lDeviceId, BYTE byDepth, BYTE byTarget);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byDepth(in)** Specifies the level to reflect the initialized setting. (LJV7IF_SETTING_DEPTH) **byTarget (in)** Specifies the setting that is the target for initialization. (LJV7IF_INIT_SETTING_TARGET)
Return value	No individual return code
Explanation	This function initializes the setting specified as the initialization target. For the procedure to reflect the setting on the controller, see "9.2.9.3 Write processing for settings".
Supported version	1.00

■ Request to reflect settings in the write settings area

Format	LONG LJV7IF_ReflectSetting(LONG lDeviceId, BYTE byDepth, DWORD*pdwError);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byDepth (in)** Specifies to what level the settings written in the write settings area will be reflected to. (LJV7IF_SETTING_DEPTH) **pdwError(out)** The buffer for receiving detailed setting errors (see "9.2.9.4 Detailed setting errors").
Return value	No individual return code
Explanation	This function reflects the settings stored in the write settings area to the running settings area. When LJV7IF_SETTING_DEPTH_SAVE is specified as a parameter, the settings in the save area can be saved. When the controller was instructed to overwrite the settings in the save area with this function, ensure that you check the access status to the save area with the LJV7IF_CheckMemoryAccess function before turning the power off.
Supported version	1.00

■ Update write settings area

Format	LONG LJV7IF_RewriteTemporarySetting(LONG lDeviceId, BYTE byDepth);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byDepth (in)** Specifies the level of the settings to update the write settings area with. (LJV7IF_SETTING_DEPTH)
Return value	No individual return code
Explanation	This function updates the contents of the write settings area with either the settings in the running settings area or the settings saved in the save area.
Supported version	1.00

■ Check the status of saving to the save area

Format	LONG LJV7IF_CheckMemoryAccess(LONG lDeviceId, BYTE* pbyBusy);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pbyBusy(out)** The buffer for receiving information on whether the save area is being accessed Other than 0: Accessing the save area, 0: no access.
Return value	No individual return code
Explanation	This function checks whether or not the controller is accessing the save area with an operation such as that to save settings. When the controller was instructed to save settings to the save area with the LJV7IF_RetrunToFactorySetting function (see "9.1.3 System control"), the LJV7IF_SetSetting function, the LJV7IF_InitializeSetting function, or the LJV7IF_ReflectSetting function, check that access to the save area has completed with this function before turning off the power.
Supported version	1.00

■ Set date/time

Format	LONG LJV7IF_SetTime(LONG lDeviceId, LJV7IF_TIME* pTime);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pTime(in)** The date/time to set. For each member, see "8 Constant, Structure Definitions".
Return value	No individual return code
Explanation	This function sets the date/time for the controller.
Supported version	1.00

■ Get date/time

Format	LONG LJV7IF_GetTime(LONG lDeviceId, LJV7IF_TIME* pTime);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pTime(out)** The buffer to store the acquired date/time. For each member, see "8 Constant, Structure Definitions".
Return value	No individual return code
Explanation	This function gets the date/time from the controller.
Supported version	1.00

■ Change program

Format	LONG LJV7IF_ChangeActiveProgram(LONG lDeviceId, BYTE byProgNo);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **byProgNo(in)** Program number after the change. Specify as 0 to 15 (0: Program 0, 1: Program 1, and so on).
Return value	0x8080: The change program setting is "terminal"
Explanation	This function changes the active program number. When specifying the same number as the active program number in byProgNo, or when an invalid program number is specified, the operation to change the program is performed (internal memory is cleared, etc.), but the active program number is not changed.
Supported version	1.00

■ Get the active program number

Format	LONG LJV7IF_GetActiveProgram(LONG lDeviceId, BYTE* pbyProgNo);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pbyProgNo(out)** The buffer to receive the active program number. It is stored as 0 to 15 (0: Program 0, 1: Program 1, and so on).
Return value	No individual return code
Explanation	This function gets the active program number.
Supported version	1.00

9.2.6 Acquiring measurement results

For communication devices, see "9.2.9.1 Communication devices".

■ Get measurement results

Format	LONG LJV7IF_GetMeasurementValue(LONG lDeviceId, LJV7IF_MEASURE_DATA* pMeasureData);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pMeasureData(out)** This buffer stores the data for all 16 OUTs including the OUTs that are not measuring. The host requires the passing of a buffer LJV7IF_MEASURE_DATA[16] in size.
Return value	0x8080: The operation mode is "high-speed (profile only)"
Explanation	This function gets the newest measurement results (measurement values and judgment results). All measurements, including OUT measurements where the measurement mode is set to "do not measure" are acquired.
Supported version	1.00

■ Get profiles (operation mode "high-speed (profile only)")

Format	LONG LJV7IF_GetProfile(LONG lDeviceId, LJV7IF_GET_PROFILE_REQ* pReq, LJV7IF_GET_PROFILE_RSP* pRsp, LJV7IF_PROFILE_INFO* pProfileInfo, DWORD* pdwProfileData, DWORD dwDataSize);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the position, etc., of the profiles to get. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Indicates the position, etc., of the profiles that were actually acquired. For each member, see "8 Constant, Structure Definitions". **pProfileInfo(out)** The profile information for the acquired profiles. For each member, see "8 Constant, Structure Definitions". **pdwProfileData (out)** The buffer to get the profile data. The profile data is stored in this buffer with "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER" as a single unit of profile data, and only the number of profiles that could be acquired are returned. **dwDataSize(in)** pdwProfileData size in BYTEs
Return value	0x8080: The operation mode is "advanced (with OUT measurement)" 0x8081: "Batch measurements on and profile compression (time axis) off" 0x80A0: No profile data
Explanation	This function gets profile data. Profile data is stored in 0.01 μm units. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement". There is a limit to the number of profiles that can be read at once. This limit depends on the measurement settings. Refer to byGetProfCnt (the number of profiles read this time) in pRsp, and confirm that the data you wish to acquire has all been acquired. If it could not be acquired, specify the following in this function to acquire the remaining data: byPosMode in pReq = LJV7IF_PROFILE_POS_SPEC dwGetProfNo in pReq = dwGetTopProfNo in pRsp + byGetProfCnt in pRsp (Specify the profile data to be read as the next profile data after the profile data that was read this time)
Supported version	1.00

■ Get batch profiles (operation mode "high-speed (profile only)")

Format	LONG LJV7IF_GetBatchProfile(LONG lDeviceId, LJV7IF_GET_BATCH_PROFILE_REQ* pReq, LJV7IF_GET_BATCH_PROFILE_RSP* pRsp, LJV7IF_PROFILE_INFO * pProfileInfo, DWORD* pdwBatchData, DWORD dwDataSize);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the position, etc., of the profiles to get. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Indicates the position, etc., of the profiles that were actually acquired. For each member, see "8 Constant, Structure Definitions". **pProfileInfo(out)** The profile information for the acquired profiles. For each member, see "8 Constant, Structure Definitions". **pdwBatchData(out)** The buffer to get the profile data. The profile data is stored in this buffer with "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER" as a single unit of profile data, and only the number of profiles that could be acquired are returned. **dwDataSize(in)** pdwProfileData size in BYTEs
Return value	0x8080: The operation mode is "advanced (with OUT measurement)" 0x8081: Not "batch measurements on and profile compression (time axis) off" 0x80A0: No batch data (batch measurements not run even once)
Explanation	This function gets profile data. Profile data is stored in 0.01 μm units. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement". To read all of the profiles in one batch, read them with the procedure below. 1. Call this function by specifying LJV7IF_BATCH_POS_CURRENT for byPosMode in pReq. Save the start position and the amount of profiles that were read and the batch number that was read. 2. Configure pReq as listed below and call this function again. byPosMode = LJV7IF_BATCH_POS_SPEC dwGetBatchNo = batch number that was read byGetProfNo = starting profile number of the unread profiles in the batch 3. Update dwGetProfNo in step 2 and call this function until all of the profiles in the batch can be read.
Supported version	1.00

■ Get profiles (operation mode "advanced (with OUT measurement)")

Format	LONG LJV7IF_GetProfileAdvance(LONG lDeviceId, LJV7IF_PROFILE_INFO* pProfileInfo, DWORD* pdwProfileData, DWORD dwDataSize, LJV7IF_MEASURE_DATA* pMeasureData);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pProfileInfo(out)** The profile information for the acquired profiles. For each member, see "8 Constant, Structure Definitions". **pdwProfileData (out)** The buffer to get the profile data. In the "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER" format, one item of profile data is stored. **dwDataSize(in)** pdwProfileData size in BYTEs **pMeasureData(out)** This buffer stores the data for all 16 OUTs including the OUTs that are not measuring. The host requires the passing of a buffer LJV7IF_MEASURE_DATA[16] in size.
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: "Batch measurements on and profile compression (time axis) off" 0x80A0: No profile data
Explanation	This function gets profile data. Profile data is stored in 0.01 μm units. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement". For pMeasureData, OUT measurements where the measurement mode is set to "do not measure" are also stored.
Supported version	1.00

■ Get batch profiles (operation mode "advanced (with OUT measurement)")

Format	LONG LJV7IF_GetBatchProfileAdvance(LONG lDeviceId, LJV7IF_GET_BATCH_PROFILE_ADVANCE_REQ* pReq, LJV7IF_GET_BATCH_PROFILE_ADVANCE_RSP* pRsp, LJV7IF_PROFILE_INFO* pProfileInfo, DWORD* pdwBatchData, DWORD dwDataSize, LJV7IF_MEASURE_DATA* pBatchMeasureData, LJV7IF_MEASURE_DATA* pMeasureData);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the position, etc., of the profiles to get. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Indicates the position, etc., of the profiles that were actually acquired. For each member, see "8 Constant, Structure Definitions" **pProfileInfo(out)** The profile information for the acquired profiles. For each member, see "8 Constant, Structure Definitions". **dwBatchData(out)** The buffer to get the profile data. The profile data is stored in this buffer with "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER - LJV7IF_MEASURE_DATA x 16 OUTs" as a single unit of profile data, and only the number of profiles that could be acquired are returned. The results of the measurement process for the relevant profile are stored in LJV7IF_MEASURE_DATA. **dwDataSize(in)** pdwProfileData size in BYTEs. **pBatchMeasureData(out)** The measurement results for the batch data that is the target to get. This buffer stores the data for all 16 OUTs including the OUTs that are not measuring. **pMeasureData(out)** The newest measurement results at the time the command was processed. This buffer stores the data for all 16 OUTs including the OUTs that are not measuring. The host requires the passing of a buffer LJV7IF_MEASURE_DATA[16] in size.
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: Not "batch measurements on and profile compression (time axis) off" 0x80A0: No batch data (batch measurements not run even once)
Explanation	This function gets profile data. Profile data is stored in 0.01 µm units. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement". For pdwBatchData/pBatchMeasureData, OUT measurements where the measurement mode is set to "do not measure" are also stored. To read all of the profiles in one batch, read them with the procedure below. 1. Call this function by specifying LJV7IF_BATCH_POS_CURRENT for byPosMode in pReq. Save the start position and the amount of profiles that were read and the batch number that was read. 2. Configure pReq as listed below and call this function again. byPosMode=LJV7IF_BATCH_POS_SPEC dwGetBatchNo = batch number that was read dwGetProfNo = starting profile number of the unread profiles in the batch 3. Update dwGetProfNo in step 2 and call this function until all of the profiles in the batch can be read. When calling this function, do not perform communication between the target controller and LJ-Navigation 2.
Supported version	1.00

9.2.7 Store function related

For communication devices, see "9.2.9.1 Communication devices".

■ Start storage

Format	LONG LJV7IF_StartStorage(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: Storage target setting is "OFF" (no storage) 0x8082: The storage condition setting is not "terminal/command"
Explanation	When the storage condition setting is terminal/command, this function requests the start of storage.
Supported version	1.00

■ Stop storage

Format	LONG LJV7IF_StopStorage(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: Storage target setting is "off" (no storage) 0x8082: The storage condition setting is not "terminal/command"
Explanation	When the storage condition setting is terminal/command, this function requests the stop (cancellation) of storage.
Supported version	1.00

■ Get storage status

Format	LONG LJV7IF_GetStorageStatus(LONG lDeviceId, LJV7IF_GET_STORAGE_STATUS_REQ* pReq, LJV7IF_GET_STORAGE_STATUS_RSP* pRsp, LJV7IF_ STORAGE_INFO* pStorageInfo);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the target of the storage status to get. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Represents the actually acquired storage status. For each member, see "8 Constant, Structure Definitions". **pStorageInfo(out)** Storage information destination. If dwActiveSurface in pRsp is 0, the storage status is not updated
Return value	0x8080: The operation mode is "high-speed (profile only)"
Explanation	When the storage target setting is not off, this function gets the storage status.
Supported version	1.00

■ Get data storage data

Format	LONG LJV7IF_GetStorageData(LONG lDeviceId, LJV7IF_GET_STORAGE_REQ* pReq, LJV7IF_STORAGE_INFO* pStorageInfo, LJV7IF_GET_STORAGE_RSP* pRsp, DWORD* pdwData, DWORD dwDataSize);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the storage data to get. For each member, see "8 Constant, Structure Definitions". **pStorageInfo(out)** Represents the actually acquired storage information. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Represents the position and size of the actually acquired storage data. For each member, see "8 Constant, Structure Definitions". **pdwData(out)** The buffer to get the storage data. The storage data is stored in this buffer with "counter value in 10 ms units from the 32-bit base time - LJV7IF_MEASURE_DATA[16]" as a single unit of storage data, and only the amount of storage data that could be acquired is returned. For each member, see "8 Constant, Structure Definitions". For details about counter value in 10 ms units, refer to "9.2.9.10 Time data added to storage data". **dwDataSize(in)** pdwData size in BYTEs
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: The storage target setting is not "OUT value"
Explanation	When the storage target setting is "OUT value", this function gets the storage data. Data can be acquired even if it is within the storage period. If there is no accumulated data, 0 is returned for the number of items read in pRsp. If the data specified by the data number to start reading in pRep has not been accumulated, one piece of the newest data that has been accumulated is returned. In pdwData, OUT measurements where the measurement mode is set to "do not measure" are also stored. There is a limit to the amount of data that can be read at once. This limit depends on the measurement settings. Refer to dwDataCnt (the number of items read) in pRsp, and confirm that the data you wish to acquire has all been acquired. If it could not be acquired, specify the following in this function to acquire the remaining data: dwStartNo in pReq = dwStartNo in pRsp + dwDataCnt (Specify the data to be read as the next piece of data after the data that was read this time)
Supported version	1.00

■ Get profile storage data

Format	LONG LJV7IF_GetStorageProfile(LONG lDeviceId, LJV7IF_GET_STORAGE_REQ* pReq, LJV7IF_STORAGE_INFO* pStorageInfo, LJV7IF_GET_STORAGE_RSP* pRsp, LJV7IF_PROFILE_INFO* pProfileInfo, DWORD* pdwData, DWORD dwDataSize);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the profiles to get. For each member, see "8 Constant, Structure Definitions". **pStorageInfo(out)** Represents the actually acquired storage information. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Indicates the position and size of the profiles that were actually acquired. For each member, see "8 Constant, Structure Definitions". **pProfileInfo(out)** Indicates the profile information that was actually acquired. For each member, see "8 Constant, Structure Definitions". **pdwData(out)** The buffer to get the storage data. The profile storage data is stored in this buffer with "counter value in 10 ms units from the 32-bit base time - LJV7IF_MEASURE_DATA[16] - LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER - LJV7IF_MEASURE_DATA[16]" as a single unit of profile storage data, and only the amount of profile storage data that could be acquired is returned. The first LJV7IF_MEASURE_DATA[16] stores the newest measurement values at that time, the second stores the measurement values for that profile. For each member, see "8 Constant, Structure Definitions". For details about counter value in 10 ms units, refer to "9.2.9.10 Time data added to storage data". **dwDataSize(in)** pdwData size in BYTEs
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: The storage target setting is not profile, or "batch measurements on and profile compression (time axis) off"
Explanation	When the storage target setting is profile, this function gets the stored profile data. (Batch setting: off) Data can be acquired even if it is within the storage period. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement". If there is no accumulated data, 0 is returned for the number of items read in pRsp. If the data specified by the data number to start reading in pRep has not been accumulated, one item of the newest data that has been accumulated is returned. In pdwData, OUT measurements where the measurement mode is set to "do not measure" are also stored. There is a limit to the amount of profile data that can be read at once. This limit depends on the measurement settings. Refer to dwDataCnt (the number of items read) in pRsp, and confirm that the profile data you wish to acquire has all been acquired. If it could not be acquired, specify the following in this function to acquire the remaining data: dwStartNo in pReq = dwStartNo in pRsp + dwDataCnt in pRsp (Specify the profile data to be read as the next profile data after the profile data that was read this time)
Supported version	1.00

■ Get batch profile storage data

Format	LONG LJV7IF_GetStorageBatchProfile (LONG lDeviceId, LJV7IF_GET_BATCH_PROFILE_STORAGE_REQ* pReq, LJV7IF_STORAGE_INFO* pStorageInfo, LJV7IF_GET_BATCH_PROFILE_STORAGE_RSP* pRsp, LJV7IF_PROFILE_INFO* pProfileInfo, DWORD* pdwData, ·DWORD dwDataSize, DWORD* pdwTimeOffset, LJV7IF_MEASURE_DATA* pMeasureData);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with. **pReq(in)** Specifies the profiles to get. For each member, see "8 Constant, Structure Definitions". **pStorageInfo(out)** Represents the actually acquired storage information. For each member, see "8 Constant, Structure Definitions". **pRsp(out)** Indicates the position and size of the profiles that were actually acquired. For each member, see "8 Constant, Structure Definitions". **pProfileInfo(out)** Indicates the profile information that was actually acquired. For each member, see "8 Constant, Structure Definitions". **pdwData (out)** The buffer to get the storage data. The storage data is stored in this buffer with "LJV7IF_PROFILE_HEADER - signed 32-bit profile data - LJV7IF_PROFILE_FOOTER - LJV7IF_MEASURE_DATA[16]" as a single unit of storage data, and only the storage data that could be acquired is returned. The measurement results for each profile are stored in LJV7IF_MEASURE_DATA[16]. For each member, see "8 Constant, Structure Definitions". **dwDataSize(in)** pdwData size in BYTEs **pdwTimeOffset(out)** The buffer for receiving the counter value in 10 ms units from the 32 bit base time. For details about counter value in 10 ms units, refer to "9.2.9.10 Time data added to storage data". **pMeasureData(out)** The measurement results for the relevant batch data. This buffer stores the data for all 16 OUTs including the OUTs that are not measuring. The host requires the passing of a buffer LJV7IF_MEASURE_DATA[16] in size.
Return value	0x8080: The operation mode is "high-speed (profile only)" 0x8081: The storage target setting is not profile, or not "batch measurements on and profile compression (time axis) off" 0x80A0: The batch data specified by the batch number to read (dwGetBatchNo) in pReq has not been accumulated yet
Explanation	When the storage target setting is profile, this function gets the stored profile data. (Batch setting: on) Data can be acquired even if it is within the storage period. For the details on the data stored in the profile data (storage order and size), see "9.2.9 Supplement" If the data specified by the number of the profile in the batch to get in pRep has not been accumulated, one item of the newest data that has been accumulated is returned. In pdwData/pMeasureData, OUT measurements where the measurement mode is set to "do not measure" are also stored. Use the following procedure to read all profiles in a batch. 1. Call this function by specifying 0 for dwGetBatchTopProfNo in pReq. Save the start position and the number of profiles that were read and the batch number that was read. 2. Configure pReq as listed below and call this function again. dwGetBatchNo = batch number that was read dwGetBatchTopProfNo = first profile number of the unread profiles in the batch 3. Update by dwGetBatchTopProfNo in step 2 and call this function until all of the profiles in the batch can be read.
Supported version	1.00

9.2.8 High-speed data communication related

For communication devices, see "9.2.9.1 Communication devices".

■ Initialize USB high-speed data communication

Format	LONG LJV7IF_HighSpeedDataUsbCommunicationInitalize (LONG lDeviceId, void (*pCallBack)(BYTE*, DWORD, DWORD, DWORD, DWORD), DWORD dwProfileCnt, DWORD dwThreadId);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	pCallBack(in)
	Specifies the callback function to call when data is received by high-speed communication.
	dwProfileCnt(in)
	Specifies the frequency to call the callback function. The callback function is called when the specified number of profiles is received.
	dwThreadId(in) (synonymous with dwUser)
	Thread ID.

Return value	No individual return code
Explanation	This function initializes high-speed communication for a USB connected controller. It is necessary to maintain a unique communication path (not used for normal command communication) for high-speed communication. This function establishes a unique high-speed communication path.
Supported version	1.00

■ Initialize Ethernet high-speed data communication

Format	LONG LJV7IF_HighSpeedDataEthernetCommunicationInitalize (LONG lDeviceId, LJV7IF_ETHERNET_CONFIG* pEthernetConfig, WORD wHighSpeedPortNo, void (*pCallBack)(BYTE*, DWORD, DWORD, DWORD, DWORD), DWORD dwProfileCnt, DWORD dwThreadId);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	pEthernetConfig(in)
	Specifies the Ethernet settings used in high-speed communication.
	wHighSpeedPortNo(in)
	Specifies the port number used in high-speed communication.
	pCallBack(in)
	Specifies the callback function to call when data is received by high-speed communication.
	dwProfileCnt(in)
	Specifies the frequency to call the callback function. The callback function is called when the specified number of profiles is received.
	dwThreadId(in) (synonymous with dwUser)
	Thread ID.

Return value	No individual return code
Explanation	This function initializes high-speed communication for an Ethernet connected controller. It is necessary to maintain a unique communication path (not used for normal command communication) for high-speed communication. This function establishes a unique high-speed communication path. It is necessary to set different TCP/IP port numbers for normal command communication and high-speed communication (see the LJ-V7000 Series User's Manual for details on setting port numbers).
Supported version	1.00

■ Request preparation before starting high-speed data communication

Format	LONG LJV7IF_PreStartHighSpeedDataCommunication (LONG lDeviceId, LJV7IF_HIGH_SPEED_PRE_START_REQ* pReq, LJV7IF_PROFILE_INFO* pProfileInfo);

Parameters	**lDeviceId(in)**
	Specifies the communication device to communicate with.
	pReq(in)
	Specifies what data to send high-speed communication from.
	pProfileInfo(out)
	Stores the profile information.

Return value	0x8080: The operation mode is "advanced (with OUT measurement)" 0x8081: The data specified as the send start position does not exist 0x80A1: Already performing high-speed data communication
Explanation	This function performs the preparation to start high-speed data communication.
Supported version	1.40

■ Start high-speed data communication

Format	LONG LJV7IF_StartHighSpeedDataCommunication(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	0x80A0: A high-speed data communication connection was not established 0x80A2: High-speed data communication was not prepared before starting 0x80A4: The send target data was cleared
Explanation	This function starts high-speed data communication. If high-speed data communication does not operate correctly, see "9.2.9.9 High-speed data communication troubleshooting".
Supported version	1.00

■ Stop high-speed data communication

Format	LONG LJV7IF_StopHighSpeedDataCommunication(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	This function stops high-speed data communication.
Supported version	1.00

■ Finalize high-speed data communication

Format	LONG LJV7IF_HighSpeedDataCommunicationFinalize(LONG lDeviceId);
Parameters	**lDeviceId(in)** Specifies the communication device to communicate with.
Return value	No individual return code
Explanation	This function finalizes high-speed data communication.
Supported version	1.00

9.2.9 Supplement

9.2.9.1 Communication devices

The controller that will communication with the PC is specified as a communication device. The maximum number of controllers that can be communicated with simultaneously is defined by LJV7IF_DEVICE_COUNT (8.1 Constant definitions).
In interfaces that involve communication, you can specify the controller to target for communication with lDeviceID. lDeviceID can be specified as 0 to (LJV7IF_DEVICE_COUNT-1).

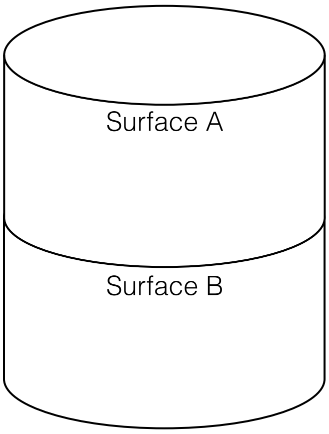
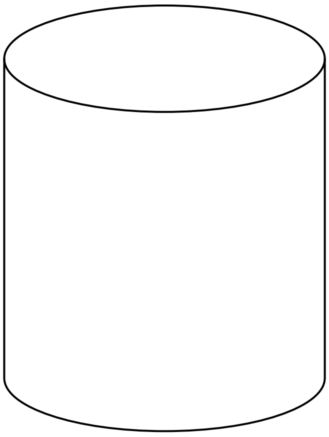
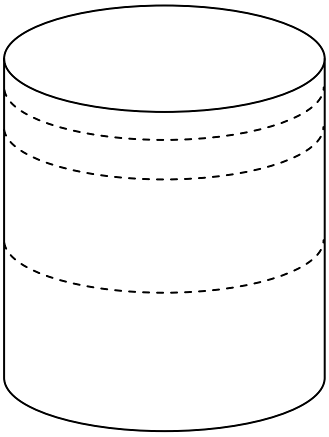
- Only one controller can communicate via USB.
  Example: When USB has been opened specifying the communication device 0, and USB is opened again specifying communication device 1, it will fail and 0x1001 (LJV7IF_RC_ERR_OPEN) is returned.
- One controller can communicate with three PCs via Ethernet communication.
  When a fourth PC connects to the controller, the PC with the oldest date/time of last communication of the three connected PCs is disconnected.
- High-speed communication is only possible between a single controller and a single PC.

9.2.9.2 Internal memory

The size of measurement data that can be saved and the save interval differs according to the memory allocation setting and the operation mode.
Use caution when using the following functions.

- LJV7IF_GetProfile (Get profiles (operation mode "high-speed (profile only)"))
- LJV7IF_GetBatchProfile (Get batch profiles (operation mode "high-speed (profile only)"))
- LJV7IF_GetStorageStatus (Get storage status)
- LJV7IF_GetStorageData (Get data storage data)
- LJV7IF_GetStorageProfile (Get profile storage data)
- LJV7IF_GetStorageBatchProfile (Get batch profile storage data)

The memory usage area for each memory allocation setting is listed below.

Double buffer	Entire area (overwrite)	Entire area (do not overwrite) [*1]
The internal memory is split into two surfaces, surface A and surface B. Each time the program changes, surface A and surface B are alternately used.[*2] 	The entire internal memory area is used. 	Of the entire internal memory area, the areas where data has not accumulated can be used to accumulate data. The save data is not deleted, even when the program is changed. 

[*1] Can only be specified when the operation mode is "advanced (with OUT measurement)".

[*2] The memory area being used by the active program is called the active surface. The unused surfaces are inactive surfaces.

9.2.9.2.1 For operation mode: high-speed (profile only)

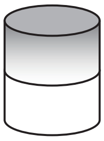
Save target data: profile

When the memory allocation is "double buffer"

The change program operation and the memory usage states transition as shown below.

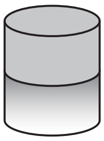
(*: Active surface)

Program number	Surface A	Surface B
1 (measuring)	Saving*	No data



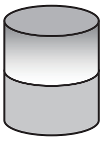
↓ Change program (save target surface is surface B)

Program number	Surface A	Surface B
2 (measuring)	Program number 1 data	Saving*



↓ Change program (save target surface is surface A. Surface A data is deleted.)

Program number	Surface A	Surface B
3 (measuring)	Saving*	Program number 2 data



Internal memory save status (white: no data ⇔ gray: saved) ↑

When the memory allocation is "entire area (overwrite)"

The memory is deleted when a change program operation is performed.

When reading profiles, if the profiles were read by specifying 1 (erase the read profiles) in byErase for LJV7IF_GET_PROFILE_REQ or LJV7IF_GET_BATCH_PROFILE_REQ, the read profiles (batch) and the profiles (batch) older than those are deleted from memory. During high-speed communication, sent profiles are deleted from memory when the controller sends the acquired profiles.
When performing an operation to continuously get profiles from the controller, make sure that the speed at which the PC reads and deletes files is faster than the speed at which data is saved on the computer.

9.2.9.2.2 For operation mode: advanced (with OUT measurement)

Save target data: storage target data (OUT measurement values/profiles)

When the memory allocation is "double buffer"

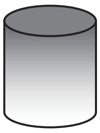
When the memory allocation is "entire area (overwrite)"

See "9.2.9.2.1 For operation mode: high-speed (profile only)".

When the memory allocation is "entire area (do not overwrite)"

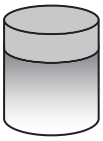
Change program operation and memory states transition example

Program number	Surface 1	Surface 2	Surface 3
1 (measuring)	Saving*	No data	No data



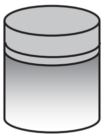
↓ Change program

Program number	Surface 1	Surface 2	Surface 3
2 (measuring)	Program number 1 data	Saving*	No data



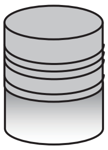
↓ Change program

Program number	Surface 1	Surface 2	Surface 3
1 (measuring)	Program number 1 data	Program number 2 data	Saving*



↓ Change program and repeat measurements

Program number	Surface 1	Surface 2	...	Surface N
1 (measuring)	Program number 1 data	Program number 2 data		Saving*



9.2.9.3 Write processing for settings

The 4 functions listed below are used when performing write processing for settings.

- LJV7IF_SetSetting (Send setting)
- LJV7IF_ReflectSetting (Request to reflect settings in the write settings area)
- LJV7IF_RewriteTemporarySetting (Update write settings area)
- LJV7IF_CheckMemoryAccess (Check the status of saving to the save area)

Depth must be specified using LJV7IF_SetSetting to send settings. The Depth options and their respective uses are as below.

Depth	Use
LJV7IF_SETTING_DEPTH_WRITE (Write settings area)	Settings written to this area are not applied to the controller. Settings written to this area are applied to RUNNING or SAVE by LJV7IF_ReflectSetting.
LJV7IF_SETTING_DEPTH_RUNNING (Running settings area)	Settings written to this area are applied to the controller, but are not saved if the power is turned off. (When rebooted, the SAVE settings are applied)
LJV7IF_SETTING_DEPTH_SAVE (Save area)	Settings written to this area are applied to the controller. The settings are saved on the controller even if the power is turned off. Writing to this area takes time. (Use LJV7IF_CheckMemoryAccess to check if settings are currently being written to this area. Make sure that writing using this function is complete before turning off the power.)

<Usage example 1>   Changing multiple settings at the same time

```
1:    Modify settings LJV7IF_SetSetting (WRITE)
2:    Modify settings LJV7IF_SetSetting (WRITE)
                          ⋮
Last: Modify settings LJV7IF_SetSetting (WRITE)
      Update write settings area LJV7IF_ReflectSetting (RUNNING)
```

The consistency of the settings is checked when writing to RUNNING or SAVE. If they are inconsistent, an error occurs. (Refer to "9.2.9.4 Detailed setting errors" for information about errors) Therefore, when multiple settings are changed and each setting is written individually to RUNNING (SAVE), there will be inconsistencies depending on the setting item, and the settings will not be applied to the controller. Write multiple settings to WRITE to create consistent settings before applying them to the controller

Use LJV7IF_RewriteTemporarySetting to rewrite inconsistent WRITE settings with the settings in the controller.

NOTICE

- Measuring is stopped when writing settings to RUNNING (SAVE).
- Do not turn the controller off when writing settings to SAVE. Use LJV7IF_CheckMemoryAccess to check if settings are written to this area.
- The same results are achieved if the last LJV7IF_SetSetting (WRITE) is replaced with LJV7IF_SetSetting (RUNNING). (Update write settings area is not required.)

<Usage example 2>   Changing 1 setting

- When not saving settings to the controller
  Modify setting LJV7IF_SetSetting (RUNNING)

- When saving settings to the controller
  Modify setting LJV7IF_SetSetting (SAVE)

NOTICE

- Measuring is stopped when writing settings to RUNNING (SAVE).
- Do not turn the controller off when writing settings to SAVE. Use LJV7IF_CheckMemoryAccess to check if settings are written to this area.

9.2.9.4 Detailed setting errors

The settings have a consistency that must be observed. The detailed setting errors (dwError) that are returned for the send setting and reflect write settings area request commands for settings that not satisfy this consistency are listed below.

dwError value	Error details
0x01000000	The operation mode is "high-speed (profile only)" and the memory allocation is set to entire area (do not overwrite)
0x1X000000 (*1)	The sampling period is outside the configurable range
0x1X06YY00 (*2)	Invalid measurement mode (when set to simple 3D measurements in batch measurement OUT, etc.)
0x1X06YY01 (*2)	Measurement mode "operator" is set for OUT1
0x1X06YY02 (*2)	When the measurement mode is operator, the operator target OUT is invalid
0x1X06YY03 (*2)	Measurement value filter cutoff frequency error
0x1X06YY04 (*2)	Prohibited combination in the measuring range settings (when configured to specify the measuring area when batch is off, etc.)
0x1X06YY05 (*2)	Measuring range setting (when the batch count and the measuring range do not match (also including simple 3D measurement Y direction measurement range))
0x1X06YY06 (*2)	When the measuring range setting is reference OUT, the consistency conditions were not fulfilled
0x1X06YY07 (*2)	Could not match the tolerance setting (upper/lower limit and hysteresis)
0x1X06YY08 (*2)	Could not match the scaling setting (measurement value 1/2, span after calculation)
0x1X070000 (*2)	Could not match the analog output scaling setting (OUT display value 1/2, span after calculation)
0x1X080000 (*2)	The amount of storage has exceeded the configurable range

*1: X indicates the program number and 0 to F is stored there.

*2: YY indicates the OUT number and 00 to 0F is stored there.

9.2.9.5 Profile data values

Profile data is output as signed 32-bit data, but for points where the profile could not be correctly detected, the following values are output.

Value (hexadecimal notation)	Name	Reason
-2147483648 (0x80000000)	Invalid data	This value is output when the peak could not be detected.
-2147483647 (0x80000001)		This value is output when the data is invalid due to a setting such as a mask having been set.
-2147483646 (0x80000002)	Dead zone data	This value is output when the data is located in a dead zone. This value is only output when dead zone processing is enabled.
-2147483645 (0x80000003)	Judgment wait data	This value is output when there is an insufficient amount of profiles for averaging.

9.2.9.6 Profile data amount calculation method

The amount of profile data to get with LJV7IF_GetProfile and other functions is a value multiplied by a correction factor determined from the settings below with 800 as the base.

Setting			Correction factor	Comment
Category	Item	Setting value		
Imaging settings	Measurement range X direction	Full	1.00	Initial value
		Middle	0.75	
		Small	0.50	
	Binning	Off	1.00	Initial value
		On	0.50	
Profile settings	Combine (wide)	Off	1.00	Initial value
		On	2.00	
	Compression (X axis)	Off	1.00	Initial value
		2	0.50	
		4	0.25	

For example, the amount of profile data with the settings below is 300 (= 800 x 0.75 x 1.00 x 1.00 x 0.50) items of data.

Measurement range X direction: Middle
Binning: Off
Combine (wide): Off
Compression (X axis): 2

However, when the amount of profile data found as a result of the equation above is less than 200, the profile compression (X axis) setting is adjusted so that the amount of profile data is 200 or higher.

For example, in a situation like that below, the amount of profile data is 300.

Measurement range X direction: Middle
Binning: Off
Combine (wide): Off
Compression (X axis): 4

The specific calculation is described below.

1. 800 x 0.75 x 1.00 x 1.00 x 0.25 = 150
2. The result is less than 200, so the profile compression (X axis) setting is adjust to 2 instead of 4
3. 800 x 0.75 x 1.00 x 1.00 x 0.5 = 300
4. The result is 200 or higher, so the amount of profile data above is confirmed

9.2.9.7 Profile data storage order

In functions that get profile data, such as LJV7IF_GetProfile, the profile data stored in the area between LJV7IF_PROFILE_HEADER and LJV7IF_PROFILE_FOOTER is 1 unit of profile data found with "Profile data amount calculation method", and the data is stored in the order below.

- Storage order
  1. 1st head profiles (when compression (time axis) is on, MAX profiles)
  2. 1st head MIN profiles
  3. 2nd head profiles (when compression (time axis) is on, MAX profiles)
  4. 2nd head MIN profiles
- Notes

  3 and 4 only exist when the number of heads is 2 and wide is off.
  2 and 4 only exist when compression (time axis) is on.

9.2.9.8 Specific examples

(i) For configuration 1 (initial settings)
Heads: 2
Measurement range X direction: Full
Binning: Off
Wide: Off
Compression (X axis): Off
Compression (time axis): Off

The amount of profile data is 800

The profile data storage order is as follows. (See profile data storage order for 1)

1. Head A profiles (800)
3. Head B profiles (800)

When getting 10 profiles with LJV7IF_GetProfile, the data below is stored in pdwProfileData.

Profile 1	LJV7IF_PROFILE_HEADER	32bit×6
	1. Head A profiles (800)	32bit×800
	3. Head B profiles (800)	32bit×800
	LJV7IF_PROFILE_FOOTER	32bit×1
⋮		
Profile 10	LJV7IF_PROFILE_HEADER	32bit×6
	1. Head A profiles (800)	32bit×800
	3. Head B profiles (800)	32bit×800
	LJV7IF_PROFILE_FOOTER	32bit×1

(ii) For configuration 2

Heads: 2
Measurement range X direction: Full
Binning: on
Wide: off
Compression (X axis): 2
Compression (time axis): On

The amount of profile data is 200
The profile data storage order is as follows. (See profile data storage order for 1)

1. Head A MAX profiles (200)
2. Head A MIN profiles (200)
3. Head B MAX profiles (200)
4. Head B MIN profiles (200)

When getting 10 profiles with LJV7IF_GetProfile, the data below is stored in pdwProfileData.

Profile 1	LJV7IF_PROFILE_HEADER	32bit×6
	1. Head A MAX profiles (200)	32bit×800 (800=200×4)
	2. Head A MIN profiles (200)	
	3. Head B MAX profiles (200)	
	4. Head B MIN profiles (200)	
	LJV7IF_PROFILE_FOOTER	32bit×1
⋮		
Profile 10	LJV7IF_PROFILE_HEADER	32bit×6
	1. Head A MAX profiles (200)	32bit×800 (800=200×4)
	2. Head A MIN profiles (200)	
	3. Head B MAX profiles (200)	
	4. Head B MIN profiles (200)	
	LJV7IF_PROFILE_FOOTER	32bit×1

9.2.9.9 High-speed data communication troubleshooting

Symptom	Item to check	Remedy
The application exits with an error after high-speed data communication starts.	Have the callback function call protocols been specified correctly?	Match the callback function call protocols with those in the header file.
The profile data to be obtained becomes abnormal at irregular intervals.	Is the data that is used by the main thread being used by a callback function without first acquiring exclusive processing access?	Acquire exclusive processing access for shared data.
The profile data to be obtained becomes abnormal at regular intervals.	Is heavy processing (such as the saving of files) being performed within a callback function?	Change the callback function so that its processing time becomes shorter.
	Is the communication speed of your communication path sufficient?	Change to a high-speed communication path such as 100BASE-T.
High-speed communication is interrupted.	Is heavy processing (such as the saving of files) being performed within a callback function?	Change the callback function so that its processing time becomes shorter.
	Is the communication speed of your communication path sufficient?	Change to a high-speed communication path such as 100BASE-T.

9.2.9.10 Time data added to storage data

Information that indicates the time that data was accumulated is added to storage data. The information added is "Base time (LJV7IF_TIME)" and "counter value in 10 ms units".

The counter value refers to the amount of time passed since the base time. If it is 1, 10 ms have passed, if it is 100, 10 ms x 100 = 1 s has passed. The time at which data was accumulated is calculated using the following formula: base time + counter value x 10 ms.

Note 1 The most application must calculate "counter value x 10 ms" and change the time so it is displayed in the following format: year, month, day, hour, minute and second.

Note 2 When the data is accumulated at a frequency greater than 10 ms, multiple pieces of data may have the same counter value. (The time cannot be shown in units smaller than 10 ms)

9.2.9.11 Data storage data storage order and specific example

Stored measurements can be acquired using GetStorageData. The acquired data has the following structure.

The structure of 1 piece of data storage data

0	Time accumulated		DWORD
1			
2			
3			
4	Measurement results (OUT1)	Measurement value information	BYTE
5	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
6		reserve	BYTE
7		reserve	BYTE
8		Measurement value	FLOAT
9			
10			
11			
~	~		
124	Measurement results (OUT16)	Measurement value information	BYTE
125	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
126		reserve	BYTE
127		reserve	BYTE
128		Measurement value	FLOAT
129			
130			
131			

9.2.9.12 Profile storage data storage order and specific example

Stored profile data, measurement values and internal measurement values (*1) can be acquired using GetStorageProfile. The acquired data has the following structure when there is 1 byte per row.

*1 "Internal measurement values" refers to measurements taken immediately before the OUT settings measuring mode processing. Each OUT internal measurement value is saved individually for each piece of profile data.

The structure of 1 piece of profile storage data

0	Time accumulated		DWORD
1			
2			
3			
4	Measurement results (OUT1)	Measurement value information	BYTE
5	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
6		reserve	BYTE
7		reserve	BYTE
8		Measurement value	FLOAT
9			
10			
11			
~	~		
124	Measurement results (OUT16)	Measurement value information	BYTE
125	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
126		reserve	BYTE
127		reserve	BYTE
128		Measurement value	FLOAT
129			
130			
131			
132	Profile data	Header LJV7IF_PROFILE_HEADER	
133			
~			
154			
155			
156		When 800 profiles * Refer to "Profile data amount calculation method" for the number of profiles. The number changes according to the settings. Unsigned 32 bit data	
~			
3355			
3356		Footer LJV7IF_PROFILE_FOOTER	
3357			
3358			
3359			
3360	Measurement results (OUT1)	Measurement value information	BYTE
3361	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
3362		reserve	BYTE
3363		reserve	BYTE
3364		Measurement value	FLOAT
3365			
3366			
3367			
~	~		
3480	Measurement results (OUT16)	Measurement value information	BYTE
3481	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
3482		reserve	BYTE
3483		reserve	BYTE
3484		Measurement value	FLOAT
3485			
3486			
3487			

Measurement value (Example: When the measuring mode is peak hold, the held value is the output measurement value)

Internal measurement value (Example: Measurements for the profile. Measurements taken immediately before measuring mode processing)

* The tolerance judgment result of internal measurement values is always 0.

9.2.9.13 Batch profile storage data storage order and specific example

Stored profile data and measurement values can be acquired using GetStorageBatchProfile The acquired data has the following structure. The acquired data has the following structure when there is 1 byte per row.

The structure of 1 piece of batch profile storage data

0	Profile data	Header LJV7IF_PROFILE_HEADER	
1			
~			
22			
23			
24		When 800 profiles * Refer to "Profile data amount calculation method" for the number of profiles. The number changes according to the settings. Unsigned 32 bit data	
~			
3223			
3224		Footer LJV7IF_PROFILE_FOOTER	
3225			
3226			
3227			
3228	Measurement results (OUT1)	Measurement value information	BYTE
3229	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
3230		reserve	BYTE
3231		reserve	BYTE
3232		Measurement value	
3233			
3234			FLOAT
3235			
~	~		
3348	Measurement results (OUT16)	Measurement value information	BYTE
3349	LJV7IF_MEASURE_DATA	Tolerance judgment result	BYTE
3350		reserve	BYTE
3351		reserve	BYTE
3352		Measurement value	
3353			
3354			FLOAT
3355			

Internal measurement value (Example: Measurements for the profile. Measurements taken immediately before measuring mode processing)

* The tolerance judgment result of internal measurement values is always 0.

10 Common Return Codes

10.1 Return codes returned by the communication library

The return codes shown below are judged in the communication library and returned to the application. Specifically, these codes are returned when the library fails to communicate with the controller or when processing could not be completed due to a state dependency in the communication library.

Definition name	Data (Lower 2 bytes)	Cause
LJV7IF_RC_OK	0x0000	Normal termination
LJV7IF_RC_ERR_OPEN	0x1000	Failed to open the communication path.
LJV7IF_RC_ERR_NOT_OPEN	0x1001	The communication path was not established.
LJV7IF_RC_ERR_SEND	0x1002	Failed to send the command.
LJV7IF_RC_ERR_RECEIVE	0x1003	Failed to receive a response.
LJV7IF_RC_ERR_TIMEOUT	0x1004	A timeout occurred while waiting for the response.
LJV7IF_RC_ERR_NOMEMORY	0x1005	Failed to allocate memory.
LJV7IF_RC_ERR_PARAMETER	0x1006	An invalid parameter was passed.
LJV7IF_RC_ERR_RECV_FMT	0x1007	The received response data was invalid.
LJV7IF_RC_ERR_HISPEED_NO_DEVICE	0x1009	High-speed communication initialization could not be performed.
LJV7IF_RC_ERR_HISPEED_OPEN_YET	0x100A	High-speed communication was initialized.
LJV7IF_RC_ERR_HISPEED_RECV_YET	0x100B	Error already occurred during high-speed communication (for high-speed communication)
LJV7IF_RC_ERR_BUFFER_SHORT	0x100C	The buffer size passed as an argument is insufficient.

10.2 Return codes returned from the controller

The return codes shown below are returned when communication with the controller was successful but the controller could not process the command.

Data (Lower 2 bytes)	Cause
0x8041	Status error (when a system error has occurred, etc.)
0x8042	Parameter error (when an invalid parameter was set, etc.)

11 Sample Program

This chapter describes the sample program which has been included as an example of how to create an application using the communication library.

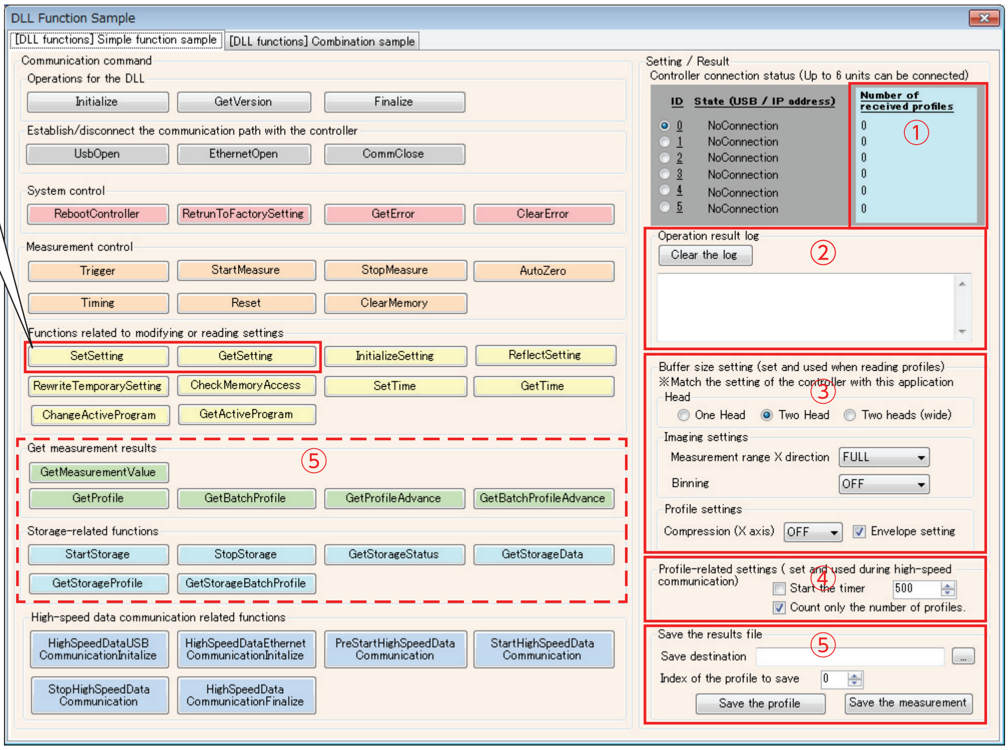
The program is fundamentally the same for C#, VB.NET, C++ and VB6. (With VB6, high-speed data communication cannot be performed.) Below is an example using C#.

* C#, VB.NET, C++ were built with Visual Studio 2008 SP1, and VB6 was built with Visual Basic 6.0 SP6.

11.1 User interface specification

[DLL functions] Simple function sample tab

Refer to 12.1 for information about changing settings.



Controller connection status (Up to 6 units can be connected)

	ID	State (USB / IP address)	Number of received profiles
●	0	NoConnection	0
○	1	NoConnection	0
○	2	NoConnection	0
○	3	NoConnection	0
○	4	NoConnection	0
○	5	NoConnection	0

Buffer size setting (set and used when reading profiles)
※Match the setting of the controller with this application

Head: ○ One Head ● Two Head ○ Two heads (wide)

Imaging settings
Measurement range X direction: FULL
Binning: OFF

Profile settings
Compression (X axis): OFF ☑ Envelope setting

Profile-related settings (set and used during high-speed communication)
□ Start the timer 500
☑ Count only the number of profiles.

Save destination:
Index of the profile to save: 0

On each button is the name of a function. Click the button to perform that function.

① Displays the number of profiles received using high-speed data communication. Up to 6 controllers can be displayed.* Profiles received using standard (not high-speed) profile receiving are not counted.

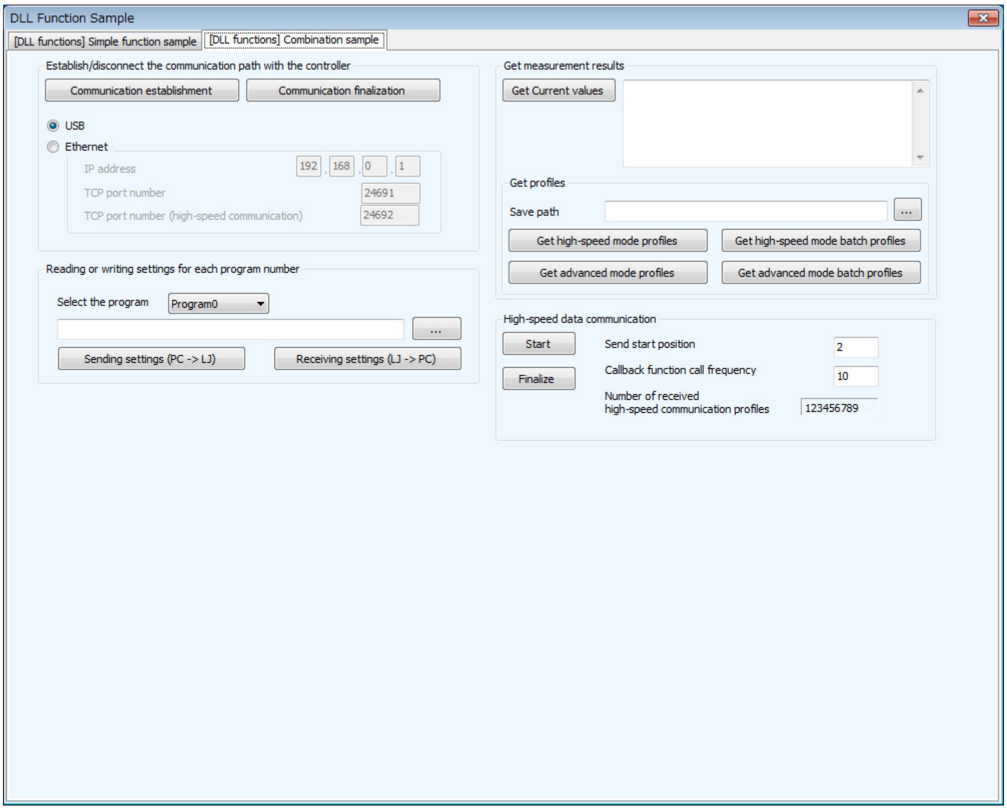
② Displays commands that have been executed and their results. Displays error codes when there is an error. For details about error codes, refer to each function's return value in "9.2 Function reference" (P.28~P.46) or the list of return codes in "10 Common Return Codes" P.60.

③ Used to change the size of arrays prepared to receive profiles in this sample program. (Used for GetProfile, GetBatchProfile, GetProfileAdvance, GetBatchProfileAdvance, GetStorageProfile and GetStorageBatchProfile.)
Align with the LJV main unit settings. If the prepared arrays are too small, profiles cannot be read correctly.

④ This must be set before high-speed data communication starts. (Set before executing HighSpeedDataUSBCommunicationInitialize or HighSpeedData EthernetCommunicationInitialize.)
<Start the timer>
Check this box to check and store received profiles at the frequency specified to the right (unit: ms). The display in ① will be updated. If "Save the profile" in ⑤ is clicked, the result data is stored in the specified save destination. The default setting is 500 ms.
<Count only the number of profiles>
Use this to check the communication speed. When the Callback function and a process are carried out, profiles may not be received at the estimated speed. When the box is checked, only the number of calls is counted in the Callback function. Use this to check if the required communication speed is acquired. If the speed is acquired, one of the processes may be heavy. If the speed is not acquired, the requested speed may be too high for the device configuration and settings.

⑤ Each command in dotted lines and measurements and profiles received during high-speed data communication can be output to a specified file.
<Example when using Save the measurement>
Results received in GetMeasurementValue or GetProfileAdvance etc. (OUT1 to 16) are saved to the file specified in "Save destination". 2 files are created. The first file contains measurements, and the second file contains internal measurement values. (Measurements for the profile. Measurements taken immediately before measuring mode processing). When using GetMeasurementValue the second file is empty as only measurement values are acquired.
<Example when using Save the profile>
Profiles received in GetProfile or GetProfileAdvance etc. are saved to the file specified in "Save destination". In this case, set "Index of the profile to save" to 0.
From profiles received with GetStorageProfile, etc., the profile specified in "Index of the profile to save" is saved to the file specified in "Save destination". (In Index of the profile to save, 0 refers to the first piece of data. If, for example, 10 pieces of data are received, enter "5" to save the 6th piece.)

[DLL functions] Combination sample tab



DLL Function Sample

[DLL functions] Simple function sample | [DLL functions] Combination sample

Establish/disconnect the communication path with the controller

● USB
○ Ethernet
IP address: 192 . 168 . 0 . 1
TCP port number: 24691
TCP port number (high-speed communication): 24692

Reading or writing settings for each program number

Select the program: Program0

Get measurement results

Get profiles

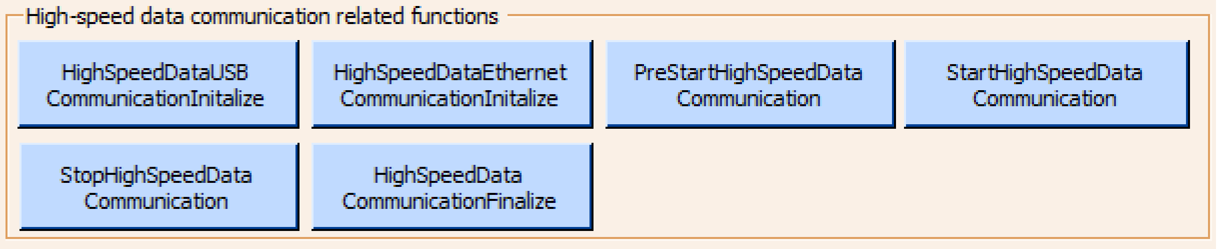
Save path:

High-speed data communication

Send start position: 2
Callback function call frequency: 10
Number of received high-speed communication profiles: 123456789

Communication establishment
Communication finalization
...
Sending settings (PC -> LJ)
Receiving settings (LJ -> PC)
Get Current values
Get high-speed mode profiles
Get high-speed mode batch profiles
Get advanced mode profiles
Get advanced mode batch profiles
Start
Finalize

The contents of previously distributed sample software can be found here. Communication establishment (finalization) to the top left is a combined sample of DLL Initialize and Establish the communication path (disconnect the communication path).
* The number of profiles that can be received at one time using the "Get high-speed mode profiles" command in the Combination sample tab is specified as 10 in the source code. If you wish change this specification refer to the "Get Profile" command in the Simple function sample tab.
"Start" and "Finalize" in High-speed data communication to the bottom right is a combined sample of the following functions.


High-speed data communication related functions
HighSpeedDataUSB CommunicationInitalize
HighSpeedDataEthernet CommunicationInitalize
PreStartHighSpeedData Communication
StartHighSpeedData Communication
StopHighSpeedData Communication
HighSpeedData CommunicationFinalize

Reading or writing settings for each program number to the bottom left is a sample of "12.2 Batch sending/ receiving". Environment settings, Common measurement settings and settings for each program can be saved in the specified file, or can be sent from the specified file.

11.2 Save file format

This section describes the file format for profiles that are saved in the get profile process.

- File format: TSV (tab separated text file)
- Values: The values acquired with the get profile interface are saved unmodified.
- Units: See "9.2.6 Acquiring measurement results".

■ Image of the arrangement of data in the save file when "Save the measurement" in the [DLL functions] Simple function sample tab is clicked

Output file

OUT1 measurement value	OUT2 measurement value	• • •	OUT16 measurement value

* 2 files are created. The second file contains internal measurement values (Measurements for the profile. Measurements taken immediately before measuring mode processing). When using GetMeasurementValue the second file is empty as only measurement values are acquired.

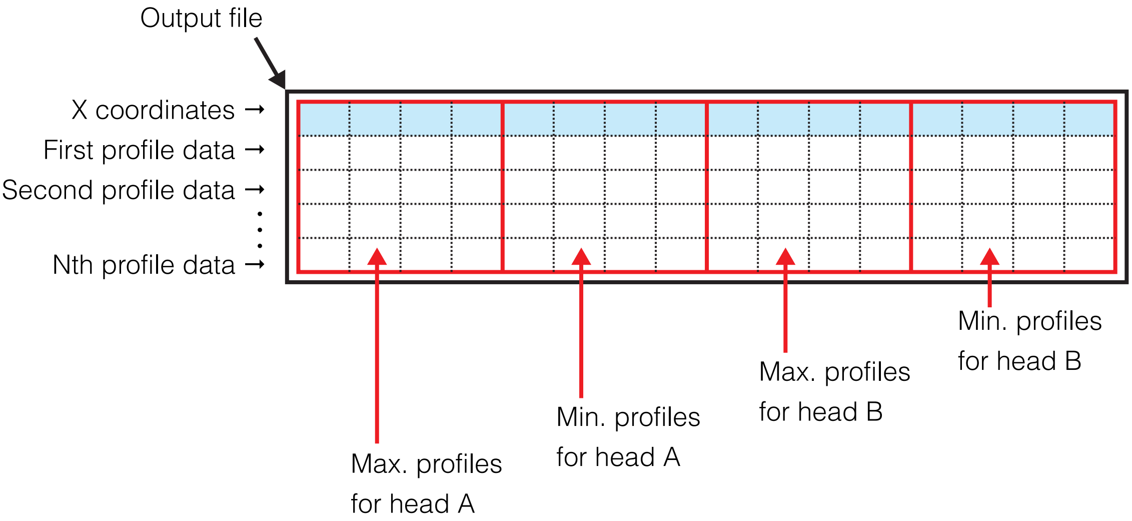
■ Image of the arrangement of data in the save file when "Save the profile" in the [DLL functions] Simple function sample tab is clicked (2 heads, compression (time axis) on)

Output file

Profile data

X coordinates	Max. profiles for head A	Min. profiles for head A	Max. profiles for head B	Min. profiles for head B

■ Image of the arrangement of data saved to "Save path" in the [DLL functions] Combination sample tab (2 heads, compression (time axis) on)



12 Appendix

12.1 Sending/Receiving Settings

With the LJ-V7000 Series, settings can be sent and received for each item using send settings (LJV7IF_SetSetting) and receive settings (LJV7IF_GetSetting). (Each item refers to set parameters for Environment settings/Common measurement settings/settings for programs 0 to 15.)
To send and receive Environment settings, Common measurement settings and settings for each program together in a batch, refer to "12.2 Batch sending/receiving".

This section explains Target Setting and pData, which are input and output into Sending/Receiving Settings commands. (Refer to "9.2.9.3 Write processing for settings" for information about byDepth)

Target Setting: Specify the item to send/receive settings for. Members are as follows. For detailed parameters for each member refer to "Details of Items for Sending/Receiving Settings" on the following page.

Type	Specify the settings to send/receive from Environment settings (01h), Common measurement settings (02h) and program 0 to program 15 (10h to 1Fh).
Category	When sending/receiving program 0 to 15 settings, specify the settings to send/receive from Trigger settings, Imaging settings etc. Specify 0 when sending/receiving Environment settings and Common measurement settings.
Item	Specify the settings to send/receive for the item specified in Category.
Target1	Specification may be required according to the setting to be sent/received. When no setting is required, specify 0.
Target2	
Target3	
Target4	

pData: Specifies the setting data to send/receive. For details, refer to "Details of Items for Sending/Receiving Settings" on the following page.

12.2 Batch sending/receiving

Specify FFh in Category above to send/receive Environment settings/Common measurement settings/settings for each program in a batch.

Example 1: Sending/receiving Common measurement setting data in a batch
Type: 02h, Category: FFh, Item: 00h, Target 1 to 4: 00h

Example 2: Sending/receiving program No. 5 settings in a batch
Type: 15h, Category: FFh, Item: 00h, Target 1 to 4: 00h

Refer to the sample program for details.

* When sent/received as a batch, Environment settings are 60 bytes, Common measurement settings are 12 bytes, and program settings are 10932 bytes.

12.3 Details of Items for Sending/Receiving Settings

Changing Environmental Settings

<Device name>

Type:01h, Category:00h, Item:00h

Target1~4:00h

byte	Setting Data
0	Device name, byte 1
1	Device name, byte 2
2	Device name, byte 3
to	to
31	Device name, byte 32

*32 characters max. 0 is not appended to the end.

*SHIFT-JIS

<Operation at next power on>

Type:01h, Category:00h, Item:01h

Target1~4:00h

byte	Setting Data
0	Operation at next power on 0:BOOT→IP addresses fixed, 1:IP address fixed, 2:BOOTP
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<High-speed communication band restriction>

Type:01h, Category:00h, Item:02h

Target1~4:00h

byte	Setting Data
0	High-speed communication bandrestriction 0:OFF, 1:500Mbps, 2:200Mbps, 3:100Mbps
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<MTU during high-speed communication>

Type:01h, Category:00h, Item:03h

Target1~4:00h

byte	Setting Data
0	MTU setting:1500~9216
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<IPaddress/Subnet mask/Gateway>

Type:01h, Category:00h

Item:04h (IP address)/05h (Subnet mask)/06h (Gateway)

Target1~4:00h

byte	Setting Data
0	IP address, 1st byte
1	IP address, 2nd byte
2	IP address, 3rd byte
3	IP address, 4th byte

The following IP addresses are treated as invalid IP addresses:
0.0.0.0./224.0.0.0~255.255.255.255

The following addresses are treated as invalid subnet masks:
0.0.0.0./255.255.255.255/ There are no consecutive[1]bits from the beginning (Example:255.255.255.64= 11111111.11111111.11111111.01000000 is an error)

The following addresses are treated as invalid gateway:
224.0.0.0~255.255.255.255

<TCP command port number/TCP high-speed port number >

Type:01h, Category:00h

Item:07h (TCP command port number)/
08h (TCP high-speed port)

Target1~4:00h

byte	Setting Data
0	Port number (1~65535)
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Do not set TCP command ports number same as TCP high-speed port number.

<Baud rate>

Type:01h, Category:00h, Item:0Ah

Target1~4:00h

byte	Setting Data
0	Baud rate: 0:9600, 1:19200, 2:38400, 3:57600, 4:115200
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Parity>

Type:01h, Category:00h, Item:0Bh

Target1~4:00h

byte	Setting Data
0	Parity:0:NONE, 1:EVEN, 2:ODD
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Common measurement settings

<Operation mode>

Type:02h, Category:00h, Item:00h

Target1~4:00h

byte	Setting Data
0	Operation mode: 0:High-speed, 1:Advanced
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Memory allocation>

Type:02h, Category:00h, Item:01h

Target1~4:00h

byte	Setting Data
0	Memory allocation setting: 0:Double buffer, 1:Entire area (overwrite), 2:Entire area (do not overwrite)
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Operation when memory full>

Type:02h, Category:00h, Item:02h

Target1~4:00h

byte	Setting Data
0	Operation when memory full: 0:Overwrite, 1:Stop
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Parallel imaging>

Type:02h, Category:00h, Item:03h

Target1~4:00h

byte	Setting Data
0	Parallel imaging: 0:Disabled, 1:Enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Strobe output time>

Type:02h, Category:00h, Item:04h

Target1~4:00h

byte	Setting Data
0	Strobe output time: 0:10μs, 1:20μs, 2:50μs, 3:100μs, 4:200μs, 5:500μs, 6:1ms, 7:2ms, 8:5ms, 9:10ms, 10:20ms
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<TRG minimum input time>

Type:02h, Category:00h, Item:06h

Target1~4:00h

byte	Setting Data
0	Constant when TRG input terminal: 0:7μs, 1:10μs, 2:20μs, 3:50μs, 4:100μs, 5:200μs, 6:500μs, 7:1ms
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<ENCODER minimum input time>

Type:02h, Category:00h, Item:07h

Target1~4:00h

byte	Setting Data
0	Constant when ENCODER input terminal: 0:120ns, 1:150ns, 2:250ns, 3:500ns, 4:1μs, 5:2μs, 6:5μs, 7:10μs, 8:20μs
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Control terminal minimum input time>

Type:02h, Category:00h, Item:08h

Target1~4:00h

byte	Setting Data
0	Control terminal minimum input time: 0:250μs, 1:1ms
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Change program>

Type:02h, Category:00h, Item:09h

Target1~4:00h

byte	Setting Data
0	Change program: 0:Terminal, 1:Command
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Change Program setting

• Trigger settings

<Trigger mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:01h

Target1~4:00h

byte	Setting Data
0	Trigger mode: 0:Continuous trigger, 1:External trigger, 2:Encoder trigger
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Sampling frequency>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:02h

Target1~4:00h

byte	Setting Data
0	Sampling frequency: 0:10Hz, 1:20Hz, 2:50Hz, 3:100Hz, 4:200Hz, 5:500Hz, 6:1KHz, 7:2KHz, 8:4KHz, 9:4.13KHz, 10:8KHz, 11:16KHz, 12:32KHz, 13:64KHz
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Batch measurement>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:03h

Target1~4:00h

byte	Setting Data
0	Batch measurement: 0:Batch OFF, 1:Batch ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Inter-trigger pitch>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:04h

Target1~4:00h

byte	Setting Data
0	Inter-trigger pitch: 0:Pitch OFF, 1:Pitch ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Inter-trigger pitch count>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:05h

Target1~4:00h

byte	Setting Data
0	Pitch count: 1~50000 (0.001mm unit, 0.001~50.000mm)
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Mutual interference prevention>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:06h

Target1~4:00h

byte	Setting Data
0	Mutual interference prevention: 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Input mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:07h

Target1~4:00h

byte	Setting Data
0	Encoder trigger input mode: 0: 1-phase 1TM (no dir_), 1:2-phase 1 times, 2:2-phase 2 times, 3:2-phase 4 times
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Skipping>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:08h

Target1~4:00h

byte	Setting Data
0	Encoder trigger skipping: 0: Skipping OFF, 1:Skipping ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Points to skip>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:09h

Target1~4:00h

byte	Setting Data
0	Encoder trigger skipping count: 2~1000
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Batch count>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:00h, Item:0Ah

Target1~4:00h

byte	Setting Data
0	Batch count: 50~15000
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

• Imaging settings

<Binning>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:01h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Binning: 0: OFF, 1: ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<X direction>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:02h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Measurement range X direction: 0:FULL, 1:MIDDLE, 2:SMALL
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Z direction>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:03h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Measurement range Z direction: 0:FULL, 1:MIDDLE, 2:SMALL
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<CMOS sensitivity>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:05h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	CMOS sensitivity characteristics: 0:High precision, 1:High dynamic range1, 2:High dynamic range2, 3:High dynamic range3
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Exposure time>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:06h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Exposure time: 0:15μs, 1:30μs, 2:60μs, 3:120μs, 4:240μs, 5:480μs, 6:960μs, 7:1920μs, 8:5ms, 9:10ms
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Imaging mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:07h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Imaging mode: 0:standard, 1:multi emission (synthesis), 2:multi emission (optimized light intensity)
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Multi emission (optimized light intensity) detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:08h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Emission times: 0:2 times, 1:4 times
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Multi emission (synthesis) detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:09h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Emission times: 0:3 times, 1:5 times
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Mask setting>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:0Ah

Traget1:00h (headA/wide), 01h (headB)
Target2:00h (Upper mask1), 01h (Upper mask2),
02h (Upper mask3), 03h (Lower mask1),
04h (Lower mask2), 05h (Lower mask3)
Target3~4:00h

byte	Setting Data
0	Enabled/disabled: 0:Mask disabled, 1:Rectangle, 2:Triangle
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	X coordinate1: 2~640
5	
6	Z coordinate1: 2~480
7	
8	X coordinate2: 2~640
9	
10	Z coordinate2: 2~480
11	
12	X coordinate3: 2~640 (invalid when Rectangle)
13	
14	Z coordinate3: 2~480 (invalid when Rectangle)
15	

<Control mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:0Bh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Control mode: 0:AUTO, 1:MANUAL
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Upper limit value/Lower limit value>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:0Ch (upper limit value), 0Dh (lower limit value)

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	FB upper/lower limit value: 1~99
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<FB target area>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:0Eh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	FB target area start: 0~639
1	
2	FB target area end: 0~639
3	

<Peak detection level>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:0Fh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Peak detection level: 1~5
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Invalid data interpolation count>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:10h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Invalid data interpolation count: 0~255
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Peak selection>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:11h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Peak selection: 0:Standard, 1:NEAR, 2:FAR, 3:Remove X multi reflection, 4:Remove Y multi reflection, 5:Make invalid data
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Peak width filter>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:01h, Item:12h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Peak width filter: 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

• Profile

<Combine (wide)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:01h

Target1~4:00h

byte	Setting Data
0	Wide setting: 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Note: This setting is not used when one sensor head.

<Compression (X axis)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:02h

Target1~4:00h

byte	Setting Data
0	Compression (X-axis): 0:OFF, 1:2, 2:4
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Compression (time axis)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:03h

Target1~4:00h

byte	Setting Data
0	Compression (time axis): 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Time axis compression count>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:04h

Target1~4:00h

byte	Setting Data
0	Time axis compression count: 2~1000
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Dead zone process valid/invalid>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:05h

Target1~4:00h

byte	Setting Data
0	Dead zone processing enabled/disabled: 0:disabled, 1:enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Reverse (X)/Reverse (Z)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:06h (ReverseX), 07h (ReverseZ)

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Reverse: 0: OFF, 1: ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Note: This setting is not used when one sensor head

<Shift(X)/Shift(Z)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:08h (Shift X), 09h (Shift Z)

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Shift amount: any value in measurement range (0.001μm unit, Sined 32-bit integer example:1mm=100000, 2mm=200000)
1	
2	
3	

Note: This setting is not used when one sensor head.

<Median (X axis)/Median (time axis)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:0Ah (Median (x axis)), 0Ch (Median (time axis))

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Median count: 0:OFF, 1:3 points, 2:5 points, 3:7 points, 4:9 points
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Smoothing>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:0Bh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Smoothing: 0:1, 1:2, 2:4, 3:8, 4:16, 5:32, 6:64
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Averaging>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:0Dh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Averaging count: 0:1, 1:2, 2:4, 3:8, 4:16, 5:32, 6:64, 7:128, 8:256
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Invalid data processing (time axis)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:0Eh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	Processing timses: 0~255
1	Resume times: 0~255
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Tilt correction>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:0Fh

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	ON/OFF: 0:Correction disabled, 1:Correction enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Linear calculation area count: 0:Area2 disabled, 1:Area2 enabled
5	Reserved (fixed as 0)
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Area start position1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
9	
10	
11	
12	Area end position1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Area start position2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Area end position2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
21	
22	
23	
24	Post-correction angle (–45.00~+45.00deg) : –4500~+4500
25	
26	Correction angle (–45.00~+45.00deg) : –4500~+4500
27	

<Height correction>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:02h, Item:10h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data
0	ON/OFF: 0:Correction disabled, 1:Correction enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Area start position1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Area end position1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
9	
10	
11	
12	Area start position2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Area end position2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Post-correction height 0~999.99mm: 0~99999
21	
22	
23	
24	Correction span: 1~131071 *The correction span is the value devided by 65536. (Condition:0<Correction span<2) example: if 98304 is set, (98304÷65536=1.5) correction span is 1.5.
25	
26	
27	

• Master regist

<Master profile>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:03h, Item:01h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data	
0	valid/invalid: 0:Master invalid, 1:Master valid	
1	Reserved (fixed as 0)	
2	Reserved (fixed as 0)	
3	Reserved (fixed as 0)	
4	Profile data amount (*1): 50,75,100,150,200,300,400,600, 800,1200,1600	
5		
6	Reserved (fixed as 0)	
7	Reserved (fixed as 0)	
8	Xcoordinate data start position (*2) (0.01μm unit Sined 32-bit integer)	
9		
10		
11		
12	X direction pitch (*2) (0.01μm unit, Sined 32-bit integer)	
13		
14		
15		
16	Profile (0.01μm unit Sined 32-bit integer)	*3
17		
18		
19		
~	~	
~		
~		
~		
3212	Profile (0.01μm unit Sined 32-bit integer)	
3213		
3214		
3215		

*1 Profile data amount depend on the setting. Refer to 10.2 "Profile data amount calculation method" or receive "MasterProfile" and confirm it.

*2 It depends on the type of sensor head and settings. Please confirm by receiving "MasterProfile"

*3 This example is in case of 800 points (It depends on Profile data amount.)

• Position correction settings

<Dual head mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:01h

Target1~4:00h

byte	Setting Data
0	Dual head mode: 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Note: This setting is not used when one sensor head.

<Dual head mode target head>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:02h

Target1~4:00h

byte	Setting Data
0	Correction target head: 0:headA, 1:headB
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

Note: This setting is not used when one sensor head.

<Dual head mode X/Z correction amount>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:03h (X correction amount), 04h (Z correctionamount)

Target1~4:00h

byte	Setting Data
0	Correction amount: -10000.0~+10000.0mm (0.001µm unit, Sined 32-bit integer example:1mm=100000, 2mm=200000)
1	
2	
3	

Note: This setting is not used when one sensor head.

<θ correction ON/OFF><Backup correction ON/OFF>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:05h (θ correctionON/OFF), 07h (preliminary correctionON/OFF)

Target1:00h (headA/wide), 01h (headB)

Target2:00h (Position correction1), 01h (Position correction2), Target3~4:00h

byte	Setting Data
0	ON/OFF: 0:OFF, 1:ON
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<θ correction detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:06h (θ correctionON/OFF)

Target1:00h (headA/wide), 01h (headB)

Target2:00h (position correction1), 01h (position correction2), Target3~4:00h

byte	Setting Data
0	Linear calculation area: 0:Area 2 disabled, 1:Area2 enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Linear calculation area1 Left: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Linear calculation area1 Right: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
9	
10	
11	
12	Linear calculation area2 Left: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Linear calculation area2 Right: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Correction standard: 0:Horizontal,1:MasterProfile
21	Reserved (fixed as 0)
22	Reserved (fixed as 0)
23	Reserved (fixed as 0)

<preliminary correction detail><Xcorrection detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:08h (preliminary correctiondetail), 0Bh (Xcorrectiondetail)

Target1:00h (headA/wide), 01h (headB)

Target2:00h (position correction1), 01h (position correction2), Target3~4:00h

byte	Setting Data
0	Edge measuring area Left: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Edge measuring area Right: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Edge direction: 0:Rising, 1:Falling
9	Detaction direction: 0:+direction, 1:–direction
10	Detection No: 1~10
11	Reserved (fixed as 0)
12	Edge level: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	

<XZcorrection selection>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:09h

Target1:00h (headA/wide), 01h (headB)

Target2:00h (position correction1), 01h (position correction2), Target3~4:00h

byte	Setting Data
0	XYcorrection selection: 0:OFF, 1:Xcorrection, 2:Zcorrection, 3:X→Zcorrection, 4:Z→Xcorrection, 5:Feature point correction
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Z correction Height measurement detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:0Ch

Target1:00h (headA/wide), 01h (headB)

Target2:00h (position correction1), 01h (position correction2), Target3~4:00h

byte	Setting Data
0	Height measuring area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Height measuring area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Height type: 0:Peak, 1:bottom, 6:Average
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)

<Featurepoint correction detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:04h, Item:0Dh

Target1:00h (headA/wide), 01h (headB)

Target2:00h (position correction1), 01h (position correction2), Target3~4:00h

byte	Setting Data
0	Correction target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), , 4:Contact (lin-arc)
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

* From 4byte, unique parameters are assigned to each correction target. For details on the unique parameters, see "Measurement area details" (page 83).

• Profilemask settings

<Profile mask area settings group>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:05h, Item:01h

Target1:00h (headA/wide), 01h (headB), Target2~4:00h

byte	Setting Data	
0	Area selection: 0:Disabled, 1:Rectangle, 2:Triangle	
1	position correction selection: 0: No position correction, 1:Position correction1, 2:Position correction2	
2	Reserved (fixed as 0)	
3	Reserved (fixed as 0)	
4	Xcoordinate1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	*1
5		
6		
7		
8	Zcoordinate1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	
9		
10		
11		
12	Xcoordinate2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	
13		
14		
15		
16	Zcoordinate2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	
17		
18		
19		
20	Xcoordinate3: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	
21		
22		
23		
24	Zcoordinate3: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)	
25		
26		
27		
~		
139		

* When Rectangle is selected, upper left (Xcoordinate1, Zcoordinate1) and bottom right (Xcoordinate2, Zcoordinate2) should be set. Xcoordinate3, Zcoordinate3 is no effect.

* When Triangle is selected, (Xcoordinate1, Zcoordinate1), (Xcoordinate2, Zcoordinate2) and (Xcoordinate3, Zcoordinate3) are used.

* 1 the number of profile mask area (×5) is continuing. (Total 140byte is used.)

<Profile Mask area settings individual>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:05h, Item:02h

Target1:00h (headA/wide), 01h (headB)

Target2:00h~04h (Profile mask area1~5) Target3~4:00h

byte	Setting Data
0	Area selection:0:Disabled, 1:Rectangle, 2:Triangle
1	position correction selection: 0: No position correction, 1:Position correction1, 2:Position correction2
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Xcoordinate1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Zcoordinate1: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
9	
10	
11	
12	Xcoordinate2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Zcoordinate2: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Xcoordinate3: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
21	
22	
23	
24	Zcoordinate3: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
25	
26	
27	

* When Rectangle is selected, upper left (Xcoordinate1, Zcoordinate1) and bottom right (Xcoordinate2, Zcoordinate2) should be set. Xcoordinate3, Zcoordinate3 is no effect.

* When Triangle is selected, (Xcoordinate1, Zcoordinate1), (Xcoordinate2, Zcoordinate2) and (Xcoordinate3, Zcoordinate3) are used.

• OUT settings

<OUT name>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:01h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

Byte	Setting Data
0	OUT name, byte1
1	OUT name, byte2
2	OUT name, byte3
~	~
19	OUT name, byte20

*20 Characters max. 0 is not appended to the end.

*SHIFT-JIS

<Minimum display unit>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:02h

Target1:00h~0Fh(OUT1~16) Target2~4:00h

byte	Setting Data
0	Minimum display unit
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

* The unit changes according to the measurement mode assigned to the OUT.

Length system, 0:1mm, 1:0.1mm, 2:0.01mm, 3:0.001mm, 4:1μm, 5:0.1μm

Area system, 0:1mm^2, 1:0.1mm^2, 2:0.01mm^2, 3:0.001mm^2, 4:0.0001mm^2, 5:0.00001mm^2

Angle system, 0:1deg, 1:0.1deg, 2:0.01deg

<Measurement mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:03h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Minimum display unit (from①)
1	Measurement mode (from②)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	From 4byte, unique parameters are assigned to each measurement mode. For details on the unique parameters, see [Unique parameters of measurement mode from 4byte]
~	
N	

① Length system ･･･ 0:1mm, 1:0.1mm, 2:0.01mm, 3:0.001mm, 4:1μm, 5:0.1μm

Area system ･･･ 0:1mm^2, 1:0.1mm^2, 2:0.01mm^2, 3:0.001mm^2, 4:0.0001mm^2, 5:0.00001mm^2

Angle system ･･･ 0:1deg, 1:0.1deg, 2:0.01deg

② 0:OFF, 1:Height, 2:Step, 3:Position, 4:Center position, 5:Width, 6:Thickness, 7:Angle, 8:R measurement, 9:Area, 10:Master comparison (Z), 11:Distance(point-point), 12:Distance(point-line), 13:Heigh (Profile compression (time axis) on), 14:Position (Profile compression (time axis) on), 15:Deflection width (Profile compression (time axis) on), 16:Height (simple 3D), 17:Step (simple 3D), 18:Position (simple 3D), 19:Calculation

[Unique parameters of measurement mode from 4byte]

Measurement mode 0:OFF

byte	Setting Data
4	Reserved (fixed as 0)
~	
91	

1:Height (when profile compression (time axis) is off)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0: Peak, 1: Bottom, 2: Knee, 3: Intsect (lines), 4: Intsect (lin-arc), 5: Center of circle, 6: Average
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

2:Step

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0: Peak, 1: Bottom, 2: Knee, 3: Intsect (lines), 4: Intsect (lin-arc), 5: Center of circle, 6: Average
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
47	
48	Reference target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
49	Referene target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5: Center of circle, 6: Average
50	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
51	Reserved (fixed as 0)
52	Unique parameters are assigned to each Reference target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

3:Position (when profile compression (time axis) is off)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5:Center of circle, 7:Edge
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

4:Center position, 5:Width

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5:Center of circle 7:Edge
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
47	
48	Reference target selection: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
49	Reference target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5:Center of circle, 7:Edge
50	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
51	Reserved (fixed as 0)
52	Unique parameters are assigned to each Reference target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

6:Thickness

byte	Setting Data
4	Measurement target selection: 8:Max thickness, 9:Min thickness, 10:Ave thickness, 11:Max thickness position, 12:Min thickness position
5	Position correction selection (HeadA): 0: No position correction, 1: Position correction1, 2: Position correction2
6	Position correction selection (HeadB): 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

7:Angle

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement reference selection: 0:Angle from X-axis, 1:Angle between lines
6	Angle range: 0:0~180deg, 1:-90~90deg
7	Measurement target position correction: 0: No position correction, 1: Position correction1, 2: Position correction2
8	Reference target position correction: 0: No position correction, 1: Position correction1, 2: Position correction2
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)
12	Measurement target Linear calculation area: 0:Area2 disabled, 1:Area2enabled
13	Reserved (fixed as 0)
14	Reserved (fixed as 0)
15	Reserved (fixed as 0)
16	Measurement target Linear calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Measurement target Linear calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
21	
22	
23	
24	Measurement target Linear calculation area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
25	
26	
27	
28	Measurement target Linear calculation area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
29	
30	
31	
32	Reference target Linear calculation area: 0:Area2 disabled, 1:Area2 enabled
33	Reserved (fixed as 0)
34	Reserved (fixed as 0)
35	Reserved (fixed as 0)
36	Reference target Linear calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
37	
38	
39	
40	Reference target Linear calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
41	
42	
43	
44	Reference target Linear calculation area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
45	
46	
47	
48	Reference target Linear calculation area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
49	
50	
51	
52	Reserved (fixed as 0)
~	
91	

8:R measurement

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Arc calculation area: 0: Area2 disabled, 1: Area2 enabled
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)
12	Arc calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Arc calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Arc calculation area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
21	
22	
23	
24	Arc calculation area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
25	
26	
27	
28	Reserved (fixed as 0)
~	
91	

9:Area

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement reference selection: 0:Reference for 1 line, 1:Reference for 2 lines, 2:Master reference
6	Measurement target position correction: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Measurement target1 position correction: 0: No position correction, 1: Position correction1, 2: Position correction2
8	Measurement target2 position correction: 0: No position correction, 1: Position correction1, 2: Position correction2
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)
12	Measurement area Area calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Measurement area Area calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
17	
18	
19	
20	Linear calculation area: 0:Area2 disabled, 1:Area2 enabled
21	Reserved (fixed as 0)
22	Reserved (fixed as 0)
23	Reserved (fixed as 0)
24	Reference straight line1 Linear calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
25	
26	
27	
28	Reference straight line1 Linear calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
29	
30	
31	
32	Reference straight line1 Area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
33	
34	
35	
36	Reference straight line1 Area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
37	
38	
39	
40	Linear calculation area: 0:Area2 disabled, 1:Area2 enabled
41	Reserved (fixed as 0)
42	Reserved (fixed as 0)
43	Reserved (fixed as 0)
44	Reference straight line2 Linear calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
45	
46	
47	
48	Reference straight line2 Linear calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
49	
50	
51	
52	Reference straight line2 area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
53	
54	
55	
56	Reference straight line2 area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
57	
58	
59	
60	Reserved (fixed as 0)
~	
91	

* Reference straight line2's settings are not needed when Reference for 1 line is selected.

* Reference straight line1, 2's settings are not needed whenMaster Reference is selected.

10:Master comparison (Z)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Area setting Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
9	
10	
11	
12	Area setting Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	
16	Reserved (fixed as 0)
~	
91	

11:Distance(point-point)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5: Center of circle
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
47	
48	Reference target selection: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
49	Reference target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5:Center of circle
50	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
51	Reserved (fixed as 0)
52	Unique parameters are assigned to each Reference target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

12:Distance (point-line)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:Bottom, 2:Knee, 3:Intsect (lines), 4:Contact (lin-arc), 5:Center of circle
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
47	
48	Reference target selection: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
49	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
50	Reserved (fixed as 0)
51	Reserved (fixed as 0)
52	Linear calculation area: 0: Area2 disabled, 1: Area2 enabled
53	Reserved (fixed as 0)
54	Reserved (fixed as 0)
55	Reserved (fixed as 0)
56	Linear calculation area Left: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
57	
58	
59	
60	Linear calculation area Right: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
61	
62	
63	
64	Area2 Left: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
65	
66	
67	
68	Area2 Right: any value in measurement range (0.01µm unit Sined 32-bit integer example: 5mm=500000)
69	
70	
71	
72	Reserved (fixed as 0)
~	
91	

13:Height (when profile compression (time axis) is on)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:bottom, 13:Middle value
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

14:Position (when profile compression (time axis) is on)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:bottom, 7:Edge
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

15:Deflection width (when profile compression (time axis) is on)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 14:P-P(Z), 15:P-P(X)
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

16:Height (simple 3D)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection:0:Peak,1:bottom,6:Average,16:P-P
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
15	
16	Ycoordinate start position: value within Batch point
17	
18	Ycoordinate endposition: value within Batch point
19	
20	Reserved (fixed as 0)
~	
91	

17:Step (simple 3D)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0:Peak, 1:bottom, 6:Average
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
15	
16	Measurement target Ycoordinate start position: value withinBatch point
17	
18	Measurement target Ycoordinate end position: value withinBatch point
19	
20	Reference target selection: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
21	Reference target selection: 0:Peak, 1:bottom, 6:Average
22	Reserved (fixed as 0)
23	Reserved (fixed as 0)
24	Unique parameters are assigned to each reference target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
31	
32	Reference target Ycoordinate start position: value withinBatch point
33	
34	Reference target Ycoordinate end position: value withinBatch point
35	
36	Reserved (fixed as 0)
~	
91	

18:Position (simple 3D)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Output coordinate: 0:Xcoordinate, 1:Ycoordinate
6	Measurement target selection: 0: Peak, 1: bottom
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
M	
M+1	Reserved (fixed as 0)
~	
15	
16	Ycoordinate start position: value withinBatch point
17	
18	Ycoordinate end position: value withinBatch point
19	
20	Reserved (fixed as 0)
~	
91	

19:Calculation

byte	Setting Data
4	Calculation mode: 0:Addition, 1:Subtraction. 2:Absolute value, 3:AVE, 4:P-P, 5:MAX, 6:MIN
5	Reserved (fixed as 0)
6	Reserved (fixed as 0)
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each calculation mode. The unique parameters are described below.
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

[Unique parameter of calculation from 8byte]

0:Addition 1:Subtraction

byte	Setting Data
8	Calculation target A: OUT number (example:OUT1:00h, OUT12:0Bh)
9	Calculation target B: OUT number (example:OUT1:00h, OUT12:0Bh)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)

2:Absolute value

byte	Setting Data
8	Target OUT: OUT number (example:OUT1:00h, OUT12:0Bh)
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)

3:AVE/ 4:P-P/ 5:MAX/ 6:MIN

byte	Setting Data
8	OUT1: 0:Do not use as calculation target, 1: Use as calculation target
~	OUT2~15: 0:Do not use as calculation target, 1: Use as calculation target
23	OUT16: 0:Do not use as calculation target, 1: Use as calculation target

<Measurement value hold count>

Type:10h~1Fh (10h~Program NO.0, 11h:Program NO.1, ･･･, 1F:Program NO.15)

Category:06h, Item:04h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Measurement value hold count: 0~999
1	
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Measurement value filter>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･, 1F:Program NO.15)

Category:06h, Item:05h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Measurement value filter: 0:OFF, 1:Moving Average, 2:Low-pass filter, 3:High-pass filter
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Measurement valuefilter detail>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:06h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

Unique parameters are assigned to each measurement value filter.

1:Moving average

byte	Setting Data
0	Averagecount: 0:4 times, 1:16 times, 2:64 times, 3:256 times, 4:1024 times, 5:4096 times
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

2:Low-pass filter

3:High-pass filter

byte	Setting Data
0	Cutoff frequency: 0:0.1Hz, 1:0.3Hz, 2:1Hz, 3:3Hz, 4:10Hz, 5:30Hz, 6:100Hz, 7:300Hz, 8:1000Hz
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Scaling measurement value1><Scaling display value1> <Scaling measurement value2><Scaling display value2>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:07h (Scaling measurement value1), 08h (Scaling display value1), 09h (Scaling measurement value2), 0Ah (Scaling display value2)

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Measurement value/Display value: 0.01µm unit Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦Measurement value/Display value ≦Display range upper limit for the minimum display unit.
1	
2	
3	

<Measuring mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:0Bh

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Measuring mode: 0:Normal, 1:Peak hold, 2:bottom hold, 3:Peak to Peak hold, 4:Average hold, 5:Sample hold, 6:Peak, 7:bottom, 8:Peak to Peak, 9:Average
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Measuring period>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:0Ch

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Measuring period: 0:Terminal/command, 1:Measurement area 2 :OUT reference, 3:Threshhold (level), 4:Threshhold (Edge)
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Unique parameters are assigned to each measuring period. The unique parameters are described below.
~	
15	

[Unique parameters of Measuring period from 4byte]

1:Measurement area

byte	Setting Data
4	Measure start position: value withinBatch point
5	
6	Measure end position: value withinBatch point
7	
8	Reserved (fixed as 0)
~	
15	

2:OUT reference

byte	Setting Data
4	Reference OUT: 0:OUT1, 1:OUT2･･･15:OUT16
5	Reserved (fixed as 0)
~	
15	

3:Threshold (level)

byte	Setting Data
4	Upper limit: 0.01µm unit. Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦Upper limit ≦display range upper limit for the minimum display unit
5	
6	
7	
8	Lower limit: 0.01µm unit. Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦Lower limit ≦dislay range upper limit for the minimum display unit
9	
10	
11	
12	Reserved (fixed as 0)
13	
14	
15	

4:Threshold (Edge)

byte	Setting Data
4	Edge threshold: 0.01µm unit. Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦Edge threshold ≦display range upper limit for the minimum display unit
5	
6	
7	
8	Edge direction: 0:Rising, 1:Falling
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)
12	Measurement count: when batch off:Integer from 1 to 999,999 when batch on:Integer from 1 to Batch point
13	
14	
15	

<Offset>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:0Dh

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Offset: 0.01µm unit. Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦Offset ≦display range upper limit for the minimum display unit
1	
2	
3	

<Tolerance upper/lower limit>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:0Eh (upper limit), 0Fh (lower limit)

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Tolerance upper/lower limit value: 0.01µm unit. Sined 32-bit integer. * Display range lower limit for the minimum display unit ≦ Tolerance upper/lower limit value ≦ display range upper limit for the minimum display unit
1	
2	
3	

<Hysteresis>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:10h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Hysteresis: 0.01µm unit. Sined 32-bit integer. *0 ≦ Hysteresis ≦ display range upper limit for the minimum display unit
1	
2	
3	

<ZERO>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:11h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	ZERO: 0:None, 1:ZERO1, 2:ZERO2
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<TIMING/RESET>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:12h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	TIMING/RESET: 0:None, 1:TIMING1/RESET1, 2:TIMING2/RESET2
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Zero reference value>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:06h, Item:13h

Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Zero reference value: 0.01µm unit. Sined 32-bit integer. *Display range lower limit for the minimum display unit ≦ Zero reference value ≦ display range upper limit for the minimum display unit
1	
2	
3	

• Terminal settings

<Judgment output setting>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:07h, Item:01h

Target1:00h~0Bh (OUT_PIN1~12) Target2~4:00h

byte	Setting Data
0	Setting method: 0:No setting, 1:AND, 2:OR
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	OUT1 judgment result: 0:Not specified, 2:HI, 4:GO, 8:LO
5	OUT2 judgment result: 0:Not specified, 2:HI, 4:GO, 8:LO
~	~
21	OUT16 judgment result: 0:Not specified, 2:HI, 4:GO, 8:LO

* The judgment result can be specified by bits with logical OR. When HI and GO are both specified, the measurement result value is “6”.

<Analog output target OUT>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:07h, Item:02h

Target1:00h (CH1), 01h (CH2) Target2~4:00h

byte	Setting Data
0	Target OUT: 0:OUT1, 1:OUT2･･･15:OUT16, 255:none
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Analog output scaling>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:07h, Item:03h

Target1:00h (CH1), 01h (CH2) Target2~4:00h

byte	Setting Data
0	OUT display value1: Sined 32-bit integer *1
1	
2	
3	
4	Output voltage1: Sined 32-bit integer -10.5V~10.5V (1mVunit)
5	
6	
7	
8	OUT display value2: Sined 32-bit integer *1
9	
10	
11	
12	Output voltage2: Sined 32-bit integer -10.5V~10.5V (1mVunit)
13	
14	
15	

* 1 setting range

Length (mm)…-999.999mm~999.999mm (0.01µm unit)

Area (mm^2)…-9999.99mm^2~9999.99mm^2 (0.00001mm^{2}unit)

Angle (deg)…-9999.99deg~9999.99deg (0.001degunit)

• Storage settings

<Storage target>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:08h, Item:01h

Target1~4:00h

byte	Setting Data
0	Storage target: 0:OFF, 1:OUT value, 2:Profile
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Storage condition>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:08h, Item:02h

Target1~4:00h

byte	Setting Data
0	Storage condition: 0:Terminal/Command, 1: OUT update, 2:OUT data (edge), 3:OUT data (level)
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)

<Storage data amount (Terminal/Command)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:08h, Item:03h

Target1~4:00h

byte	Setting Data
0	Storage data amount: 0 to buffer size upper limit. *Upper limit value is the max points can be set by LJ-Navigator2.
1	
2	
3	

<Storage data amount (OUT data (Edge))>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:08h, Item:04h

Target1~4:00h

byte	Setting Data
0	Storage data amount: 0 to buffer size upper limit. *Upper limit value is the max points can be set by LJ-Navigator2.
1	
2	
3	
4	Threshold: The range that can be input in the OUT minimum display unit. (0.01μm unit. Sined 32-bit integer)
5	
6	
7	
8	Hysteresis: 0 ≦ Hysteresis ≦ display range upper limit for the minimum display unit. (0.01μm unit. Sined 32-bit integer)
9	
10	
11	
12	Target OUT: 0:OUT1, 1:OUT2, 2:OUT3･･･15:OUT16
13	Edgedirection: 0:Rising, 1:Falling
14	Reserved (fixed as 0)
15	Reserved (fixed as 0)

<Storage data amount (OUT data (level)>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:08h, Item:05h

Target1~4:00h

byte	Setting Data
0	Upper limit value: The range that can be input in the OUT minimum display unit. (0.01μm unit. Sined 32-bit integer)
1	
2	
3	
4	Lower limit value: The range that can be input in the OUT minimum display unit. (0.01μm unit. Sined 32-bit integer)
5	
6	
7	
8	Target OUT: 0:OUT1, 1:OUT2･･･15:OUT16
9	Reserved (fixed as 0)
10	Reserved (fixed as 0)
11	Reserved (fixed as 0)

• Program name

<Program name>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)

Category:09h, Item:00h

Target1~4:00h

byte	Setting Data
0	Program name, byte0
1	Program name, byte1
2	Program name, byte2
~	~
23	Program name, byte23

* 24 characters max. 0 is not appended to the end.

* SHIFT-JIS

• Measurement area details

The correction target selection of feature point correction of position correction and the unique parameters of the measurement target of the measurement mode are shown below.The byte numbers shown here indicate the byte numbers from the start of the corresponding data block. (see the example at 12.4)

0:Peak, 1:bottom, 6:Average, 8:Maximum thickness, 9:Minimum thickness, 10:Average thickness, 11:Maximum thickness position, 12:Mimimum thickness position, 13:Middle value, 14:P-P (Z) (only when profile compression (time axis) is on),16:P-P (only for Height (simple 3D)

byte	Setting Data
0	Area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	

2:Knee

byte	Setting Data
0	Area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Knee shape: 0:Valley, 1:Peak
9	Detection direction: 0: + direction, 1: - direction
10	Detection No: 1~10
11	Sensitivity: 0~100

3:Intersection (lines)

byte	Setting Data	
0	Linear calculation area: 0: Area2 disabled, 1: Area2 enabled	Line1
1	Reserved (fixed as 0)	
2	Reserved (fixed as 0)	
3	Reserved (fixed as 0)	
4	Line calculation area Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
5		
6		
7		
8	Line calculation area Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
9		
10		
11		
12	Area2 Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
13		
14		
15		
16	Area2 Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
17		
18		
19		
20	Linear calculation area count: 0: Area2 disabled, 1: Area2 enabled	Line2
21	Reserved (fixed as 0)	
22	Reserved (fixed as 0)	
23	Reserved (fixed as 0)	
24	Line calculation area Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
25		
26		
27		
28	Line calculation area Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
29		
30		
31		
32	Area2 Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
33		
34		
35		
36	Area2 Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
37		
38		
39		

4:Contact (line-arc)

byte	Setting Data	
0	Linear calculation area: 0: Area2 disabled, 1: Area2 enabled	Line
1	Reserved (fixed as 0)	
2	Reserved (fixed as 0)	
3	Reserved (fixed as 0)	
4	Line calculation area Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
5		
6		
7		
8	Line calculation area Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
9		
10		
11		
12	Area2 Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
13		
14		
15		
16	Area2 Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
17		
18		
19		
20	Arc calculation area: 0: Area2 disabled, 1: Area2 enabled	Arc
21	Reserved (fixed as 0)	
22	Reserved (fixed as 0)	
23	Reserved (fixed as 0)	
24	Arc calculation area Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
25		
26		
27		
28	Arc calculation area Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
29		
30		
31		
32	Area2 Left: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
33		
34		
35		
36	Area2 Right: Any value in measurement range (0.01μm unit Sined 32-bit integer. example: 5mm=500000)	
37		
38		
39		

5:Centerof circle

byte	Setting Data
0	Arc calculation areacount: 0: Area2 disabled, 1: Area2 enabled
1	Reserved (fixed as 0)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Arc calculation area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm =500000)
5	
6	
7	
8	Arc calculation area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm =500000)
9	
10	
11	
12	Area2 Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm =500000)
13	
14	
15	
16	Area2 Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm =500000)
17	
18	
19	

7:Edge (when profile compression (time axis) is off)

byte	Setting Data
0	Edge measuring area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Edge measuring area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Edge direction: 0:Rising, 1:Falling
9	Detect direction: 0:+direction, 1:–direction
10	Detect No: 1~10
11	Reserved (fixed as 0)
12	Edge level: Any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	

7:Edge (when profile compression (time axis) is on)

byte	Setting Data
0	Edge measuring area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Edge measuring area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Detection target: 0:Upper profile, 1:Lower profile
9	Edge direction: 0:Rising, 1:Falling
10	Detect direction: 0:+direction, 1:–direction
11	Detect No: 1~10
12	Edge level: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	

15:P-P (X) (only for when profile compression (time axis) is on)

byte	Setting Data
0	Edge measuring area Left: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
1	
2	
3	
4	Edge measuring area Right: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
5	
6	
7	
8	Edge direction: 0:Rising, 1:Falling
9	Detect direction: 0:+direction, 1:–direction
10	Detect No: 1~10
11	Reserved (fixed as 0)
12	Edge level: any value in measurement range (0.01μm unit Sined 32-bit integer example: 5mm=500000)
13	
14	
15	

12.4 Examples of sending/receiving measurement mode settings

Example: When "Height (profile compression (time axis): OFF" is selected and "Average" height is measured.

<Measurement mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)
Category:06h, Item:03h
Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Minimum display unit (from①)
1	Measurement mode (from②)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	From 4byte, unique parameters are assigned to each measurement mode. For details on the unique parameters, see [Unique parameters of measurement mode from 4byte]
~	
N	

1:Height (when profile compression (time axis) is off)

byte	Setting Data
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0: Peak, 1: Bottom, 2: Knee, 3: Intsect (lines), 4: Intsect (lin–arc), 5: Center of circle, 6: Average
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Unique parameters are assigned to each measurement target. For details on the unique parameters, see "Measurement area details" (page 83).
~	
N	
N+1	Reserved (fixed as 0)
~	
91	

6:Average

byte	Setting Data
0	Area Left: any value in measurement range (0.01µm unit Sined 32–bit integer example: 5mm=500000)
1	
2	
3	
4	Area Right: any value in measurement range (0.01µm unit Sined 32–bit integer example: 5mm=500000)
5	
6	
7	

These can be summarized as follows:

<Measurement mode>

Type:10h~1Fh (10h:Program NO.0, 11h:Program NO.1, ･･･ , 1F:Program NO.15)
Category:06h, Item:03h
Target1:00h~0Fh (OUT1~16) Target2~4:00h

byte	Setting Data
0	Minimum display unit (from①)
1	Measurement mode (from②)
2	Reserved (fixed as 0)
3	Reserved (fixed as 0)
4	Measurement target head: 0: HeadA, 1: HeadB, 2: Combined profile (only when profile combine (wide) is on)
5	Measurement target selection: 0: Peak, 1: Bottom, 2: Knee, 3: Intsect (lines), 4: Intsect (lin–arc), 5: Center of circle, 6: Average
6	Position correction selection: 0: No position correction, 1: Position correction1, 2: Position correction2
7	Reserved (fixed as 0)
8	Area Left: any value in measurement range (0.01µm unit Sined 32–bit integer example: 5mm=500000)
9	
10	
11	
12	Area Right: any value in measurement range (0.01µm unit Sined 32–bit integer example: 5mm=500000)
13	
14	
15	
16	Reserved (fixed as 0)
~	
91	

Measurement mode
1:Height

Measurement target
6:Average

13 Using the high-speed data communication command

When the high-speed data communication command is used, data for currently measured profiles can be output from the controller to a computer at high speed. While performing measurements using the LJ-V Series, data integrated into the computer can be processed.

13.1 Preparation for high-speed data communication

Make sure of and/or carry out the following before performing high-speed data communication.
[Device]

- Use a network interface card that supports gigabit communication.
- Use a hub that supports gigabit communication.
- Use an ethernet cable that is either category 7 or above, or that supports 10GBASE-T.

[Settings]

- Change the controller operation mode to "high-speed (profile only)".

* To create programs more easily.*
When LJ-H2 (setting/monitoring software for the LJ-V7000 Series) is installed, the DLL sample program is stored in the folder that LJ-Navigator2 is installed in. Use this as a reference to make it easier to create programs.
Example) When LJ-Navigator2 is stored in C:\Program Files, the DLL sample program is stored in the following folder:
C:\Program Files\KEYENCE\LJ-Navigator 2\lib

- About Req and SendPos parameters which must be specified in request preparation before starting high-speed data communication

■ Req

Sendpos: Send start position.
0: From previous send complete position (from oldest data if first time)
1: From oldest data
2: From next data

* Specify 2 to read profiles after high-speed communication starts.
When 0 or 1 is selected, profiles stored in the controller are read before starting high-speed communication.

* Read profiles are deleted from the controller memory.

* If the sampling period is faster than the reading speed, the memory becomes FULL. If the setting specified for "Operation when memory full" in Common measurement settings is "Stop", profiles are not accumulated. If "Overwrite", data is overwritten from the oldest data.

* If there is a long period between the first high-speed data communication and the next, the memory will be overwritten (when "Operation when memory full" is set to "Overwrite"), and previously sent position data will be overwritten. In this case specifying 0 will cause an error.

[Performing high-speed data communication]
High-speed data communication uses one of two procedures listed below, according to the format and measurement contents of the target.

- Without using the batch setting
- Using the batch setting

Batch setting is a function that treats the specified number of profiles as a batch.
It can be set to on or off in LJ-navigator2.

[Checking read profiles]

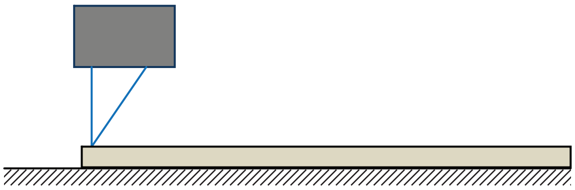
Profiles read using the callback function each have a header that contains the following information.

- Whether the encoder Z phase has been input. (Refer to "Profile header information structure" in "8.2 Structure definitions")
- Trigger counter - Indicates which number trigger from the start of measurements this profile is.
- Encoder count - The encoder count. (The encoder trigger count value can be checked in A-2 of the User's Manual.) If the encoder input period is faster than the set sampling period, the encoder input is not regarded as a trigger, but is counted as an encoder count value. (When a pulse exceeding the encoder minimum input time is input)

Use this information to specify the profile position according to the intended use.

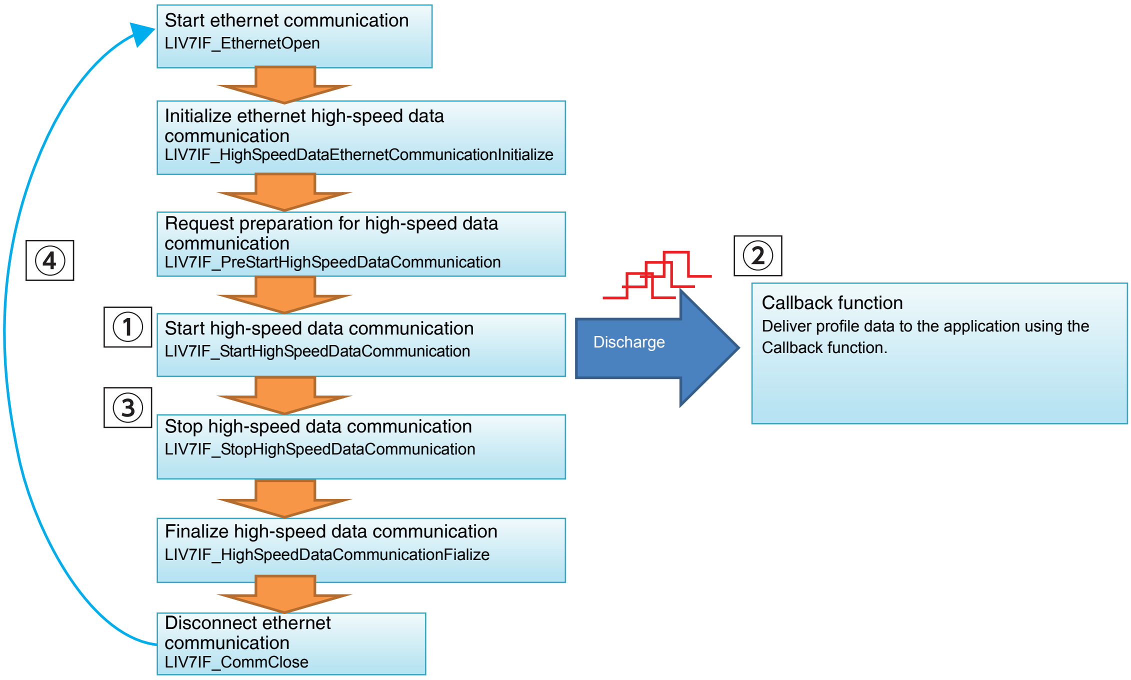
13.2 High-speed communication without using the batch setting

Use this to continuously measure the target profile, as in the diagram below.
Use this to measure and output the results of the profile with no upper data limit.



[Command procedure]

Commands are sent using the following procedure.



Start ethernet communication
LJV7IF_EthernetOpen

Initialize ethernet high-speed data communication
LJV7IF_HighSpeedDataEthernetCommunicationInitialize

Request preparation for high-speed data communication
LJV7IF_PreStartHighSpeedDataCommunication

① Start high-speed data communication
LJV7IF_StartHighSpeedDataCommunication

③ Stop high-speed data communication
LJV7IF_StopHighSpeedDataCommunication

Finalize high-speed data communication
LJV7IF_HighSpeedDataCommunicationFialize

Disconnect ethernet communication
LJV7IF_CommClose

② Callback function
Deliver profile data to the application using the Callback function.

① Start high-speed data communication

When the start high-speed data communication command is sent to the controller profile data measured on LJ-V starts to be output. Measurement is carried out on LJ-V each time a trigger is input.

② Callback function

Use the callback function to deliver profile data to the application each time the amount of profile data specified in high-speed data communication initialization is sent by the computer.

*1 The callback function is called in the following circumstances.

- The specified number of profiles are sent from the controller
- High-speed communication stops (settings are changed, a stop high-speed data communication command is sent, a clear memory command is sent)
- The program is switched

*2 When the number of profiles specified in batch settings are sent by the computer, the 16th bit of the 8.3.1.1 dwNotify parameter turns ON.

*3 When the number of profiles specified in the callback function is low, the delivery frequency becomes high and the computer load increases, which may cause the computer processing speed to decrease. Check and consider the load status of the application when setting the delivery amount.

③ Stop high-speed data communication

Stops profiles from being output from the controller to the computer.

When high-speed communication stops, the 1st bit of the 8.3.1.1 dwNotify parameter turns ON.

④ Restart high-speed data communication

To restart high-speed communication after it has stopped, start the high-speed data communication in the following order:

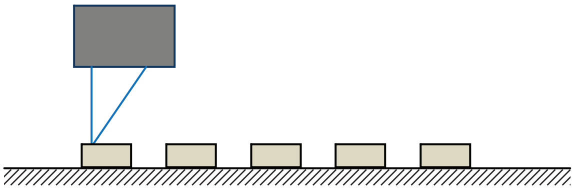
"Finalize high-speed data communication" → "Disconnect ethernet communication" → "Start ethernet communication" → "Initialize ethernet high-speed communication" → "Request preparation for Ethernet high-speed data communication"

13.3 High-speed communication using the batch function

Use this when measuring each target in the batch individually as it arrives, as in the diagram below.

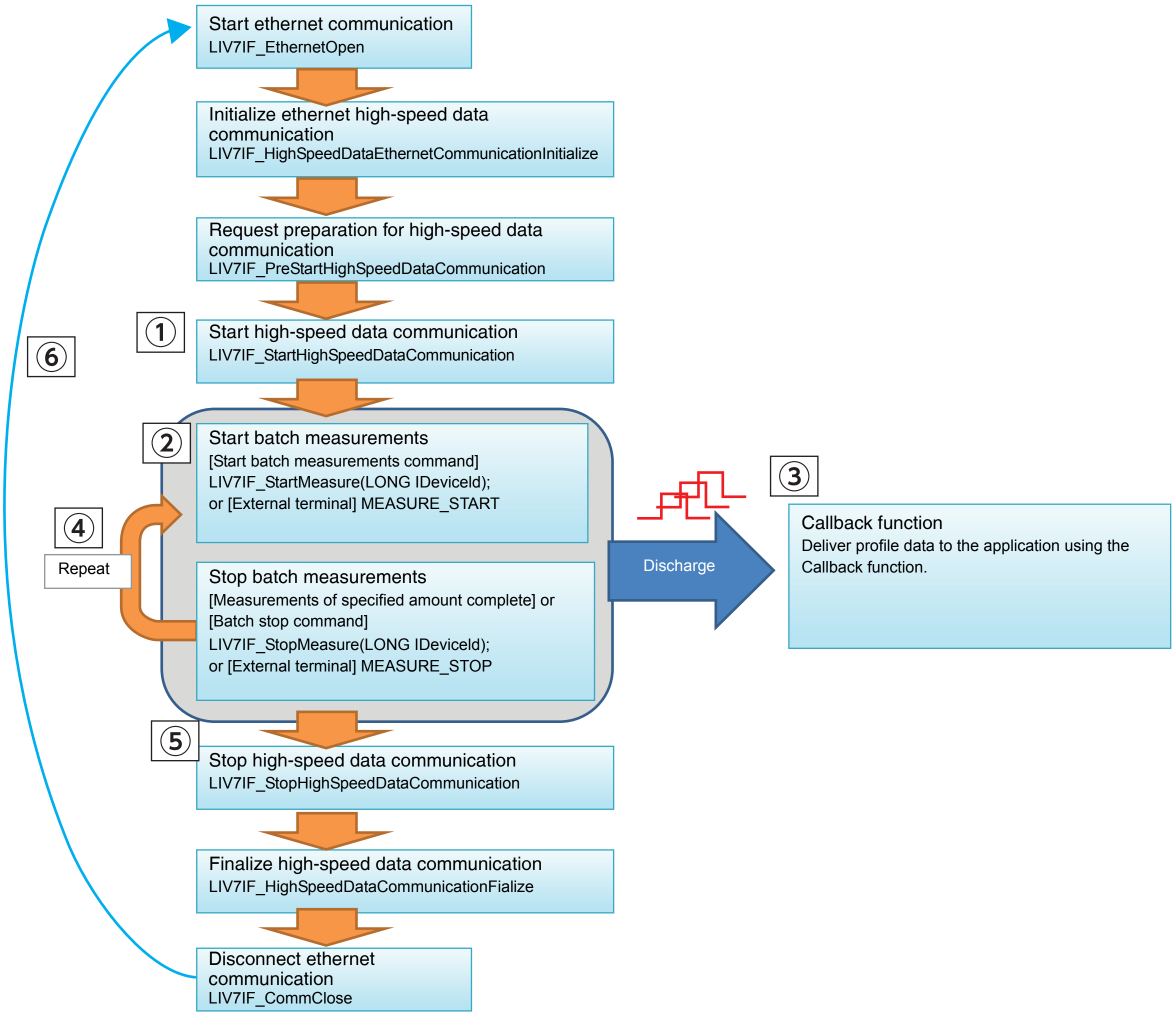
The batch function has the following advantages.

- Up to 15,000 profiles can be managed as one batch of data, making it easier to handle on the computer.
- When the previously set amount of data is measured, data measuring automatically finishes.
- When high-speed data communication starts, the timing of when profiles are sent to the computer can be controlled using the batch start timing control for the controller.



[Command procedure]

Commands are sent using the following procedure. Confirm that the controller batch setting is ON.


Start ethernet communication
LJV7IF_EthernetOpen
Initialize ethernet high-speed data communication
LJV7IF_HighSpeedDataEthernetCommunicationInitialize
Request preparation for high-speed data communication
LJV7IF_PreStartHighSpeedDataCommunication
①
⑥
Start high-speed data communication
LJV7IF_StartHighSpeedDataCommunication
②
Start batch measurements
[Start batch measurements command]
LJV7IF_StartMeasure(LONG lDeviceId);
or [External terminal] MEASURE_START
④
Repeat
Stop batch measurements
[Measurements of specified amount complete] or
[Batch stop command]
LJV7IF_StopMeasure(LONG lDeviceId);
or [External terminal] MEASURE_STOP
Discharge
③
Callback function
Deliver profile data to the application using the Callback function.
⑤
Stop high-speed data communication
LJV7IF_StopHighSpeedDataCommunication
Finalize high-speed data communication
LJV7IF_HighSpeedDataCommunicationFialize
Disconnect ethernet communication
LJV7IF_CommClose

① Start high-speed data communication

When the start high-speed data communication command is sent to the controller profile data measured on LJ-V starts to be output.

* When the batch setting is ON, trigger input cannot be received until batch measuring starts. For this reason, profiles are not output even if the start high-speed data communication command in step 1 is sent.

② Start batch measurements

Starts batch measurements. There are two ways to start batch measurements, as listed below.

- Terminal block: Page 11-10 of the LJ-V7000 User's Manual (Measure Start)
- Command: Start batch measurements command (9.2.4 Measurement control LONG LJV7IF_StartMeasure(LONG lDeviceId);)

When a trigger is input after batch measurement is started, the controller starts ouputting profiles to the computer.

③ Callback function

Use the callback function to deliver profile data to the application each time the amount of profile data specified in high-speed data communication initialization is sent by the computer.

*1 The callback function is called in the following circumstances.

- The specified number of profiles are sent from the controller
- High-speed communication stops (settings are changed, a stop high-speed data communication command is sent, a clear memory command is sent)
- The data in the batch is all acquired.
- The program is switched

*2 When the number of profiles specified in batch settings are sent by the computer, the 16th bit of the 8.3.1.1 dwNotify parameter turns ON.

*3 When the number of profiles specified in the callback function is low, the delivery frequency becomes high and the computer load increases, which may cause the computer processing speed to decrease. Check and consider the load status of the application when setting the delivery amount.

④ Repeat

Repeat batch start to continuously output profiles.

Batch measurement stops in the following circumstances.

- The specified number of profiles are acquired.
- The stop batch measurements signal is input

⑤ Stop high-speed data communication

Stops profiles from being output from the controller to the computer.

When high-speed communication stops, the 1st bit of the 8.3.1.1 dwNotify parameter turns ON.

⑥ Restart high-speed data communication

To restart high-speed communication after it has stopped, start the high-speed data communication in the following order:

"Finalize high-speed data communication" → "Disconnect ethernet communication" → "Start ethernet communication" → "Initialize ethernet high-speed communication" → "Request preparation for Ethernet high-speed data communication"

Revision History

Revision date	Revision number	Revision details
June 2012	2nd edition	Ver. 2
October 2012	1st revision	Z-phase specifications were added.
February 2014	2nd revision	
August 2014	2nd revision, 2nd edition	
July 2015	2nd revision, 3rd edition	

WARRANTIES AND DISCLAIMERS

(1) KEYENCE warrants the Products to be free of defects in materials and workmanship for a period of one (1) year from the date of shipment. If any models or samples were shown to Buyer, such models or samples were used merely to illustrate the general type and quality of the Products and not to represent that the Products would necessarily conform to said models or samples. Any Products found to be defective must be shipped to KEYENCE with all shipping costs paid by Buyer or offered to KEYENCE for inspection and examination. Upon examination by KEYENCE, KEYENCE, at its sole option, will refund the purchase price of, or repair or replace at no charge any Products found to be defective. This warranty does not apply to any defects resulting from any action of Buyer, including but not limited to improper installation, improper interfacing, improper repair, unauthorized modification, misapplication and mishandling, such as exposure to excessive current, heat, coldness, moisture, vibration or outdoors air. Components which wear are not warranted.

(2) KEYENCE is pleased to offer suggestions on the use of its various Products. They are only suggestions, and it is Buyer's responsibility to ascertain the fitness of the Products for Buyer's intended use. KEYENCE will not be responsible for any damages that may result from the use of the Products.

(3) The Products and any samples ("Products/Samples") supplied to Buyer are not to be used internally in humans, for human transportation, as safety devices or fail-safe systems, unless their written specifications state otherwise. Should any Products/Samples be used in such a manner or misused in any way, KEYENCE assumes no responsibility, and additionally Buyer will indemnify KEYENCE and hold KEYENCE harmless from any liability or damage whatsoever arising out of any misuse of the Products/Samples.

(4) **OTHER THAN AS STATED HEREIN, THE PRODUCTS/SAMPLES ARE PROVIDED WITH NO OTHER WARRANTIES WHATSOEVER. ALL EXPRESS, IMPLIED, AND STATUTORY WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF PROPRIETARY RIGHTS, ARE EXPRESSLY DISCLAIMED. IN NO EVENT SHALL KEYENCE AND ITS AFFILIATED ENTITIES BE LIABLE TO ANY PERSON OR ENTITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, PUNITIVE, SPECIAL OR CONSEQUENTIAL DAMAGES (INCLUDING, WITHOUT LIMITATION, ANY DAMAGES RESULTING FROM LOSS OF USE, BUSINESS INTERRUPTION, LOSS OF INFORMATION, LOSS OR INACCURACY OF DATA, LOSS OF PROFITS, LOSS OF SAVINGS, THE COST OF PROCUREMENT OF SUBSTITUTED GOODS, SERVICES OR TECHNOLOGIES, OR FOR ANY MATTER ARISING OUT OF OR IN CONNECTION WITH THE USE OR INABILITY TO USE THE PRODUCTS, EVEN IF KEYENCE OR ONE OF ITS AFFILIATED ENTITIES WAS ADVISED OF A POSSIBLE THIRD PARTY'S CLAIM FOR DAMAGES OR ANY OTHER CLAIM AGAINST BUYER.** In some jurisdictions, some of the foregoing warranty disclaimers or damage limitations may not apply.

BUYER'S TRANSFER OBLIGATIONS:

If the Products/Samples purchased by Buyer are to be resold or delivered to a third party, Buyer must provide such third party with a copy of this document, all specifications, manuals, catalogs, leaflets and written information provided to Buyer pertaining to the Products/Samples.

Specifications are subject to change without notice.

KEYENCE CORPORATION

1-3-14, Higashi-Nakajima, Higashi-Yodogawa-ku, Osaka, 533-8555, Japan PHONE: +81-6-6379-2211

www.keyence.com

AUSTRIA	Phone: +43 22 36-3782 66-0
BELGIUM	Phone: +32 1 528 1222
BRAZIL	Phone: +55-11-3045-4011
CANADA	Phone: +1-905-366-7655
CHINA	Phone: +86-21-68757500
CZECH REPUBLIC	Phone: +420 222 191 483
FRANCE	Phone: +33 1 56 37 78 00
GERMANY	Phone: +49 6102 36 89-0
HONG KONG	Phone: +852-3104-1010
HUNGARY	Phone: +36 1 802 73 60
INDIA	Phone: +91-44-4963-0900
INDONESIA	Phone: +62-21-2966-0120
ITALY	Phone: +39-02-6688220
KOREA	Phone: +82-31-789-4300
MALAYSIA	Phone: +60-3-2092-2211
MEXICO	Phone: +52-55-8850-0100
NETHERLANDS	Phone: +31 40 20 66 100
POLAND	Phone: +48 71 36861 60
ROMANIA	Phone: +40 269-232-808
SINGAPORE	Phone: +65-6392-1011
SLOVAKIA	Phone: +421 2 5939 6461
SLOVENIA	Phone: +386 1-4701-666
SWITZERLAND	Phone: +41 43-45577 30
TAIWAN	Phone: +886-2-2718-8700
THAILAND	Phone: +66-2-369-2777
UK & IRELAND	Phone: +44-1908-696900
USA	Phone: +1-201-930-0100
VIETNAM	Phone: +84-4-3760-6214

A4WW1-MAN-1035

122020E 1075-3 376GB Printed in Japan

376GB-3