

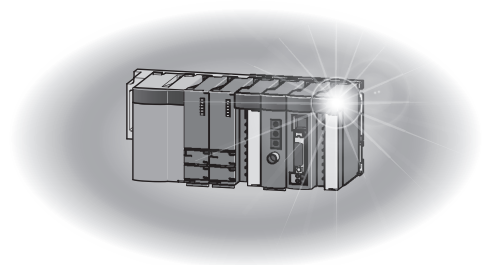


Mitsubishi Programmable Controller

MELSEC **Q** series

# High Speed Data Communication Module Programming Manual

---







# SAFETY PRECAUTIONS

(Read these precautions before using this product.)

Before using this product, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly.

Note that these precautions apply only to this product. For the safety precautions of the programmable controller system, please read the User's Manual for the CPU module used.

In this manual, the safety precautions are classified into two levels: "⚠ WARNING" and "⚠ CAUTION".


 <b>WARNING</b>	Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.
 <b>CAUTION</b>	Indicates that incorrect handling may cause hazardous conditions, resulting in minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠ CAUTION" may lead to serious consequences.

Always follow the instructions of both levels because they are important to personal safety.

Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

## [Design precautions]

 <b>WARNING</b>	<ul style="list-style-type: none"><li>● Configure safety circuits external to the programmable controller to ensure that the entire system operates safely even when a fault occurs in the external power supply or the programmable controller. Failure to do so may result in an accident due to an incorrect output or malfunction.</li><li>● For the operating status of each station after a communication failure, refer to relevant manuals for the network. Erroneous outputs and malfunctions may lead to accidents. Not doing so can cause an accident due to false output or malfunction.</li><li>● To maintain the safety of the programmable controller system against unauthorized access from external devices via the network, take appropriate measures. To maintain the safety against unauthorized access via the Internet, take measures such as installing a firewall.</li><li>● When changing data of the running programmable controller from a peripheral connected to the CPU module or from a personal computer connected to an intelligent function module or special function module, configure an interlock circuit in the sequence program to ensure that the entire system will always operate safely.</li></ul> <p>For program modification and operating status change, read relevant manuals carefully and ensure the safety before operation.</p> <p>Especially in the above mentioned control operations that are performed from an external device to a remote programmable controller, any problems on the programmable controller side may not be dealt with promptly due to abnormal data communication.</p> <p>To prevent this, configure an interlock circuit in the sequence program, and determine corrective actions to be taken between the external device and CPU module in case of a communication failure.</p> <ul style="list-style-type: none"><li>● Do not write any data in the "system area" of the buffer memory in the intelligent function module. Also, do not use any "use prohibited" signals as an output signal from the programmable controller CPU to the intelligent function module.</li></ul> <p>Doing so may cause malfunction of the programmable controller system.</p>
--	--

## [Design precautions]

---

### **CAUTION**

- To change the operating status of the programmable controller CPU from a connected device (remote run/stop, etc.), set the initial timing of the network parameter to "Always standby for OPEN (can communicate during STOP)".

If the initial timing is set to "Do not wait for OPEN (cannot communicate during STOP)", then when the connected device executes a remote STOP, the communication line will be closed.

Subsequently, it will not be possible to re-open the connection from the programmable controller CPU, and the connected equipment will not be able to execute a remote RUN either.

- During registering each setting, do not power OFF the mounted module or reset the CPU module. Otherwise, data in the CompactFlash card will be undefined. Therefore, resetting and re-registering data are required.

This may also cause a module failure or malfunctions.

---

## [Operating precautions]

---

### **WARNING**

- Ensure safety before controlling a running programmable controller (e.g. data modification).
- Do not write any data in the "system area" of the buffer memory in the intelligent function module. Also, do not use any "use prohibited" signals as an output signal from the CPU module to the intelligent function module.

Doing so may cause malfunction of the programmable controller system.

---

# CONDITIONS OF USE FOR THE PRODUCT

---

(1) Mitsubishi programmable controller ("the PRODUCT") shall be used in conditions;

- i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and
- ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI'S USER, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above, restrictions Mitsubishi may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTS are required. For details, please contact the Mitsubishi representative in your region.

## INTRODUCTION

---

Thank you for purchasing the Mitsubishi MELSEC-Q series programmable controllers.

This manual describes the functions and programming to use the following module.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC-Q series programmable controller to handle the product correctly.

# CONTENTS

SAFETY PRECAUTIONS .....	1
CONDITIONS OF USE FOR THE PRODUCT .....	3
INTRODUCTION .....	3
RELATED MANUALS .....	7
HOW TO READ THIS MANUAL .....	8
TERMS .....	9

## PART 1 High Speed Data Communication Library

<b>CHAPTER 1 OVERVIEW</b> .....	<b>12</b>
<b>CHAPTER 2 PROCEDURE FROM PROGRAM CREATION TO EXECUTION</b> .....	<b>14</b>
<b>CHAPTER 3 FUNCTION LIST</b> .....	<b>16</b>
<b>CHAPTER 4 PROGRAM CREATION OVERVIEW</b> .....	<b>18</b>
4.1 Processing Flow for Streaming Transfer .....	18
4.2 Processing Flow for Data Reading .....	19
4.3 Processing Flow for Data Writing .....	19

## PART 2 CREATING C# PROGRAMS

<b>CHAPTER 5 CREATING C# PROGRAMS</b> .....	<b>22</b>
5.1 Preparing Development Environment .....	22
Importing High Speed Data Communication Library .....	22
Using High Speed Data Communication Library .....	23
5.2 Designing Programs .....	24
Streaming transfer .....	24
Receiving streaming transfer data .....	25
Receiving notifications of module operation status changes .....	26
Reading data using label specification .....	27
Writing data using label specification .....	28
Reading data using device specification .....	29
Writing data using device specification .....	30
Programming precautions .....	31
5.3 Class and Method .....	32
Class list .....	33
Communication class .....	34
LabelGroupList class .....	56
LabelGroup class .....	57
DataLabel class .....	59
RegisteredLabelName class .....	61
ReceiveData class .....	63
RecordData class .....	64
DataValue class .....	68
ConnectionDeviceList class .....	74

ConnectionDevice class . . . . .	75
ConnectionDeviceDetail class . . . . .	78
DataCommunicationException class . . . . .	81
DataCommunicationError enumerator . . . . .	82
<b>5.4 Build C# Program . . . . .</b>	<b>85</b>

## **CHAPTER 6 EXECUTING C# USER PROGRAM . . . . . 87**

<b>6.1 Copying Execution File for User Program to Server Personal Computer . . . . .</b>	<b>87</b>
<b>6.2 Executing Processing from Server Personal Computer . . . . .</b>	<b>87</b>

## **PART 3 CREATING Java PROGRAMS**

### **CHAPTER 7 CREATING Java PROGRAMS . . . . . 90**

<b>7.1 Preparing Development Environment . . . . .</b>	<b>90</b>
Copying High Speed Data Communication Library . . . . .	90
Using High Speed Data Communication Library . . . . .	90
<b>7.2 Designing Programs . . . . .</b>	<b>91</b>
Streaming transfer . . . . .	91
Receiving streaming transfer data . . . . .	92
Receiving notifications of module operation status changes . . . . .	93
Reading data using label specification . . . . .	94
Writing data using label specification . . . . .	95
Reading data using device specification . . . . .	96
Writing data using device specification . . . . .	97
Programming precautions . . . . .	98
<b>7.3 Class and Method . . . . .</b>	<b>99</b>
Class list . . . . .	99
Communication class . . . . .	101
Notification class . . . . .	123
LabelGroupList class . . . . .	125
LabelGroup class . . . . .	126
DataLabel class . . . . .	128
RegisteredLabelName class . . . . .	130
ReceiveData class . . . . .	132
RecordData class . . . . .	133
DataValue class . . . . .	137
ConnectionDeviceList class . . . . .	147
ConnectionDevice class . . . . .	148
ConnectionDeviceDetail class . . . . .	151
DataCommunicationException class . . . . .	154
DataCommunicationError enumerator . . . . .	155
<b>7.4 Compiling Java User Program . . . . .</b>	<b>158</b>

### **CHAPTER 8 EXECUTING Java USER PROGRAM . . . . . 159**

<b>8.1 Copying Execution File for User Program to Server Personal Computer . . . . .</b>	<b>159</b>
<b>8.2 Executing Processing from Server Personal Computer . . . . .</b>	<b>159</b>

# PART 4 TROUBLESHOOTING

---

## CHAPTER 9 TROUBLESHOOTING 162

---

9.1	Troubleshooting related to user programs . . . . .	162
9.2	Troubleshooting related to network connection . . . . .	163
9.3	Troubleshooting related to label functions . . . . .	165
9.4	Troubleshooting related to streaming transfer . . . . .	165
9.5	Troubleshooting related to data read and data write functions . . . . .	167

## APPENDIX 168

---

Appendix 1	Sample Program . . . . .	168
	How to use the C# sample program . . . . .	168
	How to use the Java sample program . . . . .	171

## INDEX 176

---

REVISIONS . . . . .	178
WARRANTY . . . . .	179
TRADEMARKS . . . . .	180



# RELATED MANUALS

The manuals related to this product are shown below.

Refer to the following tables when ordering required manuals.

Manuals in printed form are sold separately for single purchase. Order a manual by quoting the manual number (model code) listed in the table above.

## CPU module user's manual to be used

Manual name <Manual number, Model code>	Description
High Speed Data Communication Module User's Manual <SH-081162ENG, 13JZ86>	Explains the system configuration, functions, and operating methods of High Speed Data Communication Module.
QCPU User's Manual (Hardware Design, Maintenance and Inspection) <SH-080483ENG, 13JR73>	Specifications of the hardware (CPU modules, power supply modules, base units, batteries, and memory cards), system maintenance and inspection, and troubleshooting.
Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals) <SH-080808ENG, 13JZ28>	Explains the programming methods, devices, and functions of Qn(H)/QnPH/QnPRHCPU module.
QnUCPU User's Manual (Function Explanation, Program Fundamentals) <SH-080807ENG, 13JZ27>	Explains the programming methods, devices, and functions of QnUCPU module.
MELSEC-L CPU Module User's Manual (Hardware Design, Maintenance and Inspection) <SH-080890ENG, 13JZ36>	Specifications of the CPU modules, power supply modules, display unit, branch module, extension module, SD memory cards, and batteries, information on how to establish a system, maintenance and inspection, and troubleshooting.
MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals) <SH-080889ENG, 13JZ35>	Explains the programming methods, devices, and functions of LCPU module.
C Controller Module User's Manual (Hardware Design, Function Explanation) <SH-080766ENG, 13JZ17>	Explains the programming methods, and functions of C controller module.
MELSEC-Q C Controller Module User's Manual <SH-081130ENG, 13JZ75>	Explains the system configuration, specifications, functions, handling methods, wiring, troubleshooting, and programming and function of the C Controller module.

## Operating manual

Manual name <Manual number, Model code>	Description
GX Works2 Version 1 Operating Manual (Common) <SH-080779ENG, 13JU63>	Explains the system configuration of GX Works2 and the functions common to Simple project and Structured project such as parameter setting, operation method for the online function.
Setting/Monitoring Tools for the C Controller Module Version 4 Operating Manual <SH-081131ENG, 13JU76>	Explains the procedures to configure and operate a system for Setting/monitoring tools for the C Controller module are described.

# HOW TO READ THIS MANUAL

The following explains the page composition and symbols in this manual. The contents of the example page used here are different from the actual contents for the intention of explaining how to use this manual.

"" indicates screen name or screen item.

indicates the reference manual.

[ ] indicates the items which are displayed on the window.

1. indicates an operation procedure.

indicates an example such as setting or operation.

indicates the reference page.

### 5.3 Intelligent Function Module Switch Setting

With the GX Works2 intelligent function module switch setting, configure the mode setting, default operation setting, response monitoring time setting, and compatibility setting.

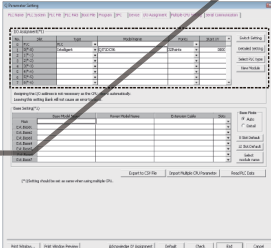
After writing data to a personal computer, the content of the intelligent function module switch settings are enabled by powering ON from OFF only by resetting the programmable controller CPU.

For the operation of GX Works2, refer to the following manual.

GX Works2 Version 1 Operating Manual (Common)

**Screen display**

• Select [Parameter] → [PLC parameter] → [I/O Assignment] tab in the project window of GX Works2.



**Operating procedure**

1. On the slot where the High Speed Data Communication Module is mounted, set the items in the table below and click the [switch setting] button

Item	Description
Type	Select "Intelligent".
Module name	Enter the model name.
Points	Select 32 points.
Start XY	Enter the High Speed Data Communication Module start I/O number.
[Detailed Setting] button	For a multiple CPU system, specify the control CPU of the High Speed Data Communication Module.

Press to D511 g auto refresh area 0000 to 00FF

**Point** The default operation setting is used to change the settings of the High Speed Data Communication Module connected to the configuration personal computer on a 1:1 basis.

Page 64, Connection system equipment

**Restriction** If the IP address duplication detection function is not supported by the target device, the duplicate IP address is not detected.

Indicates the section of currently open page.

5

**Point** indicates the useful tip.

**Restriction** indicates the particular attention.

5 INSTALLATION AND WIRING

5.3 Intelligent Function Module Switch Setting

113

# TERMS

This manual uses the terms listed in the following table unless otherwise noted

Term	Description
Account	Designates the right to use the High Speed Data Communication Module or the ID necessary when using the module
CompactFlash card	A storage card regulated by the "CF+ and CompactFlash Specification" issued by the CompactFlash Association The memory card required for operating the High Speed Data Communication Module.
Configuration Tool	An abbreviation for High Speed Data Communication Module Configuration Tool This tool configures and maintains High Speed Data Communication Module. Configuration Tool is included in High Speed Data Communication Module Configuration Tool.
Connection via a hub	A method of connecting the High Speed Data Communication Module and a personal computer to a local area network The High Speed Data Communication Module's IP address must be specified when connecting. Multiple High Speed Data Communication Modules can be accessed from a personal computer over a network.
CPU module	A generic term for MELSEC-Q series and MELSEC-L series CPU module.
Daylight saving time (summer time)	A system where clocks are set ahead for a specified period during summer
Development personal computer	Indicates that a personal computer for creating user applications using High Speed Data Communication Library
Direct connection	A connection method using an Ethernet cable to connect the High Speed Data Communication Module and a personal computer on a 1:1 basis They can be easily connected without knowing the IP address.
High Speed Data Communication Library	An abbreviation for Visual C# library and Java library The library to communicate server personal computer to High Speed Data Communication Module High Speed Data Communication Library is included in High Speed Data Communication Module Tool.
High Speed Data Communication Module	An abbreviation for QJ71DC96 MELSEC-Q Series-compatible High Speed Data Communication Module
High Speed Data Communication Module Tool	A generic term for High Speed Data Communication Module Configuration Tool (SW1DNN-DCUTL) and High Speed Data Communication Library
JDK	An abbreviation for Java™ Platform, Standard Edition Development Kit
Local time	The local time that people use from day to day. It is affected by time zone and whether daylight saving (summer time) is in effect
Server personal computer	Indicates that the transfer destination personal computer from High Speed Data Communication Module
Streaming transfer	A method of transferring data by sending a continuous stream of data over the network from the server to the transfer destination, which processes this stream continuously
Time zone	The standard time zones for each region of the world Each nation uses the time difference (within ± 12 hours) from the time at the Greenwich Observatory in England (GMT) as the standard time. Regions using the same time difference are called a time zone. The standard time for Japan is 9 hours ahead of GMT. Depending on the country, they may also use daylight saving time in summer.
UTC (coordinated universal time)	This is a universal time that does not change regardless of time zone or whether daylight saving (summer time) is in effect. It is used to compare data times across time zones.
Windows Vista® or later	A generic term for Windows Vista®, Windows® 7, Windows 8, Windows 8.1, and Windows 10.

# MEMO

---

# PART 1

# High Speed Data Communication Library

This part explains the functions of High Speed Data Communication Library and overview of the programs.

1 OVERVIEW

---

2 PROCEDURE FROM PROGRAM CREATION TO EXECUTION

---

3 FUNCTION LIST

---

4 PROGRAM CREATION OVERVIEW

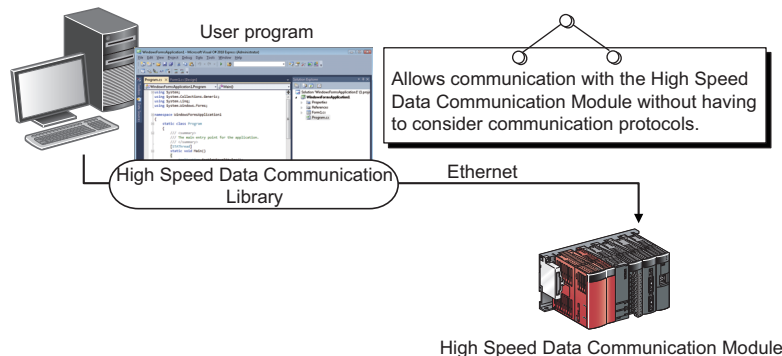
---

# 1 OVERVIEW


High Speed Data Communication Library prepares communication methods between the server personal computer and High Speed Data Communication Module.

Use High Speed Data Communication Library to create user programs allowing High Speed Data Communication Module to communicate.


This allows communication with High Speed Data Communication Module without having to consider communication protocols.




Use High Speed Data Communication Library when creating the following programs.

- Streaming transfer programs (  Page 18 Processing Flow for Streaming Transfer)

Programs to receive the streaming transfer data transferred from High Speed Data Communication Module via Ethernet.

- Data reading programs (  Page 19 Processing Flow for Data Reading)

Programs to read data from the target CPU module on demand.

- Data writing programs (  Page 19 Processing Flow for Data Writing)

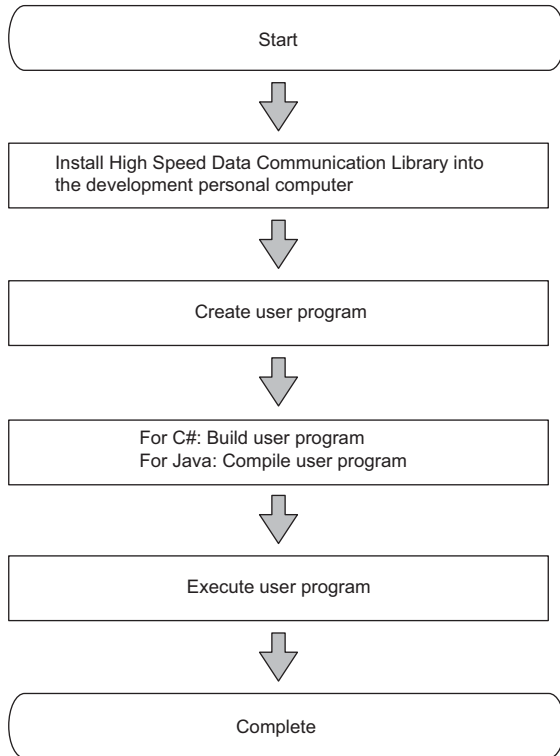
Programs to write data to the target CPU module on demand.



## 2 PROCEDURE FROM PROGRAM CREATION TO EXECUTION

---

The following shows the steps from creating a user program using High Speed Data Communication Library to execute the program.







# 3 FUNCTION LIST

The major functions of High Speed Data Communication Library are shown below.

○: Required, △: Execute if needed, —: Unnecessary

Item		Description	Supported program <sup>*1</sup>			Reference
			Transfer	Reading	Writing	
Connection function		A function to connect the server personal computer to High Speed Data Communication Module.	○	○	○	Page 34 Communication class Page 101 Communication class
Label list acquisition function		A function to acquire a list of labels set on High Speed Data Communication Module.	○	△	△	Page 41 GetLabelList Page 42 GetLabelListWithComment Page 107 getLabelList Page 108 getLabelListWithComment
Streaming transfer setting/receiving function	Transfer label registration	A function to register the target labels for streaming transfer to High Speed Data Communication Module.	○	—	—	Page 18 Processing Flow for Streaming Transfer
	Streaming transfer start	A function to request High Speed Data Communication Module to start streaming transfer to a server personal computer.	○	—	—	
	Streaming transfer stop	A function to request High Speed Data Communication Module to stop streaming transfer to a server personal computer.	○	—	—	
	Streaming transfer notification	A function to notify a user of a reception of data sent by streaming transfer from High Speed Data Communication Module.	○	—	—	
	Re-transfer start function	A function to execute the communication processing or streaming transfer start processing with fixed scan interval when connecting with buffering data transfer.	△	—	—	—
State change notification function		A function to notify a user of state changes of connected High Speed Data Communication Module.	△	△	△	Page 26 Receiving notifications of module operation status changes
On-demand function	Data read	A function to read specified data from a CPU module to a server personal computer.	—	○	—	Page 19 Processing Flow for Data Reading
	Data write	A function to write specified data from a server personal computer to a CPU module.	—	—	○	Page 19 Processing Flow for Data Writing

<sup>\*1</sup> The supported program by each function is written as follow.

Transfer: Streaming transfer program

Reading: Data reading program

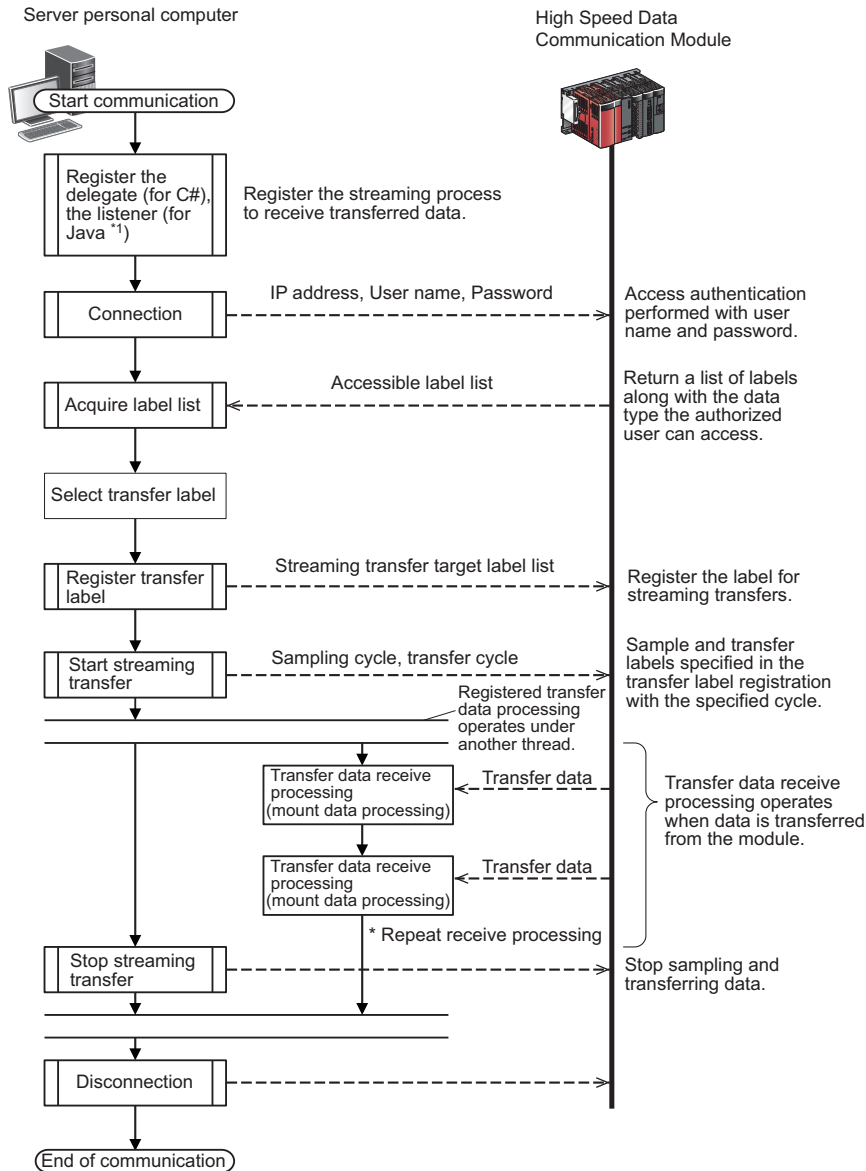
Writing: Data writing programs



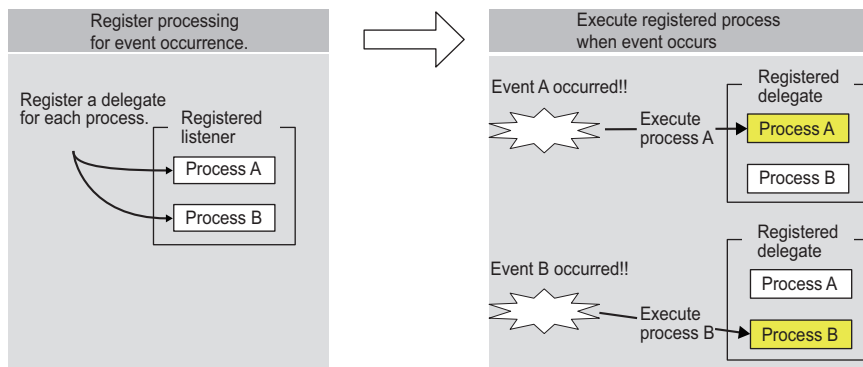
# 4 PROGRAM CREATION OVERVIEW

## 4.1 Processing Flow for Streaming Transfer

A streaming transfer program executes processes in the order listed below.



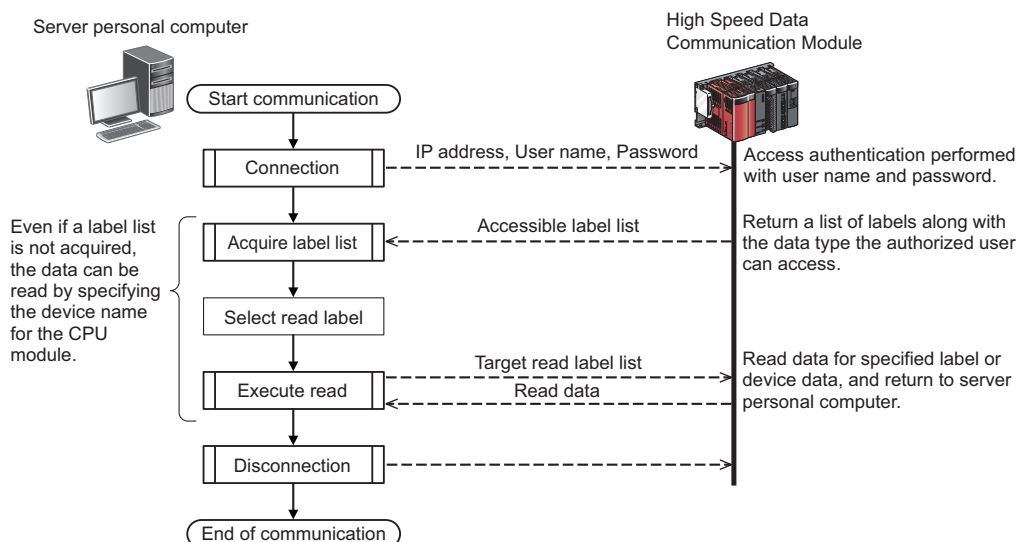
\*1 A delegate and listener recognize an event occurrence, and determine how to handle the event.  
Register the processing for event occurrence to execute registered processing when the event occurs.



\* For Java, consider delegate as listener.

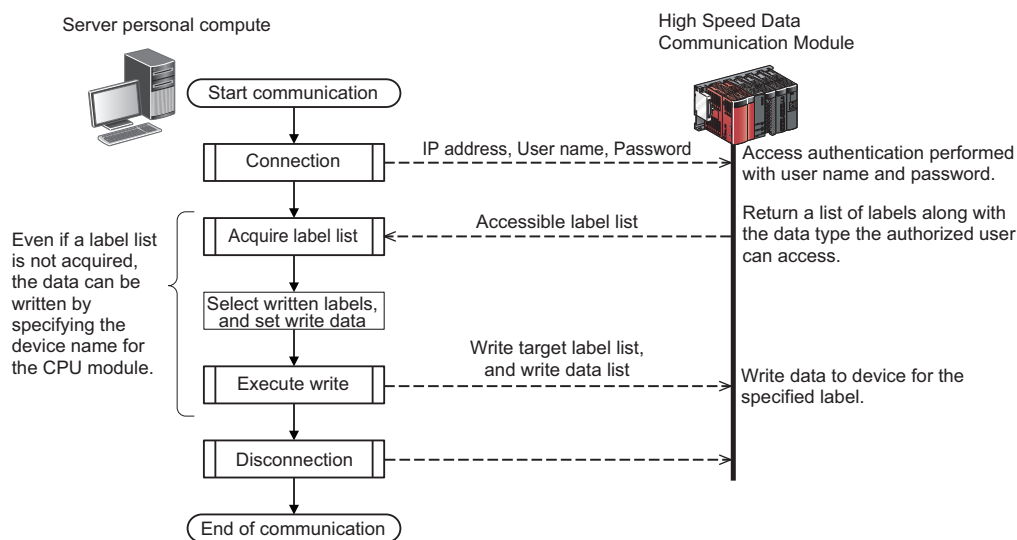
## 4.2 Processing Flow for Data Reading

A data reading program executes processes in the order listed below.



## 4.3 Processing Flow for Data Writing

A data writing program executes processes in the order listed below.





## PART 2

# CREATING C# PROGRAMS

This part explains the creating method of C# program and class method.

5 CREATING C# PROGRAMS

---

6 EXECUTING C# USER PROGRAM

---

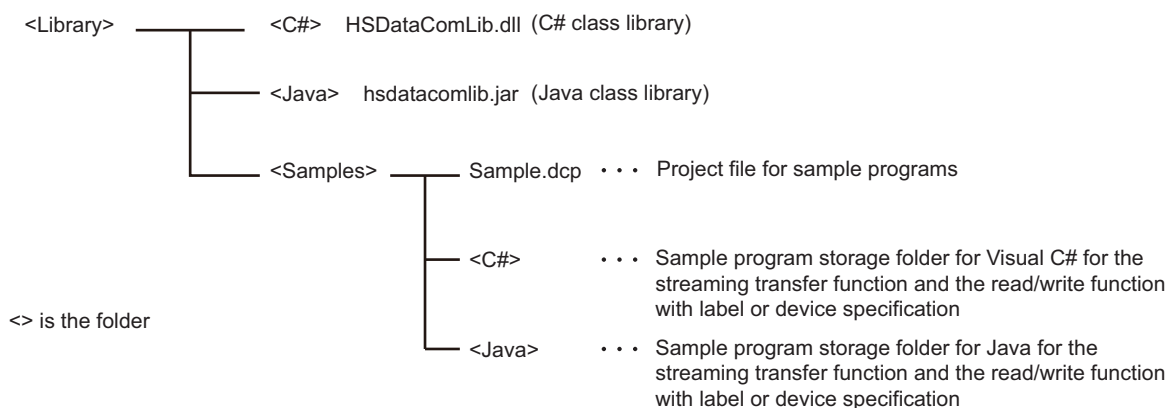
# 5 CREATING C# PROGRAMS

## 5.1 Preparing Development Environment

### Importing High Speed Data Communication Library

Copy High Speed Data Communication Library under the "Library" folder in the compressed file of this product to a folder (the working folder when creating the user program) on a personal computer.

The following diagram shows the structure of the "Library" folder.





# Using High Speed Data Communication Library

To use the classes and methods provided by High Speed Data Communication Library, High Speed Data Communication Library needs to be referred on the source code of the user program.

## Using C# class library

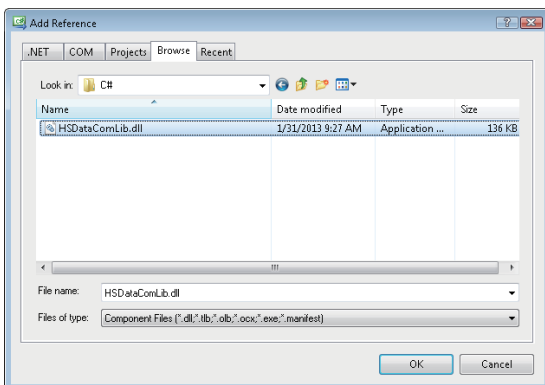
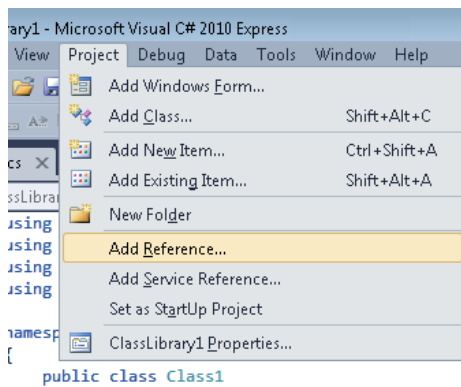
The following explains the procedure for using High Speed Data Communication Library.

- 1) Add a reference for High Speed Data Communication Library to the project of user program.
- 2) Use the using Directive to refer High Speed Data Communication Library.

### ■Adding a reference for High Speed Data Communication Library

Visual Studio® 2010 Express is used in the following example.

1. Start Visual Studio® 2010 Express, and create a new project.
2. Select [Project] ⇒ [Add Reference].



3. Select "HSDDataComLib.dll" on the [Browse] tab, and click the [OK] button.

Specify the folder to which High Speed Data Communication Library is copied for the storage location of "HSDDataComLib.dll".

### ■using Directive

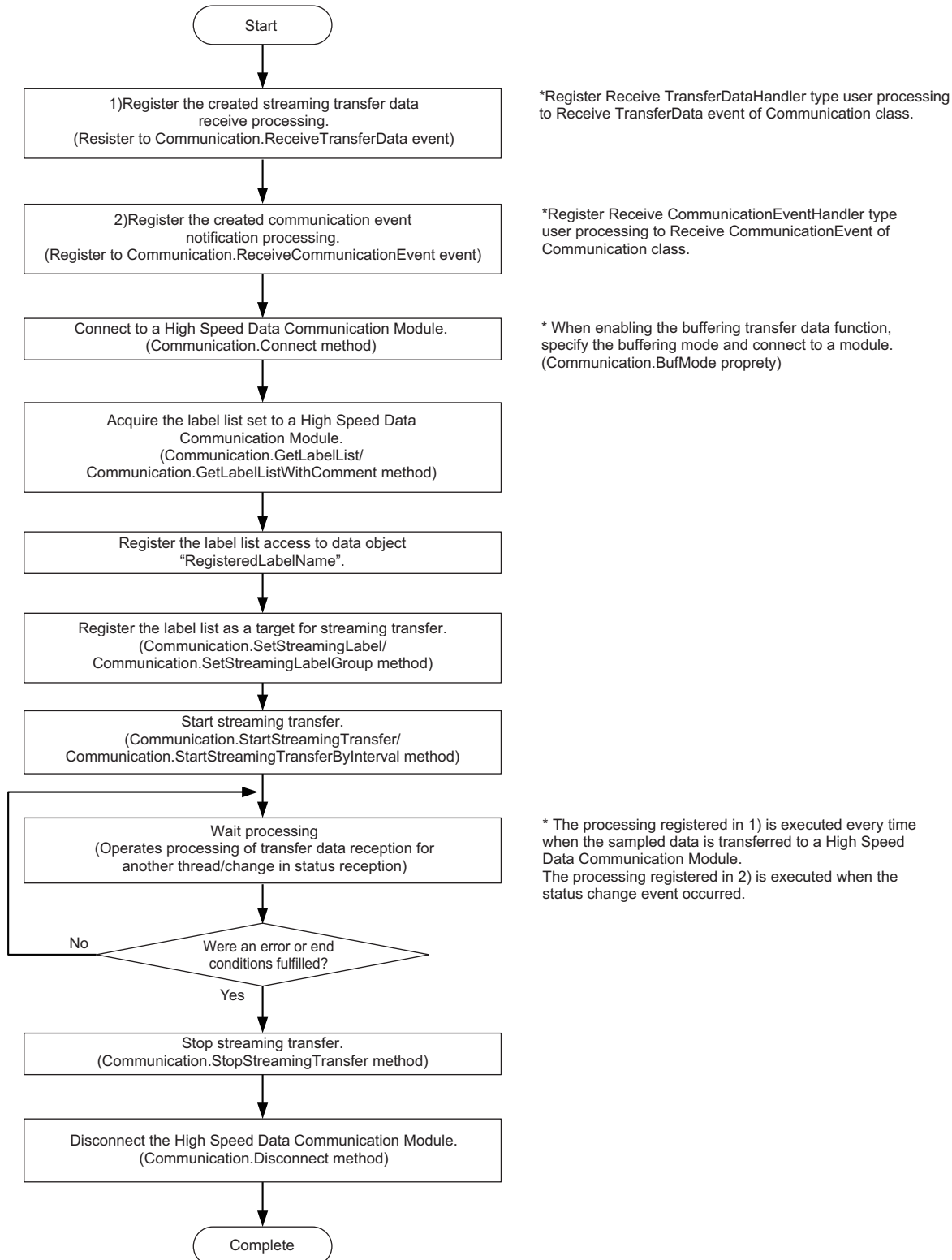
Refer the namespace "MitsubishiElectric.Melsoft.HSDDataCommunication" at the top of the user program.

```
//Refer to the High Speed Data Communication Library  
using MitsubishiElectric.Melsoft.HSDDataCommunication;
```

## 5.2 Designing Programs

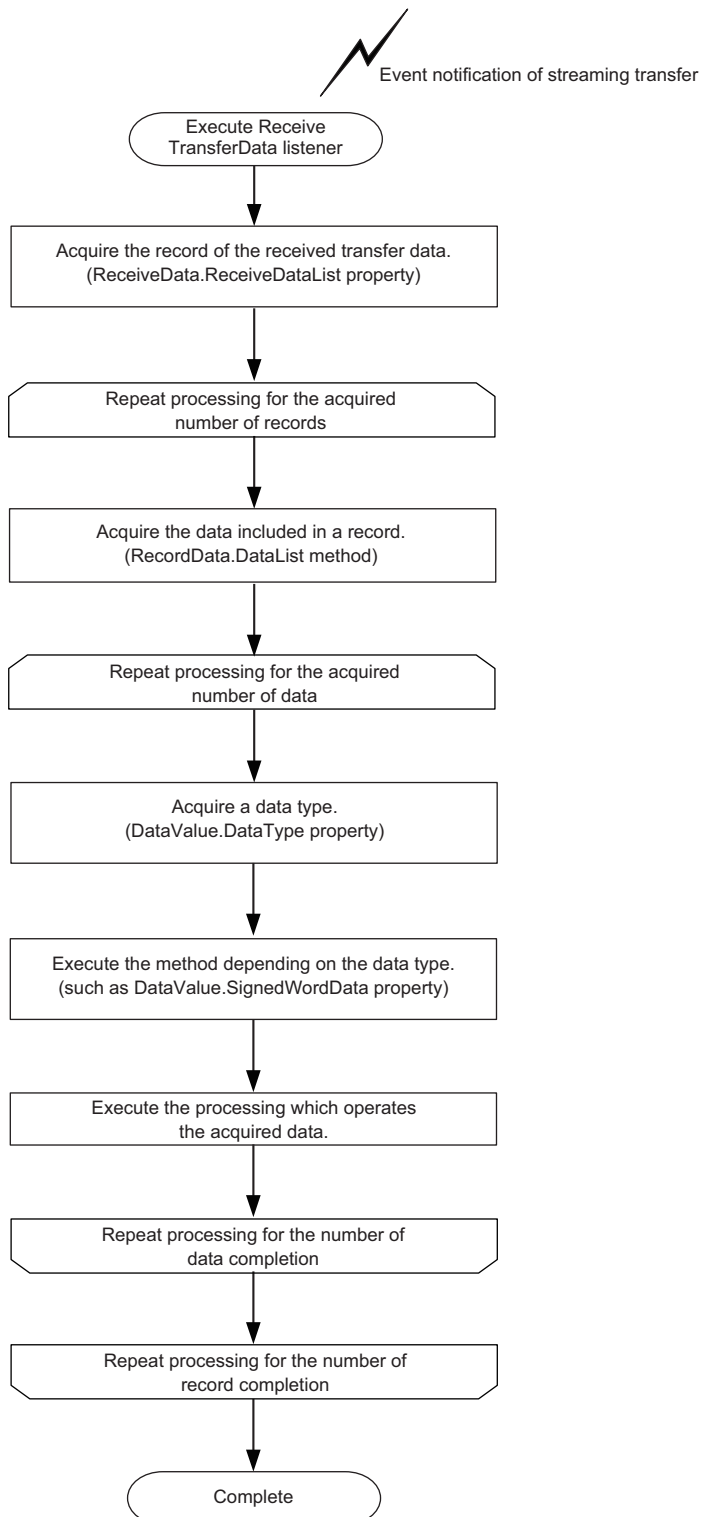
### Streaming transfer

The following flow chart shows the processes to execute the streaming transfer after connecting to a High Speed Data Communication Module and registering the target label list of the streaming transfer. In order to receive the data to be transferred, register the transfer data receive processing for Communication class event.



## Receiving streaming transfer data

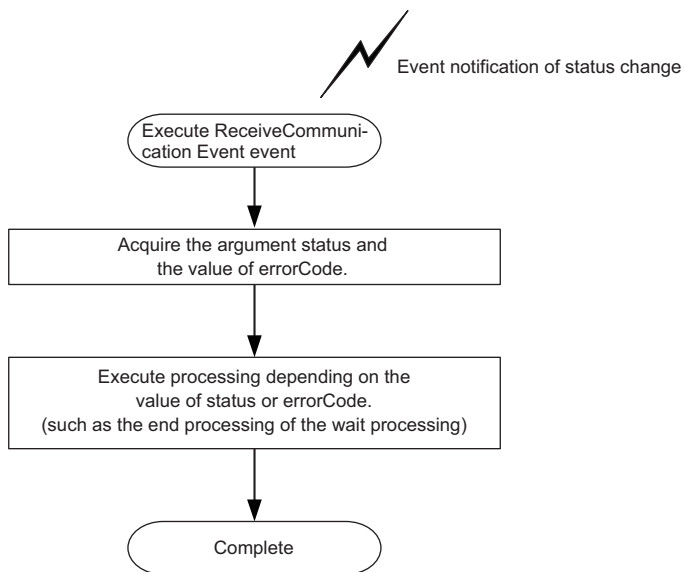
The following flow chart shows the processes of the user program when received data transfer result from a High Speed Data Communication Module after starting streaming transfer.



## Receiving notifications of module operation status changes

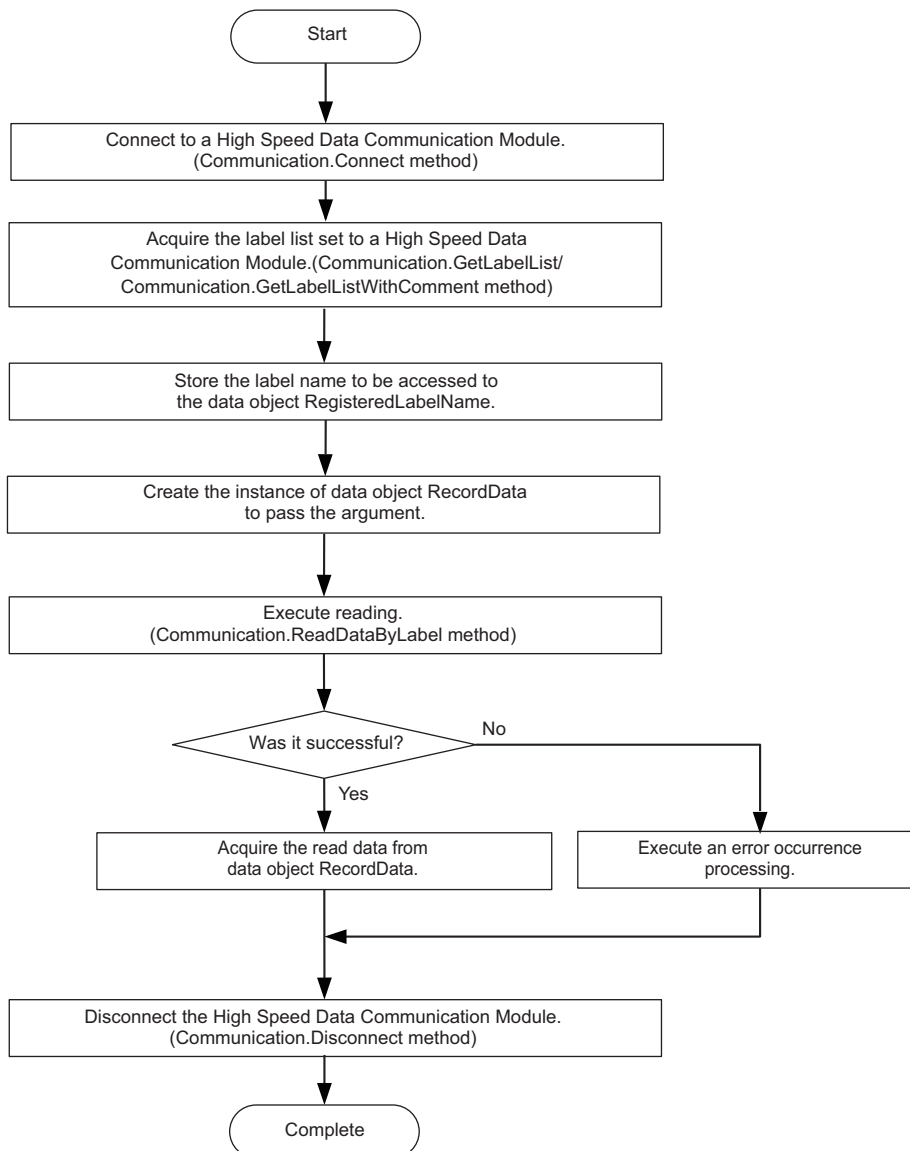
The following flow chart shows the processes when received the notification of module status changes from a High Speed Data Communication Module.

End the wait process of the streaming transfer as necessary.



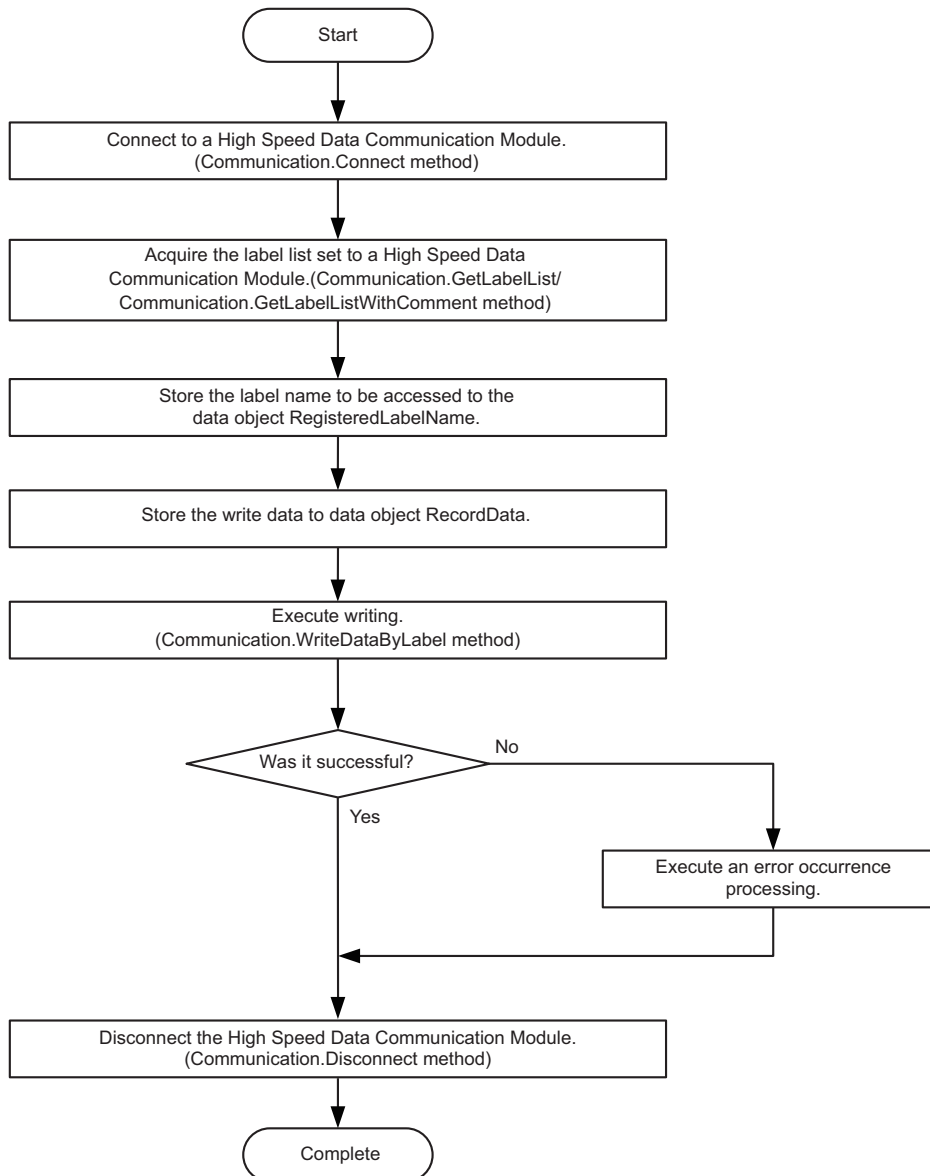
## Reading data using label specification

The following flow chart shows the operations to read data by specifying the target label list for the data read function after connecting to a High Speed Data Communication Module.



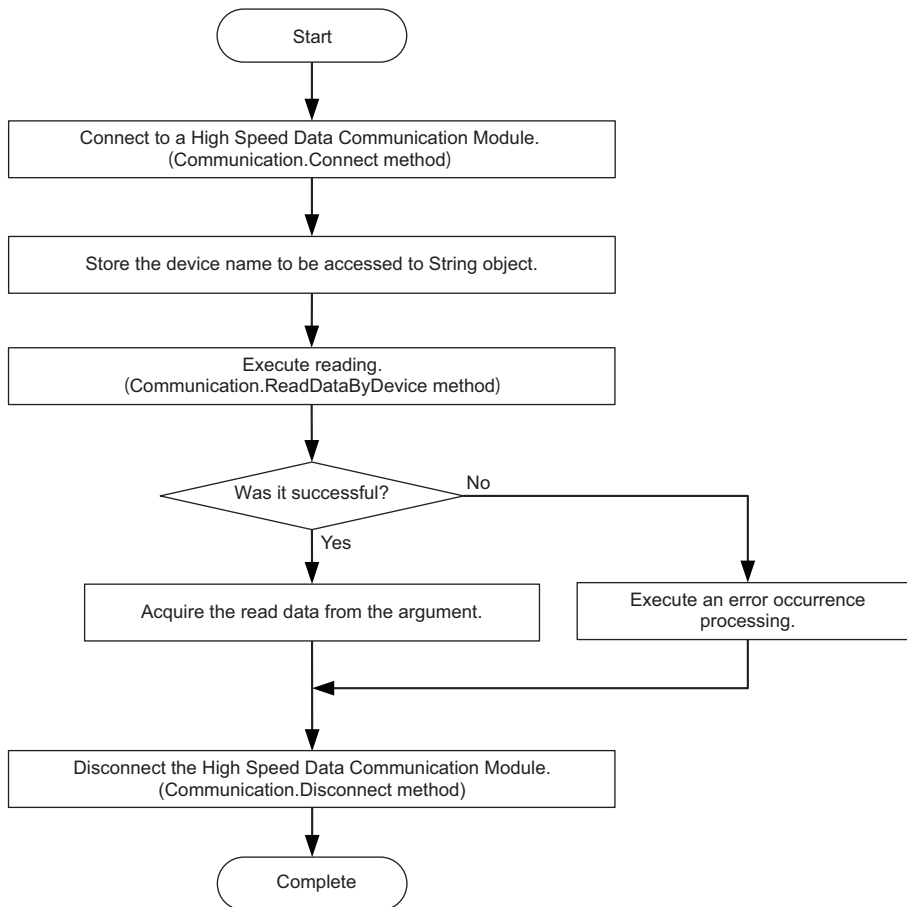
## Writing data using label specification

The following flow chart shows the operations to write data by specifying the target label list for the data write function after connecting to a High Speed Data Communication Module.



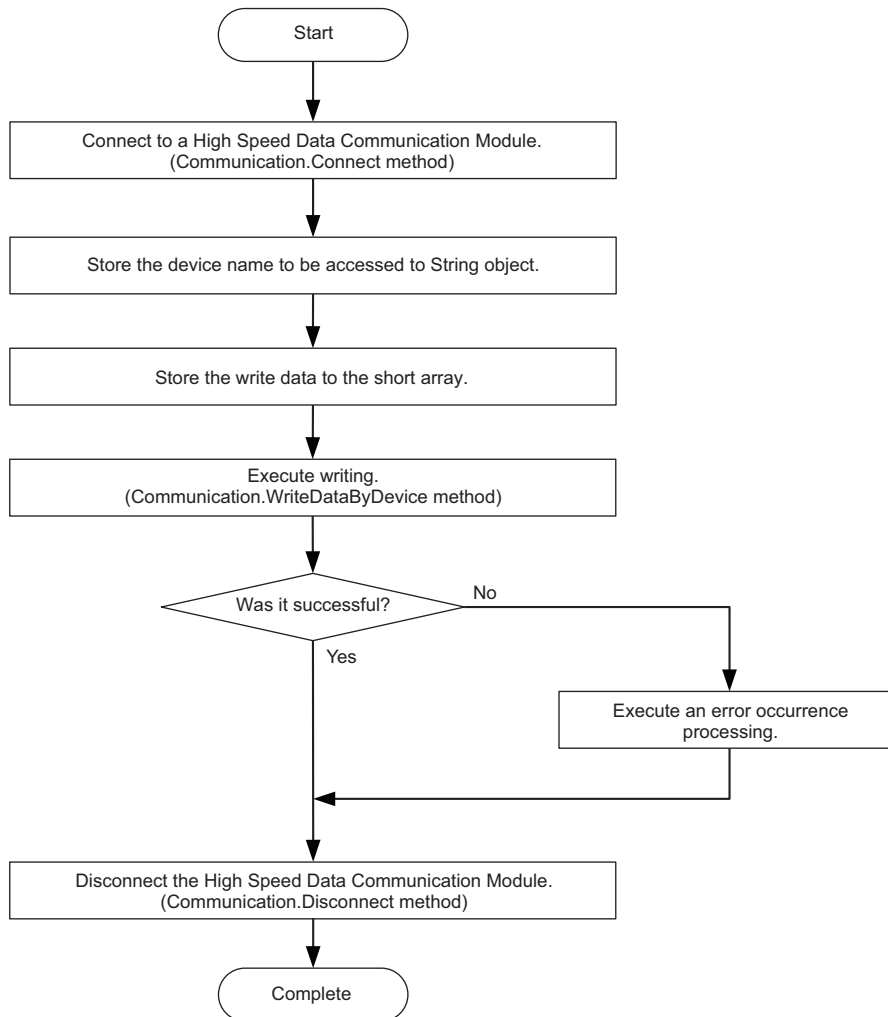
## Reading data using device specification

The following flow chart shows the operations to read data by specifying the target device list for the data read function after connecting to a High Speed Data Communication Module.



## Writing data using device specification

The following flow chart shows the operations to write data by specifying the target device list for the data write function after connecting to a High Speed Data Communication Module.





# Programming precautions

## Connect/Disconnect processing for Communication class

The efficient communication can be established by executing this method only once at the beginning and the end of the program.

## Streaming transfer data receive processing

In data receive processing of the streaming transfer, if a large amount of data are received and thus the longer processing time is required, the transfer data received from the module during the processing are discarded.

Confirm the continuous data reception by checking the index information of each record exists in the reception data are in sequence.

Note that the sequential order of the index information is corrupted even when a data miss occurs on the module side.

For details of conditions for data miss occurrences, refer to the following section.

 High Speed Data Communication Module User's Manual

## ReceiveTransferData registration processing for Communication class

Execute this method before executing the Connect processing for the Communication class.

If the ReceiveTransferData registration processing is executed after executing the Connect processing, the streaming transfer from the High Speed Data Communication Module and module status changes cannot be received.

## get accessor of DataValue class property

The get accessors other than DataType acquires data based on the data type of data stored in the DataValue class.

Therefore, if data is acquired with a data type that differs from the data type of the stored data, a cast conversion is automatically performed.

The data type of the acquired data may differ from the data type of the stored data depending on the data type after the cast conversion.

Since the source data information may be lost or a longer processing time may be required due to the cast conversion, execute the get accessor matches with the data type to be acquired.

## Communication during streaming transfer

A 06B0H error occurs when any of the following Communication class methods is executed for a single instance after executing the StartStreamingTransfer/StartStreamingTransferByInterval method.

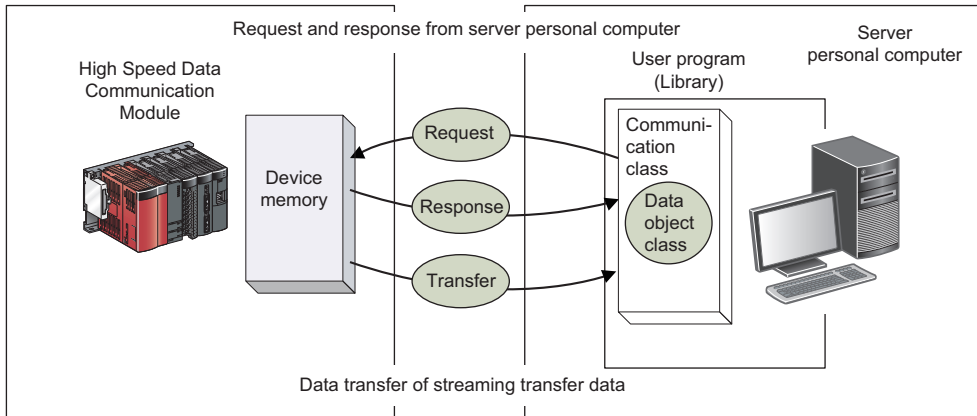
Execute the following methods after executing the StopStreamingTransfer method.

- GetConnectionDeviceList
- GetLabelList
- GetLabelListWithComment
- SetStreamingLabel
- SetStreamingLabelGroup
- StartStreamingTransferByInterval
- StartStreamingTransfer
- StopStreamingDataBuffering
- ReadDataByLabel
- ReadDataByDevice
- WriteDataByLabel
- WriteDataByDevice

## 5.3 Class and Method

The classes in High Speed Data Communication Library is the communication class that is used for receiving streaming transfer data and module status changes by realizing the streaming transfer function and the data read/write function, and the data object class that stores data for communication.

The following figure shows the functions of each class.



# Class list

## Communication class

Class name	Overview	Reference
Communication	This class processes the communication between the server personal computer and High Speed Data Communication Module. Performs the connection to High Speed Data Communication Module, the streaming transfer, and the data read/write function. The receive processing for streaming-transferred data and module status changes are implemented to notify data transfer and status changes from High Speed Data Communication Module to user programs.	Page 34 Communication class

## Data object class

Class name	Overview	Reference
LabelGroupList	This class stores information of the label group list. Stores information of the label group acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 56 LabelGroupList class
LabelGroup	This class stores information of the label group. Stores information of the label acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 57 LabelGroup class
DataLabel	This class stores label names, output data types, and data sizes. Stores information of the label acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 59 DataLabel class
RegisteredLabelName	This class stores registration target label names and data read/write label names. Stores label group names and label names used for the streaming transfer function and the data read/write function.	Page 61 RegisteredLabelName class
ReceiveData	This class stores the data list sent by the streaming transfer.	Page 63 ReceiveData class
RecordData	This class stores data used for the streaming transfer function and the data read/write function. Data received from the High Speed Data Communication Module is stored in the streaming transfer function. Values read from the specified labels are stored in the data read function. Values written to the specified labels are stored in the data write function.	Page 64 RecordData class
DataValue	This class store values for the specified data type. Stores data match with the output data type of label registered as the target of streaming transfer when performing the streaming transfer.	Page 68 DataValue class
ConnectionDeviceList	This class stores the list of the connected device information.	Page 74 ConnectionDeviceList class
ConnectionDevice	This class stores the connected device information.	Page 75 ConnectionDevice class
ConnectionDeviceDetail	This class stores the details of the connected device information.	Page 78 ConnectionDeviceDetail class

## Exception class

Class name	Overview	Reference
DataCommunicationException	This class stores enumerators of exceptions in the methods for the communication class and the data object class. The user program implements the exception handler.	Page 81 DataCommunicationException class
DataCommunicationError	This class defines the detailed information at the time of error occurrence in the methods for the communication class and the data object class.	Page 82 DataCommunicationError enumerator

# Communication class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class Communication: System.IDisposable
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDDataCommunication.Communication

## Constructor

Format	Content
Communication()	Default constructor

## Details of the constructor

### ■Communication

Item		Content
Function		Constructor
Call format		public Communication()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to communicate between the server personal computer and the High Speed Data Communication Module.

## Property list

Type	Property name	Accessor	Overview	Reference
byte	AdditionTimeType	set/get	Specify the time type received from the module at the streaming transfer to this Communication class.	Page 34 AdditionTimeType
uint	BufferingTimeOut	set/get	Specify the timeout period of the buffering data transfer function to this Communication class.	Page 35 BufferingTimeOut
byte	BufMode	set/get	Specify the buffering data transfer mode of the streaming transfer to this Communication class.	Page 35 BufMode
ushort	TimeOut	set/get	Specify the timeout period of processing request for the High Speed Data Communication Module to this Communication class.	Page 36 TimeOut


## Property details

### ■AdditionTimeType

Item	Content
Function	Acquires or stores the type of time appended to each transfer record.
Syntax	public byte AdditionTimeType { get; set; }
Property value	A byte value indicates the type of time to be appended Default: 1
Comment	Stores the type of time appended to the transfer record to this Communication class.*1 The following are the time types that can be specified. 0: Local time 1: UTC An exception occurs when a negative number is stored. If a value other than 0 or 1 is stored, the Connect method fails.

\*1 The types of time appended to each transfer record are determined when the Connect method is executed. The type of time received from the module is not changed even if this method is executed after executing the Connect method. To change the setting, execute the Disconnect method, execute this method, and then execute the Connect method again.

## ■BufferingTimeOut

Item	Content
Function	Acquires or stores the type of time appended to each transfer record.
Syntax	public uint BufferingTimeOut{ get; set; }
Property value	A uint value indicates the timeout period of the buffering data transfer function Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_TIMEOUT For details, refer to the following section.  Page 81 ErrorCode
Comment	Stores the timeout period when the buffering data transfer function is executed to this Communication class.*1 Specify the timeout period of the buffering data transfer function. (0, 60 to 86400 seconds) When using the buffering data transfer function, specify a timeout period to stop the buffering data transfer processing of the High Speed Data Communication Module in a specified period. An exception occurs if a value out of the range of 60 to 86400 is stored. When 0 is specified, the buffering data transfer function continues its operation until the network recovery without the timeout.


- \*1 The timeout for the buffering data transfer function is determined when the StartStreamingTransfer/StartStreamingTransferByInterval method is executed.  
The timeout period of the buffering data transfer function is not changed even if this method is executed after executing the StartStreamingTransfer/StartStreamingTransferByInterval method.  
To change the setting, execute the StopStreamingTransfer method, execute this method, and then execute the StartStreamingTransfer/StartStreamingTransferByInterval method again.

## ■BufMode

Item	Content
Function	Acquires or stores the buffering data transfer mode of the streaming transfer.
Syntax	public byte BufMode { get; set; }
Property value	A byte value indicates the buffering data transfer mode Default: 0
Comment	Stores the buffering data transfer mode of the streaming transfer to this Communication class. The following are the values that can be specified. Invalidity (without buffering data transfer): Other than 1 Validity (with buffering data transfer): 1 Specify "1" to validate the buffering transfer data function of High Speed Data Communication Module.*1

- \*1 The buffering transfer data mode is determined when the Connect method is executed.  
The buffering data transfer function is not validated even if this method is executed after executing the Connect method.  
To validate the buffering transfer data function, execute the Disconnect method, execute this method, and then execute the Connect method again.

## ■TimeOut

Item	Content
Function	Acquires or stores the response timeout period for communication. *1,*2
Syntax	public ushort TimeOut { get; set; }
Property value	A ushort value indicates the response timeout period. Default: 20
Exception	DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_TIMEOUT For details, refer to the following section.  Page 81 ErrorCode
Comment	Stores the response timeout period for communication to this Communication class. Specify the response timeout period of the communication request for the High Speed Data Communication Module. *3,*4,*5 1 to 255 seconds can be specified. (In seconds)

- \*1 In some Communication class methods, data are divided to process the communication.  
Therefore, a timeout may not occur with these methods even when the response timeout period specified in the TimeOut property elapses.
- \*2 A longer communication time may be required for the on-demand function when communicating to CPUs on other stations other than the control CPU of the High Speed Data Communication Module.  
Execute the on-demand function after performing the following measures when a timeout occurs in the on-demand function.
- Check the module status and the network connection status of the High Speed Data Communication Module.
  - Check the network connection status of CPUs on other stations.
  - Set a longer timeout period for the TimeOut property.
  - Reduce the number of CPUs on other stations that are accessed simultaneously.
  - Reduce the number of device points that are accessed simultaneously.
- \*3 If the network is disconnected after the connection, a notification is sent by the ReceiveCommunicationEvent event about 10 seconds after the disconnection.
- \*4 The response timeout period for communication is determined when the Connect method is executed.  
The response timeout period for communication is not changed even if this method is executed after executing the Connect method.  
To change the setting, execute the Disconnect method, execute this method, and then execute the Connect method again.
- \*5 An exception may occur regardless of the timeout period that is set during the connection when the network cable is not connected to the server personal computer.

## Method list

Type	Method name	Overview	Reference
void	Connect( string ipAddress)	Opens a communication line to the High Speed Data Communication Module, and establishes a connection.	Page 38 Connect
void	Connect( string ipAddress, string userName, string password)	Opens a communication line to the High Speed Data Communication Module in which the access authentication is enabled, and establishes a connection.	Page 39 Connect
void	Disconnect()	Closes the communication line to the High Speed Data Communication Module, and disconnects the connection.	Page 40 Disconnect
short	GetConnectionDeviceList( ConnectionDeviceList connectionDeviceList)	Acquires the list of the connected device information.	Page 40 GetConnectionDeviceList
short	GetLabelList( LabelGroupList labelGroupList)	Acquires the label list set for the High Speed Data Communication Module.	Page 41 GetLabelList
short	GetLabelListWithComment( LabelGroupList labelGroupList)	Acquires the label list, including comments, set on the High Speed Data Communication Module.	Page 42 GetLabelListWithComment
short	ReadDataByDevice( string[] deviceList, short[] deviceValue)	Reads data by specifying the device name from the programmable controller CPU to the server personal computer.	Page 43 ReadDataByDevice
short	ReadDataByLabel( RegisteredLabelName[] readLabelList, RecordData readDeviceList)	Reads data by specifying the label name from the programmable controller CPU to the server personal computer.	Page 44 ReadDataByLabel
short	SetStreamingLabel( RegisteredLabelName[] labelList)	Registers the data list streamed from the programmable controller CPU to the server personal computer by specifying the label name to the High Speed Data Communication Module.	Page 45 SetStreamingLabel

Type	Method name	Overview	Reference
short	SetStreamingLabelGroup( string[] labelGroupList)	Registers the data list streamed from the programmable controller CPU to the server personal computer by specifying the label name to the High Speed Data Communication Module.	Page 46 SetStreamingLabelGroup
short	StartStreamingTransfer( uint samplingTime) *1	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle.	Page 47 StartStreamingTransfer
short	StartStreamingTransfer( uint samplingTime, ushort transferCount) *1	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle and the number of transfer records	Page 48 StartStreamingTransfer
short	StartStreamingTransferByInterval( uint samplingTime, uint transferTime) *1	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle and the transfer time.	Page 49 StartStreamingTransferByInterval
short	StopStreamingDataBuffering( ushort connectNum)	Discards the transfer data buffered with the buffering data transfer function, and disconnects the line.	Page 50 StopStreamingDataBuffering
short	StopStreamingTransfer()	Stops the streaming transfer.	Page 51 StopStreamingTransfer
short	WriteDataByDevice( string[] deviceList, short[] deviceValue)	Writes data by specifying the device name from the server personal computer to the programmable controller CPU.	Page 51 WriteDataByDevice
short	WriteDataByLabel( RegisteredLabelName[] writeLabelList, RecordData writeDeviceList)	Writes data by specifying the label name from the server personal computer to the programmable controller CPU.	Page 52 WriteDataByLabel


\*1 A 06B0H error occurs when any of the following Communication class methods is executed for a single instance after executing the StartStreamingTransfer/StartStreamingTransferByInterval method.

Execute the following methods after executing the StopStreamingTransfer method.

- GetConnectionDeviceList
- GetLabelList
- GetLabelListWithComment
- SetStreamingLabel
- SetStreamingLabelGroup
- StartStreamingTransferByInterval
- StartStreamingTransfer
- StopStreamingDataBuffering
- ReadDataByLabel
- ReadDataByDevice
- WriteDataByLabel
- WriteDataByDevice

## Method details


### ■Connect

Item		Content
Function		Opens a communication line to a High Speed Data Communication Module whose access authentication is enabled, and establishes a connection.
Call format		public void Connect(string ipAddress) throws DataCommunicationException
Argument	Type name	string
	Variable name	ipAddress
	Content	[IN] IP address for the High Speed Data Communication Module
Return value		None
Exception		<p>DataCommunicationException — High Speed Data Communication Library error</p> <p>The following are the exceptions for High Speed Data Communication Library.</p> <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• AUTHENTICATION_FAILED</li> <li>• BUFFERING_CONNECT_OVER</li> <li>• CONNECT_OVER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• INTERRUPTED_ERROR</li> <li>• INVALID_CONNECT</li> <li>• INVALID_IPADDRESS</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• INVALID_STREAMING</li> <li>• IO_ERROR</li> <li>• IPADDRESS_NULL</li> <li>• NOT_PERMITTED_IPADDRESS</li> <li>• OUT_RANGE_ADDITIONAL_TIMETYPE</li> <li>• OUT_RANGE_TIMEOUT</li> <li>• RECONNECT</li> <li>• STREAMING_CONNECT_OVER</li> <li>• TIMEOUT</li> </ul> <p>For details, refer to the following section.</p> <p> Page 81 ErrorCode</p>
Comment		<p>Opens a communication line to the High Speed Data Communication Module.*1</p> <p>When using multiple lines to communicate with a single High Speed Data Communication Module, create communication class instances for each communication and execute the Connect method.</p>

\*1 To receive the streaming transfer data and module status changes, register the delegate before executing the Connect method.




## ■Connect

Item		Content		
Function		Opens a communication line to the High Speed Data Communication Module, and establishes a connection.		
Call format		public void Connect(string ipAddress, string userName, string password) throws DataCommunicationException		
Argument	Type name	string	string	string
	Variable name	ipAddress	userName	password
	Content	[IN] IP address for the target device	[IN] Connection user name (up to 20 characters) <sup>*1</sup>	[IN] Password for the connection user name (up to 16 characters) <sup>*2</sup>
Return value		None		
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• AUTHENTICATION_FAILED</li> <li>• BUFFERING_CONNECT_OVER</li> <li>• CONNECT_OVER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• INTERRUPTED_ERROR</li> <li>• INVALID_CONNECT</li> <li>• INVALID_IPADDRESS</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• INVALID_STREAMING</li> <li>• IO_ERROR</li> <li>• IPADDRESS_NULL</li> <li>• NOT_PERMITTED_IPADDRESS</li> <li>• OUT_RANGE_ADDITIONAL_TIMETYPE</li> <li>• OUT_RANGE_PASSWORD</li> <li>• OUT_RANGE_TIMEOUT</li> <li>• OUT_RANGE_USER_NAME</li> <li>• PASSWORD_NULL</li> <li>• RECONNECT</li> <li>• STREAMING_CONNECT_OVER</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode		
Comment		Opens a communication line to the High Speed Data Communication Module in which the account authentication is enabled. <sup>*2</sup> When using multiple lines to communicate with a single High Speed Data Communication Module, create communication class instances for each communication and execute the Connect method.		



\*1 To receive the streaming transfer data and module status changes, register the delegate before executing the Connect method.

\*2 When a High Speed Data Communication Module on which the account authentication is disabled is connected, the specified user name and the password are ignored.



## ■Disconnect

Item		Content
Function		Disconnects the connection to the High Speed Data Communication Module.
Call format		public void Disconnect() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • NOT_CONNECTED_YET For details, refer to the following section.  Page 81 ErrorCode
Comment		Closes the established communication line, and disconnects the connection.

## ■GetConnectionDeviceList

Item		Content
Function		Acquires the list of status for devices connected with High Speed Data Communication Module.
Call format		public short GetConnectionDeviceList(ConnectionDeviceList connectionDeviceList) throws DataCommunicationException
Argument	Type name	ConnectionDeviceList
	Variable name	connectionDeviceList
	Content	[IN/OUT] List of status for devices connected with High Speed Data Communication Module
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • ABORTED_BY_PEER • ARGUMENT_NULL • DISCONNECTED • DISPOSED_ERROR • EXECUTING • GENERAL_ERROR • INVALID_REQUEST • INVALID_RESPONSE • IO_ERROR • NOT_CONNECTED_YET • TIMEOUT For details, refer to the following section.  Page 81 ErrorCode
Comment		Acquires the list of status for devices connected with High Speed Data Communication Module. Use this method to check the connections when executing the StopStreamingDataBuffering method.

## ■GetLabelList

Item		Content
Function		Acquires the label list set on the High Speed Data Communication Module.
Call format		public short GetLabelList(LabelGroupList labelGroupList) throws DataCommunicationException
Argument	Type name	LabelGroupList
	Variable name	labelGroupList
	Content	[IN/OUT] Stores the label list set on the High Speed Data Communication Module.*1
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXCHANGE_TO_STRING_FAILED</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Acquires the label list set on the High Speed Data Communication Module.*2,*3 Labels of label groups which are accessible with the account used for the connection can be acquired.



\*1 "NULL" is stored to the label group comments and label comments in the acquired label list.

\*2 When settings are not written to the High Speed Data Communication Module, or when label groups which are accessible with the account used for the connection do not exist, the number of label groups stored in labelGroupList is set to 0.  
When the number of label groups is 0 and the array element is accessed, the exception IndexOutOfRangeException occurs.  
Therefore, access the array element after checking the acquired number of label groups is not 0.

\*3 When a label group without labels is set, the number of labels in the label group without labels is set to 0 among the label groups stored in labelGroupList.



When the number of labels is 0 and the array element is accessed, the exception IndexOutOfRangeException occurs.  
Therefore, access the array element after checking the acquired number of labels is not 0.

## ■GetLabelListWithComment

Item		Content
Function		Acquires the label list, including comments, set on the High Speed Data Communication Module.
Call format		public short GetLabelListWithComment(LabelGroupList labelGroupList) throws DataCommunicationException
Argument	Type name	LabelGroupList
	Variable name	labelGroupList
	Content	[IN/OUT] Stores the label list set on the High Speed Data Communication Module.
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXCHANGE_TO_STRING_FAILED</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Acquires the label list, including comments, set on the High Speed Data Communication Module.*1,*2 Labels of label groups which are accessible with the account used for the connection can be acquired.


- \*1 When settings are not written to the High Speed Data Communication Module, or when label groups which are accessible with the account used for the connection do not exist, the number of label groups stored in labelGroupList is set to 0.  
When the number of label groups is 0 and the array element is accessed, the exception IndexOutOfRangeException occurs.  
Therefore, access the array element after checking the acquired number of label groups is not 0.
- \*2 When a label group without labels is set, the number of labels in the label group without labels is set to 0 among the label groups stored in labelGroupList.  
When the number of labels is 0 and the array element is accessed, the exception IndexOutOfRangeException occurs.  
Therefore, access the array element after checking the acquired number of labels is not 0.

## ■ReadDataByDevice

Item		Content	
Function		Reads device values of the specified devices from the programmable controller CPU to the server personal computer.*1	
Call format		public short ReadDataByDevice(string[] deviceList, short[] deviceValue) throws DataCommunicationException	
Argument	Type name	string[]	short[]
	Variable name	deviceList	deviceValue
	Content	[IN] List of device names to be read*2 (up to 32 characters for each device name)	[IN/OUT] Storage destination for read device value
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_DEVICE_COUNT</li> <li>• OUT_RANGE_DEVICE_NAME_LENGTH</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Reads device values stored in the specified devices from the programmable controller CPU.*3,*4,*5	

\*1 An exception occurs if the number of words for the labels to be read exceeds 65536 words.

\*2 K5 to K8 cannot be specified for a device name. When specifying K5 to K8, combine and specify the digit specifications for K1 to K4.  
(Example) For K5M0  
Specify as K4M0 and K1M16.

\*3 If a word device or the digit specification of device is specified, one-word device value is acquired. If a bit device or the bit specification of device is specified, 0 or 1 is acquired for the corresponding bit.  
For accessible devices, refer to the following section.  
 High Speed Data Communication Module User's Manual

When the consecutive data of 2 words or more are acquired, read the device values by specifying multiple devices.



(Example) When a two-word device value is acquired from D0

Acquire the device value specifying D0 and D1, and process the acquired device value to be two-word data.

\*4 An error occurs when the account authentication function is enabled and if the read authority for device specification is not applicable with the connected account.

\*5 Device values other than those of the control CPU of the High Speed Data Communication Module cannot be read. When reading device values from programmable controller CPUs other than the control CPU of the High Speed Data Communication Module, specify the labels.

## ■ReadDataByLabel

Item		Content	
Function		Reads device values of the specified labels from the programmable controller CPU to the server personal computer.*1	
Call format		public short ReadDataByLabel(RegisteredLabelName[] readLabelList, RecordData readDeviceList) throws DataCommunicationException	
Argument	Type name	RegisteredLabelName[]	RecordData
	Variable name	readLabelList	readDeviceList
	Content	[IN] Label list to be read	[IN/OUT] Storage destination for read data
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Reads the device value from the programmable controller CPU to the server personal computer by specifying the labels set on the High Speed Data Communication Module.*2,*3,*4	



\*1 An exception occurs if the number of words for the labels to be read exceeds 65536 words.

\*2 The combination of general sampling labels and high speed sampling labels can be specified.

\*3 An exception occurs if the number of labels to be read exceeds 65536.

\*4 An error occurs when the account authentication function is enabled and if a label group without the read authority is specified with the connected account.

## ■SetStreamingLabel

Item		Content
Function		Registers the target labels for the streaming transfer on the High Speed Data Communication Module.
Call format		public short SetStreamingLabel(RegisteredLabelName[] labelList) throws DataCommunicationException
Argument	Type name	RegisteredLabelName[]
	Variable name	labelList
	Content	[IN] Target label list for streaming transfer* <sup>1</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Specify the labels to be streamed to the server personal computer.* <sup>2</sup> * <sup>3</sup> * <sup>4</sup> * <sup>5</sup>

\*<sup>1</sup> When the account authentication function is enabled, an error occurs if a registration of a label group without the read authority is attempted with the connected account.



\*<sup>2</sup> The combination of general sampling labels and high speed sampling labels cannot be specified in a single registration.

\*<sup>3</sup> Overwrites data if the streaming transfer data are already registered.

\*<sup>4</sup> The total data size for labels that can be registered as the target of streaming transfer is 65536 words.

\*<sup>5</sup> The number of labels that can be registered as the target of streaming transfer is 65536.



## ■SetStreamingLabelGroup

Item		Content
Function		Registers the target label groups for the streaming transfer on the High Speed Data Communication Module.
Call format		public short SetStreamingLabelGroup(string[] labelGroupList) throws DataCommunicationException
Argument	Type name	string[]
	Variable name	labelGroupList
	Content	[IN] Target label group list for streaming transfer* <sup>1</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Specify the label groups to be streamed to the server personal computer. By specifying the label groups, all labels of the specified label groups are registered.* <sup>2</sup> ,* <sup>3</sup> ,* <sup>4</sup> ,* <sup>5</sup>

- \*<sup>1</sup> When the account authentication function is enabled, an error occurs if a registration of a label group without the read authority is attempted with the connected account.
- \*<sup>2</sup> The combination of general sampling labels and high speed sampling labels cannot be specified in a single registration.
- \*<sup>3</sup> Overwrites data if the streaming transfer data are already registered.
- \*<sup>4</sup> The total data size for labels of label groups that can be registered as the target of streaming transfer is 65536 words.
- \*<sup>5</sup> The number of labels of the label groups that can be registered as the target of streaming transfer is 65536.



## ■StartStreamingTransfer

Item		Content
Function		Specify the sampling cycle, and start the streaming transfer of the data registered to a High Speed Data Communication Module.
Call format		public short StartStreamingTransfer(uint samplingTime) throws DataCommunicationException
Argument	Type name	uint
	Variable name	samplingTime
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3,*4</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by synchronizing the transfer cycle with the sampling cycle. <sup>*5</sup> Use this method when executing the streaming transfer by specifying general sampling labels. <sup>*6,*7,*8</sup>

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down. When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 An error occurs if a value larger than 32,767,000 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to general sampling.

\*5 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.


\*6 The value as the same as the sampling cycle is processed for the transfer cycle, and 1 is processed for the number of transfer records.

\*7 The streaming transfer function is a best effort delivery.

This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices.

Run the system by fully verifying the processing time of each function when constructing it.



For the processing time, refer to the following section.

 High Speed Data Communication Module User's Manual

\*8 The transfer data may be received by ReceiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (Disconnect), expand the timeout cycle with TimeOut property, reconnect the line (Connect), and start streaming transfer (StartStreamingTransferByInterval).

## ■StartStreamingTransfer

Item		Content	
Function		Specify the sampling cycle and a number of transfer record, and start the streaming transfer of the data registered to a High Speed Data Communication Module.	
Call format		public short StartStreamingTransfer (uint samplingTime, ushort transferCount) throws DataCommunicationException	
Argument	Type name	uint	ushort
	Variable name	samplingTime	transferCount
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3,*4</sup>	[IN] Transfer number of records Specify the number of records to execute the data transfer with the specified number of sampling execution. High speed sampling: 1 to 100 <sup>*5</sup> General sampling: fixed to 1 <sup>*6</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by synchronizing the transfer cycle with the sampling cycle. <sup>*7</sup> Data are transferred when the specified number of transfer records is sampled. <sup>*8,*9</sup>	

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down. When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 An error occurs if a value larger than 32,767,000 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to general sampling.

\*5 An error occurs if the specified number of transfer records exceeds the range when the label group registered as the target for streaming transfer is set to high speed sampling.

\*6 Any value other than 1 cannot be specified for the number of transfer records when the label group registered as the target for streaming transfer is set to general sampling. An error occurs if any value other than 1 is specified.


\*7 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.

\*8 The streaming transfer function is a best effort delivery.

This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices.

Run the system by fully verifying the processing time of each function when constructing it.

For the processing time, refer to the following section.

 High Speed Data Communication Module User's Manual

\*9 The transfer data may be received by ReceiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (Disconnect), expand the timeout cycle with Timeout property, reconnect the line (Connect), and start streaming transfer (StartStreamingTransferByInterval).

## ■StartStreamingTransferByInterval

Item		Content	
Function		Specify the sampling cycle and transfer cycle, and start the streaming transfer of data registered to a High Speed Data Communication Module.	
Call format		public short StartStreamingTransferByInterval(uint samplingTime, uint transferTime) throws DataCommunicationException	
Argument	Type name	uint	uint
	Variable name	samplingTime	transferTime
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3</sup>	[IN] Transfer time (milliseconds) High speed sampling: 0.1 to 100 ms <sup>*4,*5</sup> General sampling: fixed to 0 <sup>*6</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section. High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_TRANSFER_TIME</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section. Page 81 ErrorCode	
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by specifying the sampling cycle and the transfer cycle. <sup>*7,*8,*9</sup>	

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down. When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 By specifying 0 for the transfer cycle, data can be transferred in synchronization with the sampling cycle.

\*5 A transfer cycle shorter than the sampling cycle cannot be specified.

\*6 The transfer cycle cannot be specified when the label group registered as the target for streaming transfer is set to general sampling. An error occurs if any value other than 0 is specified.

\*7 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.

\*8 The streaming transfer function is a best effort delivery.

This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices.

Run the system by fully verifying the processing time of each function when constructing it.

For the processing time, refer to the following section.

High Speed Data Communication Module User's Manual

\*9 The transfer data may be received by ReceiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (Disconnect), expand the timeout cycle with TimeOut property, reconnect the line (Connect), and start streaming transfer (StartStreamingTransferByInterval).

When the label group registered as the target of streaming transfer is set to high speed sampling, the following combinations of sampling cycle and the transfer cycle can be specified.

○: Applicable, ×: Error

Transfer cycle (milliseconds)	Sampling time (milliseconds)							
	0	1	2	...	100	...	32767	32768 or higher
0	○	○	○	○	○	○	○	×
1	○	○	×	×	×	×	×	×
2	○	○	○	×	×	×	×	×
...	○	○	○	○/×	×	×	×	×
100	○	○	○	○	○	×	×	×
101 or higher	×	×	×	×	×	×	×	×



\*1 An error occurs when a value smaller than the sampling cycle is specified for the transfer cycle.

(Example) Sampling time: 40, Transfer cycle: 30 → Error

Sampling time: 40, Transfer cycle: 40 → Applicable

Sampling time: 30, Transfer cycle: 40 → Applicable

## ■ StopStreamingDataBuffering

Item		Content
Function		Discards the buffering transfer data and disconnects the line.
Call format		public short StopStreamingDataBuffering(ushort connectNum) throws DataCommunicationException
Argument	Type name	ushort
	Variable name	connectNum
	Content	[IN] Connection number to be disconnected Access status 1 to 5 is specified: 1 to 5
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_CONNECTION_NO</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		When the access status of the specified connection number is buffering the transfer data, the buffered transfer data are discarded and the line is disconnected.*1 An error occurs when the access status of the specified connection number is not buffering the transfer data. The connection numbers correspond to the access status 1 to 5 in the access status area of the buffer memory. (When the GetConnectionDeviceList method is executed, the array elements 0 to 4 in the property ConnectionDeviceArray[] of argument connectionDeviceList are synonymous with the access status 1 to 5.) An exception OUT_RANGE_CONNECTION_NO is returned if a value other than 1 to 5 is specified.


\*1 An error occurs when the account authentication function is enabled and the administrative right does not apply to the connected account.

## ■StopStreamingTransfer



Item		Content
Function		Stops the streaming transfer to the server personal computer.
Call format		public short StopStreamingTransfer() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section. High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section. Page 81 ErrorCode
Comment		Stops the streaming transfer to the server personal computer. Stop the re-transfer start function which operates on a user program after detected the disconnection by executing at the buffering transfer data function.

## ■WriteDataByDevice

Item		Content	
Function		Writes device values of the specified devices from the server personal computer to the programmable controller CPU.* <sup>1</sup>	
Call format		public short WriteDataByDevice(string[] deviceList, short[] deviceValue) throws DataCommunicationException	
Argument	Type name	string[]	short[]
	Variable name	deviceList	deviceValue
	Content	[IN] List of device names to be written* <sup>2</sup> (up to 32 characters for each device name)	[IN] Storage destination for write device value
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section. High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_DEVICE_COUNT</li> <li>• OUT_RANGE_DEVICE_NAME_LENGTH</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section. Page 81 ErrorCode	
Comment		Writes device values from the server personal computer to the devices specified on the programmable controller CPU.* <sup>3,4,*5</sup>	

- \*1 An exception occurs if the number of words for the devices to be written exceeds 65536 words.
- \*2 K5 to K8 cannot be specified for a device name. When specifying K5 to K8, combine and specify the digit specifications for K1 to K4.  
(Example) For K5M0  
Specify as K4M0 and K1M16.
- \*3 If a word device or the digit specification of the device is specified, one-word device value is stored. If a bit device or the bit specification of device is specified, the value for the writing device value's least significant bit is stored.  
For accessible devices, refer to the following section.  
 High Speed Data Communication Module User's Manual  
When the consecutive data of 2 words or more are acquired, write the device values by specifying multiple devices.  
(Example) When a two-word device value is stored from D0  
Write the device values by specifying D0 and D1.
- \*4 An error occurs when the account authentication function is enabled and if the read authority for device specification is not applicable with the connected account, or if the read authority is applicable but the write authority is not applicable with the connected account.
- \*5 Device values other than those of the control CPU of the High Speed Data Communication Module cannot be written. When writing device values from programmable controller CPUs other than the control CPU of the High Speed Data Communication Module, specify the labels.

## ■WriteDataByLabel

Item		Content	
Function		Writes device values of the specified labels from the server personal computer to the programmable controller CPU.*1	
Call format		public short WriteDataByLabel(RegisteredLabelName[] writeLabelList, RecordData writeDeviceList) throws DataCommunicationException	
Argument	Type name	RegisteredLabelName[]	RecordData
	Variable name	writeLabelList	writeDeviceList
	Content	[IN] Label list to be written	[IN] Storage destination for write data
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_DATASIZE</li> <li>• INVALID_DATATYPE</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Writes device values from the server personal computer to the programmable controller CPU by specifying the labels set on the High Speed Data Communication Module.*2,*3,*4	

- \*1 An exception occurs if the number of words for the labels to be written exceeds 65536 words.
- \*2 The combination of general sampling labels and high speed sampling labels can be specified.
- \*3 An exception occurs if the number of labels to be written exceeds 65536.
- \*4 An error occurs when the account authentication function is enabled and if a label group without the read authority or a label group with the read authority but not with the write authority is specified with the connected account.

## Setting data

The following describes the operations when writing data to labels whose data type is string type/raw type.

- When the data size of the set label is an odd number,

A null code (00H) is stored to the upper byte of the last device.

**Ex.**

When writing "ABC" to the label with the start device: "D0", data type: "string type", and data size: "3" (odd number), the data are written as shown below.

The character code to handle a single character as one byte (such as "iso-8859-1") is specified for the string.

D0		D1	
41H	42H	43H	00H
A	B	C	Null code

- When the write data size or the number of elements is smaller than that of the set label.

Null codes (00H) are stored to the devices following the amount of the write data size or the number of elements.

**Ex.**

When writing "ABCD" to the label with the start device: "D0", data type: "string type", and data size: "8", the data are written as shown below.

The character code to handle a single character as one byte (such as "iso-8859-1") is specified for the string.

D0		D1		D2		D3	
41H	42H	43H	44H	00H	00H	00H	00H
A	B	C	D	Null code	Null code	Null code	Null code

- When the write data size or the number of elements is larger than that of the set label.

The exception "INVALID\_DATASIZE" occurs.

Check the data size set to the label and correct the write data size not to be larger than the data size.

Note that the data sizes of the same strings are different depending on the encoding code.

Specify the encoding code for each specific purpose.

**Ex.**

When writing "ABCD" to the label with the start device: "D0", data type: "string type", and size: "8" (odd number), the data are written depending on the encoding code as shown below.

"UTF-16LE" (two-byte single character)

D0		D1		D2		D3	
41H	00H	42H	00H	43H	00H	44H	00H
A	Null code	B	Null code	C	Null code	D	Null code

"iso-8859-1" (one-byte single character)

D0		D1		D2		D3	
41H	42H	43H	44H	00H	00H	00H	00H
A	B	C	D	null code	null code	null code	null code

## Event list

Type	Event name	Overview	Reference
ReceiveCommunicationEventHandler	ReceiveCommunicationEvent	The event to register the communication event notification processing.	Page 54 ReceiveCommunicationEvent
ReceiveTransferDataHandler	ReceiveTransferData	The event to register the streaming transfer data receive processing.	Page 54 ReceiveTransferData

## Event details

### ■ReceiveCommunicationEvent

Item	Content
Function	Registers the communication event notification processing.
Call format	public event ReceiveCommunicationEventHandler ReceiveCommunicationEvent
Comment	Registers the delegate contains the user processing in which the processing at the time of receiving status changes such as a line disconnection is written. Match the user processing with the call format of ReceiveCommunicationEventHandler.

### ■ReceiveTransferData

Item	Content
Function	Registers the streaming transfer data receive processing.
Call format	public event ReceiveTransferDataHandler ReceiveTransferData
Comment	Registers the delegate contains the user processing in which the processing at the time of receiving the streaming transfer data is written. Match the user processing with the call format of ReceiveTransferDataHandler.

## Delegate list

Type	Method name	Overview	Reference
void	ReceiveCommunicationEventHandler(short status, short errorCode)	The method which is called when a status change such as a line disconnection is received.*1	Page 54 ReceiveCommunicationEventHandler
void	ReceiveTransferDataHandler(ReceiveData recordList)	The method which is called when the streaming transfer data sent from the High Speed Data Communication Module is received.*1	Page 55 ReceiveTransferDataHandler

\*1 Since the default is no-operation, it is necessary to create actual processing and register it to the event.

## Delegate details

### ■ReceiveCommunicationEventHandler

Item		Content	
Function		Notifies status changes of High Speed Data Communication Module to the server personal computer.	
Call format		public delegate void ReceiveCommunicationEventHandler (short status, short errorCode)	
Argument	Type name	short	short
	Variable name	status	errorCode
	Content	[IN/OUT] Notified events**1 Disconnection: 0 Update settings: 3 Stop module operation: 1 Library processing overload: 4 Transfer error stop: 5 Re-transfer start failure: 6 Re-transfer start: 7	[IN] Error code of occurred error (Refer to the error code when an event such as disconnection, transfer error stop, re-transfer start failure, and re-transfer start is notified. "0" is stored to events other than noted.)
Return value		—	
Comment		Notifies status changes such as a line disconnection to the server personal computer. Implement the processing for the status changes in the user program.	



\*1 The following are the actions for the stored events.

Module status	Action
Disconnection: 0	<p>The High Speed Data Communication Module is disconnected, the programmable controller CPU is reset, the power is turned OFF, or the settings are updated.</p> <p>Check the LAN cable connections between the server personal computer and the High Speed Data Communication Module and the system operations.</p> <p>If no problems are found, execute the Connect processing again and establish the communication.</p> <p>(If the Connect processing is not executed again, the communication may not be established normally.)</p> <p>When the StartStreamingTransfer/StartStreamingTransferByInterval processing is executed by enabling the buffering mode for the buffering transfer data function, "1" is stored to errorCode at the time of the disconnection event occurrence.</p> <p>When "1" is stored to errorCode, the execution of Connect processing again is unnecessary.</p> <p>When an incorrect packet is received and the line is disconnected, "2" is stored to errorCode at the time of disconnection event occurrence.</p> <p>A "0" is stored to errorCode when disconnected the line other than above.</p>
Update settings: 3	<p>The module settings are changed.</p> <p>After executing the Disconnect processing and checking the module settings, execute the Connect processing again and establish the communication.</p> <p>The disconnection event is notified after the setting update is notified.</p>
Stop module operation: 1	<p>A module stop error occurred.</p> <p>Execute the Disconnect processing and check the status of the module.</p> <p>After starting the module in operation, execute the Connect processing again and establish the communication.</p>
Library processing overload: 4	<p>The speed to process the streaming transfer data is slow.</p> <p>Execute the Disconnect processing, and perform the following actions.</p> <ul style="list-style-type: none"> <li>• Reduce the number of labels registered to the streaming transfer.</li> <li>• Change the output data type and the data size of the labels registered to the streaming transfer, and reduce the number of points.</li> <li>• Increase the sampling cycle and the transfer cycle.</li> <li>• Reduce the number of transfer records.</li> <li>• Review the processing content for ReceiveTransferData.</li> </ul> <p>After performing the action, execute the Connect processing again and establish the communication.</p> <p>A library processing overload may occur when the server personal computer is overloaded, such as when running Java VM for the first time.</p>
Transfer error stop: 5	<p>The streaming transfer is stopped.</p> <p>Execute the Disconnect processing, and perform the following actions.</p> <ul style="list-style-type: none"> <li>• Check the value of errorCode, and perform the actions provided in the error code list.</li> <li>• Check the status of the network accessing another station/other stations programmable controller CPU.</li> <li>• Check if the parameters of the programmable controller CPU from which data are sampled are not changed.</li> </ul> <p>After performing the action, execute the Connect processing again and establish the communication.</p>
Re-transfer start failure: 6	<p>Restarting of the streaming transfer failed after the network connection between the server personal computer and the High Speed Data Communication Module is recovered.</p> <p>The cause of the restarting failure is stored to errorCode.</p> <p>The following are the values to be stored.</p> <p>0: Cancelling request</p> <p>1: Timeout occurrence</p> <p>2: Error occurrence (Occurs when data buffering is not executed in High Speed Data Communication Module after the operation to recover the network such as resetting the CPU module or stopping buffering transfer data from the High Speed Data Communication Module.)</p> <p>Check the LAN cable connections between the server personal computer and the High Speed Data Communication Module and the system operations.</p> <p>If no problems are found, execute the Connect processing again and establish the communication.</p>
Re-transfer start: 7	<p>The network connection between the server personal computer and the High Speed Data Communication Module is recovered.</p> <p>When an processing overload occurs during the buffering data transfer and the old data are overwritten, or the buffer clear is requested, "1" is stored to errorCode at the time of the re-transfer start event occurrence.</p>

## ■ReceiveTransferDataHandler

Item	Content
Function	Notifies the reception of the streaming transfer data sent from the High Speed Data Communication Module to the server personal computer.
Call format	public delegate void ReceiveTransferDataHandler (ReceiveData recordList)
Argument	Type name
	Variable name
	Content
Return value	—
Comment	<p>Notifies the reception of the streaming transfer data sent from the High Speed Data Communication Module to the server personal computer.</p> <p>Implement the processing for the receive data in the user program.</p>

# LabelGroupList class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDataCommunication
Class syntax	public class LabelGroupList
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDataCommunication.LabelGroupList

## Constructor

Format	Content
LabelGroupList()	Default constructor

## Details of the constructor

### ■LabelGroupList

Item		Content
Function		Constructor
Call format		public LabelGroupList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the label group list. The label group list maintained previously is initialized.

## Property list

Type	Property name	Accessor	Overview	Reference
LabelGroup[]	LabelGroupArray	get	Acquires the label group list from this LabelGroupList class.	Page 56 LabelGroupArray

## Property details

### ■LabelGroupArray

Item	Content
Function	Acquires the label group list.
Syntax	public LabelGroup[] LabelGroupArray { get; }
Property value	A LabelGroup object array indicates the label group list Default: null
Exception	None
Comment	Acquires the label group list from this LabelGroupList class. "null" is acquired by default.

# LabelGroup class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDataCommunication
Class syntax	public class LabelGroup
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDataCommunication.LabelGroup

## Constructor

Format	Content
LabelGroup()	Default constructor

## Details of the constructor

### ■LabelGroup

Item	Content						
Function	Constructor						
Call format	public LabelGroup()						
Argument	<table> <tr> <td>Type name</td><td>—</td></tr> <tr> <td>Variable name</td><td>—</td></tr> <tr> <td>Content</td><td>—</td></tr> </table>	Type name	—	Variable name	—	Content	—
Type name	—						
Variable name	—						
Content	—						
Return value	None						
Comment	Constructs a class to store the information of the label group. The label name, the label comment, the output data type, and the data size maintained previously are initialized.						

## Property list

Type	Property name	Accessor	Overview	Reference
string	LabelGroupComment	get	Acquires the label group comment from this LabelGroup class.	Page 57 LabelGroupComment
string	LabelGroupName	get	Acquires the label group name from this LabelGroup class.	Page 58 LabelGroupName
DataLabel[]	LabelList	get	Acquires the label list set for the label group from this LabelGroup class.	Page 58 LabelList
byte	SamplingType	get	Acquires the type of label from this LabelGroup class.	Page 58 SamplingType

## Property details

### ■LabelGroupComment

Item	Content
Function	Acquires the label group comment.
Syntax	public string LabelGroupComment { get; }
Property value	A string indicates the label group comment Default: null
Exception	None
Comment	Acquires the label group comment from this LabelGroup class.

## ■LabelGroupName

Item	Content
Function	Acquires the label group name.
Syntax	public string LabelGroupName { get; }
Property value	A string indicates the label group name Default: null
Exception	None
Comment	Acquires the label group name from this LabelGroup class. Store the acquired label group names in the RegisteredLabelName class, and use them for the arguments of the SetStreamingLabel, ReadDataByLabel, WriteDataByLabel methods in the Communication class.

## ■LabelList

Item	Content
Function	Acquires the label list.
Syntax	public DataLabel[] LabelList { get; }
Property value	A DataLabel object array indicates the label list. Default: null
Exception	None
Comment	Acquires the label list from this LabelGroup class.

## ■SamplingType

Item	Content
Function	Acquires the type of label.
Syntax	public byte SamplingType { get; }
Property value	A byte value indicates the type of label Default: 1
Exception	None
Comment	Acquires the type of label from this LabelGroup class. High speed sampling label: 0 General sampling label: 1

# DataLabel class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class DataLabel
Class inheritance	System.Object ↳ MitsubishiElectric.Melsoft.HSDDataCommunication.DataLabel

## Constructor

Format	Content
DataLabel()	Default constructor

## Details of the constructor

### ■DataLabel

Item	Content						
Function	Constructor						
Call format	public DataLabel()						
Argument	<table> <tr> <td>Type name</td><td>—</td></tr> <tr> <td>Variable name</td><td>—</td></tr> <tr> <td>Content</td><td>—</td></tr> </table>	Type name	—	Variable name	—	Content	—
Type name	—						
Variable name	—						
Content	—						
Return value	None						
Comment	Constructs a class to store the information of the label. The label name, the label comment, the output data type, and the data size maintained previously are initialized.						

## Property list

Type	Property name	Accessor	Overview	Reference
short	DataSize	get	Acquires the data size of label from this DataLabel class.	Page 59 DataSize
short	DataType	get	Acquires the output data type of label from this DataLabel class.	Page 60 DataType
string	LabelComment	get	Acquires the label comment from this DataLabel class.	Page 60 LabelComment
string	LabelName	get	Acquires the label name from this DataLabel class.	Page 60 LabelName

## Property details

### ■DataSize

Item	Content
Function	Acquires the data size.
Syntax	public short DataSize { get; }
Property value	A short value indicates the data size. Default: 0
Exception	None
Comment	Acquires the data size of label from this DataLabel class. The following are the values of data size for each output data type. (Unit: byte) Bit: 1 Word [signed]: 2 Double word [signed]: 4 Word [unsigned]: 2 Double word [unsigned]: 4 Float [single precision]: 4 Float [double precision]: 8 String: Size* <sup>1</sup> Raw: Size* <sup>1</sup>

\*1 Stores the value of size specified with Configuration Tool for "String" and "Raw".

## ■DataType

Item	Content
Function	Acquires the output data type.
Syntax	public short DataType { get; }
Property value	A short value indicates the output data type. Default: 0
Exception	None
Comment	Acquires the output data type of label from this DataLabel class. The following are the values for each output data type. Bit: 1 Word [signed]: 10 Double word [signed]: 11 Word [unsigned]: 12 Double word [unsigned]: 13 Float [single precision]: 20 Float [double precision]: 21 String: 30 Raw: 40

## ■LabelComment

Item	Content
Function	Acquires the label comment.
Syntax	public string LabelComment { get; }
Property value	A string indicates the label comment Default: null
Exception	None
Comment	Acquires the label comment from this DataLabel class.

## ■LabelName

Item	Content
Function	Acquires the label name.
Syntax	public string LabelName { get; }
Property value	A string indicates the label name Default: null
Exception	None
Comment	Acquires the label group name from this DataLabel class. Store the acquired label group names in the RegisteredLabelName class, and use them for the arguments of the SetStreamingLabel, ReadDataByLabel, WriteDataByLabel methods in the Communication class.

# RegisteredLabelName class


Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class RegisteredLabelName
Class inheritance	System.Object ↳ MitsubishiElectric.Melsoft.HSDDataCommunication.RegisteredLabelName

## Constructor

Format	Content
RegisteredLabelName(string labelGroupName, string labelName)	Stores the specified label group name and label name to this RegisteredLabelName class.

## Details of the constructor

### RegisteredLabelName


Item		Content	
Function		Constructor	
Call format		public RegisteredLabelName(string labelGroupName, string labelName) throws DataCommunicationException	
Argument	Type name	string	string
	Variable name	labelGroupName	labelName
	Content	[IN] Target label group name (up to 32 characters)	[IN] Target label name (up to 32 characters)
Return value		None	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_LABELGROUP_NAME_LENGTH</li><li>• OUT_RANGE_LABEL_NAME_LENGTH</li></ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Stores the target label group names and label names for the streaming transfer function, the data read function, and the data write function to this RegisteredLabelName class.	

## Property list


Type	Property name	Accessor	Overview	Reference
string	LabelGroupName	set/get	Stores the label group name to this RegisteredLabelName class. Acquires the label group name from this RegisteredLabelName class.	Page 62 LabelGroupName
string	LabelName	set/get	Stores the label name to this RegisteredLabelName class. Acquires the label name from this RegisteredLabelName class.	Page 62 LabelName

## Property details

### ■LabelGroupName

Item	Content
Function	Acquires or stores the label group name.
Syntax	public string LabelGroupName { get; set; }
Property value	A string indicates the label group name Default: The string specified for Constructor
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_LABELGROUP_NAME_LENGTH</li></ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the label group name from this RegisteredLabelName class. Stores the label group name specified to this RegisteredLabelName class.

### ■LabelName

Item	Content
Function	Acquires or stores the label name.
Syntax	public string LabelName { get; set; }
Property value	A string indicates the label name Default: The string specified for Constructor
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_LABEL_NAME_LENGTH</li></ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the label name from this RegisteredLabelName class. Stores the label name specified to this RegisteredLabelName class.



## ReceiveData class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class ReceiveData
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDDataCommunication.ReceiveData

### Constructor

Format	Content
ReceiveData()	Default constructor

### Details of the constructor

#### ■ReceiveData

Item		Content
Function		Constructor
Call format		public ReceiveData()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the transfer data list. The transfer data list maintained previously is initialized.

### Property list

Type	Property name	Accessor	Overview	Reference
RecordData[]	ReceiveDataList	get	Acquires the label group name from this RegisteredLabelName class.	Page 63 ReceiveDataList

### Property details

#### ■ReceiveDataList

Item	Content
Function	Acquires the transfer data list.
Syntax	public RecordData[] ReceiveDataList { get; }
Property value	A RecordData object array indicates the transfer data list Default: null
Exception	None
Comment	Acquires the transfer data list from this ReceiveData class.

# RecordData class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class RecordData
Class inheritance	System.Object ↳ MitsubishiElectric.Melsoft.HSDDataCommunication.RecordData

## Constructor

Format	Content
RecordData()	Default constructor

## Details of the constructor

### ■RecordData


Item		Content
Function		Constructor
Call format		public RecordData()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store data used for the streaming transfer function, the data read function, and the data write function. The data list, the time synchronization flag, the data sampling time, and the index information maintained previously are initialized.

## Property list


Type	Property name	Accessor	Overview	Reference
DataValue[]	DataList	set/get	Stores the data list to this RecordData class. Acquires the data list from this RecordData class.	Page 64 DataList
uint	Index	get	Acquires the index information at the data sampling from this RecordData class.	Page 65 Index
uint	NanoTime	get	Acquires the time (less than a second) when data are sampled from this RecordData.	Page 65 NanoTime
uint	SecTime	get	Acquires the time when data are sampled from this RecordData.	Page 65 SecTime
byte	TimeSyncFlag	get	Acquires the time synchronization execution flag from this RecordData class.	Page 65 TimeSyncFlag

## Property details

### ■DataList

Item	Content
Function	Acquires or stores the data list.
Syntax	public DataValue[] DataList{ get; set; }
Property value	A DataValue object array indicates the data list. Default: null
Exception	DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • ARGUMENT_NULL For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data list from this RecordData class. Stores the specified data list to this RecordData class.

## ■Index

Item	Content
Function	Acquires the index information.
Syntax	public uint Index{ get;}
Property value	A uint value indicates the index information Default: 0
Exception	None
Comment	Acquires the index information at the data sampling from this RecordData class. Outputs a numerical value starting from 1 incremented in ascending order. When the value exceeds the upper limit of 4,294,967,295, it returns to 0 and increments again in the range of 0 to 4,294,967,295. When a data miss occurs, the index starts again from 1. For details, refer to the following section.  High Speed Data Communication Module User's Manual

## ■NanoTime

Item	Content
Function	Acquires the time (less than a second) when data are sampled.
Syntax	public uint NanoTime{ get;}
Property value	A uint value indicates the time (less than a second) when data are sampled Default: 0
Exception	None
Comment	Acquires the time when data are sampled from this RecordData class. Acquires the time less than a second from the number of elapsed seconds from January 1, 1970 in nanoseconds.

## ■SecTime

Item	Content
Function	Acquires the time when data are sampled.
Syntax	public uint SecTime{ get;}
Property value	A uint value indicates the time when data are sampled Default: 0
Exception	None
Comment	Acquires the time when data are sampled from this RecordData class. Acquires the number of elapsed seconds from January 1, 1970.

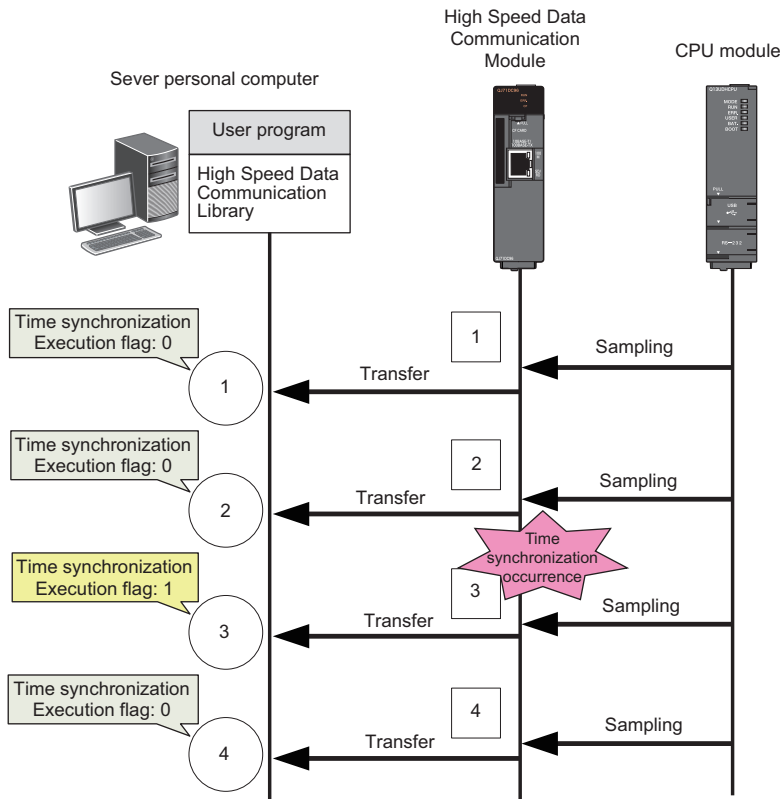
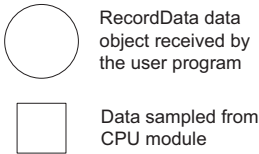
## ■TimeSyncFlag

Item	Content
Function	Acquires the time synchronization execution flag.
Syntax	public byte TimeSyncFlag { get;}
Property value	A byte value indicates the time synchronization execution flag Default: 0
Exception	None
Comment	Acquires the time synchronization execution flag from this RecordData class. Time synchronization execution: 1 Time synchronization not executed: 0 The time synchronization execution flag becomes "1" only for RecordData transferred after the success of the time query with the time synchronization function of High Speed Data Communication Module. When multiple records are transferred, the time synchronization execution flag becomes "1" only for the RecordData object to which the sampled data are stored after the time synchronization.

The following is an example when the time synchronization occurs while performing the streaming transfer.

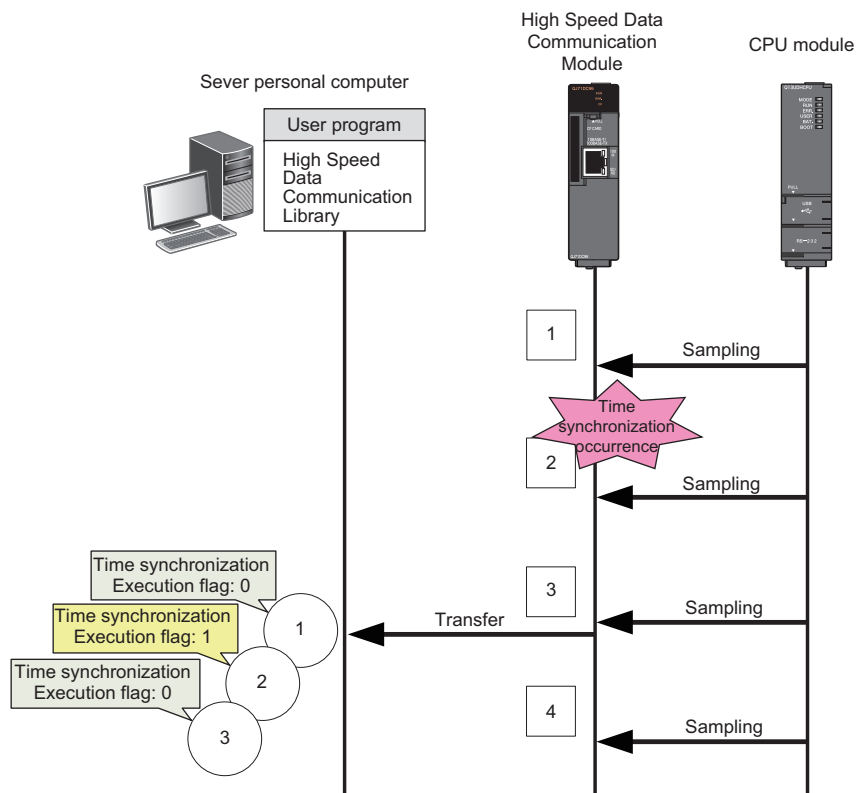
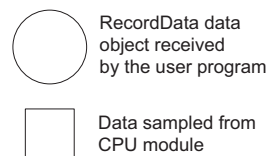
- When time synchronization occurs at the streaming transfer in sampling synchronization

\* Executing streaming transfer  
(Sampling synchronization of transfer)



- When the time synchronization occurs at the streaming transfer in record specification

\* Executing streaming transfer  
(Transfer 3 record specification)



# DataValue class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class DataValue
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDDataCommunication.DataValue

## Constructor

Format	Content
DataValue()	Default constructor

## Constructor details

### ■DataValue

Item		Content
Function		Constructor
Call format		public DataValue()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store label values. The data types and values of labels maintained previously are initialized.

## Property list


Type	Property name	Accessor	Overview	Reference
bool	BitData	set/get *1	Stores the bit data to this DataValue class. Acquires the bit data from this DataValue class.	Page 69 BitData
short	DataType	get	Acquires the data type of the stored data from this DataValue class.	Page 69 DataType
double	DoubleFloatData	set/get *1	Stores the float [double precision] data to this DataValue class. Acquires the float [double precision] data from this DataValue class.	Page 70 DoubleFloatData
byte[]	RawData	set/get *1	Stores the raw data to this DataValue class. Acquires the raw data from this DataValue class.	Page 70 RawData
int	SignedDoubleWordData	set/get *1	Stores the double word [signed] data to this DataValue class. Acquires the double word [signed] data from this DataValue class.	Page 70 SignedDoubleWordData
short	SignedWordData	set/get *1	Stores the word [signed] data to this DataValue class. Acquires the word [signed] data from this DataValue class.	Page 71 SignedWordData
float	SingleFloatData	set/get *1	Stores the float [single precision] data to this DataValue class. Acquires the float [single precision] data from this DataValue class.	Page 71 SingleFloatData
string	StringData	set/get *1	Stores the character string data to this DataValue class. Acquires the character string data from this DataValue class.	Page 71 StringData
uint	UnsignedDoubleWordData	set/get *1	Stores the double word [unsigned] data to this DataValue class. Acquires the double word [unsigned] data from this DataValue class.	Page 72 UnsignedDoubleWordData

Type	Property name	Accessor	Overview	Reference
ushort	UnsignedWordData	set/get <sup>*1</sup>	Stores the word [unsigned] data to this DataValue class. Acquires the word [unsigned] data from this DataValue class.	Page 72 UnsignedWordData

\*1 The get accessors other than DataType acquires data based on the data type of data stored in the DataValue class. Therefore, if data is acquired with a data type that differs from the data type of the stored data, a cast conversion is automatically performed.  
The data type of the acquired data may differ from the data type of the stored data depending on the data type after the cast conversion. Since the source data information may be lost or a longer processing time may be required due to the cast conversion, execute the get accessor matches with the data type to be acquired.

## Property details


### ■BitData

Item	Content
Function	Acquires or stores the bit data.
Syntax	public bool BitData{ get; set; }
Property value	A bool value indicates the bit data Default: false
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_BIT_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in bit unit. Use this property when acquiring bit type data. An exception occurs when the data type stored in the DataValue class is a character string. Stores the bit data to this DataValue class. Execute this method when storing bit type data.


### ■DataType

Item	Content
Function	Acquires the data type of the stored data.
Syntax	public short DataType { get; }
Property value	A short value indicates the data type Default: 0
Exception	None
Comment	Acquires the data type of the data stored in this DataValue class. The following are the values for each output data type. Bit: 1 Word [signed]: 10 Double word [signed]: 11 Word [unsigned]: 12 Double word [unsigned]: 13 Float [single precision]: 20 Float [double precision]: 21 String: 30 Raw: 40


## ■DoubleFloatData

Item	Content
Function	Acquires or stores the data in float [double precision] unit.
Syntax	public double DoubleFloatData { get; set; }
Property value	A double value indicates the float [double precision] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in float [double precision] unit. Use this property when acquiring float [double precision] data. Stores the float [double precision] data to this DataValue class. Execute this method when storing float [double precision] data.

## ■RawData


Item	Content
Function	Acquires or stores the raw data.
Syntax	public byte[] RawData { get; set; }
Property value	A byte array indicates the raw data Default: null
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> <li>• OUT_RANGE_RAW_LENGTH</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in raw unit. Use this property when acquiring raw data. When data acquired from the module are stored, the same amount of data set to the acquired label is acquired. (For example, when the data size is 4, the array element to be stored is 4.) Stores the raw data to this DataValue class. An exception occurs if the number of array elements of data to be stored exceeds 8192.

## ■SignedDoubleWordData


Item	Content
Function	Acquires or stores the double word [signed] data.
Syntax	public int SignedDoubleWordData{ get; set; }
Property value	An int value indicates the double word [signed] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in double word unit. Use this property when acquiring double word [signed] data. Stores the double word [signed] data to this DataValue class. Execute this method when storing double word [signed] data.




## ■SignedWordData

Item	Content
Function	Acquires or stores the word [signed] data.
Syntax	public short SignedWordData{ get; set; }
Property value	A short value indicates the word [signed] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in word unit. Use this property when acquiring word [signed] data. Stores the word [signed] data to this DataValue class. Execute this method when storing word [signed] data.

## ■SingleFloatData

Item	Content
Function	Acquires or stores data in float [single precision] unit.
Syntax	public float SingleFloatData { get; set; }
Property value	A float value indicates the float [single precision] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in float [single precision] unit. Use this property when acquiring float [single precision] data. Stores the float [single precision] data to this DataValue class. Execute this method when storing float [single precision] data.

## ■StringData

Item	Content
Function	Acquires or stores the character string data.
Syntax	public string StringData { get; set; }
Property value	A string indicates the character string data. Default: null
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• NOT_DATA_SET</li> <li>• OUT_RANGE_STRING_LENGTH</li> <li>• UNSUPPORTED_ENCODING</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in character string unit. Use this property when acquiring data in character string despite of the data type of the stored data. The encoding code of the acquired character string data is "UTF-16LE". <sup>*1</sup> Stores the character string data to this DataValue class. Execute this method when storing character string data. An exception occurs if the number of characters of data to be stored exceeds 4096. The encoding code of the stored character string data is "UTF-16LE". <sup>*2</sup>

\*1 When character string data is acquired with the "UTF-16LE" method, a single character is acquired as a two-byte data.

Therefore, the incorrect values are read if data are stored with one-byte ASCII characters.


When acquiring character string data in one-byte unit, specify the encoding code (such as "iso-8859-1") to set data in one-byte unit with GetStringData (string encode).

\*2 When character string data are stored with the "UTF-16LE" method, a single character is stored as a two-byte data. (For example, when a character "A" is stored, it is stored as a two-byte data.)


Therefore, when the stored data is written to the label of data size 1, the exception INVALID\_DATASIZE occurs.

When storing character string data in one-byte unit, specify the encoding code (such as "iso-8859-1") to set data in one-byte unit with SetStringData (string stringData, string encode).

## ■UnsignedDoubleWordData

Item	Content
Function	Acquires or stores the double word [unsigned] data.
Syntax	public uint UnsignedDoubleWordData { get; set; }
Property value	A uint value indicates the double word [unsigned] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in double word unit. Use this property when acquiring double word [unsigned] data. Stores the double word [unsigned] data to this DataValue class. Execute this method when storing double word [unsigned] data.

## ■UnsignedWordData

Item	Content
Function	Acquires or stores the word [unsigned] data.
Syntax	public ushort UnsignedWordData { get; set; }
Property value	A ushort value indicates the word [unsigned] data Default: 0
Exception	DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment	Acquires the data stored in this DataValue class in word unit. Use this property when acquiring word [unsigned] data. Stores the word [unsigned] data to this DataValue class. Execute this method when storing word [unsigned] data.


## Method list

Type	Method name	Overview	Reference
string	GetStringData(string encode) <sup>*1</sup>	Acquires the character string data from this DataValue class by specifying the character code.	Page 73 GetStringData
void	SetStringData(string stringData, string encode)	Stores the character string data to this DataValue class by specifying the character code.	Page 73 SetStringData

<sup>\*1</sup> The GetStringData processing acquires data based on the data type of data stored in the DataValue class. Therefore, if the data type of the stored data is not character string, a cast conversion is automatically performed. The data type of the acquired data may differ from the data type of the stored data depending on the data type after the cast conversion. Since the source data information may be lost or a longer processing time may be required due to the cast conversion, execute this method when the data type is character string.


## Method details

### ■GetStringData

Item		Content
Function		Acquires the character string data.
Call format		public string GetStringData(string encode) throws DataCommunicationException
Argument	Type name	string
	Variable name	encode
	Content	[IN] Encoding code of the character string data to be acquired
Return value		Returns the character string data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• OUT_RANGE_STRING_LENGTH</li> <li>• UNSUPPORTED_ENCODING</li> </ul> For details, refer to the following section.  Page 81 ErrorCode
Comment		Acquires data stored in this DataValue class in character string unit. Character string data are acquired with the specified encoding code.*1 Use this method when acquiring stored character string data by specifying the encoding code. "null" is stored to the return value by default.

\*1 Use this method when the data type of data stored in the DataValue class is character string.  
If this method is used for the data type other than character string, the specified character code is ignored.

### ■SetStringData

Item		Content	
Function		Stores the character string data.	
Call format		public void SetStringData(string stringData, string encode) throws DataCommunicationException	
Argument	Type name	string	string
	Variable name	stringData	encode
	Content	[IN] Character string data	[IN] Encoding code for character string
Return value		None	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• OUT_RANGE_STRING_LENGTH</li> <li>• UNSUPPORTED_ENCODING</li> </ul> For details, refer to the following section.  Page 81 ErrorCode	
Comment		Stores the character string data to this DataValue class by specifying the encoding code. An exception occurs if the number of characters of data to be stored exceeds 8192. (Example) <ul style="list-style-type: none"> <li>• Specifying the encoding code "UTF-16LE" (two-byte single character)</li> </ul> An exception occurs when the character string data with the number of characters exceeding 4096 is stored. <ul style="list-style-type: none"> <li>• Specifying the encoding code "iso-8859-1" (one-byte single character)</li> </ul> An exception occurs when the character string data with the number of characters exceeding 8192 is stored.	

# ConnectionDeviceList class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class ConnectionDeviceList
Class inheritance	System.Object └─ MitsubishiElectric.Melsoft.HSDDataCommunication.ConnectionDeviceList

## Constructor

Format	Content
ConnectionDeviceList()	Default constructor

## Details of the constructor

### ■ConnectionDeviceList

Item		Content
Function		Constructor
Call format		public ConnectionDeviceList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the list of the connected device information. The information maintained previously is initialized.

## Property list

Type	Property name	Accessor	Overview	Reference
ConnectionDevice[]	ConnectionDeviceArray	get	Acquires the list of the connected device information from this ConnectionDeviceList class.	Page 74 ConnectionDeviceList

## Property details

### ■ConnectionDeviceArray

Item	Content
Function	Acquires the list of the connected device information.
Syntax	public ConnectionDevice[] ConnectionDeviceArray{ get; }
Property value	A ConnectionDevice object array indicates the list of the connected device information Default: null
Exception	None
Comment	Acquires the list of the connected device information from this ConnectionDeviceList class.*1 "null" is acquired by default.

\*1 When values are stored using the GetConnectionDeviceList method in the Communication class, "null" is stored to the ConnectionDeviceArray array elements in this class for the information of devices without a connection.

# ConnectionDevice class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDataCommunication
Class syntax	public class ConnectionDevice
Class inheritance	System.Object ↳ MitsubishiElectric.Melsoft.HSDataCommunication.ConnectionDevice

## Constructor

Format	Content
ConnectionDevice()	Default constructor

## Details of the constructor

### ■ConnectionDevice

Item		Content
Function		Constructor
Call format		public ConnectionDevice()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the list of the connected device information.

## Property list

Type	Property name	Accessor	Overview	Reference
short	AccessStatus	get	Acquires the access status from this ConnectionDevice class.	Page 76 AccessStatus
ConnectionDeviceDetail	ConnectionDeviceDetail	get	Acquires the details of the connected device information from the ConnectionDevice class.	Page 76 ConnectionDeviceDetail
string	ConnectionIPAddress	get	Acquires the connection IP address from this ConnectionDevice class.	Page 77 ConnectionIPAddress
uint	ConnectionTime	get	Acquires the connection time from this ConnectionDevice class.	Page 77 ConnectionTime
string	ConnectionUsername	get	Acquires the connection user name from this ConnectionDevice class.	Page 77 ConnectionUsername

## Property details

### ■AccessStatus

Item	Content
Function	Acquires the access status.
Syntax	public short AccessStatus { get; }
Property value	A value for the access status Default: 0
Exception	None
Comment	Acquires the access status from this ConnectionDevice class.

The following are the values for the access status.

Access status	Buffering invalid	Buffering enabled
No connection	0	
Connecting (The connection is established.)	10	11
Reading device data	20	21
Writing device data	30	31
On transfer (High speed sampling)	40	41
On transfer (General sampling)	60	61
Buffering transfer data (High speed sampling)	—	51
Buffering transfer data (General sampling)	—	71
On communication with tool	255	—

### ■ConnectionDeviceDetail

Item	Content
Function	Acquires the details of the connected device information.
Syntax	public ConnectionDeviceDetail ConnectionDeviceDetail { get; }
Property value	A value for the connected device information Default: null
Exception	None
Comment	Acquires the details of the connected device information from the ConnectionDevice class.*1

\*1 When values are stored using the GetConnectionDeviceList method in the Communication class, "null" is acquired for the access status other than the ones indicated below.

- On connection: 10
- On connection (Buffering enabled): 11
- Reading device data: 20
- Reading device data (Buffering enabled): 21
- Writing device data: 30
- Writing device data (Buffering enabled): 31
- On transfer (High speed sampling): 40
- On transfer (High speed sampling) (Buffering enabled): 41
- Buffering transfer data (High speed sampling): 51
- On transfer (General sampling): 60
- On transfer (General sampling) (Buffering enabled): 61
- Buffering transfer data (General sampling): 71

## ■ConnectionIPAddress

Item	Content
Function	Acquires the connection IP address.
Syntax	public string ConnectionIPAddress { get; }
Property value	A value for the connection IP address Default: null
Exception	None
Comment	Acquires the connection IP address from this ConnectionDevice class.

## ■ConnectionTime

Item	Content
Function	Acquires the connection time.
Syntax	public uint ConnectionTime { get; }
Property value	Connection time (the number of elapsed seconds from January 1, 1970) Default: 0
Exception	None
Comment	Acquires the connection time from this ConnectionDevice class.

## ■ConnectionUsername

Item	Content
Function	Acquires the connection user name.
Syntax	public string ConnectionUsername { get; }
Property value	A value for the connection user name Default: null
Exception	None
Comment	Acquires the connection user name from this ConnectionDevice class.*1

\*1 When the access authentication is disabled, "null" is stored to the connection user name.

## ConnectionDeviceDetail class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public class ConnectionDeviceDetail
Class inheritance	System.Object ↳ MitsubishiElectric.Melsoft.HSDDataCommunication.ConnectionDeviceDetail

### Constructor

Format	Content
ConnectionDeviceDetail()	Default constructor

### Details of the constructor

#### ■ConnectionDeviceDetail

Item		Content
Function		Constructor
Call format		public ConnectionDeviceDetail()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the details of the connected device information.

### Property list

Type	Property name	Accessor	Overview	Reference
uint	BufferingDataCountCurrent	get	Acquires the unprocessed data count (present) from this ConnectionDeviceDetail class.	Page 79 BufferingDataCountCurrent
uint	BufferingDataCountMaximum	get	Acquires the unprocessed data count (maximum) from this ConnectionDeviceDetail class.	Page 79 BufferingDataCountMaximum
ushort	HighSpeedSamplingFailureCount	get	Acquires the number of high speed sampling failures from this ConnectionDeviceDetail class.	Page 79 HighSpeedSamplingFailureCount
short	LatestErrorCode	get	Acquires the current error code from this ConnectionDeviceDetail class.	Page 79 LatestErrorCode
uint	PossibleBufferingDataCount	get	Acquires the number of data that can be buffered from this ConnectionDeviceDetail class.	Page 79 PossibleBufferingDataCount
uint	PossibleBufferingTime	get	Acquires the total time that data can be buffered from this ConnectionDeviceDetail class.	Page 80 PossibleBufferingTime
ushort	ProcessingOverloadCount	get	Acquires the number of processing overload occurrences from this ConnectionDeviceDetail class.	Page 80 ProcessingOverloadCount
uint	SamplingIntervalAverage	get	Acquires the sampling time (moving average) from this ConnectionDeviceDetail class.	Page 80 SamplingIntervalAverage
uint	SamplingIntervalMaximum	get	Acquires the sampling time (maximum) from this ConnectionDeviceDetail class.	Page 80 SamplingIntervalMaximum



## Property details

### ■BufferingDataCountCurrent

Item	Content
Function	Acquires the unprocessed data count (present).
Syntax	public uint BufferingDataCountCurrent { get; }
Property value	A value of the unprocessed data count (present) Default: 0
Exception	None
Comment	Acquires the unprocessed data count (present) from this ConnectionDeviceDetail class.

### ■BufferingDataCountMaximum

Item	Content
Function	Acquires the unprocessed data count (maximum).
Syntax	public uint BufferingDataCountMaximum { get; }
Property value	The value of the unprocessed data count (maximum) Default: 0
Exception	None
Comment	Acquires the unprocessed data count (maximum) from this ConnectionDeviceDetail class.

### ■HighSpeedSamplingFailureCount

Item	Content
Function	Acquires the number of high speed sampling failures.
Syntax	public ushort HighSpeedSamplingFailureCount { get; }
Property value	A value for the number of high speed sampling failures Default: 0
Exception	None
Comment	Acquires the number of high speed sampling failures from this ConnectionDeviceDetail class.*1

\*1 When the access status is "Streaming transfer (General sampling)" or "Buffering transfer data (General sampling)", "0" is stored to the number of high speed sampling failures.

### ■LatestErrorCode

Item	Content
Function	Acquires the current error code.
Syntax	public short LatestErrorCode { get; }
Property value	A value for the current error code Default: 0
Exception	None
Comment	Acquires the current error code from this ConnectionDeviceDetail class.

### ■PossibleBufferingDataCount

Item	Content
Function	Acquires the number of data that can be buffered.
Syntax	public uint PossibleBufferingDataCount { get; }
Property value	A value for the number of data that can be buffered Default: 0
Exception	None
Comment	Acquires the number of data that can be buffered from this ConnectionDeviceDetail class.

## ■PossibleBufferingTime

Item	Content
Function	Acquires the total time that data can be buffered.
Syntax	public uint PossibleBufferingTime { get; }
Property value	A value for the total time that data can be buffered Default: 0
Exception	None
Comment	Acquires the total time that data can be buffered from this ConnectionDeviceDetail class.

## ■ProcessingOverloadCount

Item	Content
Function	Acquires the number of processing overload occurrences.
Syntax	public ushort ProcessingOverloadCount { get; }
Property value	A value for the number of processing overload occurrences Default: 0
Exception	None
Comment	Acquires the number of processing overload occurrences from this ConnectionDeviceDetail class.

## ■SamplingIntervalAverage

Item	Content
Function	Acquires the sampling time (moving average).
Syntax	public uint SamplingIntervalAverage { get; }
Property value	A value for the sampling time (moving average) Default: 0
Exception	None
Comment	Acquires the sampling time (moving average) from this ConnectionDeviceDetail class.

## ■SamplingIntervalMaximum

Item	Content
Function	Acquires the sampling time (maximum).
Syntax	public uint SamplingIntervalMaximum { get; }
Property value	A value for the sampling time (maximum) Default: 0
Exception	None
Comment	Acquires the sampling time (maximum) from this ConnectionDeviceDetail class.

# DataCommunicationException class

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDataCommunication
Class syntax	public class DataCommunicationException
Class inheritance	System.Object └─ System.Exception └─ System.ApplicationException └─ MitsubishiElectric.Melsoft.HSDataCommunication.DataCommunicationException

## Constructor

Format	Content
DataCommunicationException()	Default constructor

## Details of the constructor

### ■DataCommunicationException

Item		Content
Function		Constructor
Call format		public DataCommunicationException()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the enumerators of the exceptions at the method execution. The enumerators of the exceptions maintained previously are initialized.

## Property list

Type	Property name	Accessor	Overview	Reference
DataCommunicationError	ErrorCode	get	Acquires the enumerators of the exceptions from this DataCommunicationException class.	Page 81 ErrorCode

## Property details

### ■ErrorCode

Item	Content
Function	Acquires the data type of the stored data.
Syntax	public DataCommunicationError ErrorCode { get; }
Property value	A DataCommunicationError enumerator indicates the data type Default: NONE
Exception	None
Comment	Acquires the enumerators of the exceptions from this DataCommunicationException class. For the exception handler occurrence, refer to the following section. ☞ Page 82 DataCommunicationError enumerator




# DataCommunicationError enumerator

Item	Content
Namespace	MitsubishiElectric.Melsoft.HSDDataCommunication
Class syntax	public enum DataCommunicationError
Class inheritance	System.Object └ System.Enum<DataCommunicationError> └── MitsubishiElectric.Melsoft.HSDDataCommunication.DataCommunicationError

## Enumerator list

Enumerator	Description	Corrective action
ABORTED_BY_PEER	Method suspension error	Check if a method other than StopStreamingTransfer is executed during the streaming transfer.
ARGUMENT_NULL	Null was specified for the argument. A null object exists in the object specified for the argument.	Review the value specified for the argument, and execute the method again.
AUTHENTICATION_FAILED	The access authentication failed.	Check the account authentication setting and check if the specified user name and password are correct.
BUFFERING_CONNECT_OVER	The number of connections in which the buffering data transfer is enabled exceeded the maximum number of applicable connections.	Refer to the event history/connection history, check if there are two connections which are enabled buffering transfer data. Take the following corrective action as necessary. <ul style="list-style-type: none"> <li>• Disconnect the connection of the user program executing Connect</li> <li>• Stop buffering transfer data</li> </ul>
CONNECT_OVER	The maximum number of devices that can be connected exceeded.	Refer to the event history/connection history, check if the connection for High Speed Data Communication Module is more than 5. Also, check if there is an unexpected access when CONNECT_OVER occurred. Take the following corrective action as necessary. <ul style="list-style-type: none"> <li>• Stop accessing from Configuration Tool</li> <li>• Disconnect the connection of the user program executing Connect</li> <li>• Stop buffering transfer data</li> </ul>
CONNECT_REFUSED	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
DISCONNECTED	A disconnection was attempted to the connection of the High Speed Data Communication Module which is not connected. A communication was attempted without the reconnection after the line disconnection event occurred.	Connect the line.
DISPOSED_ERROR	The line was disconnected.	Execute the connection processing again.
EXCHANGE_FROM_STRING_FAILED	The conversion from the character string failed.	Check if .NET Framework 3.5 is installed.
EXCHANGE_TO_BIT_FAILED	A conversion from character string data to bit device was attempted.	Check the data type of the stored data. If the data type is character string, do not acquire the data as a bit device.
EXCHANGE_TO_NUMBER_FAILED	A conversion from character string data to raw data is failed.	Check that the stored character string can be converted to acquirable data (integers or decimals) with the specified method. Check that the stored character string can be converted to acquirable data with the specified method.
EXCHANGE_TO_STRING_FAILED	A conversion to character string data failed.	Check if .NET Framework 3.5 is installed.
EXECUTING	Another command was executed from one instance while the module was in processing.	Execute the method again after canceling or completing the processing being executed.
GENERAL_ERROR	Unknown error	Check if .NET Framework 3.5 is installed.
INTERRUPTED_ERROR	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
INVALID_CONNECT		
INVALID_DATASIZE	A data size that exceeds the label data size specified in Configuration Tool.	Check that the specified label is correct. Check that the stored data size (byte) does not exceed the data size set in Configuration Tool.

Enumerator	Description	Corrective action
INVALID_DATATYPE	A data type that differs from the data type of the label written in Configuration Tool was specified.	Check that the specified label is correct. Check that the data type of the stored data is correct.
INVALID_IPADDRESS	The specified IP address is invalid.	Check that the specified IP address is within the IP address range.
INVALID_REQUEST	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
INVALID_RESPONSE		
INVALID_STREAMING		
IO_ERROR	A communication error occurred.	Check the following. <ul style="list-style-type: none"> <li>• High Speed Data Communication Module status, network connection status</li> <li>• Cable connection.</li> </ul> Review the value specified for the IP address, and then execute the method again.
IPADDRESS_NULL	Null was specified for the IP address.	Check the value specified for the IP address, and then execute the method again.
NONE	No errors	—
NOT_CONNECTED_YET	The connection processing is not executed.	Execute the connection processing, and then executed the method again.
NOT_DATA_SET	Data is not stored.	Store the data and execute the method again.
NOT_PERMITTED_IPADDRESS	An IP address which is not authorized to connect was specified.	Check the specified IP address for the connection, and check the access authentication. Check that the specified IP address is correct.
NOT_SAME_ELEMENT_COUNT	The number of specified labels or devices does not match with the number of data.	Check the specified argument, and set each number of elements to be the same amount.
OUT_RANGE_ADDITIONAL_TIMETYPE	The specified type of time information to be appended is out of the range.	Check that the specified type of additional time information is within the range.
OUT_RANGE_CONNECT_MODE	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
OUT_RANGE_CONNECTION_NO	The specified connection number is out of the range.	Check that the specified connection number is within the range of 1 to 5.
OUT_RANGE_DEVICE_COUNT	The specified number of devices is out of the range.	Check that the specified number of devices is within the range.
OUT_RANGE_DEVICE_NAME_LENGTH	The length of the specified device name is out of the range.	Check that the length of the specified device name is within the range.
OUT_RANGE_LABEL_COUNT	The specified number of labels is out of the range.	Check that the specified number of labels is within the range.
OUT_RANGE_LABEL_NAME_LENGTH	The length of the specified label name is out of the range.	Check that the length of the specified label name is within the range.
OUT_RANGE_LABELGROUP_NAME_LENGTH	The length of the specified label group name is out of the range.	Check that the length of the specified label group name is within the range.
OUT_RANGE_PASSWORD	The length of the specified password is out of the range.	Check that the length of the specified password is within the range.
OUT_RANGE_RAW_LENGTH	The number of the specified raw data is out of the range.	Check that the number of the specified raw data is within the range.
OUT_RANGE_STRING_LENGTH	The length of the specified character string is out of the range.	Check that the length of the specified character string is within the range.
OUT_RANGE_TIMEOUT	The specified timeout period is out of the range.	Check that the specified timeout period is within the range.
OUT_RANGE_TRANSFER_TIME	The specified transfer cycle is out of the range.	Check that the specified transfer cycle is within the range.
OUT_RANGE_USER_NAME	The length of the specified user name is out of the range.	Check that the length of the specified user name is within the range.
OUT_RANGE_WORD_SIZE	The total number of words of the specified label is out of the range.	Check that the number of words of the specified label is within the range.
PASSWORD_NULL	Null was specified for the password.	Review the value of the specified password, and then execute the method again.
RECONNECT	A connection is attempted to the module which is already connected.	To reconnect, disconnect the line and then connect again.

Enumerator	Description	Corrective action
STREAMING_CONNECT_OVER	The maximum number of devices that can be connected for the streaming transfer exceeded.	<p>Refer to the event history/connection history, check if the connection of the user program is more than 4. Also, check if there is an unexpected access when STREAMING_CONNECT_OVER occurred.</p> <p>Take the following corrective action as necessary.</p> <ul style="list-style-type: none"> <li>• Stop accessing from Configuration Tool</li> <li>• Disconnect the connection of the user program executing Connect</li> <li>• Stop buffering transfer data</li> </ul>
TIMEOUT	The communication was timed-out.	<p>Check the following:</p> <ul style="list-style-type: none"> <li>• High Speed Data Communication Module status, network connection status</li> <li>• Cable connection</li> <li>• Network connection status of CPUs on other stations</li> </ul> <p>Execute the method again after performing the following actions.</p> <ul style="list-style-type: none"> <li>• Set a longer timeout period for the TimeOut property.</li> <li>• Reduce the number of CPUs on other stations that are accessed simultaneously.</li> <li>• Reduce the number of device points that are accessed simultaneously.</li> </ul> <p>Review the value of the specified IP address, and then execute the method again.</p> <p>When using the StartStreamingTransferByInterval method or StartStreamingTransfer method, refer to the details of the method to be used.</p> <p> Page 47 StartStreamingTransfer</p> <p> Page 48 StartStreamingTransfer</p> <p> Page 49 StartStreamingTransferByInterval</p>
UNSUPPORTED_ENCODING	The unsupported character code is specified.	<p>Check if .NET Framework 3.5 is installed.</p> <p>Check if the specified character code is supported by .NET Framework 3.5.</p>

## 5.4 Build C# Program

---

This section explains how to build the user program created with C#.

### Operating procedure

- Open the solution file to build the program, and select [Build] ⇨ [Build Solution].

The alternative operation to execute the build: select the project to be build on the solution explorer, and select [Build] on the right-click menu.

# MEMO

---



# 6 EXECUTING C# USER PROGRAM

---

This chapter explains how to execute a user program on a server personal computer.

## 6.1 Copying Execution File for User Program to Server Personal Computer

---

Copy the execution file (XXXX.exe) created by a development personal computer and High Speed Data Communication Library (HSDDataComLib.dll) to a server personal computer.

Copying the user program (such as XXXX.cs) used for programming on the development personal computer is not needed.

## 6.2 Executing Processing from Server Personal Computer

---

Execute the execution file to start processing.

Execute the execution file according to the format.



# PART 3

# CREATING Java PROGRAMS

## PART 3

This part explains the creating method of Java program and class method.

7 CREATING Java PROGRAMS

---

8 EXECUTING Java USER PROGRAM

---

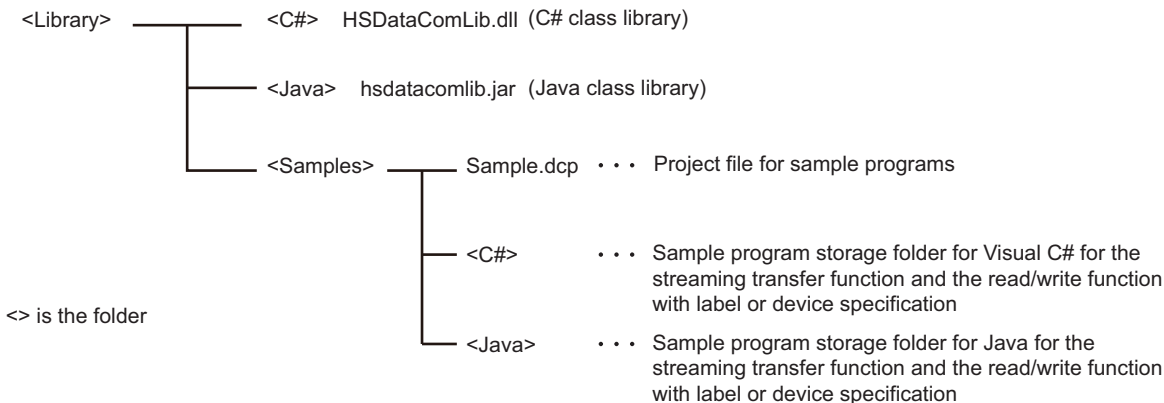
## 7

## 7.1

# Copying High Speed Data Communication Library

Copy High Speed Data Communication Library under the "Library" folder in the compressed file of this product to a folder (the working folder when creating the user program) on a personal computer.

The following diagram shows the structure of the "Library" folder.



## Using High Speed Data Communication Library

To use the classes and methods provided by High Speed Data Communication Library, High Speed Data Communication Library needs to be imported on the source code of the user program.

## Using Java class library

Write the import declaration on the user program to use High Speed Data Communication Library.

For the import declaration, write "jp.co.mitsubishielectric.hsdatacommunication.\*" or write only classes necessary for the user program.

For details of the import declaration, refer to the Java language specifications.

The following is an example of the import declaration.

**Ex.**

When importing "jp.co.mitsubishielectric.hsdatacommunication.\*"

```
//Import High Speed Data Communication Library
import jp.co.mitsubishielectric.hsdatacommunication.*;
```

**Ex.**

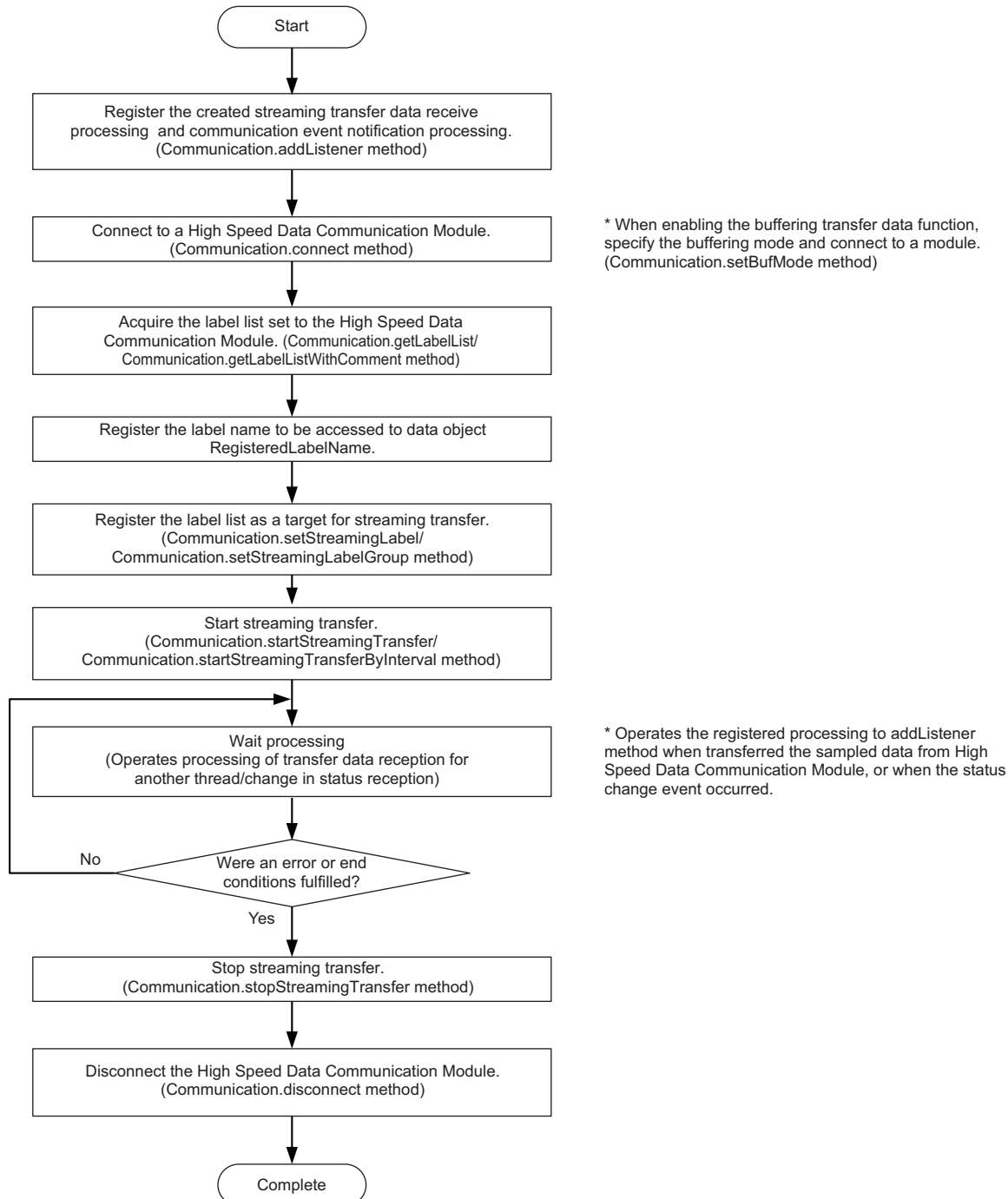
## When importing only necessary classes

```
//Import High Speed Data Communication Library
import jp.co.mitsubishielectric.hsdatacommunication.Communication;
import jp.co.mitsubishielectric.hsdatacommunication.DataCommunicationException;
import jp.co.mitsubishielectric.hsdatacommunication.DataCommunicationError;
```

## 7.2 Designing Programs

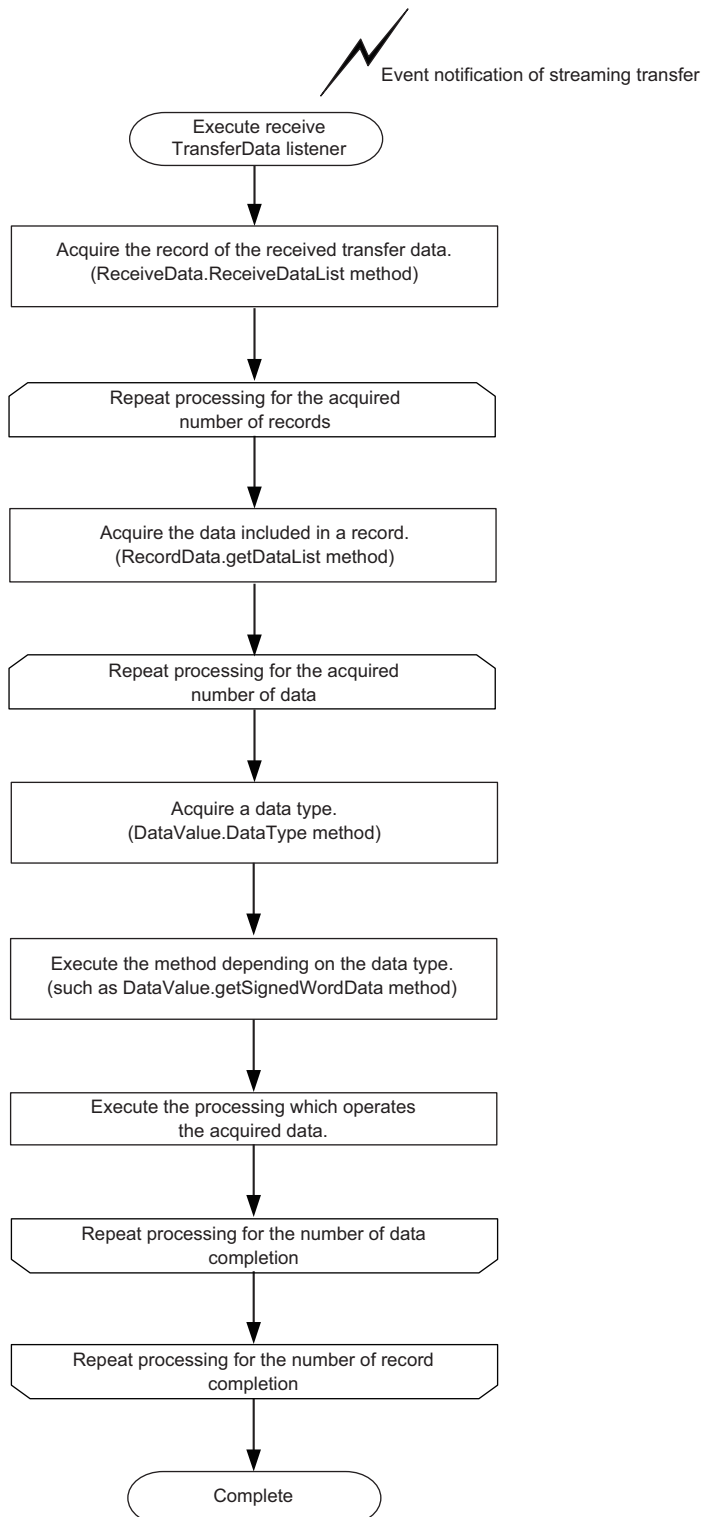
### Streaming transfer

The following flow chart shows the processes to execute the streaming transfer after connecting to a High Speed Data Communication Module and registering the target label list of the streaming transfer. In order to receive the data to be transferred, register the transfer data receive processing with the addListener method of Communication class.



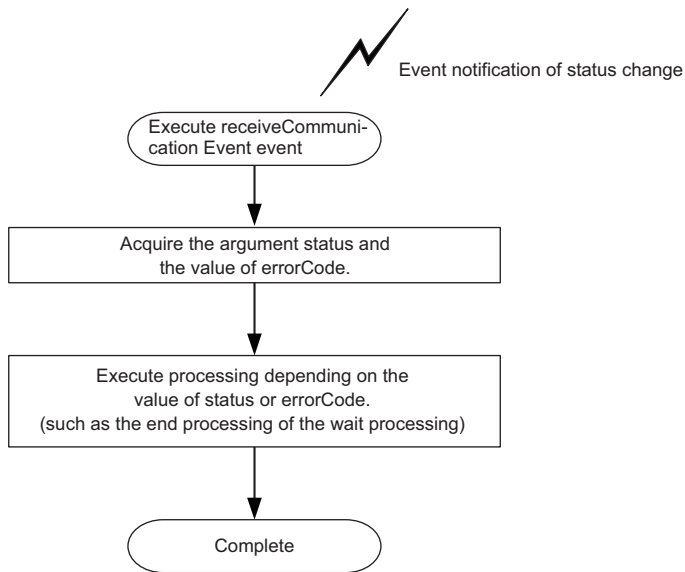
## Receiving streaming transfer data

The following flow chart shows the processes of the user program when received data transfer result from a High Speed Data Communication Module after starting streaming transfer.



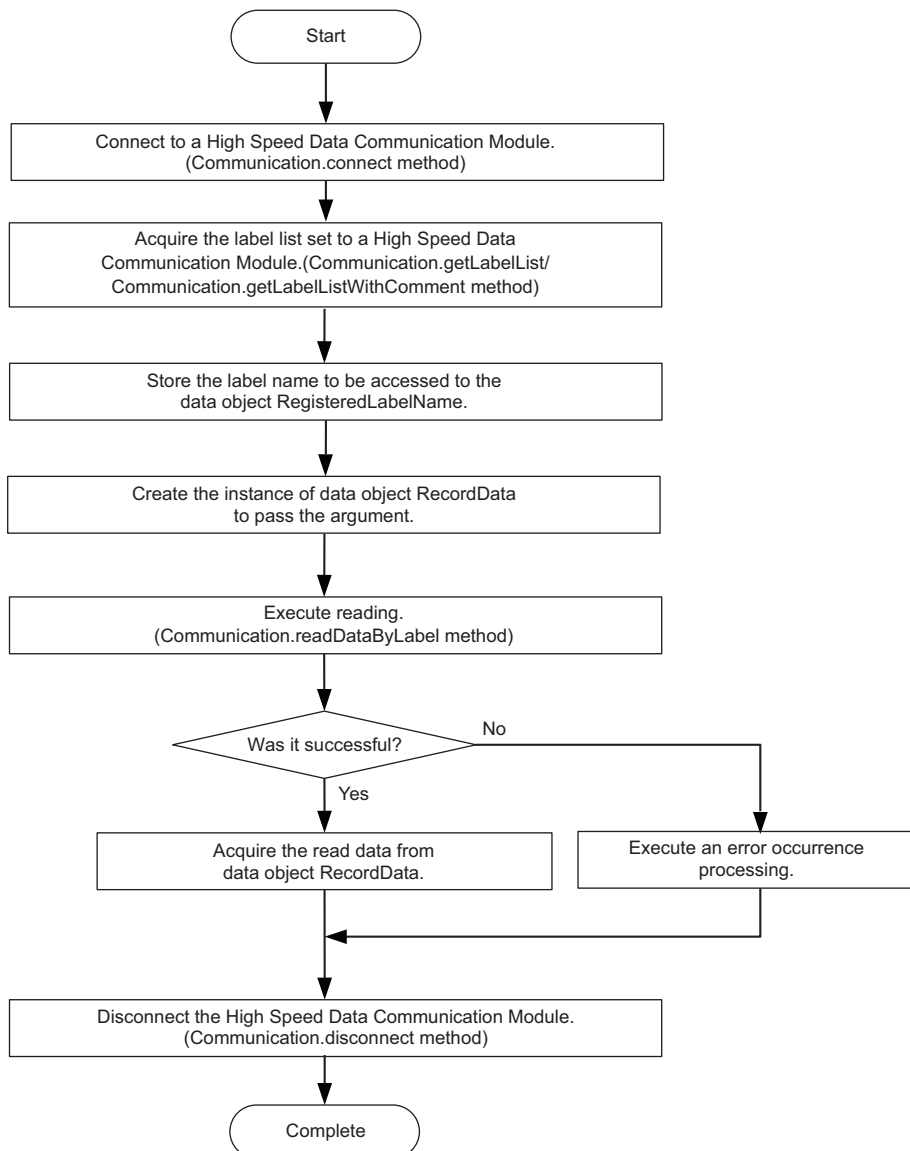
## Receiving notifications of module operation status changes

The following flow chart shows the processes when received the notification of module status changes from a High Speed Data Communication Module. End the wait process of the streaming transfer as necessary.



## Reading data using label specification

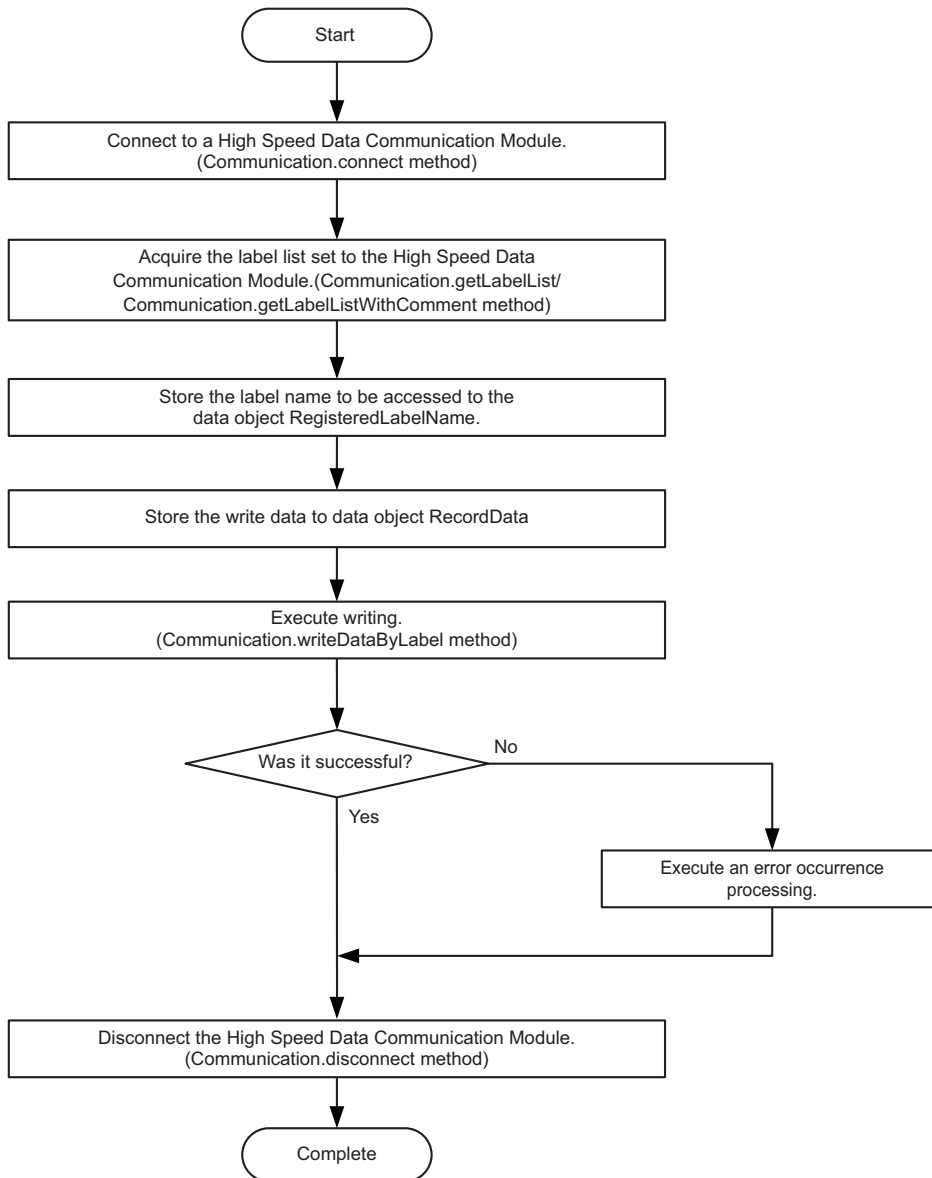
The following flow chart shows the operations to read data by specifying the target label list for the data read function after connecting to a High Speed Data Communication Module.





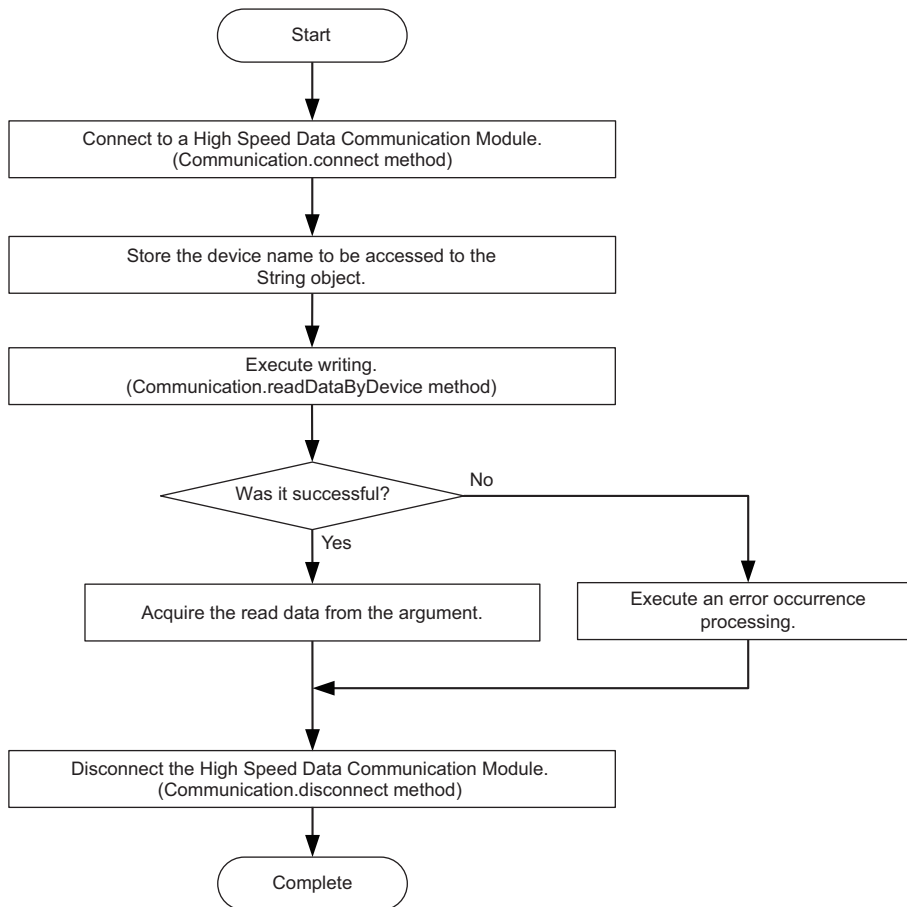
## Writing data using label specification

The following flow chart shows the operations to write data by specifying the target label list for the data read function after connecting to a High Speed Data Communication Module.



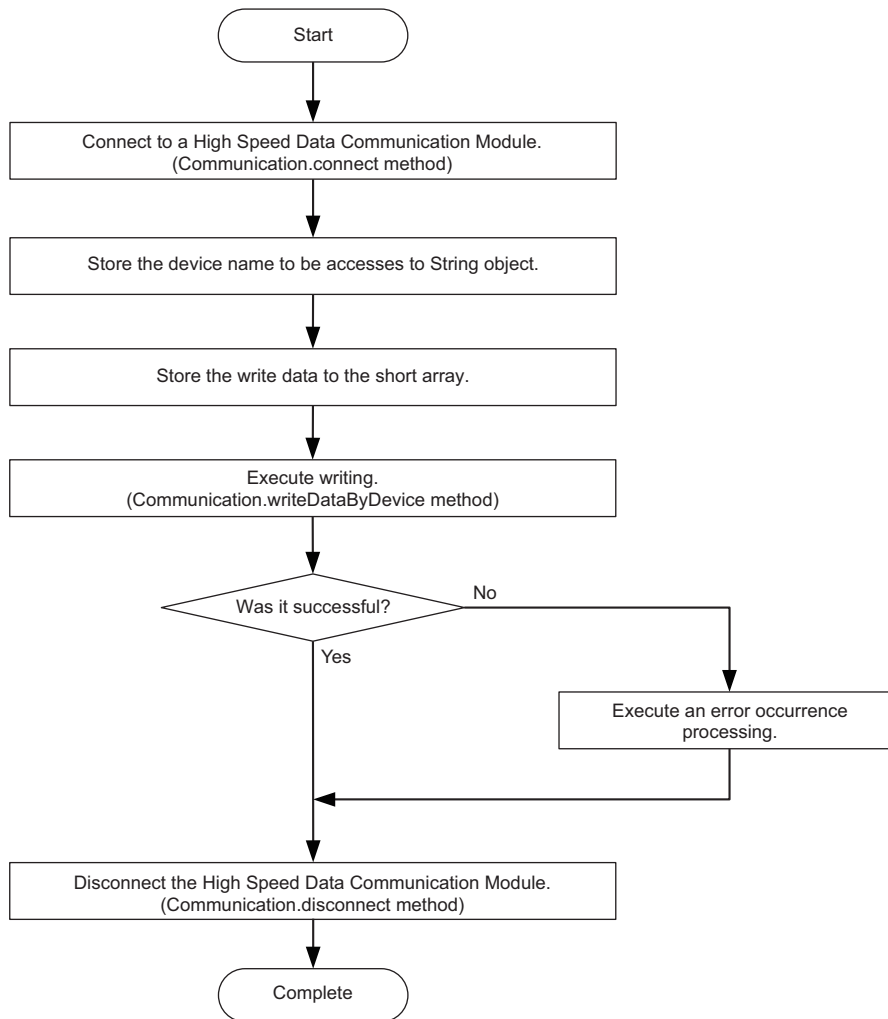
## Reading data using device specification

The following flow chart shows the operations to read data by specifying the target device list for the data read function after connecting to a High Speed Data Communication Module.



## Writing data using device specification

The following flow chart shows the operations to write data by specifying the target device list for the data read function after connecting to a High Speed Data Communication Module.



# Programming precautions

## connect/disconnect processing for Communication class

The efficient communication can be established by executing this method only once at the beginning and the end of the program.

## Streaming transfer data receive processing

In data receive processing of the streaming transfer, if a large amount of data are received and thus the longer processing time is required, the transfer data received from the module during the processing are discarded.

Confirm the continuous data reception by checking the index information of each record exists in the reception data are in sequence.

Note that the sequential order of the index information is corrupted even when a data miss occurs on the module side.

For details of conditions for data miss occurrences, refer to the following section.

 High Speed Data Communication Module User's Manual

## addListener processing for Communication class

Execute this method before executing the connect processing for Communication class.

If the addListener processing is executed after executing the connect processing, the streaming transfer from the High Speed Data Communication Module and module status changes cannot be received.

## get processing of DataValue class

The get processing other than getDataType of the DataValue class acquires data based on the data type of data stored in the DataValue class.

Therefore, if data is acquired with a data type that differs from the data type of the stored data, a cast conversion is automatically performed.

The data type of the acquired data may differ from the data type of the stored data depending on the data type after the cast conversion.

Since the source data information may be lost or a longer processing time may be required due to the cast conversion, execute the get processing matches with the data type to be acquired.

## Communication during streaming transfer

A 06B0H error occurs when any of the following Communication class methods is executed for a single instance after executing the startStreamingTransfer/startStreamingTransferByInterval method.

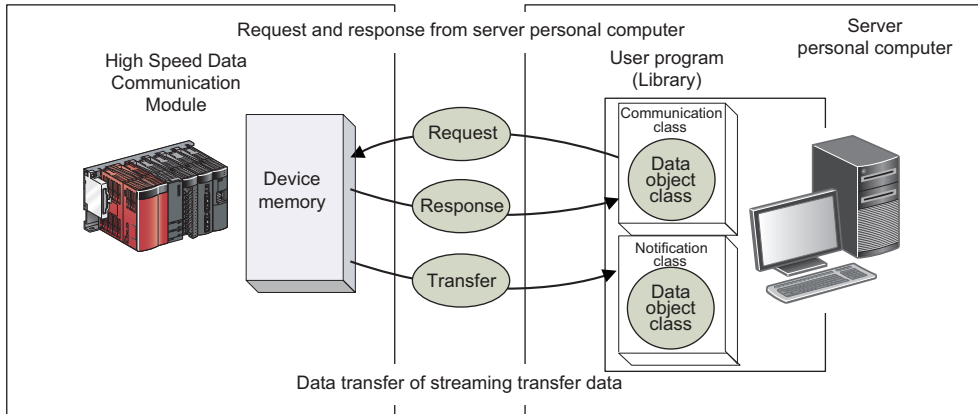
Execute the following methods after executing the stopStreamingTransfer method.

- getConnectionDeviceList
- getLabelList
- getLabelListWithComment
- setStreamingLabel
- setStreamingLabelGroup
- startStreamingTransferByInterval
- startStreamingTransfer
- stopStreamingDataBuffering
- readDataByLabel
- readDataByDevice
- writeDataByLabel
- writeDataByDevice

## 7.3 Class and Method

The classes in High Speed Data Communication Library are the communication class that realizes the streaming transfer function and the data read/write function, the notification class to receive streaming transfer data and module status changes, and the data object class that stores data for communication.

The following figure shows the functions of each class.



### Class list

7

#### Communication class

Class name	Overview	Reference
Communication	This class processes the communication between the server personal computer and High Speed Data Communication Module. Performs the connection to High Speed Data Communication Module, the streaming transfer, and the data read/write function.	Page 101 Communication class

#### Notification class

Class name	Overview	Reference
Notification	This class notifies data transfers and status changes from High Speed Data Communication Module to user programs. Inherit this class to the class executing the streaming transfer receive processing. The receive processing for streaming-transferred data and module status changes are implemented.	Page 123 Notification class

## Data object class

Class name	Overview	Reference
LabelGroupList	This class stores information of the label group list. Stores information of the label group acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 125 LabelGroupList class
LabelGroup	This class stores information of the label group. Stores information of the label acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 126 LabelGroup class
DataLabel	This class stores label names, output data types, and data sizes. Stores information of the label acquired from the High Speed Data Communication Module using the label list acquisition function.	Page 128 DataLabel class
RegisteredLabelName	This class stores registration target label names and data read/write label names. Stores label group names and label names used for the streaming transfer function and the data read/write function.	Page 130 RegisteredLabelName class
ReceiveData	This class stores the data list sent by the streaming transfer.	Page 132 ReceiveData class
RecordData	This class stores data used for the streaming transfer function and the data read/write function. Data received from the High Speed Data Communication Module is stored in the streaming transfer function. Values read from the specified labels are stored in the data read function. Values written to the specified labels are stored in the data write function.	Page 133 RecordData class
DataValue	This class store values for the specified data type. Stores data match with the output data type of label registered as the target of streaming transfer when performing the streaming transfer.	Page 137 DataValue class
ConnectionDeviceList	This class stores the list of the connected device information.	Page 147 ConnectionDeviceList class
ConnectionDevice	This class stores the connected device information.	Page 148 ConnectionDevice class
ConnectionDeviceDetail	This class stores the details of the connected device information.	Page 151 ConnectionDeviceDetail class

## Exception class

Class name	Overview	Reference
DataCommunicationException	This class stores exceptions of listed enumerators in the methods for the communication class and the data object class. The user program implements the exception handler.	Page 154 DataCommunicationException class
DataCommunicationError	This class defines the detailed information at the time of error occurrence in the methods for the communication class and the data object class.	Page 155 DataCommunicationError enumerator

# Communication class

Item	Content
Class syntax	public class Communication extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatabcommunication.Communication

## Constructor

Format	Content
Communication()	Default constructor

## Details of the constructor

### ■Communication

Item		Content
Function		Constructor
Call format		public Communication()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to communicate between the server personal computer and the High Speed Data Communication Module.

## Method list

Type	Method name	Overview	Reference
void	addListener( Notification listener)	Registers a listener to receive streaming transfer notifications and module status changes to this Communication class.	Page 103 addListener
void	connect( java.lang.String ipAddress)	Opens a communication line to the High Speed Data Communication Module, and establishes a connection.	Page 103 connect
void	connect( java.lang.String ipAddress, java.lang.String userName, java.lang.String password)	Opens a communication line to the High Speed Data Communication Module in which the access authentication is enabled, and establishes a connection.	Page 104 connect
void	disconnect()	Closes the communication line to the High Speed Data Communication Module, and disconnects the connection.	Page 105 disconnect
byte	getAdditionTimeType()	Acquires the type of time appended to each transfer record from this Communication class.	Page 105 getAdditionTimeType
int	getBufferingTimeOut()	Acquires the timeout period of the buffering transfer data function from this Communication class.	Page 105 getBufferingTimeOut
byte	getBufMode()	Acquires the buffering transfer data mode from this Communication class.	Page 105 getBufMode
short	getConnectionDeviceList( ConnectionDeviceList connectionDeviceList)	Acquires the list of status for devices connected with High Speed Data Communication Module.	Page 106 getConnectionDeviceList
short	getLabelList( LabelGroupList labelGroupList)	Acquires the label list set for the High Speed Data Communication Module.	Page 107 getLabelList
short	getLabelListWithComment( LabelGroupList labelGroupList)	Acquires the label list, including comments, set on the High Speed Data Communication Module.	Page 108 getLabelListWithComment
short	getTimeOut()	Acquires the timeout period from this Communication class.	Page 108 getTimeOut
short	readDataByDevice( java.lang.String[] deviceList, short[] deviceValue)	Reads data by specifying the device name from the programmable controller CPU to the server personal computer.	Page 109 readDataByDevice

Type	Method name	Overview	Reference
short	readDataByLabel( RegisteredLabelName[] readLabelList, RecordData readDeviceList)	Reads data by specifying the label name from the programmable controller CPU to the server personal computer.	Page 110 readDataByLabel
void	setAdditionTimeType( byte type)	Stores the type of time appended to each transfer record to this Communication class.	Page 110 setAdditionTimeType
void	setBufferingTimeOut( int timeOut)	Stores the timeout period of the buffering transfer data function to this Communication class.	Page 111 setBufferingTimeOut
void	setBufMode( byte bufMode)	Stores the buffering transfer data mode at the streaming transfer to this Communication class.	Page 111 setBufMode
short	setStreamingLabel( RegisteredLabelName[] labelList)	Registers the data list streamed from the programmable controller CPU to the server personal computer by specifying the label name to the High Speed Data Communication Module.	Page 112 setStreamingLabel
short	setStreamingLabelGroup( java.lang.String[] labelGroupList)	Registers the data list streamed from the programmable controller CPU to the server personal computer by specifying the label name to the High Speed Data Communication Module.	Page 113 setStreamingLabelGroup
void	setTimeout( short timeOut)	Stores the timeout period to this Communication class. Specifies the response timeout period for communication.	Page 114 setTimeout
short	startStreamingTransfer( int samplingTime)* <sup>1</sup>	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle.	Page 115 startStreamingTransfer
short	startStreamingTransfer( int samplingTime, short transferCount)* <sup>1</sup>	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle and the number of transfer records	Page 116 startStreamingTransfer
short	startStreamingTransferByInterval( int samplingTime, int transferTime)* <sup>1</sup>	Starts the streaming transfer of data registered on the High Speed Data Communication Module. Specify the sampling cycle and the transfer time.	Page 117 startStreamingTransferByInterval
short	stopStreamingDataBuffering( short connectNum)	Discards the transfer data buffered with the buffering transfer data function, and disconnects the line.	Page 118 stopStreamingDataBuffering
short	stopStreamingTransfer()	Stops the streaming transfer.	Page 119 stopStreamingTransfer
short	writeDataByDevice( java.lang.String[] deviceList, short[] deviceValue)	Writes data by specifying the device name from the server personal computer to the programmable controller CPU.	Page 120 writeDataByDevice
short	writeDataByLabel( RegisteredLabelName[] writeLabelList, RecordData writeDeviceList)	Writes data by specifying the label name from the server personal computer to the programmable controller CPU.	Page 121 writeDataByLabel


\*1 A 06B0H error occurs when any of the following Communication class methods is executed for a single instance after executing the startStreamingTransfer/startStreamingTransferByInterval method. Execute the following methods after executing the stopStreamingTransfer method.

- getConnectionDeviceList
- getLabelList
- getLabelListWithComment
- setStreamingLabel
- setStreamingLabelGroup
- startStreamingTransferByInterval
- startStreamingTransfer
- stopStreamingDataBuffering
- readDataByLabel
- readDataByDevice
- writeDataByLabel
- writeDataByDevice




## Method details

### ■addListener

Item		Content
Function		Adds a listener to the user program.
Call format		public void addListener(Notification listener) throws DataCommunicationException
Argument	Type name	Notification
	Variable name	listener
	Content	[IN/OUT] Registration destination for listener
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • ARGUMENT_NULL For details, refer to the following section.  Page 154 getErrorCode
Comment		Registers a listener to receive streaming transfer data and to notify module status changes to this Communication class.*1


\*1 Execute this method before executing the connect method.

### ■connect

Item		Content
Function		Opens a communication line to a High Speed Data Communication Module whose access authentication is enabled, and establishes a connection.
Call format		public void connect(java.lang.String ipAddress) throws DataCommunicationException
Argument	Type name	java.lang.String
	Variable name	ipAddress
	Content	[IN] IP address for the High Speed Data Communication Module
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • ARGUMENT_NULL • AUTHENTICATION_FAILED • BUFFERING_CONNECT_OVER • CONNECT_OVER • CONNECT_REFUSED • DISCONNECTED • DISPOSED_ERROR • EXECUTING • INTERRUPTED_ERROR • INVALID_CONNECT • INVALID_IPADDRESS • INVALID_REQUEST • INVALID_RESPONSE • INVALID_STREAMING • IO_ERROR • IPADDRESS_NULL • NOT_PERMITTED_IPADDRESS • OUT_RANGE_ADDITIONAL_TIMETYPE • OUT_RANGE_TIMEOUT • RECONNECT • STREAMING_CONNECT_OVER • TIMEOUT For details, refer to the following section.  Page 154 getErrorCode
Comment		Opens a communication line to the High Speed Data Communication Module.*1 When using multiple lines to communicate with a single High Speed Data Communication Module, create communication class instances for each communication and execute the connect method.

\*1 To receive the streaming transfer data and module status changes, execute the addListener method before executing the connect method.


## ■connect

Item		Content		
Function		Opens a communication line to the High Speed Data Communication Module, and establishes a connection.		
Call format		public void connect(java.lang.String ipAddress, java.lang.String userName, java.lang.String password) throws DataCommunicationException		
Argument	Type name	java.lang.String	java.lang.String	java.lang.String
	Variable name	ipAddress	userName	password
	Content	[IN] IP address for the target device	[IN] Connection user name (up to 20 characters) <sup>*1</sup>	[IN] Password for the connection user name (up to 16 characters) <sup>*2</sup>
Return value		None		
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ARGUMENT_NULL</li> <li>• AUTHENTICATION_FAILED</li> <li>• BUFFERING_CONNECT_OVER</li> <li>• CONNECT_OVER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• INTERRUPTED_ERROR</li> <li>• INVALID_CONNECT</li> <li>• INVALID_IPADDRESS</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• INVALID_STREAMING</li> <li>• IO_ERROR</li> <li>• IPADDRESS_NULL</li> <li>• NOT_PERMITTED_IPADDRESS</li> <li>• OUT_RANGE_ADDITIONAL_TIMETYPE</li> <li>• OUT_RANGE_PASSWORD</li> <li>• OUT_RANGE_TIMEOUT</li> <li>• OUT_RANGE_USER_NAME</li> <li>• PASSWORD_NULL</li> <li>• RECONNECT</li> <li>• STREAMING_CONNECT_OVER</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode		
Comment		Opens a communication line to the High Speed Data Communication Module in which the account authentication is enabled. <sup>*2</sup> When using multiple lines to communicate with a single High Speed Data Communication Module, create communication class instances for each communication and execute the connect method.		

\*1 To receive the streaming transfer data and module status changes, execute the addListener method before executing the connect method.

\*2 When a High Speed Data Communication Module on which the account authentication is disabled is connected, the specified user name and the password are ignored.

## ■disconnect

Item		Content
Function		Disconnects the connection to the High Speed Data Communication Module.
Call format		public void disconnect() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • NOT_CONNECTED_YET For details, refer to the following section.  Page 154 getErrorCode
Comment		Closes the established communication line, and disconnects the connection.

## ■getAdditionTimeType

Item		Content
Function		Acquires the type of time appended to each transfer record.
Call format		public byte getAdditionTimeType()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the time type. 0: local time 1: UTC
Comment		Acquires the type of time appended to each transfer record from this Communication class. "1: UTC" is stored to the return value by default.



## ■getBufferingTimeOut

Item		Content
Function		Acquires the timeout period of the buffering transfer data function.
Call format		public int getBufferingTimeOut()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Timeout period of the buffering transfer data function: 0.60 to 86400 seconds
Comment		Acquires the timeout period of the buffering transfer data function from this Communication class. "0" is stored to the return value by default.



## ■getBufMode

Item		Content
Function		Acquires the buffering transfer data mode of the streaming transfer.
Call format		public void getBufMode()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Buffering transfer data mode Invalidity (without buffering transfer data): Other than 1 Validity (with buffering transfer data): 1
Comment		Acquires the buffering transfer data mode from this Communication class. "0: invalidity (without buffering transfer data)" is stored to the return value by default.

## ■getConnectionDeviceList

Item		Content
Function		Acquires the list of status for devices connected with High Speed Data Communication Module.
Call format		public short getConnectionDeviceList(ConnectionDeviceList connectionDeviceList) throws DataCommunicationException
Argument	Type name	ConnectionDeviceList
	Variable name	connectionDeviceList
	Content	[IN/OUT] List of status for devices connected with High Speed Data Communication Module
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the list of status for devices connected with High Speed Data Communication Module. Use this method to check the connections when executing the stopStreamingDataBuffering method.

## ■getLabelList



Item		Content
Function		Acquires the label list set on the High Speed Data Communication Module.
Call format		public short getLabelList(LabelGroupList labelGroupList) throws DataCommunicationException
Argument	Type name	LabelGroupList
	Variable name	labelGroupList
	Content	[IN/OUT] Stores the label list set on the High Speed Data Communication Module.*1
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXCHANGE_TO_STRING_FAILED</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the label list set on the High Speed Data Communication Module.*2,*3 Labels of label groups which are accessible with the account used for the connection can be acquired.

\*1 "NULL" is stored to the label group comments and label comments in the acquired label list.

\*2 When settings are not written to the High Speed Data Communication Module, or when label groups which are accessible with the account used for the connection do not exist, the number of label groups stored in labelGroupList is set to 0.  
When the number of label groups is 0 and the array element is accessed, the exception `ArrayIndexOutOfBoundsException` occurs.  
Therefore, access the array element after checking the acquired number of label groups is not 0.

\*3 When a label group without labels is set, the number of labels in the label group without labels is set to 0 among the label groups stored in labelGroupList.  
When the number of labels is 0 and the array element is accessed, the exception `ArrayIndexOutOfBoundsException` occurs.  
Therefore, access the array element after checking the acquired number of labels is not 0.

## ■getLabelListWithComment



Item		Content
Function		Acquires the label list, including comments, set on the High Speed Data Communication Module.
Call format		public short getLabelListWithComment(LabelGroupList labelGroupList) throws DataCommunicationException
Argument	Type name	LabelGroupList
	Variable name	labelGroupList
	Content	[IN/OUT] Stores the label list set on the High Speed Data Communication Module.
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXCHANGE_TO_STRING_FAILED</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the label list, including comments, set on the High Speed Data Communication Module. <sup>*1,*2</sup> Labels of label groups which are accessible with the account used for the connection can be acquired.

- \*1 When settings are not written to the High Speed Data Communication Module, or when label groups which are accessible with the account used for the connection do not exist, the number of label groups stored in labelGroupList is set to 0.  
When the number of label groups is 0 and the array element is accessed, the exception ArrayIndexOutOfBoundsException occurs.  
Therefore, access the array element after checking the acquired number of label groups is not 0.
- \*2 When a label group without labels is set, the number of labels in the label group without labels is set to 0 among the label groups stored in labelGroupList.  
When the number of labels is 0 and the array element is accessed, the exception ArrayIndexOutOfBoundsException occurs.  
Therefore, access the array element after checking the acquired number of labels is not 0.

## ■getTimeout


Item		Content
Function		Acquires the timeout period for processing.
Call format		public short getTimeout()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Timeout period: returns 1 to 255 (In seconds).
Comment		Acquires the response timeout period for the connected processing request from this Communication class. "20" is stored to the return value by default.

## ■readDataByDevice

Item		Content	
Function		Reads device values of the specified devices from the programmable controller CPU to the server personal computer.* <sup>1</sup>	
Call format		public short readDataByDevice(java.lang.String[] deviceList, short[] deviceValue) throws DataCommunicationException	
Argument	Type name	java.lang.String[]	short[]
	Variable name	deviceList	deviceValue
	Content	[IN] List of device names to be read* <sup>2</sup> (up to 32 characters for each device name)	[IN/OUT] Storage destination for read device value
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_DEVICE_COUNT</li> <li>• OUT_RANGE_DEVICE_NAME_LENGTH</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Reads device values stored in the specified devices from the programmable controller CPU.* <sup>3</sup> * <sup>4</sup> * <sup>5</sup>	

\*<sup>1</sup> An exception occurs if the number of words for the labels to be read exceeds 65536 words.



\*<sup>2</sup> K5 to K8 cannot be specified for a device name. When specifying K5 to K8, combine and specify the digit specifications for K1 to K4.  
(Example) For K5M0  
Specify as K4M0 and K1M16.

\*<sup>3</sup> If a word device or the digit specification of device is specified, one-word device value is acquired. If a bit device or the bit specification of device is specified, 0 or 1 is acquired for the corresponding bit.  
For accessible devices, refer to the following section.  
 High Speed Data Communication Module User's Manual  
When the consecutive data of 2 words or more are acquired, read the device values by specifying multiple devices.  
(Example) When a two-word device value is acquired from D0  
Acquire the device value specifying D0 and D1, and process the acquired device value to be two-word data.

\*<sup>4</sup> An error occurs when the account authentication function is enabled and if the read authority for device specification is not applicable with the connected account.

\*<sup>5</sup> Device values other than those of the control CPU of the High Speed Data Communication Module cannot be read. When reading device values from programmable controller CPUs other than the control CPU of the High Speed Data Communication Module, specify the labels.

## ■readDataByLabel

Item		Content	
Function		Reads device values of the specified labels from the programmable controller CPU to the server personal computer.*1	
Call format		public short readDataByLabel(RegisteredLabelName[] readLabelList, RecordData readDeviceList) throws DataCommunicationException	
Argument	Type name	RegisteredLabelName[]	RecordData
	Variable name	readLabelList	readDeviceList
	Content	[IN] Label list to be read	[IN/OUT] Storage destination for read data
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Reads the device value from the programmable controller CPU to the server personal computer by specifying the labels set on the High Speed Data Communication Module.*2,*3,*4	


\*1 An exception occurs if the number of words for the labels to be read exceeds 65536 words.

\*2 The combination of general sampling labels and high speed sampling labels can be specified.

\*3 An exception occurs if the number of labels to be read exceeds 65536.

\*4 An error occurs when the account authentication function is enabled and if a label group without the read authority is specified with the connected account.


## ■setAdditionTimeType

Item		Content	
Function		Stores the type of time appended to each transfer record.	
Call format		public void setAdditionTimeType(byte type) throws DataCommunicationException	
Argument	Type name	byte	
	Variable name	type	
	Content	[IN] Type of time information to be appended 0: local time 1: UTC	
Return value		None	
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• MINUS_ARGUMENT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Stores the type of time appended to the transfer record to this Communication class.*1 If a value other than 0 or 1 is specified to type, the connect method fails.	

\*1 The types of time appended to each transfer record are determined when the connect method is executed.  
The type of time received from the module is not changed even if this method is executed after executing the connect method.  
To change the setting, execute the disconnect method, execute this method, and then execute the connect method again.




## ■setBufferingTimeOut

Item		Content
Function		Specify the timeout period of the buffering transfer data function.* <sup>1</sup>
Call format		public void setBufferingTimeOut(int timeOut) throws DataCommunicationException
Argument	Type name	int
	Variable name	timeOut
	Content	[IN] Timeout period of the buffering transfer data function 0.60 to 86400 seconds
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_TIMEOUT For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the timeout period for the execution of the buffering transfer data function to this Communication class. Specify the timeout period of the buffering transfer data function. When using the buffering transfer data function, specify a timeout period to stop the buffering transfer data processing of the High Speed Data Communication Module in a specified period. An exception occurs if a value out of the range of 60 to 86400 is stored. When 0 is specified, the buffering transfer data function continues its operation until the network recovery without the timeout.



- \*<sup>1</sup> The timeout for the buffering transfer data function is determined when the startStreamingTransfer/startStreamingTransferByInterval method is executed.  
The timeout period of the buffering transfer data function is not changed even if this method is executed after executing the startStreamingTransfer/startStreamingTransferByInterval method.  
To change the setting, execute the stopStreamingTransfer method, execute this method, and then execute the startStreamingTransfer/startStreamingTransferByInterval method again.

## ■setBufMode

Item		Content
Function		Specify the transfer buffering mode of the streaming transfer.
Call format		public void setBufMode(byte bufMode) throws DataCommunicationException
Argument	Type name	byte
	Variable name	bufMode
	Content	[IN] Buffering transfer data mode Invalidity (without buffering transfer data): Other than 1 Validity (with buffering transfer data): 1
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • MINUS_ARGUMENT For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the buffering transfer data mode of the streaming transfer to this Communication class. Specify "1" to validate the transfer buffering function of High Speed Data Communication Module.* <sup>1</sup>



- \*<sup>1</sup> the buffering transfer data mode is determined when the connect method is executed.  
the buffering transfer data function is not validated even if this method is executed after executing the connect method.  
To validate the buffering transfer data function, execute the disconnect method, execute this method, and then execute the connect method again.

## ■setStreamingLabel

Item		Content
Function		Registers the target labels for the streaming transfer on the High Speed Data Communication Module.
Call format		public short setStreamingLabel(RegisteredLabelName[] labelList) throws DataCommunicationException
Argument	Type name	RegisteredLabelName[]
	Variable name	labelList
	Content	[IN] Target label list for streaming transfer* <sup>1</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Specify the labels to be streamed to the server personal computer.* <sup>2</sup> ,* <sup>3</sup> ,* <sup>4</sup> ,* <sup>5</sup>

- \*<sup>1</sup> When the account authentication function is enabled, an error occurs if a registration of a label group without the read authority is attempted with the connected account.
- \*<sup>2</sup> The combination of general sampling labels and high speed sampling labels cannot be specified in a single registration.
- \*<sup>3</sup> Overwrites data if the streaming transfer data are already registered.
- \*<sup>4</sup> The total data size for labels that can be registered as the target of streaming transfer is 65536 words.
- \*<sup>5</sup> The number of labels that can be registered as the target of streaming transfer is 65536.

## ■setStreamingLabelGroup

Item		Content
Function		Registers the target labelgroups for the streaming transfer on the High Speed Data Communication Module.
Call format		public short setStreamingLabelGroup(java.lang.String[] labelGroupList) throws DataCommunicationException
Argument	Type name	java.lang.String[]
	Variable name	labelGroupList
	Content	[IN] Target labelgroup list for streaming transfer* <sup>1</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Specify the label groups to be streamed to the server personal computer. By specifying the label groups, all labels of the specified label groups are registered. * <sup>2</sup> ,* <sup>3</sup> ,* <sup>4</sup> ,* <sup>5</sup>

\*<sup>1</sup> When the account authentication function is enabled, an error occurs if a registration of a label group without the read authority is attempted with the connected account.


\*<sup>2</sup> The combination of general sampling labels and high speed sampling labels cannot be specified in a single registration.

\*<sup>3</sup> Overwrites data if the streaming transfer data are already registered.

\*<sup>4</sup> The total data size for labels of label groups that can be registered as the target of streaming transfer is 65536 words.

\*<sup>5</sup> The number of labels of the label groups that can be registered as the target of streaming transfer is 65536.

## ■setTimeout

Item		Content
Function		Specifies the response timeout period for communication.*1,*2
Call format		public void setTimeout(short timeout) throws DataCommunicationException
Argument	Type name	short
	Variable name	timeout
	Content	[IN] Timeout period 1 to 255 seconds can be specified. (In seconds)
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_TIMEOUT For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the response timeout period for communication to this Communication class. Specify the response timeout period of the communication request for the High Speed Data Communication Module.*3,*4,*5

\*1 In some Communication class methods, data are divided to process the communication.

Therefore, a timeout may not occur with these methods even when the response timeout period specified in the setTimeout method elapses.

\*2 A longer communication time may be required for the on-demand function when communicating to CPUs on other stations other than the control CPU of the High Speed Data Communication Module.

Execute the on-demand function after performing the following measures when a timeout occurs in the on-demand function.

- Check the module status and the network connection status of the High Speed Data Communication Module.
- Check the network connection status of CPUs on other stations.
- Set a longer timeout period for the setTimeout method.
- Reduce the number of CPUs on other stations that are accessed simultaneously.
- Reduce the number of device points that are accessed simultaneously.

\*3 If the network is disconnected after the connection, a notification is sent by the receiveCommunicationEvent method of the Notification class about 10 seconds after the disconnection.



\*4 The response timeout period for communication is determined when the connect method is executed.

The response timeout period for communication is not changed even if this method is executed after executing the connect method.

To change the setting, execute the disconnect method once, execute this method, and then execute the connect method again.

\*5 An exception may occur regardless of the timeout period that is set during the connection when the network cable is not connected to the server personal computer.

## ■startStreamingTransfer

Item		Content
Function		Starts the streaming transfer of the data registered on the High Speed Data Communication Module.
Call format		public short startStreamingTransfer(int samplingTime) throws DataCommunicationException
Argument	Type name	int
	Variable name	samplingTime
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3,*4</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• MINUS_ARGUMENT</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by synchronizing the transfer cycle with the sampling cycle. <sup>*5</sup> Use this method when executing the streaming transfer by specifying general sampling labels. <sup>*6,*7,*8</sup>

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down. When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 An error occurs if a value larger than 32,767,000 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to general sampling.

\*5 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.


\*6 The value as the same as the sampling cycle is processed for the transfer cycle, and 1 is processed for the number of transfer records.

\*7 The streaming transfer function is a best effort delivery.

This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices.

Run the system by fully verifying the processing time of each function when constructing it.



For the processing time, refer to the following section.

 High Speed Data Communication Module User's Manual

\*8 The transfer data may be received by receiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (disconnect), expand the timeout cycle with setTimeout method, reconnect the line (connect), and start streaming transfer (startStreamingTransferByInterval).

## ■startStreamingTransfer

Item		Content	
Function		Specify a sampling cycle and a number of transfer record, and start streaming transfer of the data registered to a High Speed Data Communication Module.	
Call format		public short startStreamingTransfer(int samplingTime, short transferCount) throws DataCommunicationException	
Argument	Type name	int	short
	Variable name	samplingTime	transferCount
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3,*4</sup>	[IN] Transfer number of records Specify the number of records to execute the data transfer with the specified number of sampling execution. High speed sampling: 1 to 100 <sup>*5</sup> General sampling: fixed to 1 <sup>*6</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• MINUS_ARGUMENT</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by synchronizing the transfer cycle with the sampling cycle. <sup>*7</sup> Data are transferred when the specified number of transfer records is sampled. <sup>*8,*9</sup>	

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767,000 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down.  
When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 An error occurs if a value larger than 32,767,000 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to general sampling.

\*5 An error occurs if the specified number of transfer records exceeds the range when the label group registered as the target for streaming transfer is set to high speed sampling.

\*6 Any value other than 1 cannot be specified for the number of transfer records when the label group registered as the target for streaming transfer is set to general sampling. An error occurs if any value other than 1 is specified.

\*7 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.

\*8 The streaming transfer function is a best effort delivery.



This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices.  
Run the system by fully verifying the processing time of each function when constructing it.  
For the processing time, refer to the following section.

 High Speed Data Communication Module User's Manual

\*9 The transfer data may be received by receiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (disconnect), expand the timeout cycle with setTimeout method, reconnect the line (connect), and start streaming transfer (startStreamingTransferByInterval).

## ■startStreamingTransferByInterval

Item		Content	
Function		Specify the sampling cycle and transfer cycle, and start the streaming transfer of data registered to a High Speed Data Communication Module.	
Call format		public short startStreamingTransferByInterval(int samplingTime, int transferTime) throws DataCommunicationException	
Argument	Type name	int	int
	Variable name	samplingTime	transferTime
	Content	[IN] Sampling time (milliseconds) High speed sampling: 0, 1 to 32,767 ms <sup>*1,*2</sup> General sampling: 100 to 32,767,000 ms <sup>*3</sup>	[IN] Transfer cycle (milliseconds) High speed sampling: 0.1 to 100 ms <sup>*4,*5</sup> General sampling: fixed to 0 <sup>*6</sup>
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• MINUS_ARGUMENT</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_TRANSFER_TIME</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Starts the streaming transfer of data registered on the High Speed Data Communication Module by specifying the sampling cycle and the transfer cycle. <sup>*7,*8,*9</sup>	

\*1 Data can be sampled in each sequence scan by specifying 0 for the sampling cycle.

\*2 An error occurs if a value larger than 32,767 ms is specified for the sampling cycle when the label group registered as the target for streaming transfer is set to high speed sampling.

\*3 The sampling time can be specified in units of 100 ms. The milliseconds in tens place and ones place are rounded down. When a value less than 100 ms is specified for a sampling cycle, data are sampled in 100 ms cycle.

\*4 By specifying 0 for the transfer cycle, data can be transferred in synchronization with the sampling cycle.


\*5 A transfer cycle shorter than the sampling cycle cannot be specified.

\*6 The transfer cycle cannot be specified when the label group registered as the target for streaming transfer is set to general sampling. An error occurs if any value other than 0 is specified.

\*7 An error occurs if three or more streaming transfers to which a high speed sampling label group is specified are performed.

\*8 The streaming transfer function is a best effort delivery.

This function may not operate with the specified data sampling cycle or transfer cycle because the module processing time changes according to the data settings, parameters specified by High Speed Data Communication Library, and status of other devices. Run the system by fully verifying the processing time of each function when constructing it.  
For the processing time, refer to the following section.

 High Speed Data Communication Module User's Manual

\*9 The transfer data may be received by receiveTransferData when exception of TIMEOUT occurred.

In that case, disconnect the line (disconnect), expand the timeout cycle with setTimeout method, reconnect the line (connect), and start streaming transfer (startStreamingTransferByInterval).

When the label group registered as the target of streaming transfer is set to high speed sampling, the following combinations of sampling cycle and the transfer cycle can be specified.

Transfer cycle (milliseconds)	Sampling time (milliseconds)							
	0	1	2	...	100	...	32767	32768 or higher
0	○	○	○	○	○	○	○	×
1	○	○	×	×	×	×	×	×
2	○	○	○	×	×	×	×	×
...	○	○	○	○/×	×	×	×	×
100	○	○	○	○	○	×	×	×
101 or higher	×	×	×	×	×	×	×	×

○: Applicable, ×: Error



\*1 An error occurs when a value smaller than the sampling cycle is specified for the transfer cycle.

(Example) Sampling time: 40, Transfer cycle: 30 → Error

Sampling time: 40, Transfer cycle: 40 → Applicable

Sampling time: 30, Transfer cycle: 40 → Applicable



## ■stopStreamingDataBuffering

Item		Content
Function		Discards the buffering transfer data and disconnects the line.
Call format		public short stopStreamingDataBuffering(short connectNum) throws DataCommunicationException
Argument	Type name	short
	Variable name	connectNum
	Content	[IN] Connection number to be disconnected Access status 1 to 5 is specified: 1 to 5
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• OUT_RANGE_CONNECTION_NO</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		When the access status of the specified connection number is buffering the transfer data, the buffered transfer data are discarded and the line is disconnected.*1 An error occurs when the access status of the specified connection number is not buffering the transfer data. The connection numbers correspond to the access status 1 to 5 in the access status area of the buffer memory. (When the getConnectionDeviceList method is executed, the array elements 0 to 4 in the property ConnectionDeviceArray[] of argument connectionDeviceList are synonymous with the access status 1 to 5.) An exception OUT_RANGE_CONNECTION_NO is returned if a value other than 1 to 5 is specified.



\*1 An error occurs when the account authentication function is enabled and the administrative right does not apply to the connected account.



## ■stopStreamingTransfer

Item		Content
Function		Stops the streaming transfer to the server personal computer.
Call format		public short stopStreamingTransfer() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Stops the streaming transfer to the server personal computer. Stop the re-transfer start function which operates on a user program after detected the disconnection by executing at the buffering transfer data function.

## ■writeDataByDevice

Item		Content	
Function		Writes device values of the specified devices from the server personal computer to the programmable controller CPU.*1	
Call format		public short writeDataByDevice(java.lang.String[] deviceList, short[] deviceValue) throws DataCommunicationException	
Argument	Type name	java.lang.String[]	short[]
	Variable name	deviceList	deviceValue
	Content	[IN] List of device names to be written*2 (up to 32 characters for each device name)	[IN] Storage destination for write device value
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_DEVICE_COUNT</li> <li>• OUT_RANGE_DEVICE_NAME_LENGTH</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Writes device values from the server personal computer to the devices specified on the programmable controller CPU.*3,*4,*5	

\*1 An exception occurs if the number of words for the devices to be written exceeds 65536 words.

\*2 K5 to K8 cannot be specified for a device name. When specifying K5 to K8, combine and specify the digit specifications for K1 to K4.  
(Example) For K5M0  
Specify as K4M0 and K1M16.

\*3 If a word device or the digit specification of the device is specified, one-word device value is stored. If a bit device or the bit specification of device is specified, the value for the writing device value's least significant bit is stored.  
For accessible devices, refer to the following section.

 High Speed Data Communication Module User's Manual

When the consecutive data of 2 words or more are acquired, write the device values by specifying multiple devices.



(Example) When a two-word device value is stored from D0

Write the device values by specifying D0 and D1.

\*4 An error occurs when the account authentication function is enabled and if the read authority for device specification is not applicable with the connected account, or if the read authority is applicable but the write authority is not applicable.

\*5 Device values other than those of the control CPU of the High Speed Data Communication Module cannot be written. When writing device values from programmable controller CPUs other than the control CPU of the High Speed Data Communication Module, specify the label.

## ■writeDataByLabel

Item		Content	
Function		Writes device values of the specified labels from the server personal computer to the programmable controller CPU.*1	
Call format		public short writeDataByLabel(RegisteredLabelName[] writeLabelList, RecordData writeDeviceList) throws DataCommunicationException	
Argument	Type name	RegisteredLabelName[]	RecordData
	Variable name	writeLabelList	writeDeviceList
	Content	[IN] Label list to be written	[IN] Storage destination for write data
Return value		Normal end: Returns 0. Error end: Returns other than 0. For error codes, refer to the following section.  High Speed Data Communication Module User's Manual	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• ABORTED_BY_PEER</li> <li>• ARGUMENT_NULL</li> <li>• CONNECT_REFUSED</li> <li>• DISCONNECTED</li> <li>• DISPOSED_ERROR</li> <li>• EXCHANGE_FROM_STRING_FAILED</li> <li>• EXECUTING</li> <li>• GENERAL_ERROR</li> <li>• INVALID_DATASIZE</li> <li>• INVALID_DATATYPE</li> <li>• INVALID_REQUEST</li> <li>• INVALID_RESPONSE</li> <li>• IO_ERROR</li> <li>• NOT_CONNECTED_YET</li> <li>• NOT_SAME_ELEMENT_COUNT</li> <li>• OUT_RANGE_LABEL_COUNT</li> <li>• OUT_RANGE_WORD_SIZE</li> <li>• TIMEOUT</li> </ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Writes device values from the server personal computer to the programmable controller CPU by specifying the labels set on the High Speed Data Communication Module.*2,*3,*4	

\*1 An exception occurs if the number of words for the labels to be written exceeds 65536 words.

\*2 The combination of general sampling labels and high speed sampling labels can be specified.

\*3 An exception occurs if the number of labels to be written exceeds 65536.

\*4 An error occurs when the account authentication function is enabled and if a label group without the read authority or a label group with the read authority but not with the write authority is specified with the connected account.

## Setting data

The following describes the operations when writing data to labels whose data type is string type/raw type.

- When the data size of the set label is an odd number,

A null code (00H) is stored to the upper byte of the last device.

**Ex.**

When writing "ABC" to the label with the start device: "D0", data type: "string type", and data size: "3" (odd number), the data are written as shown below.

The character code to handle a single character as one byte (such as "iso-8859-1") is specified for the string.

D0		D1	
41H	42H	43H	00H
A	B	C	Null code

- When the write data size or the number of elements is smaller than that of the set label.

Null codes (00H) are stored to the devices following the amount of the write data size or the number of elements.

**Ex.**

When writing "ABCD" to the label with the start device: "D0", data type: "string type", and data size: "8", the data are written as shown below.

The character code to handle a single character as one byte (such as "iso-8859-1") is specified for the string.

D0		D1		D2		D3	
41H	42H	43H	44H	00H	00H	00H	00H
A	B	C	D	Null code	Null code	Null code	Null code

- When the write data size or the number of elements is larger than that of the set label.

The exception "INVALID\_DATASIZE" occurs.

Check the data size set to the label and correct the write data size not to be larger than the data size.

Note that the data sizes of the same strings are different depending on the encoding code.

Specify the encoding code for each specific purpose.

**Ex.**

When writing "ABCD" to the label with the start device: "D0", data type: "string type", and size: "8" (odd number), the data are written depending on the encoding code as shown below.

"UTF-16LE" (two-byte single character)

D0		D1		D2		D3	
41H	00H	42H	00H	43H	00H	44H	00H
A	Null code	B	Null code	C	Null code	D	Null code

"iso-8859-1" (one-byte single character)

D0		D1		D2		D3	
41H	42H	43H	44H	00H	00H	00H	00H
A	B	C	D	Null code	Null code	Null code	Null code

# Notification class

Item	Content
Class syntax	public class Notification
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.Notification

## Constructor

Format	Content
Notification()	Default constructor

## Details of the constructor

### ■Notification

Item		Content
Function		Constructor
Call format		public Notification()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to notify data transfer and status changes from High Speed Data Communication Module to user programs. This constructor is automatically called from the constructor of the user created class on which the Notification class is implemented.

## Method list

Type	Method name	Overview	Reference
void	receiveCommunicationEvent( short status, short errorCode)	The method which is called when a status change such as a line disconnection is received.*1	Page 123 receiveCommunicationEvent
void	receiveTransferData( ReceiveData recordList)	The method which is called when the streaming transfer data sent from the High Speed Data Communication Module is received.*1	Page 124 receiveTransferData

\*1 Since the default is no-operation, it is necessary to create a processing to be executed by the override.

## Method details

### ■receiveCommunicationEvent

Item		Content	
Function		Notifies status changes of High Speed Data Communication Module to the server personal computer.	
Call format		public void receiveCommunicationEvent(short status, short errorCode)	
Argument	Type name	short	short
	Variable name	status	errorCode
	Content	[IN/OUT] Notified events**1 Disconnection: 0 Update settings: 3 Stop module operation: 1 Library processing overload: 4 Transfer error stop: 5 Re-transfer start failure: 6 Re-transfer start: 7	[IN] Error code of occurred error (Refer to the error code when an event such as disconnection, transfer error stop, re-transfer start failure, and re-transfer start is notified. "0" is stored to events other than noted.)
Return value		None	
Comment		Notifies status changes such as a line disconnection to the server personal computer. Implement the processing for the status changes in the user program.	

\*1 The following are the actions for the stored events.

Module status	Action
Disconnection: 0	<p>The High Speed Data Communication Module is disconnected, the programmable controller CPU is reset, the power is turned OFF, or the settings are updated.</p> <p>Check the LAN cable connections between the server personal computer and the High Speed Data Communication Module and the system operations.</p> <p>If no problems are found, execute the connect processing again and establish the communication.</p> <p>(If the connect processing is not executed again, the communication may not be established normally.)</p> <p>When the startStreamingTransfer/startStreamingTransferByInterval processing is executed by enabling the buffering mode for the buffering transfer data function, "1" is stored to errorCode at the time of the disconnection event occurrence.</p> <p>When "1" is stored to errorCode, the execution of connect processing again is unnecessary.</p> <p>When an incorrect packet is received and the line is disconnected, "2" is stored to errorCode at the time of disconnection event occurrence.</p> <p>A "0" is stored to errorCode when disconnected the line other than above.</p>
Update settings: 3	<p>The module settings are changed.</p> <p>After executing the disconnect processing and checking the module settings, execute the connect processing again and establish the communication.</p> <p>The disconnection event is notified after the setting update is notified.</p>
Stop module operation: 1	<p>A module stop error occurred.</p> <p>Execute the disconnect processing and check the status of the module.</p> <p>After starting the module in operation, execute the connect processing again and establish the communication.</p>
Library processing overload: 4	<p>The speed to process the streaming transfer data is slow.</p> <p>Execute the disconnect processing, and perform the following actions.</p> <ul style="list-style-type: none"> <li>• Reduce the number of labels registered to the streaming transfer.</li> <li>• Change the output data type and the data size of the labels registered to the streaming transfer, and reduce the number of points.</li> <li>• Increase the sampling time and the transfer cycle.</li> <li>• Reduce the number of transfer records.</li> <li>• Review the processing content for receiveTransferData.</li> </ul> <p>After performing the action, execute the connect processing again and establish the communication.</p> <p>A library processing overload may occur when the server personal computer is overloaded, such as when running Java VM for the first time.</p>
Transfer error stop: 5	<p>The streaming transfer is stopped.</p> <p>Execute the disconnect processing, and perform the following actions.</p> <ul style="list-style-type: none"> <li>• Check the value of errorCode, and perform the actions provided in the error code list.</li> <li>• Check the status of the network accessing another station/other stations programmable controller CPU.</li> <li>• Check if the parameters of the programmable controller CPU from which data are sampled are not changed.</li> </ul> <p>After performing the action, execute the connect processing again and establish the communication.</p>
Re-transfer start failure: 6	<p>Restarting of the streaming transfer failed after the network connection between the server personal computer and the High Speed Data Communication Module is recovered.</p> <p>The cause of the restarting failure is stored to errorCode.</p> <p>The following are the values to be stored.</p> <p>0: Cancelling request</p> <p>1: Timeout occurrence</p> <p>2: Error occurrence (Occurs when data buffering is not executed in High Speed Data Communication Module after the operation to recover the network such as resetting the CPU module or stopping buffering transfer data from the High Speed Data Communication Module.)</p> <p>Check the LAN cable connections between the server personal computer and the High Speed Data Communication Module and the system operations.</p> <p>If no problems are found, execute the connect processing again and establish the communication.</p>
Re-transfer start: 7	<p>The network connection between the server personal computer and the High Speed Data Communication Module is recovered.</p> <p>When an processing overload occurs during the buffering transfer data and the old data are overwritten, or the buffer clear is requested, "1" is stored to errorCode at the time of the re-transfer start event occurrence.</p>

## ■receiveTransferData

Item	Content
Function	Notifies the reception of the streaming transfer data sent from the High Speed Data Communication Module to the server personal computer.
Call format	public void receiveTransferData(ReceiveData recordList)
Argument	Type name
	Variable name
	Content
Return value	None
Comment	<p>Notifies the reception of the streaming transfer data sent from the High Speed Data Communication Module to the server personal computer.</p> <p>Implement the processing for the receive data in the user program.</p>

# LabelGroupList class

Item	Content
Class syntax	public class LabelGroupList extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatacommunication.LabelGroupList

## Constructor

Format	Content
LabelGroupList()	Default constructor

## Details of the constructor

### ■LabelGroupList

Item		Content
Function		Constructor
Call format		public LabelGroupList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the label group list. The label group list maintained previously is initialized.

7

## Method list

Type	Method name	Overview	Reference
LabelGroup[]	getLabelGroupList()	Acquires the label group list from this LabelGroupList class.	Page 125 getLabelGroupList

## Method details

### ■getLabelGroupList

Item		Content
Function		Acquires the label group list.
Call format		public LabelGroup[] getLabelGroupList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the label group list.
Comment		Acquires the label group list from this LabelGroupList class. "null" is stored to the return value by default.

# LabelGroup class

Item	Content
Class syntax	public class LabelGroup extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.LabelGroup

## Constructor

Format	Content
LabelGroup()	Default constructor

## Details of the constructor

### ■LabelGroup

Item		Content
Function		Constructor
Call format		public LabelGroup()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the label group. The label name, the label comment, the output data type, and the data size maintained previously are initialized.

## Method list

Type	Method name	Overview	Reference
java.lang.String	getLabelGroupComment()	Acquires the label group comment from this LabelGroup class.	Page 126 getLabelGroupComment
java.lang.String	getLabelGroupName()	Acquires the label group name from this LabelGroup class.	Page 127 getLabelGroupName
DataLabel[]	getLabelList()	Acquires the label list from this LabelGroup class.	Page 127 getLabelList
byte	getSamplingType()	Acquires the type of label from this LabelGroup class.	Page 127 getSamplingType

## Method details

### ■getLabelGroupComment

Item		Content
Function		Acquires the label group comment.
Call format		public java.lang.String getLabelGroupComment()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label group comment.
Comment		Acquires the label group comment from this LabelGroup class. "null" is stored to the return value by default.



## ■getLabelGroupName

Item		Content
Function		Acquires the label group name.
Call format		public java.lang.String getLabelGroupName()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label group name.
Comment		Acquires the label group name from this LabelGroup class. Store the acquired label group names in the RegisteredLabelName class, and use them for the arguments of the setStreamingLabel, readDataByLabel, writeDataByLabel methods in the Communication class. "null" is stored to the return value by default.

## ■getLabelList

Item		Content
Function		Acquires the label list.
Call format		public DataLabel[] getLabelList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label list.
Comment		Acquires the label list from this LabelGroup class. "null" is stored to the return value by default.

## ■getSamplingType

Item		Content
Function		Acquires the type of label.
Call format		public byte getSamplingType()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored data sampling method.
Comment		Acquires the type of label from this LabelGroup class. High speed sampling label: 0 General sampling label: 1 "1" is stored to the return value by default.

# DataLabel class

Item	Content
Class syntax	public class DataLabel extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.DataLabel

## Constructor

Format	Content
DataLabel()	Default constructor

## Details of the constructor

### ■DataLabel

Item		Content
Function		Constructor
Call format		public DataLabel()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the label. The label name, the label comment, the output data type, and the data size maintained previously are initialized.

## Method list

Type	Method name	Overview	Reference
short	getDataSize()	Acquires the data size from this DataLabel class.	Page 128 getDataSize
short	getDataType()	Acquires the output data type from this DataLabel class.	Page 129 getDataType
java.lang.String	getLabelComment()	Acquires the label comment from this DataLabel class.	Page 129 getLabelComment
java.lang.String	getLabelName()	Acquires the label name from this DataLabel class.	Page 129 getLabelName

## Method details

### ■getDataSize

Item		Content
Function		Acquires the data size.
Call format		public short getDataSize()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored data size.
Comment		Acquires the data size of label from this DataLabel class. The following are the values of data size for each output data type. (Unit: byte) Bit: 1 Word [signed]: 2 Double word [signed]: 4 Word [unsigned]: 2 Double word [unsigned]: 4 Float [single precision]: 4 Float [double precision]: 8 String: Size * <sup>1</sup> Raw: Size * <sup>1</sup> "0" is stored to the return value by default.

\*1 Stores the value of Size specified with Configuration Tool for "String" and "Raw".

## ■getDataType

Item		Content
Function		Acquires the output data type.
Call format		public short getDataType()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored output data type.
Comment		<p>Acquires the output data type of label from this DataLabel class.</p> <p>The following are the values for each output data type.</p> <p>Bit: 1</p> <p>Word [signed]: 10</p> <p>Double word [signed]: 11</p> <p>Word [unsigned]: 12</p> <p>Double word [unsigned]: 13</p> <p>Float [single precision]: 20</p> <p>Float [double precision]: 21</p> <p>String: 30</p> <p>Raw: 40</p> <p>"0" is stored to the return value by default.</p>

## ■getLabelComment

Item		Content
Function		Acquires the label comment.
Call format		public java.lang.String getLabelComment()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label comment.
Comment		<p>Acquires the label comment from this DataLabel class.</p> <p>"null" is stored to the return value by default.</p>

## ■getLabelName

Item		Content
Function		Acquires the label name.
Call format		public java.lang.String getLabelName()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label name.
Comment		<p>Acquires the label name from this DataLabel class.</p> <p>Store the acquired label group names in the RegisteredLabelName class, and use them for the arguments of the setStreamingLabel, readDataByLabel, writeDataByLabel methods in the Communication class.</p> <p>"null" is stored to the return value by default.</p>

# RegisteredLabelName class


Item	Content
Class syntax	public class RegisteredLabelName extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.RegisteredLabelName

## Constructor

Format	Content
RegisteredLabelName(java.lang.String labelGroupName, java.lang.String labelName)	Stores the specified label group name and label name to this RegisteredLabelName class.

## Details of the constructor

### ■RegisteredLabelName

Item		Content	
Function		Constructor	
Call format		public RegisteredLabelName(java.lang.String labelGroupName, java.lang.String labelName) throws DataCommunicationException	
Argument	Type name	java.lang.String	java.lang.String
	Variable name	labelGroupName	labelName
	Content	[IN] Target label group name (up to 32 characters)	[IN] Target label name (up to 32 characters)
Return value		None	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_LABELGROUP_NAME_LENGTH</li><li>• OUT_RANGE_LABEL_NAME_LENGTH</li></ul> For details, refer to the following section.  Page 154 <code>getErrorCode</code>	
Comment		Stores the target label group names and label names for the streaming transfer function, the data read function, and the data write function to this RegisteredLabelName class.	

## Method list

Type	Method name	Overview	Reference
java.lang.String	getLabelGroupName()	Acquires the label group name from this RegisteredLabelName class.	Page 130 getLabelGroupName
java.lang.String	getLabelName()	Acquires the label name from this RegisteredLabelName class.	Page 131 getLabelName
void	setLabelGroupName( java.lang.String labelGroupName)	Stores the label group name to this RegisteredLabelName class.	Page 131 setLabelGroupName
void	setLabelName( java.lang.String labelName)	Stores the label name to this RegisteredLabelName class.	Page 131 setLabelName

## Method details


### ■getLabelGroupName

Item		Content
Function		Acquires the label group name.
Call format		public java.lang.String getLabelGroupName()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label group name.
Comment		Acquires the label group name from this RegisteredLabelName class. "null" is stored to the return value by default.


## ■getLabelName

Item		Content
Function		Acquires the label name.
Call format		public java.lang.String getLabelName()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the stored label name.
Comment		Acquires the label name from this RegisteredLabelName class. "null" is stored to the return value by default.

## ■setLabelGroupName

Item		Content
Function		Stores the label group name.
Call format		public void setLabelGroupName(java.lang.String labelGroupName) throws DataCommunicationException
Argument	Type name	java.lang.String
	Variable name	labelGroupName
	Content	[IN] Label group name (up to 32 characters)
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • ARGUMENT_NULL • OUT_RANGE_LABELGROUP_NAME_LENGTH For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the label group name to this RegisteredLabelName class.

## ■setLabelName

Item		Content
Function		Stores the label name.
Call format		public void setLabelName(java.lang.String labelName) throws DataCommunicationException
Argument	Type name	java.lang.String
	Variable name	labelName
	Content	[IN] Label name (up to 32 characters)
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • ARGUMENT_NULL • OUT_RANGE_LABEL_NAME_LENGTH For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the label name to this RegisteredLabelName class.

## ReceiveData class

Item	Content
Class syntax	public class ReceiveData extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatacommunication.ReceiveData

### Constructor

Format	Content
ReceiveData()	Default constructor

### Details of the constructor

#### ■ReceiveData

Item		Content
Function		Constructor
Call format		public ReceiveData()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the information of the transfer data list. The transfer data list maintained previously is initialized.

### Method list

Type	Method name	Overview	Reference
ReceiveData()	getReceiveDataList()	Acquires the transfer data list from this ReceiveData class.	Page 132 getReceiveDataList

### Method details

#### ■getReceiveDataList

Item		Content
Function		Acquires the transfer data list.
Call format		public RecordData[] getReceiveDataList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the transfer data list.
Comment		Acquires the transfer data list from this ReceiveData class. "null" is stored to the return value by default.

# RecordData class

Item	Content
Class syntax	public class RecordData extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatacommunication.RecordData

## Constructor

Format	Content
RecordData()	Default constructor

## Details of the constructor

### RecordData

Item		Content
Function		Constructor
Call format		public RecordData()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store data used for the streaming transfer function, the data read function, and the data write function. The data list, the time synchronization flag, the data sampling time, and the index information maintained previously are initialized.

7

## Method list

Type	Method name	Overview	Reference
DataValue[]	getDataList()	Acquires the data list from this RecordData.	Page 133 getDataList
long	getIndex()	Acquires the index information at the data sampling from this RecordData class.	Page 134 getIndex
long	getNanoTime()	Acquires the time (less than a second) when data are sampled from this RecordData.	Page 134 getNanoTime
long	getSecTime()	Acquires the time when data are sampled from this RecordData.	Page 134 getSecTime
byte	getTimeSyncFlag()	Acquires the time synchronization execution flag from this RecordData class.	Page 134 getTimeSyncFlag
void	setDataList(DataValue[] dataList)	Stores the data list to this RecordData.	Page 136 setDataList


## Method details

### getDataList

Item		Content
Function		Acquires the data list. *1
Call format		public DataValue[] getDataList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the data list.
Comment		Acquires the data list from this RecordData class. "null" is stored to the return value by default.

- \*1 Store or acquire data according to the function being used.  
When using the streaming transfer function: The streamed data list is stored.  
When using the data read function: The data list read from the programmable controller CPU is stored.  
When using the data write function: The data list to be written to the programmable controller CPU is stored.

## ■getIndex

Item		Content
Function		Acquires the index information.
Call format		public long getIndex()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the index information.
Comment		<p>Acquires the index information at the data sampling from this RecordData class.</p> <p>Outputs a numerical value starting from 1 incremented in ascending order.</p> <p>When the value exceeds the upper limit of 4,294,967,295, it returns to 0 and increments again in the range of 0 to 4,294,967,295.</p> <p>"0" is stored to the return value by default.</p> <p>When a data miss occurs, the index starts again from 1.</p> <p>For details, refer to the following section.</p> <p> High Speed Data Communication Module User's Manual</p>

## ■getNanoTime

Item		Content
Function		Acquires the time (less than a second) when data are sampled.
Call format		public long getNanoTime()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the data sampling time (less than a second).
Comment		<p>Acquires the time when data are sampled from this RecordData class.</p> <p>Acquires the time less than a second from the number of elapsed seconds from January 1, 1970 in nanoseconds.</p> <p>"0" is stored to the return value by default.</p>

## ■getSecTime

Item		Content
Function		Acquires the time when data are sampled.
Call format		public long getSecTime()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the data sampling time.
Comment		<p>Acquires the time when data are sampled from this RecordData class.</p> <p>Acquires the number of elapsed seconds from January 1, 1970.</p> <p>"0" is stored to the return value by default.</p>

## ■getTimeSyncFlag

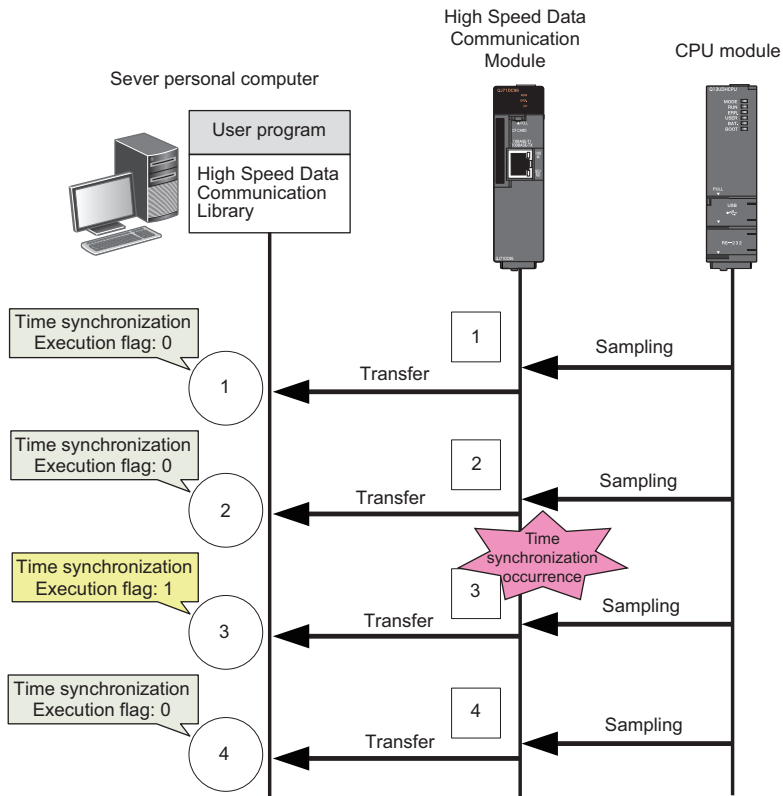
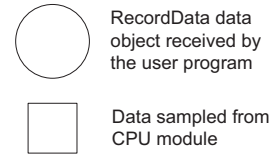
Item		Content
Function		Acquires the time synchronization execution flag.
Call format		public byte getTimeSyncFlag()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the time synchronization execution flag.
Comment		<p>Acquires the time synchronization execution flag from this RecordData class.</p> <p>Time synchronization execution: 1</p> <p>Time synchronization not executed: 0</p> <p>"0" is stored to the return value by default.</p> <p>The time synchronization execution flag becomes "1" only for RecordData transferred after the success of the time query with the time synchronization function of High Speed Data Communication Module.</p> <p>When multiple records are transferred, the time synchronization execution flag becomes "1" only for the RecordData object to which the sampled data are stored after the time synchronization.</p>



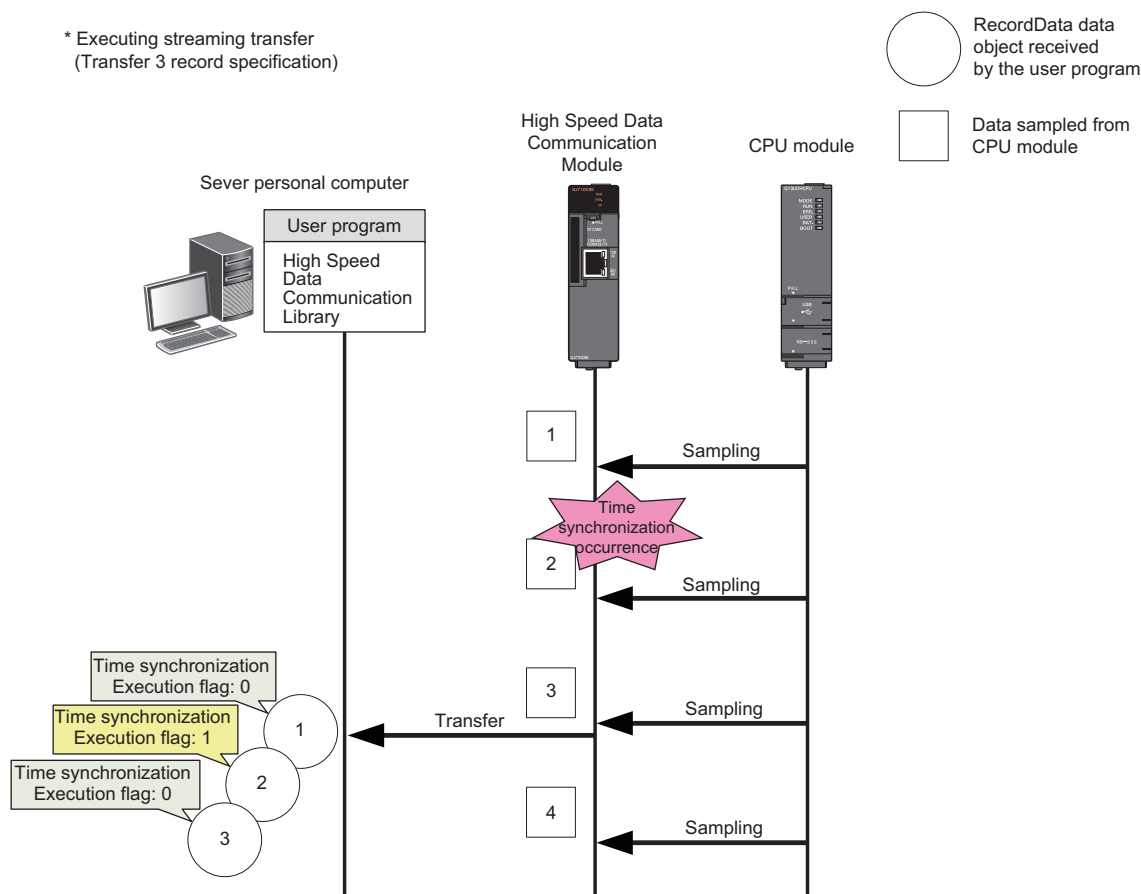
The following is an example when the time synchronization occurs while performing the streaming transfer.

- When time synchronization occurs at the streaming transfer in sampling synchronization

\* Executing streaming transfer  
(Sampling synchronization of transfer)



- When the time synchronization occurs at the streaming transfer in record specification



## ■ setDataList

Item		Content
Function		Stores the data list.*1
Call format		public void setDataList(DataValue[] dataList) throws DataCommunicationException
Argument	Type name	DataValue[]
	Variable name	dataList
	Content	[IN] Label group name
Return value		—
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • ARGUMENT_NULL For details, refer to the following section. ☞ Page 154 getErrorCode
Comment		Stores the data list to this RecordData class.

- \*1 Store or acquire data according to the function being used.  
 When using the streaming transfer function: The streamed data list is stored.  
 When using the data read function: The data list read from the programmable controller CPU is stored.  
 When using the data write function: The data list to be written to the programmable controller CPU is stored.

# DataValue class

Item	Content
Class syntax	public class DataValue extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatacommunication.DataValue

## Constructor

Format	Content
DataValue()	Default constructor

## Details of the constructor

### ■DataValue

Item		Content
Function		Constructor
Call format		public DataValue()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store label values. The data types and values of labels maintained previously are initialized.

## Method list

Type	Method name	Overview	Reference
boolean	getBitData() <sup>*1</sup>	Acquires the bit data from this DataValue class.	Page 139 getBitData
short	getDataType()	Acquires the data type of the stored data from this DataValue class. Bit: 1 Word [signed]: 10 Double word [signed]: 11 Word [unsigned]: 12 Double word [unsigned]: 13 Float [single precision]: 20 Float [double precision]: 21 String: 30 Raw: 40	Page 139 getDataType
double	getDoubleFloatData() <sup>*1</sup>	Acquires the float [double precision] data from this DataValue class.	Page 140 getDoubleFloatData
byte[]	getRawData() <sup>*1</sup>	Acquires the raw data from this DataValue class.	Page 140 getRawData
int	getSignedDoubleWordData() <sup>*1</sup>	Acquires the double word [signed] data from this DataValue class.	Page 141 getSignedDoubleWordData
short	getSignedWordData() <sup>*1</sup>	Acquires the word [signed] data from this DataValue class.	Page 141 getSignedWordData
float	getSingleFloatData() <sup>*1</sup>	Acquires the float [single precision] data from this DataValue class.	Page 141 getSingleFloatData
java.lang.String	getStringData() <sup>*1</sup>	Acquires the character string data from this DataValue class.	Page 142 getStringData
java.lang.String	getStringData( java.lang.String encode) <sup>*1</sup>	Acquires the character string data from this DataValue class by specifying the character code.	Page 142 getStringData
long	getUnsignedDoubleWordData() <sup>*1</sup>	Acquires the double word [unsigned] data from this DataValue class.	Page 143 getUnsignedDoubleWordData
int	getUnsignedWordData() <sup>*1</sup>	Acquires the word [unsigned] data from this DataValue class.	Page 143 getUnsignedWordData
void	setBitData( boolean bitData)	Stores the bit data to this DataValue class.	Page 143 setBitData

Type	Method name	Overview	Reference
void	setDoubleFloatData( double doubleFloatData)	Stores the float [double precision] data to this DataValue class.	Page 144 setDoubleFloatData
void	setRawData( byte[] rawData)	Stores the raw data to this DataValue class.	Page 144 setRawData
void	setSignedDoubleWordData( int signedDoubleWordData)	Stores the double word [signed] data to this DataValue class.	Page 144 setSignedDoubleWordData
void	setSignedWordData( short signedWordData)	Stores the word [signed] data to this DataValue class.	Page 144 setSignedWordData
void	setSingleFloatData( float singleFloatData)	Stores the float [single precision] data to this DataValue class.	Page 145 setSingleFloatData
void	setStringData( java.lang.String stringData)	Stores the character string data to this DataValue class.	Page 145 setStringData
void	setStringData( java.lang.String stringData, java.lang.String encode)	Stores the character string data to this DataValue class by specifying the character code.	Page 145 setStringData
void	setUnsignedDoubleWordData( long unsignedDoubleWordData)	Stores the double word [unsigned] data to this DataValue class.	Page 146 setUnsignedDoubleWordData
void	setUnsignedWordData( int unsignedWordData)	Stores the word [unsigned] data to this DataValue class.	Page 146 setUnsignedWordData


\*1 The get processing other than getDataType of the DataValue class acquires data based on the data type of data stored in the DataValue class.

Therefore, if data is acquired with a data type that differs from the data type of the stored data, a cast conversion is automatically performed.

The data type of the acquired data may differ from the data type of the stored data depending on the data type after the cast conversion. Since the source data information may be lost or a longer processing time may be required due to the cast conversion, execute the get processing matches with the data type to be acquired.

## Method details


### ■getBitData

Item		Content
Function		Acquires the data in bit unit.
Call format		public boolean getBitData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the bit data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_BIT_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in bit unit. Use this property when acquiring bit type data. An exception occurs when the data type stored in the DataValue class is a character string. "false" is stored to the return value by default.


### ■getDataType

Item		Content
Function		Acquires the data type of the stored data.
Call format		public short getDataType()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the data type of the stored data. Bit: 1 Word [signed]: 10 Double word [signed]: 11 Word [unsigned]: 12 Double word [unsigned]: 13 Float [single precision]: 20 Float [double precision]: 21 String: 30 Raw: 40
Comment		Acquires the data type of the data stored in this DataValue class. When multiple data are registered, the data type of the most recently registered method is stored. "0" is stored to the return value by default.


## ■getDoubleFloatData

Item		Content
Function		Acquires the float [double precision] data.
Call format		public double getDoubleFloatData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the float [double precision] data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in float [double precision] unit. Use this property when acquiring float [double precision] data. "0.0" is stored to the return value by default.


## ■getRawData

Item		Content
Function		Acquires the raw data.
Call format		public byte[] getRawData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the raw data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"> <li>• EXCHANGE_TO_NUMBER_FAILED</li> <li>• NOT_DATA_SET</li> </ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in raw unit. Use this property when acquiring raw data. When data acquired from the module are stored, the same amount of data set to the acquired label is acquired. (For example, when the data size is 4, the array element to be stored is 4.) "null" is stored to the return value by default.


## ■getSignedDoubleWordData

Item		Content
Function		Acquires the double word [signed] data.
Call format		public int getSignedDoubleWordData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the double word data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_NUMBER_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in double word unit. Use this property when acquiring double word [signed] data. "0" is stored to the return value by default.


## ■getSignedWordData

Item		Content
Function		Acquires the word [signed] data.
Call format		public short getSignedWordData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the word data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_NUMBER_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in word unit. Use this property when acquiring word [signed] data. "0" is stored to the return value by default.

## ■getSingleFloatData


Item		Content
Function		Acquires the float [single precision] data.
Call format		public float getSingleFloatData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the float [single precision] data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_NUMBER_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in float [single precision] unit. Use this property when acquiring float [single precision] data. "0.0" is stored to the return value by default.

## ■getStringData

Item		Content
Function		Acquires the character string data.
Call format		public java.lang.String getStringData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the character string data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • NOT_DATA_SET • UNSUPPORTED_ENCODING For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in character string unit. Use this property when acquiring data in character string despite of the data type of the stored data. The encoding code of the character string data acquired by this method is "UTF-16LE".*1 "null" is stored to the return value by default.

- \*1 When character string data is acquired with the "UTF-16LE" method, a single character is acquired as a two-byte data. Therefore, the incorrect values are read if data are stored with one-byte ASCII characters.  
When acquiring character string data in one-byte unit, specify the encoding code (such as "iso-8859-1") to set data in one-byte unit with getStringData(java.lang.String encode).


## ■getStringData

Item		Content
Function		Acquires the character string data.
Call format		public java.lang.String getStringData(java.lang.String encode) throws DataCommunicationException
Argument	Type name	java.lang.String
	Variable name	encode
	Content	[IN] Encoding code of the character string data to be acquired
Return value		Returns the character string data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library error. • ARGUMENT_NULL • NOT_DATA_SET • UNSUPPORTED_ENCODING For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires data stored in this DataValue class in character string unit. Character string data are acquired with the specified encoding code.*1 Use this method when acquiring stored character string data by specifying the encoding code. "null" is stored to the return value by default.


- \*1 Use this method when the data type of data stored in the DataValue class is character string.  
If this method is used for the data type other than character string, the specified character code is ignored.



## ■getUnsignedDoubleWordData

Item		Content
Function		Acquires the double word [unsigned] data.
Call format		public long getUnsignedDoubleWordData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the double word data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_NUMBER_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in double word unit. Use this property when acquiring double word [unsigned] data. "0" is stored to the return value by default.

## ■getUnsignedWordData

Item		Content
Function		Acquires the word [unsigned] data.
Call format		public int getUnsignedWordData() throws DataCommunicationException
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the word data.
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • EXCHANGE_TO_NUMBER_FAILED • NOT_DATA_SET For details, refer to the following section.  Page 154 getErrorCode
Comment		Acquires the data stored in this DataValue class in word unit. Use this property when acquiring word [unsigned] data. "0" is stored to the return value by default.


## ■setBitData

Item		Content
Function		Stores the bit type data.
Call format		public void setBitData(boolean bitData)
Argument	Type name	boolean
	Variable name	bitData
	Content	[IN] Bit type data
Return value		None
Comment		Stores the bit type data to this DataValue class. Execute this method when storing bit type data.

## ■setDoubleFloatData

Item		Content
Function		Stores the data in float [double precision] unit.
Call format		public void setDoubleFloatData(double doubleFloatData)
Argument	Type name	double
	Variable name	doubleFloatData
	Content	[IN] Float [double precision] data
Return value		None
Comment		Stores the data in float [double precision] unit to this DataValue class. Execute this method when storing float [double precision] data.

## ■setRawData

Item		Content
Function		Stores the data in raw unit.
Call format		public void setRawData(byte[] rawData) throws DataCommunicationException
Argument	Type name	byte[]
	Variable name	rawData
	Content	[IN] Raw data
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_RAW_LENGTH</li></ul> For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the raw data to this DataValue class. An exception occurs if the number of array elements of data to be stored exceeds 8192.

## ■setSignedDoubleWordData

Item		Content
Function		Stores the double word [signed] data.
Call format		public void setSignedDoubleWordData(int signedDoubleWordData)
Argument	Type name	int
	Variable name	signedDoubleWordData
	Content	[IN] Data in double word unit
Return value		None
Comment		Stores the data in double word unit to this DataValue class. Execute this method when storing double word [signed] data.


## ■setSignedWordData

Item		Content
Function		Stores the word [signed] data.
Call format		public void setSignedWordData(short signedWordData)
Argument	Type name	short
	Variable name	signedWordData
	Content	[IN] Data in word unit
Return value		None
Comment		Stores the data in word unit to this DataValue class. Execute this method when storing word [signed] data.

## ■setSingleFloatData


Item		Content
Function		Stores the data in float [single precision] unit.
Call format		public void setSingleFloatData(float singleFloatData)
Argument	Type name	float
	Variable name	singleFloatData
	Content	[IN] Data in float [single precision] unit
Return value		None
Comment		Stores the data in float [single precision] unit to this DataValue class. Execute this method when storing float [single precision] data.

## ■setStringData


Item		Content
Function		Stores the character string data.
Call format		public void setStringData(java.lang.String stringData) throws DataCommunicationException
Argument	Type name	java.lang.String
	Variable name	stringData
	Content	[IN] Character string data
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. • ARGUMENT_NULL • OUT_RANGE_STRING_LENGTH • UNSUPPORTED_ENCODING For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the character string data to this DataValue class. An exception occurs if the number of characters of data to be stored exceeds 4096. The encoding code of the character string data stored by this method is "UTF-16LE".*1

- \*1 When character string data are stored with the "UTF-16LE" method, a single character is stored as a two-byte data. (For example, when a character "A" is stored, it is stored as a two-byte data.)  
Therefore, when the stored data is written to the label of data size 1 with this method, the exception INVALID\_DATASIZE occurs.  
When storing character string data in one-byte unit, specify the encoding code (such as "iso-8859-1") to set data in one-byte unit with setStringData(java.lang.String stringData, java.lang.String encode).


## ■setStringData

Item		Content	
Function		Stores the character string data.	
Call format		public void setStringData(java.lang.String stringData, java.lang.String encode) throws DataCommunicationException	
Argument	Type name	java.lang.String	java.lang.String
	Variable name	stringData	encode
	Content	[IN] Character string data	[IN] Encoding code for character string
Return value		None	
Exception		DataCommunicationException — High Speed Data Communication Library error The following are the exceptions for High Speed Data Communication Library. <ul style="list-style-type: none"><li>• ARGUMENT_NULL</li><li>• OUT_RANGE_STRING_LENGTH</li><li>• UNSUPPORTED_ENCODING</li></ul> For details, refer to the following section.  Page 154 getErrorCode	
Comment		Stores the character string data to this DataValue class by specifying the encoding code. An exception occurs if the number of characters of data to be stored exceeds 8192. (Example) <ul style="list-style-type: none"><li>• Specifying the encoding code "UTF-16LE" (two-byte single character)</li></ul> An exception occurs when the character string data with the number of characters exceeding 4096 is stored. <ul style="list-style-type: none"><li>• Specifying the encoding code "iso-8859-1" (one-byte single character)</li></ul> An exception occurs when the character string data with the number of characters exceeding 8192 is stored.	

## ■setUnsignedDoubleWordData

Item		Content
Function		Stores the double word [unsigned] data.
Call format		public void setUnsignedDoubleWordData(long unsignedDoubleWordData) throws DataCommunicationException
Argument	Type name	long
	Variable name	unsignedDoubleWordData
	Content	[IN] Data in double word unit
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_UNSIGNED_DOUBLE_WORD For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the data in double word unit to this DataValue class. An exception occurs if a value out of the range of 0 to 4,294,967,295 is stored. Execute this method when storing double word [unsigned] data.

## ■setUnsignedWordData

Item		Content
Function		Stores the word [unsigned] data.
Call format		public void setUnsignedWordData(int unsignedWordData) throws DataCommunicationException
Argument	Type name	int
	Variable name	unsignedWordData
	Content	[IN] Data in word unit
Return value		None
Exception		DataCommunicationException — High Speed Data Communication Library error The following is the exception for High Speed Data Communication Library. • OUT_RANGE_UNSIGNED_WORD For details, refer to the following section.  Page 154 getErrorCode
Comment		Stores the data in word unit to this DataValue class. An exception occurs if a value out of the range of 0 to 65535 is stored. Execute this method when storing word [unsigned] data.

# ConnectionDeviceList class

Item	Content
Class syntax	public class ConnectionDeviceList extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.ConnectionDeviceList

## Constructor

Format	Content
ConnectionDeviceList()	Default constructor

## Details of the constructor

### ■ConnectionDeviceList

Item		Content
Function		Constructor
Call format		public ConnectionDeviceList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the list of the connected device information. The information maintained previously is initialized.

7

## Method list

Type	Method name	Overview	Reference
ConnectionDevice[]	getConnectionDeviceList()	Acquires the list of the connected device information from this ConnectionDeviceList class.	Page 147 getConnectionDeviceList

### ■getConnectionDeviceList

Item		Content
Function		Acquires the list of the connected device information.
Call format		public ConnectionDevice[] getConnectionDeviceList()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the list of the connected device information.
Comment		Acquires the list of the connected device information from this ConnectionDeviceList class. "null" is stored to the return value by default.*1

\*1 When values are stored using the getConnectionDeviceList method in the Communication class, "null" is stored to the array elements stored in the return value of getConnectionDeviceList in this class.

# ConnectionDevice class

Item	Content
Class syntax	public class ConnectionDevice extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hldatacommunication.ConnectionDevice

## Constructor

Format	Content
ConnectionDevice()	Default constructor

## Details of the constructor

### ■ConnectionDevice

Item		Content
Function		Constructor
Call format		public ConnectionDevice()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the list of the connected device information. The information maintained previously is initialized.

## Method list

Type	Method name	Overview	Reference
short	getAccessStatus()	Acquires the access status from this ConnectionDevice class.	Page 148 getAccessStatus
ConnectionDeviceDetail	getConnectionDeviceDetail()	Acquires the details of the connected device information from the ConnectionDevice class.	Page 149 getConnectionDeviceDetail
java.lang.String	getConnectionIPAddress()	Acquires the connection IP address from this ConnectionDevice class.	Page 149 getConnectionIPAddress
long	getConnectionTime()	Acquires the connection time from this ConnectionDevice class.	Page 150 getConnectionTime
java.lang.String	getConnectionUsername()	Acquires the connection user name from this ConnectionDevice class.	Page 150 getConnectionUsername

### ■getAccessStatus

Item		Content
Function		Acquires the access status.
Call format		public short getAccessStatus()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the access status.
Comment		Acquires the access status from this ConnectionDevice class. "0" is stored to the return value by default.

The following are the values for the access status.

Access status	Buffering invalid	Buffering enabled
No connection	0	
(The connection is established.)	10	11
Reading device data	20	21
Writing device data	30	31
On transfer (High speed sampling)	40	41
On transfer (General sampling)	60	61
Buffering transfer data (High speed sampling)	—	51
Buffering transfer data (General sampling)	—	71
On communication with tool	255	—

## ■getConnectionDeviceDetail

Item	Content
Function	Acquires the details of the connected device information.
Call format	public ConnectionDeviceDetail getConnectionDeviceDetail()
Argument	Type name
	Variable name
	Content
Return value	Returns the details of the connected device information.
Comment	Acquires the details of the connected device information from the ConnectionDevice class. "null" is stored to the return value by default. *1

\*1 When values are stored using the getConnectionDeviceList method in the Communication class, "null" is acquired to the return values of getConnectionDeviceDetail in this class for the access status other than the ones indicated below.

- On connection: 10
- On connection (Buffering enabled): 11
- Reading device data: 20
- Reading device data (Buffering enabled): 21
- Writing device data: 30
- Writing device data (Buffering enabled): 31
- On transfer (High speed sampling): 40
- On transfer (High speed sampling) (Buffering enabled): 41
- Buffering transfer data (High speed sampling) (Buffering enabled): 51
- On transfer (General sampling): 60
- On transfer (General sampling) (Buffering enabled): 61
- Buffering transfer data (General sampling) (Buffering enabled): 71

## ■getConnectionIPAddress

Item	Content
Function	Acquires the connection IP address.
Call format	public java.lang.String getConnectionIPAddress()
Argument	Type name
	Variable name
	Content
Return value	Returns the connection IP address.
Comment	Acquires the connection IP address from this ConnectionDevice class. "null" is stored to the return value by default.

## ■getConnectionTime

Item		Content
Function		Acquires the connection time.
Call format		public long getConnectionTime()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the connection time.
Comment		Acquires the connection time (the number of elapsed seconds from January 1, 1970) from this ConnectionDevice class. "0" is stored to the return value by default.

## ■getConnectionUsername

Item		Content
Function		Acquires the connection user name.
Call format		public java.lang.String getConnectionUsername()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the connection user name.
Comment		Acquires the connection user name from this ConnectionDevice class. "null" is stored to the return value by default. *1

\*1 When the access authentication is disabled, "null" is stored to the connection user name.



# ConnectionDeviceDetail class

Item	Content
Class syntax	public class ConnectionDeviceDetail extends java.lang.Object
Class inheritance	java.lang.Object └─jp.co.mitsubishielectric.hsdatacommunication.ConnectionDeviceDetail

## Constructor

Format	Content
ConnectionDeviceDetail()	Default constructor

## Details of the constructor

### ■ConnectionDeviceDetail

Item		Content
Function		Constructor
Call format		public ConnectionDeviceDetail()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the details of the connected device information. The information maintained previously is initialized.

## Method list

Type	Method name	Overview	Reference
long	getBufferingDataCountCurrent()	Acquires the number of buffering data (current) from this ConnectionDeviceDetail class.	Page 152 getBufferingDataCountCurrent
long	getBufferingDataCountMaximum()	Acquires the number of buffering data (maximum) from this ConnectionDeviceDetail class.	Page 152 getBufferingDataCountMaximum
int	getHighSpeedSamplingFailureCount()	Acquires the number of high speed sampling failures from this ConnectionDeviceDetail class.	Page 152 getHighSpeedSamplingFailureCount
short	getLatestErrorCode()	Acquires the current error code from this ConnectionDeviceDetail class.	Page 152 getLatestErrorCode
long	getPossibleBufferingDataCount()	Acquires the number of data that can be buffered from this ConnectionDeviceDetail class.	Page 153 getPossibleBufferingDataCount
long	getPossibleBufferingTime()	Acquires the total time that data can be buffered from this ConnectionDeviceDetail class.	Page 153 getPossibleBufferingTime
int	getProcessingOverloadCount()	Acquires the number of processing overload occurrences from this ConnectionDeviceDetail class.	Page 153 getProcessingOverloadCount
long	getSamplingIntervalAverage()	Acquires the sampling time (moving average) from this ConnectionDeviceDetail class.	Page 153 getSamplingIntervalAverage
long	getSamplingIntervalMaximum()	Acquires the sampling time (maximum) from this ConnectionDeviceDetail class.	Page 153 getSamplingIntervalMaximum

## Method details

### ■getBufferingDataCountCurrent

Item		Content
Function		Acquires the number of buffering data (current).
Call format		public long getBufferingDataCountCurrent()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the number of buffering data (current).
Comment		Acquires the number of buffering data (current) from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

### ■getBufferingDataCountMaximum

Item		Content
Function		Acquires the number of buffering data (maximum).
Call format		public long getBufferingDataCountMaximum()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the number of buffering data (maximum).
Comment		Acquires the number of buffering data (maximum) from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

### ■getHighSpeedSamplingFailureCount

Item		Content
Function		Acquires the number of high speed sampling failures.
Call format		public int getHighSpeedSamplingFailureCount()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the number of high speed sampling failures.
Comment		Acquires the number of high speed sampling failures from this ConnectionDeviceDetail class. "0" is stored to the return value by default. *1

\*1 When the access status is "On transfer (General sampling)" or "Buffering transfer data (General sampling)", "0" is stored to the number of high speed sampling failures.

### ■getLatestErrorCode

Item		Content
Function		Acquires the current error code.
Call format		public short getLatestErrorCode()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the current error code.
Comment		Acquires the current error code from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

## ■getPossibleBufferingDataCount

Item		Content
Function		Acquires the number of data that can be buffered.
Call format		public long getPossibleBufferingDataCount()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the number of data that can be buffered.
Comment		Acquires the number of data that can be buffered from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

## ■getPossibleBufferingTime

Item		Content
Function		Acquires the total time that data can be buffered.
Call format		public long getPossibleBufferingTime()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the total time that data can be buffered.
Comment		Acquires the total time that data can be buffered from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

## ■getProcessingOverloadCount

Item		Content
Function		Acquires the number of processing overload occurrences.
Call format		public int getProcessingOverloadCount()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the number of processing overload occurrences.
Comment		Acquires the number of processing overload occurrences from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

## ■getSamplingIntervalAverage

Item		Content
Function		Acquires the sampling time (moving average).
Call format		public long getSamplingIntervalAverage()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the sampling time (moving average).
Comment		Acquires the sampling time (moving average) from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

## ■getSamplingIntervalMaximum

Item		Content
Function		Acquires the sampling time (maximum).
Call format		public long getSamplingIntervalMaximum()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the sampling time (maximum).
Comment		Acquires the sampling time (maximum) from this ConnectionDeviceDetail class. "0" is stored to the return value by default.

# DataCommunicationException class

Item	Content
Class syntax	public class DataCommunicationException extends Exception
Class inheritance	java.lang.Object └─ java.lang.Throwable └─ java.lang.Exception └─ jp.co.mitsubishielectric.hsdatacommunication.DataCommunicationException

## Constructor

Format	Content
DataCommunicationException()	Default constructor

## Details of the constructor

### ■DataCommunicationException


Item		Content
Function		Constructor
Call format		public DataCommunicationException()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		None
Comment		Constructs a class to store the enumerators of the exceptions at the method execution. The enumerators of the exceptions maintained previously are initialized.

## Method list

Type	Method name	Overview	Reference
DataCommunicationError	getErrorCode()	Acquires the enumerators of the exceptions from this DataCommunicationException class.	Page 154 getErrorCode

## Method details

### ■getErrorCode

Item		Content
Function		Acquires the enumerators of the exceptions.
Call format		public DataCommunicationError getErrorCode()
Argument	Type name	—
	Variable name	—
	Content	—
Return value		Returns the enumerators of the exceptions.
Comment		Acquires the enumerators of the exceptions from this DataCommunicationException class. For the exception handler occurrence, refer to the following section.  Page 155 DataCommunicationError enumerator "NONE" is stored to the return value by default.

# DataCommunicationError enumerator

Item	Content
Class syntax	public enum DataCommunicationError
Class inheritance	java.lang.Object └ java.lang.Enum<DataCommunicationError> └ └ jp.co.mitsubishielectric.hsdatacommunication.DataCommunicationError

## Enumerator list

Enumerator	Description	Corrective action
ABORTED_BY_PEER	Method suspension error	Check if a method other than stopStreamingTransfer is executed during the streaming transfer.
ARGUMENT_NULL	Null was specified for the argument. A null object exists in the object specified for the argument.	Review the value specified for the argument, and execute the method again.
AUTHENTICATION_FAILED	The access authentication failed.	Check the account authentication setting and check if the specified user name and password are correct.
BUFFERING_CONNECT_OVER	The number of connections in which the buffering transfer data is enabled exceeded the maximum number of applicable connections.	Refer to the event history/connection history, check if there are two connections which are enabled buffering transfer data. Take the following corrective action as necessary. <ul style="list-style-type: none"> <li>• Disconnect the connection of the user program executing connect</li> <li>• Stop buffering transfer data</li> </ul>
CONNECT_OVER	The maximum number of devices that can be connected exceeded.	Refer to the event history/connection history, check if the connection for High Speed Data Communication Module is more than 5. Also, check if there is an unexpected access when CONNECT_OVER occurred. Take the following corrective action as necessary. <ul style="list-style-type: none"> <li>• Stop accessing from Configuration Tool</li> <li>• Disconnect the connection of the user program executing connect</li> <li>• Stop buffering transfer data</li> </ul>
CONNECT_REFUSED	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
DISCONNECTED	A disconnection was attempted to the connection of the High Speed Data Communication Module which is not connected. A communication was attempted without the reconnection after the line disconnection event occurred.	Connect the line.
DISPOSED_ERROR	The line was disconnected.	Execute the connection processing again.
EXCHANGE_FROM_STRING_FAILED	The conversion from the character string failed.	Check that JDK provided by Oracle is used.
EXCHANGE_TO_BIT_FAILED	A conversion from character string data to bit device was attempted.	Check the data type of the stored data. If the data type is character string, do not acquire the data as a bit device.
EXCHANGE_TO_NUMBER_FAILED	A conversion from character string data to raw data is failed.	Check that the stored character string can be converted to acquirable data (integers or decimals) with the specified method. Check that the stored character string is within the range of acquirable data with the specified method.
EXCHANGE_TO_STRING_FAILED	A conversion to character string data failed.	Check that JDK provided by Oracle is used.
EXECUTING	Another command was executed from one instance while the module was in processing.	Execute the method again after canceling or completing the processing being executed.
GENERAL_ERROR	Unknown error	Check that JDK provided by Oracle is used.
INTERRUPTED_ERROR	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
INVALID_CONNECT		
INVALID_DATASIZE	A data size that exceeds the label data size specified in Configuration Tool.	Check that the specified label is correct. Check that the stored data size (byte) does not exceed the data size set in Configuration Tool.
INVALID_DATATYPE	A data type that differs from the data type of the label written in Configuration Tool was specified.	Check that the specified label is correct. Check that the data type of the stored data is correct.

Enumerator	Description	Corrective action
INVALID_IPADDRESS	The specified IP address is invalid.	Check that the specified IP address is within the IP address range.
INVALID_REQUEST	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
INVALID_RESPONSE		
INVALID_STREAMING		
IO_ERROR	A communication error occurred.	Check the following. <ul style="list-style-type: none"> <li>• High Speed Data Communication Module status, network connection status</li> <li>• Cable connection.</li> </ul> Review the value specified for the IP address, and then execute the method again.
IPADDRESS_NULL	Null was specified for the IP address.	Check the value specified for the IP address, and then execute the method again.
MINUS_ARGUMENT	A negative value was specified for the argument.	Review the specified argument, and then execute the method again.
NONE	No errors	—
NOT_CONNECTED_YET	The connection processing is not executed.	Execute the connection processing, and then executed the method again.
NOT_DATA_SET	Data is not stored.	Store the data and execute the method again.
NOT_PERMITTED_IPADDRESS	An IP address which is not authorized to connect was specified.	Check the specified IP address for the connection, and check the access authentication. Check that the specified IP address is correct.
NOT_SAME_ELEMENT_COUNT	The number of specified labels or devices does not match with the number of data.	Check the specified argument, and set each number of elements to be the same amount.
OUT_RANGE_ADDITIONAL_TIMETYPE	The specified type of time information to be appended is out of the range.	Check that the specified type of additional time information is within the range.
OUT_RANGE_CONNECT_MODE	System error	Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
OUT_RANGE_CONNECTION_NO	The specified connection number is out of the range.	Check that the specified connection number is within the range of 1 to 5.
OUT_RANGE_DEVICE_COUNT	The specified number of devices is out of the range.	Check that the specified number of devices is within the range.
OUT_RANGE_DEVICE_NAME_LENGTH	The length of the specified device name is out of the range.	Check that the length of the specified device name is within the range.
OUT_RANGE_LABEL_COUNT	The specified number of labels is out of the range.	Check that the specified number of labels is within the range.
OUT_RANGE_LABEL_NAME_LENGTH	The length of the specified label name is out of the range.	Check that the length of the specified label name is within the range.
OUT_RANGE_LABELGROUP_NAME_LENGTH	The length of the specified label group name is out of the range.	Check that the length of the specified label group name is within the range.
OUT_RANGE_PASSWORD	The length of the specified password is out of the range.	Check that the length of the specified password is within the range.
OUT_RANGE_RAW_LENGTH	The number of the specified raw data is out of the range.	Check that the number of the specified raw data is within the range.
OUT_RANGE_STRING_LENGTH	The length of the specified character string is out of the range.	Check that the length of the specified character string is within the range.
OUT_RANGE_TIMEOUT	The specified timeout period is out of the range.	Check that the specified timeout period is within the range.
OUT_RANGE_TRANSFER_TIME	The specified transfer cycle is out of the range.	Check that the specified transfer cycle is within the range.
OUT_RANGE_UNSIGNED_DOUBLE_WORD	The specified data is out of the range of double word [unsigned].	Check that the specified data is within the range of double word [unsigned].
OUT_RANGE_UNSIGNED_WORD	The specified data is out of the range of word [unsigned].	Check that the specified data is within the range of word [unsigned].
OUT_RANGE_USER_NAME	The length of the specified user name is out of the range.	Check that the length of the specified user name is within the range.
OUT_RANGE_WORD_SIZE	The total number of words of the specified label is out of the range.	Check that the number of words of the specified label is within the range.
PASSWORD_NULL	Null was specified for the password.	Review the value of the specified password, and then execute the method again.

Enumerator	Description	Corrective action
RECONNECT	A connection is attempted to the module which is already connected.	To reconnect, disconnect the line, and then connect again.
STREAMING_CONNECT_OVER	The maximum number of devices that can be connected for the streaming transfer exceeded.	Refer to the event history/connection history, check if the connection of the user program is more than 4. Also, check if there is an unexpected access when STREAMING_CONNECT_OVER occurred. Take the following corrective action as necessary. <ul style="list-style-type: none"> <li>• Stop accessing from Configuration Tool</li> <li>• Disconnect the connection of the user program executing connect</li> <li>• Stop buffering transfer data</li> </ul>
TIMEOUT	The communication was timed-out.	Check the following: <ul style="list-style-type: none"> <li>• High Speed Data Communication Module status, network connection status</li> <li>• Cable connection</li> <li>• Network connection status of CPUs on other stations</li> </ul> Execute the method again after performing the following actions. <ul style="list-style-type: none"> <li>• Set a longer timeout period for the TimeOut property.</li> <li>• Reduce the number of CPUs on other stations that are accessed simultaneously.</li> <li>• Reduce the number of device points that are accessed simultaneously.</li> </ul> Review the value of the specified IP address, and then execute the method again. When using startStreamingTransferByInterval or startStreamingTransfer, refer to the details of the method to be used. <ul style="list-style-type: none"> <li>☞ Page 115 startStreamingTransfer</li> <li>☞ Page 116 startStreamingTransfer</li> <li>☞ Page 117 startStreamingTransferByInterval</li> </ul>
UNSUPPORTED_ENCODING	The unsupported character code is specified.	Check that JDK provided by Oracle is used. Check if the specified character code is supported by JDK provided by Oracle.

## 7.4 Compiling Java User Program

This section explains how to compile the user program created with Java.

When compiling the user program to which High Speed Data Communication Library is imported, the classpath to High Speed Data Communication Library needs to be specified.

Specify the classpath to High Speed Data Communication Library using "-classpath(-cp)" when compiling the program.

When another class file is referred, enter a delimiter to the specified target of "-classpath".

For a delimiter, use a semicolon (;) for Windows®, and a colon (:) for the UNIX based operating system.

The following is the command format when compiling the program.

```
javac -classpath ○○○○XXXX.java
```

○○○○ = Path to High Speed Data Communication Library copied to the development personal computer

XXXX.java = User program name

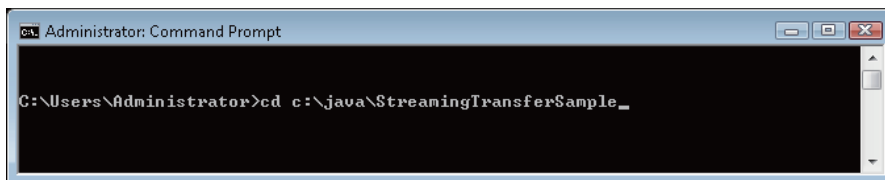
### Operating procedure

1. Start a command prompt.
2. Change the work folder for the command prompt to the folder to which the sample program is stored.

**Ex.**

When the work folder at the command prompt activation is "C:\Users\Administrator"

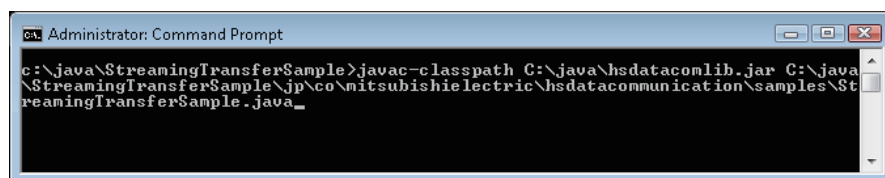
Execute "cd C:\java\StreamingTransferSample".



3. The user program is compiled.

**Ex.**

When compiling "StreamingTransferSample.java"



→ After compiling the program, the executable file StreamingTransferSample.class etc.) is created in the same folder as the user program.



# 8 EXECUTING Java USER PROGRAM

This chapter explains how to execute a user program from the server personal computer.

## 8.1 Copying Execution File for User Program to Server Personal Computer

Copy the execution files (XXXX.class) and High Speed Data Communication Library (hsdatacomlib.jar) created by a development personal computer to a server personal computer.

Copying the user program (XXXX.java) used for programming on the development personal computer is not needed.

## 8.2 Executing Processing from Server Personal Computer

Execute the execution file to start processing.

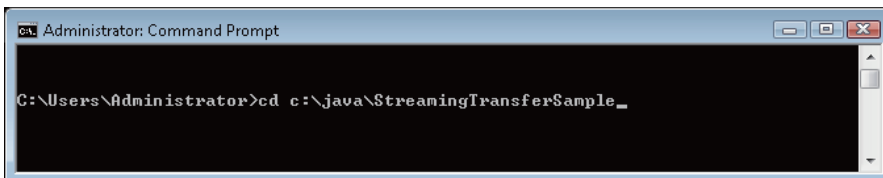
This section explains how to execute a data processing by executing the execution file named XXXX.class copied into the C:\java folder on the server personal computer (Windows) as an example.

### Operating procedure

1. Start a command prompt.
2. Change the operation folder for the command prompt to the folder that stores the execution file.

**Ex.**

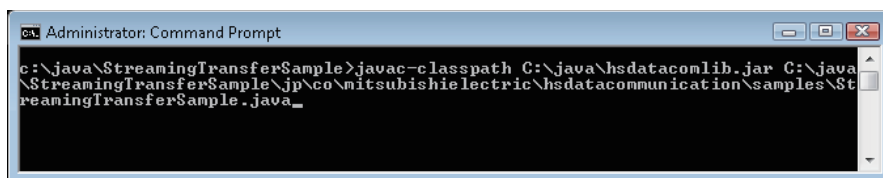
When the execution file is stored in the java folder and the work folder for the command prompt is "C:\Users\Administrator". Execute "cd C:\java\StreamingTransferSample".



3. Execute the execution file.

**Ex.**

When executing the execution file "XXXX.class" attaching the option "-classpath" with java command. Execute "java -classpath .;C:\java\hsdatacomlib.jar XXXX".



Processing starts.



# PART 4

# TROUBLESHOOTING

## PART 4

This part explains the troubleshooting at creating programs.




## 9 TROUBLESHOOTING

---

# 9 TROUBLESHOOTING

This section explains the error definition and corrective action by function and symptom.  
The method name is written based on C# in this chapter unless otherwise noted.

## 9.1 Troubleshooting related to user programs

Symptom	Check point	Corrective action
Build fails. (C#)	Does the reference of High Speed Data Communication Library add to the project?	<ul style="list-style-type: none"><li>• Check if "HSDDataComLib.dll" is referred on the solution explorer, and add the reference.</li></ul>
Compilation fails. (Java)	Has High Speed Data Communication Library been imported?	<ul style="list-style-type: none"><li>• Import High Speed Data Communication Library at the top of the user program.  High Speed Data Communication Module User's Manual</li></ul>
	Is the classpath specified at compilation?	<ul style="list-style-type: none"><li>• Specify '-classpath' for compilation option.  Page 158 Compiling Java User Program</li></ul>
	Is the classpath specification destination correct?	<ul style="list-style-type: none"><li>• Check the location where High Speed Data Communication Library is installed, and change the classpath specification.  Page 158 Compiling Java User Program</li></ul>
Execution fails. (C#)	Is the High Speed Data Communication Library copied to a server personal computer?	Check the storage folder of the execution file, and copy "HSDDataComLib.dll" to the server personal computer.
Execution fails. (Java)	Is the High Speed Data Communication Library copied to a server personal computer?	Copy "hsdatacomlib.jar" to the server personal computer.
	Is the classpath specification destination correct?	Check the storage folder of the execution file, and change the specification destination of the class path.

## 9.2 Troubleshooting related to network connection

Symptom	Check point	Corrective action
Cannot access the High Speed Data Communication Module. (The Connect method does not complete successfully.)	Is the mode of the intelligent function module switch setting "Online"?	<ul style="list-style-type: none"> <li>Set the mode to "Online".</li> </ul> High Speed Data Communication Module User's Manual
	Is the High Speed Data Communication Module connected to the network? (X4 = ON)	<ul style="list-style-type: none"> <li>Connect the High Speed Data Communication Module to the network.</li> </ul>
	Is there a disconnection along the connection route?	<ul style="list-style-type: none"> <li>Connect the cables properly.</li> <li>Replace the cables to new ones.</li> </ul>
	Is there a terminal on the network with the same IP address as the High Speed Data Communication Module?	<ul style="list-style-type: none"> <li>Make sure that overlapping terminals are not exist on the network.</li> </ul> High Speed Data Communication Module User's Manual
	Does a firewall or proxy server exist along the connection route?	<ul style="list-style-type: none"> <li>Ask your network administrator about the firewall and proxy server settings.</li> </ul>
	Is Windows firewall enabled on the personal computer?	<ul style="list-style-type: none"> <li>Disable Windows firewall on the personal computer when using the module search function or direct connection.</li> </ul>
	Is antivirus software blocking Ethernet communications?	<ul style="list-style-type: none"> <li>Change the antivirus software settings to allow Ethernet communications.</li> <li>Lower the antivirus software's security settings level.</li> <li>Stop the antivirus software.</li> </ul>
	Is there any problem on the personal computer?	<ul style="list-style-type: none"> <li>Replace it with another personal computer.</li> </ul>
	Are the authorities of the user logged on to Windows® sufficient?	<ul style="list-style-type: none"> <li>For Windows® XP, logon as a user with a "limited" or higher user account.</li> <li>For Windows Vista® or later, logon as a user with a "standard" or higher user account.</li> </ul>
	Is the IP address of the High Speed Data Communication Module specified in transfer setup correct? (When the power is turned ON without a CompactFlash card inserted, the module operates with the IP address in the initial status (192.168.3.3).)	<ul style="list-style-type: none"> <li>Review the IP address of the High Speed Data Communication Module specified in transfer setup.</li> </ul> High Speed Data Communication Module User's Manual <ul style="list-style-type: none"> <li>Directly connect the personal computer and specify direct connection in transfer setup.</li> </ul> High Speed Data Communication Module User's Manual
	Has an error occurred on the own station's programmable controller CPU?	<ul style="list-style-type: none"> <li>Check the error code for the programmable controller CPU, take corrective action according to the error code.</li> </ul>
	Is the High Speed Data Communication Module mounting slot set to other than "Intelli." (other CPU in multiple CPU, I/O module) in the PLC parameter (I/O assignment) of the host programmable controller CPU?	<ul style="list-style-type: none"> <li>Review the PLC parameter (I/O assignment) of the programmable controller CPU.</li> </ul>
	Are the High Speed Data Communication Module and personal computer connected to each other via a hub?	<ul style="list-style-type: none"> <li>For a direct connection, connect the High Speed Data Communication Module to the personal computer on a 1:1 basis.</li> </ul> High Speed Data Communication Module User's Manual

Symptom	Check point	Corrective action
Cannot access the High Speed Data Communication Module. (The Connect method does not complete successfully.)	Are multiple IP addresses enabled at the same time in the personal computer side?	<ul style="list-style-type: none"> <li>For a direct connection, make sure the multiple IP addresses are not enabled at the same time in the personal computer. Disable the wireless LAN function.</li> </ul>
	Are there more than 5 connections?	<p>Check the module's access status with the following, and check whether there are 5 connections to the module.</p> <ul style="list-style-type: none"> <li>Configuration Tool diagnostics (access status)</li> <li>Access status area of buffer memory If there are 5 connections, perform the measures below, and then try again.</li> <li>Close the diagnostics window of Configuration Tool.</li> <li>Close a user program.</li> </ul>
	Is the account authentication function enabled?	<ul style="list-style-type: none"> <li>Check whether the account authentication function of the High Speed Data Communication Module is enabled.</li> </ul>
	Is the IP address specified by the user program correct?	<ul style="list-style-type: none"> <li>Review the IP address of the High Speed Data Communication Module, and review whether the specification of the user program is correct.</li> </ul>
	Is the number of connections greater than the number of transfer-enabled personal computers?	<p>Check the module's access status with the following, and check whether there are four connections to the module other than "On communication with tool".</p> <ul style="list-style-type: none"> <li>Configuration Tool diagnostics (access status)</li> <li>Access status area of buffer memory</li> </ul> <p>If there are four connections other than "On communication with tool", close a user program, and try again.</p>
	Are there more than 2 connections which enable buffering?	<ul style="list-style-type: none"> <li>Disabled the buffering and execute user program.</li> <li>End the user program which enables buffering, and execute again.</li> </ul>
	Is the specified value of buffering mode correct?	<ul style="list-style-type: none"> <li>Check if the value stored to the buffering mode is within the range, and execute again.</li> </ul>
	Does the specified time information to add have the correct type?	<ul style="list-style-type: none"> <li>Specify a value for the time information to add that is within the range, and try again.</li> </ul>
	Is the specified user name correct? Or is the length of the user name in bounds?	<ul style="list-style-type: none"> <li>Make sure that the account specified in the user program exists.</li> <li>Review whether the length of the specified user name is within bounds.</li> </ul>
	Is the specified password correct? Or is the length of the password in bounds?	<ul style="list-style-type: none"> <li>Make sure that the combination of the specified account and password are correct.</li> <li>Review whether the length of the specified password is within bounds.</li> <li>Make sure that the specified password is not null, and then try again.</li> </ul>
	Is the IP filter function enabled?	<ul style="list-style-type: none"> <li>Check whether the access restriction function of the High Speed Data Communication Module is enabled, and make sure that the server personal computer's IP address is accessible.</li> </ul>
	Was the communication right after disconnection?	<p>Disconnect the line, and reconnect after a while.</p>

## 9.3 Troubleshooting related to label functions

Symptom	Check point	Corrective action
The GetLabelList/GetLabelListWithComment method does not complete successfully.	Has streaming transfer been started?	Check the module's access status with the following, and check whether the connection that is executing the method is performing the streaming transfer. <ul style="list-style-type: none"> <li>• Configuration Tool diagnostics (access status)</li> <li>• Establish the access area of the buffer memory or a new connection, and execute on the connection.</li> </ul>
In the GetLabelList/GetLabelListWithComment method, there is no label group list/label list to store to.	Are the High Speed Data Communication Module and personal computer connected with an account where the accessible label group exists?	• Check the settings written to the High Speed Data Communication Module, and make sure that the label is set in the label group.
	Connected with the account that accessible label group exist	• Check the connected account, and make sure that the checkbox for authority to read the label group list is selected.

## 9.4 Troubleshooting related to streaming transfer

Symptom	Check point	Corrective action
The SetStreamingLabel/SetStreamingLabelGroup method do not complete successfully. Is the total number of word exceeded 8192 word?	Is non-existent label group or label name specified?	• Review that the label group specified in the user program matches the label group name or group name set in the High Speed Data Communication Module.
	Are a high-speed sampling label group and general sampling label group specified simultaneously?	• Review that the user program does not specify both a high-speed sampling label group and general sampling label group.
	Does the total number of high speed sampling labels to register in a High Speed Data Communication Module exceed 8192?	• Reduce the total number of labels or words in the high-speed sampling labels specified by the user program. • If the total number of high-speed sampling labels or words registered by multiple user programs is greater than 8192, then reduce the total number of labels or words. Alternatively, stop a user program.
	Are there more than 16384 general sampling labels per connection to register to the High Speed Data Communication Module?	• Reduce the number of general sampling labels specified in the user program.
	Does the total number of labels to register to the High Speed Data Communication Module exceed the per-module limit of 65536? Is the total number of words greater than 65536?	• Reduce the total number of labels or words specified by the user program. • If the total number of labels or words registered by multiple user programs is greater than 65536, then reduce the total number of labels or words. Alternatively, stop a user program.
	Has streaming transfer been started?	Check the module's access status with the following, and check whether the connection that is executing the method is performing the streaming transfer. <ul style="list-style-type: none"> <li>• Configuration Tool diagnostics (access status)</li> <li>• Establish the access area of the buffer memory or a new connection, and execute on the connection.</li> </ul>
	The number of label groups or labels specified in the user program is 0?	• Check if the number of labels acquired with the GetLabelList/GetLabelListWithComment method exist more than one. • Check the number of label groups or labels specified in the user program, and specify more than one label.

Symptom	Check point	Corrective action
The StartStreamingTransfer/ StartStreamingTransferByInterval method does not complete normally.	Are three or more streaming transfers for the specified high-speed sampling label group executed on a single module?	<ul style="list-style-type: none"> <li>• Reduce the number of streaming transfers per module for the specified high-speed sampling label group.</li> <li>• Check the module's access status by follows, and check the buffering transfer data (high speed sampling) count.</li> <li>• Configuration Tool diagnostics (access status)</li> <li>• Access status area of buffer memory</li> </ul>
	Has streaming transfer been started?	<p>Check the module's access status with the following, and check whether the connection that is executing the method is performing the streaming transfer.</p> <ul style="list-style-type: none"> <li>• Configuration Tool diagnostics (access status)</li> <li>• Establish the access area of the buffer memory or a new connection, and execute on the connection.</li> </ul>
The StartStreamingTransfer/ StartStreamingTransferByInterval method cannot be received the transferred data.	<ul style="list-style-type: none"> <li>• For C# Is the ReceiveTransferData registration processing executed before executed the Connect method?</li> <li>• For Java Is the addListener method executed before executing the connect method?</li> </ul>	<ul style="list-style-type: none"> <li>• For C# Execute the ReceiveTransferData registration processing before executed the Connect method of the user program.</li> <li>• For Java Run addListener method before the user program's connect method.</li> </ul>
The streaming transfer of the module which is executing buffering transfer data is not restarted.	Is there a disconnection along the connection route?	<ul style="list-style-type: none"> <li>• Connect the cables properly.</li> <li>• Replace the cables to new ones.</li> </ul>
	Does a firewall or proxy server exist along the connection route?	Ask your network administrator about the firewall and proxy server settings.
	Is antivirus software blocking Ethernet communications?	<ul style="list-style-type: none"> <li>• Change the antivirus software settings to allow Ethernet communications.</li> <li>• Lower the antivirus software's security settings level.</li> <li>• Stop the antivirus software.</li> </ul>
	Is the timeout period elapsed?	Start streaming transfer again after stopping buffering and set a longer timeout period for the TimeOut property.



## 9.5 Troubleshooting related to data read and data write functions

Symptom	Check point	Corrective action
The ReadDataByLabel/ WriteDataByLabel method does not complete normally.	Is the total number of specified labels greater than 65536?	<ul style="list-style-type: none"> <li>Reduce the number of labels specified in the user program.</li> </ul>
	Is the total size in words of the specified labels greater than 65536?	<ul style="list-style-type: none"> <li>Reduce the total size in words of the specified labels in the user program.</li> </ul>
	Does the connected account have Label group access authority?	<ul style="list-style-type: none"> <li>Make sure that the connected account has access authority (Read from CPU/Write to CPU) for label groups.</li> </ul>
	Did access to the other module fail?	<ul style="list-style-type: none"> <li>Check the access log file, and check if there was a problem with the other module's access network.</li> <li>If there was an error, take measures in accordance with the error code.</li> </ul>
	Has streaming transfer been started?	<p>Check the module's access status with the following, and check whether the connection that is executing the method is performing the streaming transfer. When performing the streaming transfer, execute the method after stopping the streaming transfer.</p> <ul style="list-style-type: none"> <li>Configuration Tool diagnostics (access status)</li> <li>Establish the access area of the buffer memory or a new connection, and execute on the connection.</li> </ul>
	Does the time exceed the timeout period?	<p>Execute the following, and then try the action again.</p> <ul style="list-style-type: none"> <li>Set a longer timeout period for the SetTimeOut method.</li> <li>Check the network connection status of CPUs on other stations.</li> <li>Reduce the number of CPUs on other stations that are accessed simultaneously.</li> <li>Reduce the number of device points that are accessed simultaneously.</li> </ul>
The readDataByDevice/ writeDataByDevice method does not complete normally.	Is the total number of specified devices greater than 65536?	<ul style="list-style-type: none"> <li>Reduce the number of devices specified in the user program.</li> </ul>
	Is a device specified in columns K5 to K8 specified?	<ul style="list-style-type: none"> <li>Check the specified device. If it is specified in columns K5 to K8, change it to a combination specifying columns K1 to K4.</li> </ul>
	Does the connected account have device access authority?	<ul style="list-style-type: none"> <li>Make sure that the connected account has device specification access authority (Read from CPU/Write to CPU).</li> </ul>
	Is nonexistent device accessed?	<ul style="list-style-type: none"> <li>Review whether the specified device name is a device that is accessible to the control CPU of own station.</li> </ul>
	Has streaming transfer been started?	<p>Check the module's access status with the following, and check whether the connection that is executing the method is performing the streaming transfer.</p> <ul style="list-style-type: none"> <li>Configuration Tool diagnostics (access status)</li> <li>Establish the access area of the buffer memory or a new connection, and execute on the connection.</li> </ul>

# APPENDIX

## Appendix 1 Sample Program

The sample program is stored in the "Samples" folder under "Library" in the compressed file of this product.  
The sample program is used at the user's own risk.

### How to use the C# sample program

#### Sample Program List

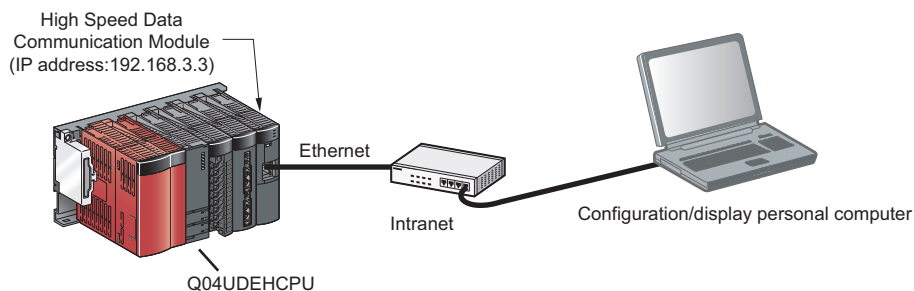
Folder Name	Details
StreamingTransferSample	Performs the stream transfer to transfer values of LABEL_B0 (B0 device) and LABEL_D0 (D0 device) in 100 ms cycles. Receives data for approximately 60 seconds.
DataReadWriteSample	Read the current value of B10 device and D10 device, and write to the module with adding "1" (bit device is inverted). Read the current value of the LABEL_B0 (B0 device) and LABEL_D0 (D0 device), and write to the module with adding "1" (bit device is inverted).

#### How to use the sample program

This section explains the configuration and procedures for operating the sample program.

##### ■System Configuration for Operating the Sample Program

Connect the High Speed Data Communication Module to the configuration personal computer on a 1:1 basis via hub.  
Set the IP address of the configuration personal computer to "192.168.3.1".

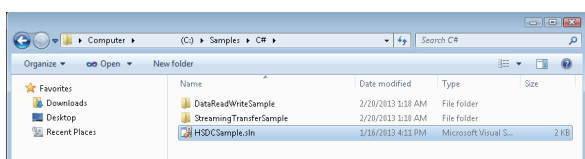
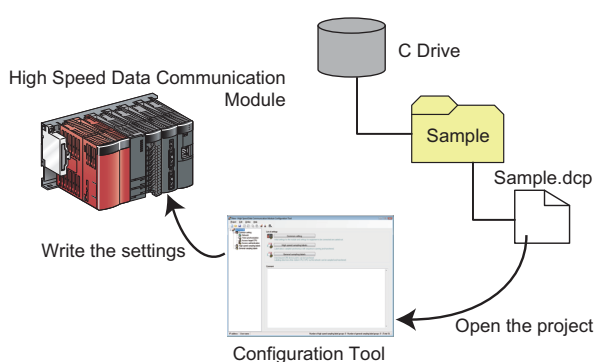
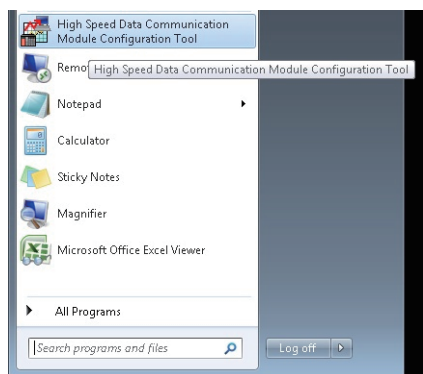
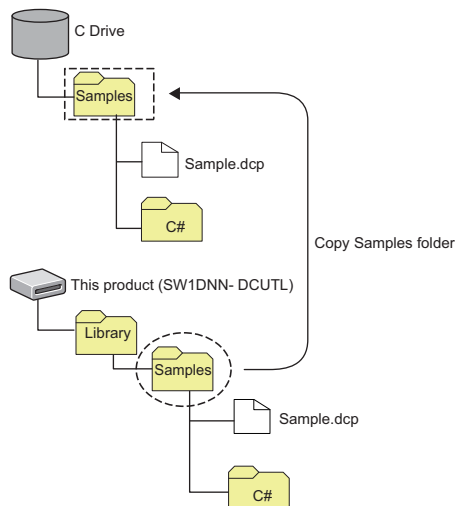


## ■Operating Procedures for Sample Program

The following explain the procedure to operate the sample program that executes the streaming transfer.

1. Copy the "Library\Samples" folder in the compressed file to the c: drive.

The contents under the java folder, however, delete them if unnecessary.



2. Start Configuration Tool.

(Example: Execute [Start] ⇒ [All Programs] ⇒ [MELSOFT Application] ⇒ [High Speed Data Communication] ⇒ [High Speed Data Communication Module Configuration Tool].)

3. Write the settings to the module.

Write the project file included in the sample program to be executed.

(Example: Open "C:\Sample\Sample.dcp" with Configuration Tool, and write to the module.)

4. Start up the solution of the sample program.

Start up the solution to build the sample program.

(Example: Execute "C:\Sample\C#\HSDCSample.sln".)



5. When the screen on the left is displayed after start the solution, click the [Finish] button.

6. Build the sample program to be executed.
7. After build the sample program, execute the created exe file.

### ■Source code of the sample program

The sample program of the streaming transfer function is stored in the "C#" folder under "Library" - "Samples" in the compressed file of this product.

# How to use the Java sample program

## Sample Program List

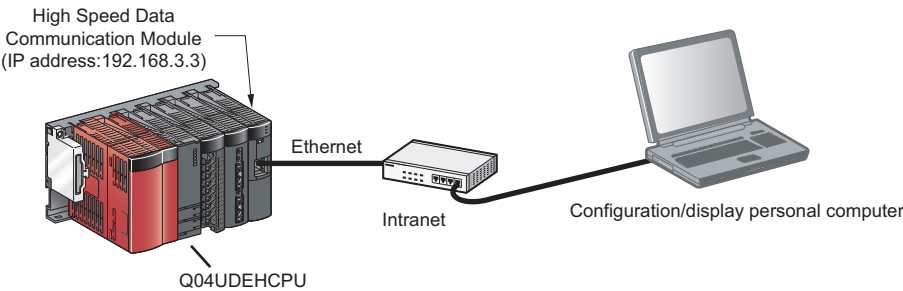
Folder Name	Details
StreamingTransferSample	Performs the stream transfer to transfer values of LABEL_B0 (B0 device) and LABEL_D0 (D0 device) in 100 ms cycles. Receives data for approximately 60 seconds.
DataReadWriteSample	Read the current value of B10 device and D10 device, and write to the module with adding "1" (bit device is inverted). Read the current value of the LABEL_B0 (B0 device) and LABEL_D0 ( D0 device), and write to the module with adding "1" (bit device is inverted).

## How to user the sample program

This section explains the configuration and procedures for operating the sample program.  
Set the environment variable to be needed to execute "javac", "java" on the command prompt.  
☞ Page 174 Setting the environment variable

### ■System Configuration for Operating the Sample Program

Connect the High Speed Data Communication Module to the configuration personal computer on a 1:1 basis via hub.  
Set the IP address of the configuration personal computer to "192.168.3.1".

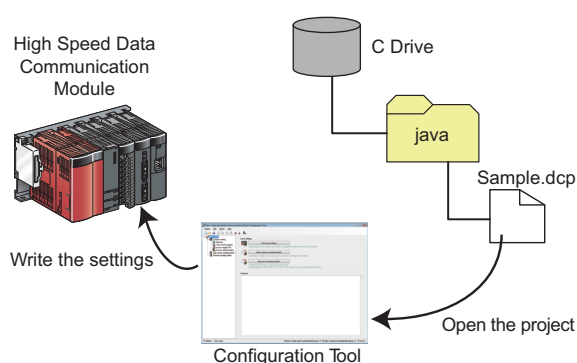
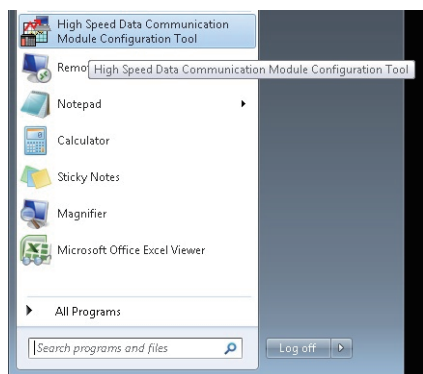
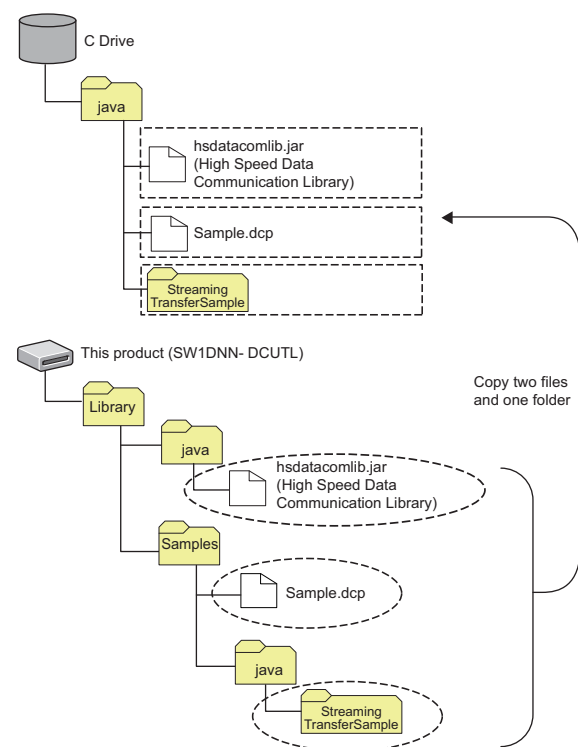


A

## ■Operating Procedures for Sample Program

The following explain the procedure to operate the sample program that executes the streaming transfer.

Install JDK to a personal computer to set up the environment to compile Java programs.



1. Copy the "hsdatacomlib.jar" folder, "Sample.dcp" folder, and "StreamingTransferSample" folder in the compressed "Library" file to the "C:\java".

Do not change the configuration of the folder under the "StreamingTransferSample" folder.

Since the folders are configured on the name of the sample project package, the compilation is failed if the folder configuration is changed.

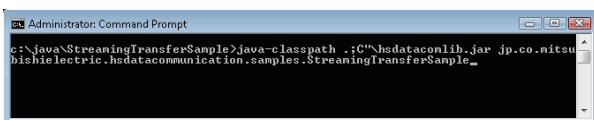
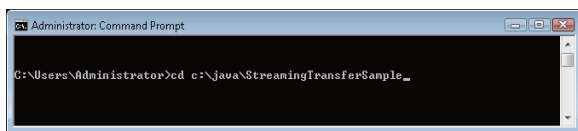
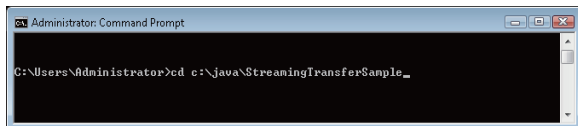
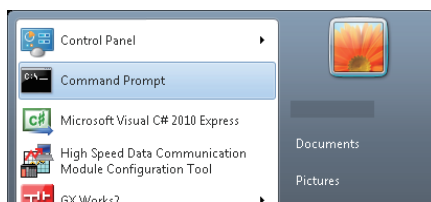
2. Start Configuration Tool.

(Example: Execute [Start] ⇒ [All Programs] ⇒ [MELSOFT Application] ⇒ [High Speed Data Communication] ⇒ [High Speed Data Communication Module Configuration Tool].)

3. Write the settings to the module.

Write the project file included in the sample program to be executed.

(Example: Open "C:\java\Sample.dcp" with Configuration Tool, and write to the module.)



#### 4. Start a command prompt.

Start a command prompt to compile the sample program.  
(Example: Execute [Start] ⇒ [All Programs] ⇒ [Accessory] ⇒ [Command Prompt].)

#### 5. Change the work folder.

Change the work folder for the command prompt to the folder to which the sample program is stored.  
(Example: Execute "cd c:\java\StreamingTransferSample" when the working folder at starting command prompt is "C:\Users\Administrator".)

#### 6. Compile the sample program.

Specify a High Speed Data Communication Library to "classpath", and compile the sample program.  
(Example: Execute `javac -classpath C:\java\hsdatacomlib.jar C:\java\StreamingTransferSample\jp.co.mitsubishielectric.hsdatacommunication.samples.StreamingTransferSample.java`.)

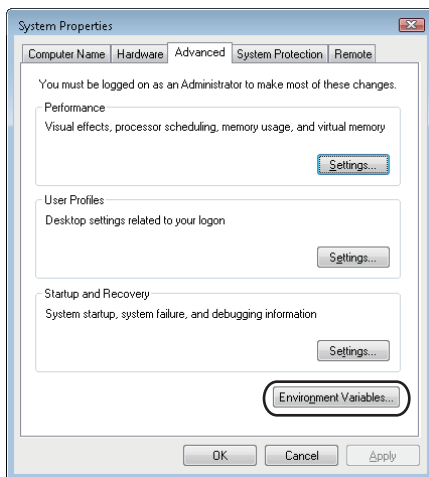
#### 7. Execute the sample program.

Specify a High Speed Data Communication Library to ClassPath, and execute the sample program.  
(Example: Execute `java -classpath .;C:\java\hsdatacomlib.jar jp.co.mitsubishielectric.hsdatacommunication.samples.StreamingTransferSample`.)  
Execute the command prompt in the same directory as the work folder changed in the step 6.

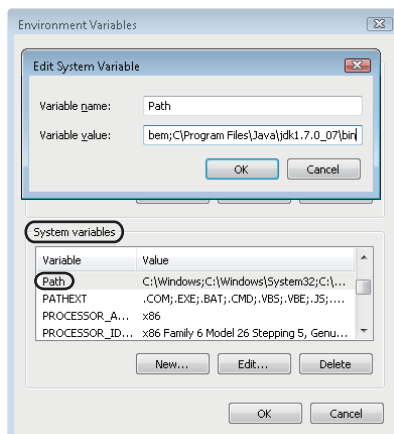
## ■Setting the environment variable

### Operating procedure

1. Install JDK to a personal computer downloaded from Oracle website.  
For the installation method, refer to the JDK installation guide on the ORACLE website.  
(Example: Install JDK 7 Update 7 to C:\Program Files\Java\jdk1.7.0\_07.)
2. Start up the control panel.
3. Select "System and Security".
4. Select "System".
5. Select "Advanced system settings".
6. Click the [Environment Variables] button on the "System Property" screen.



7. Select "Path" from "User variables for User", and click the [New] button.  
If the "Path" is not exist, click the [New] button to create the Path.
8. Add "JDK installed folder\bin" to "Value".  
If the value is already entered, enter ";" to delimit them.  
(Example: Add ";C:\Program Files\Java\jdk1.7.0\_07\bin." after the entered value.)



## ■Source code of the sample program

The sample program of the streaming transfer function is stored in the "Java" folder under "Library" - "Samples" in the compressed file of this product.



# MEMO

---

A

# INDEX

---

## B

---

Build . . . . . 85

## C

---

Class . . . . . 33,99

Compiling . . . . . 158

## D

---

Data reading . . . . . 19

Data writing . . . . . 19

Development environment . . . . . 22,90

Device specification . . . . . 29,30,96,97

## E

---

Receiving streaming transfer data . . . . . 92

Enumerator . . . . . 82,155

Executing C# user program . . . . . 87

Executing Java user program . . . . . 159

## L

---

Label specification . . . . . 27,28,94,95

## R

---

Receiving notifications of module operation status  
changes . . . . . 26,93

Receiving streaming transfer data . . . . . 25

## S

---

Streaming transfer . . . . . 18,24,91

## U

---

Using C# class library . . . . . 23

using Directive . . . . . 23

Using Java class library . . . . . 90

# MEMO

---

# REVISIONS

\*The manual number is written at the bottom left of the back cover.

Revision date	*Manual number	Revision
April 2013	SH(NA)-081163ENG-A	First edition
April 2014	SH(NA)-081163ENG-B	Windows® 8.1 is supported.
March 2016	SH(NA)-081163ENG-C	Windows® 10 is supported.

Japanese manual number: SH-081161-C

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

© 2013 MITSUBISHI ELECTRIC CORPORATION

# WARRANTY

---

Please confirm the following product warranty details before using this product.

## **1. Gratis Warranty Term and Gratis Warranty Range**

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

- (1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
- (2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
  1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
  2. Failure caused by unapproved modifications, etc., to the product by the user.
  3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
  4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
  5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
  6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
  7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

## **2. Onerous repair term after discontinuation of production**

- (1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
- (2) Product supply (including repair parts) is not available after production is discontinued.

## **3. Overseas service**

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

## **4. Exclusion of loss in opportunity and secondary loss from warranty liability**

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

- (1) Damages caused by any cause found not to be the responsibility of Mitsubishi.
- (2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.
- (3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.
- (4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

## **5. Changes in product specifications**

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

# TRADEMARKS

---

CompactFlash is either a registered trademark or a trademark of SanDisk Corporation.

Ethernet is a registered trademark of Fuji Xerox Corporation in Japan.

Microsoft, Visual C#, Visual Studio, Windows, Windows Vista, and Windows XP are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.

In some cases, trademark symbols such as <sup>™</sup> or <sup>®</sup> are not specified in this manual.



SH(NA)-081163ENG-C(1603)KWIX

MODEL: QJ71DC96-P-E

MODEL CODE: 13JC26

## **mitsubishi electric corporation**

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN  
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN

When exported from Japan, this manual does not require application to the  
Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.