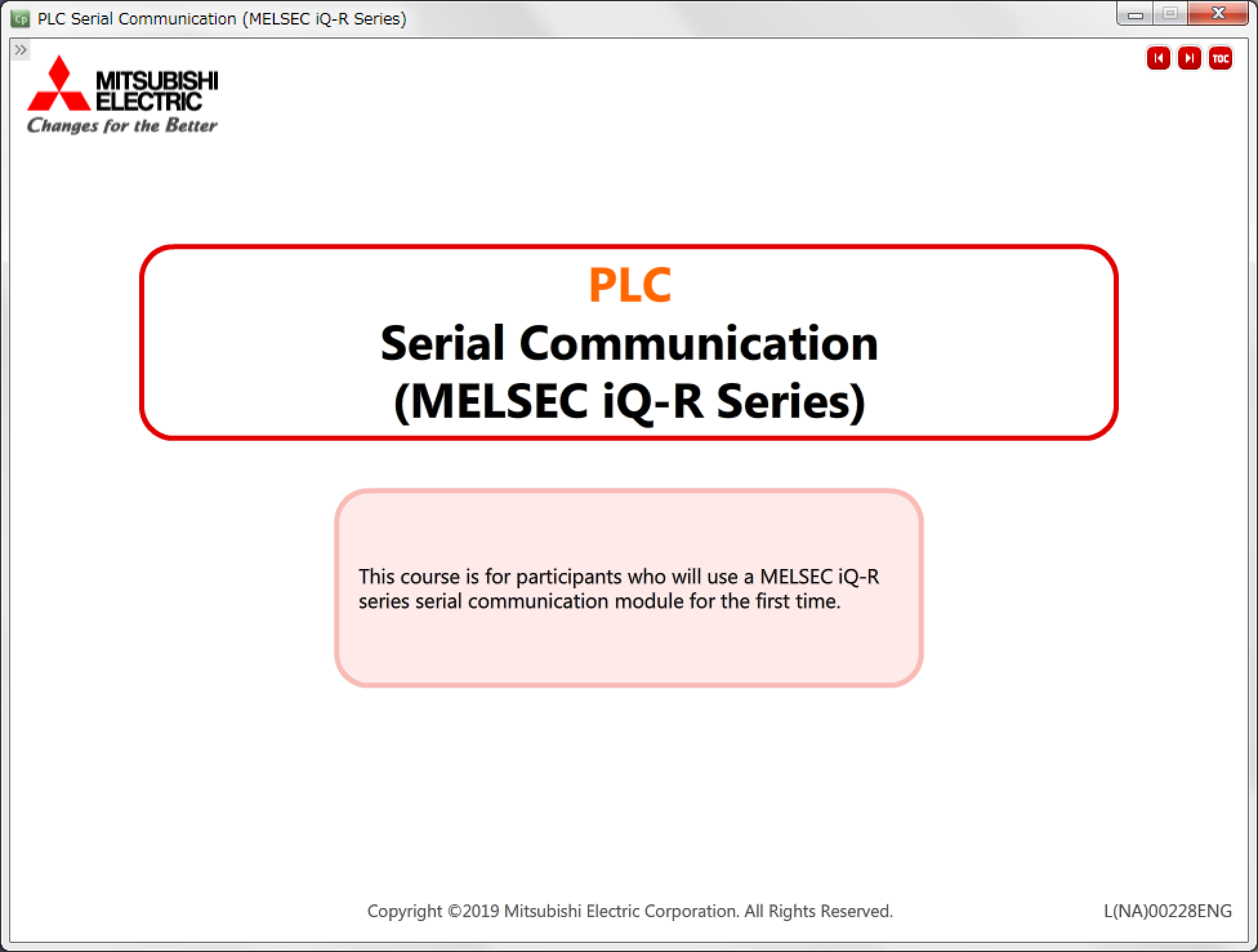
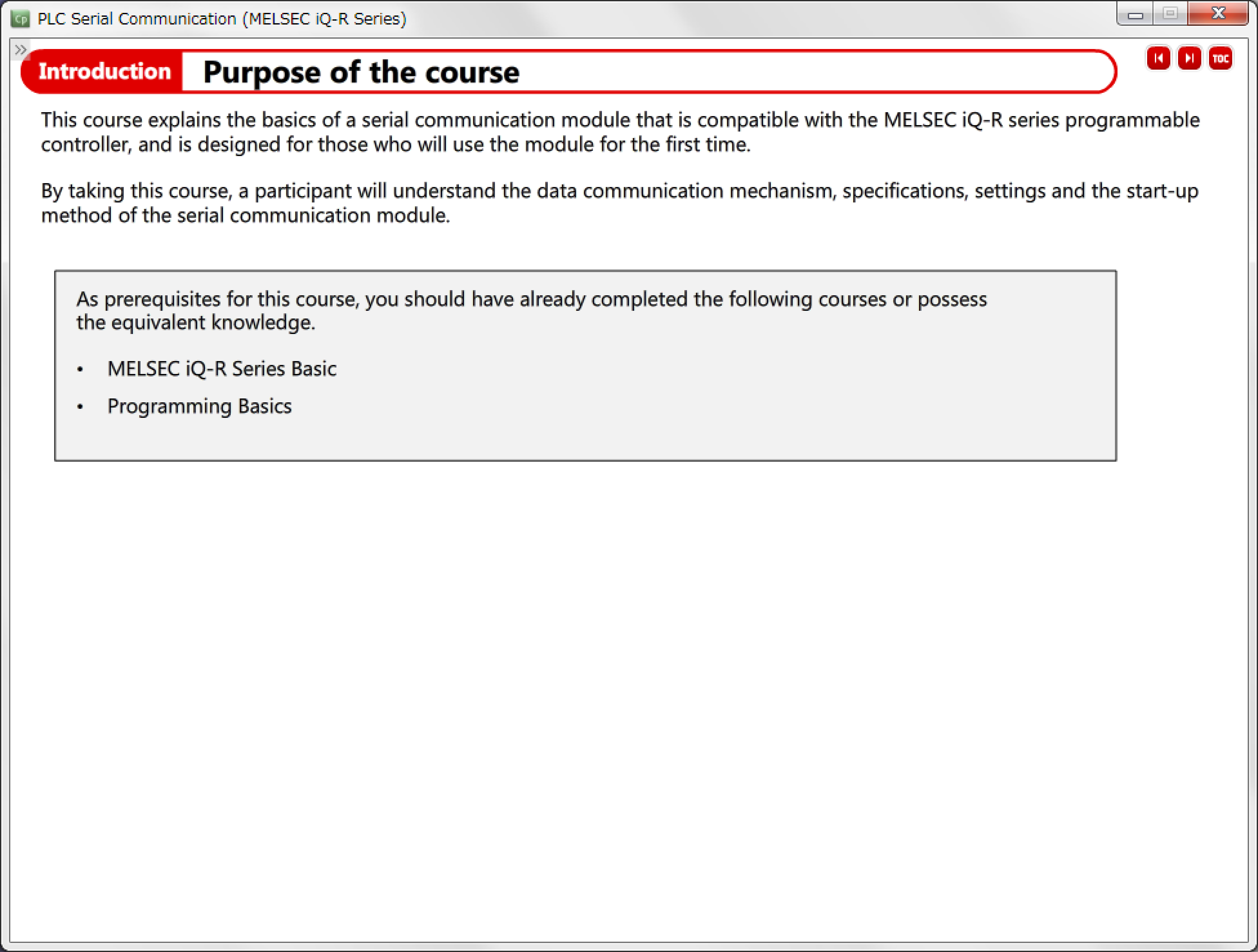


MITSUBISHI ELECTRIC

Changes for the Better

PLC
Serial Communication (MELSEC iQ-R Series)

This course is for participants who will use a MELSEC iQ-R series serial communication module for the first time.



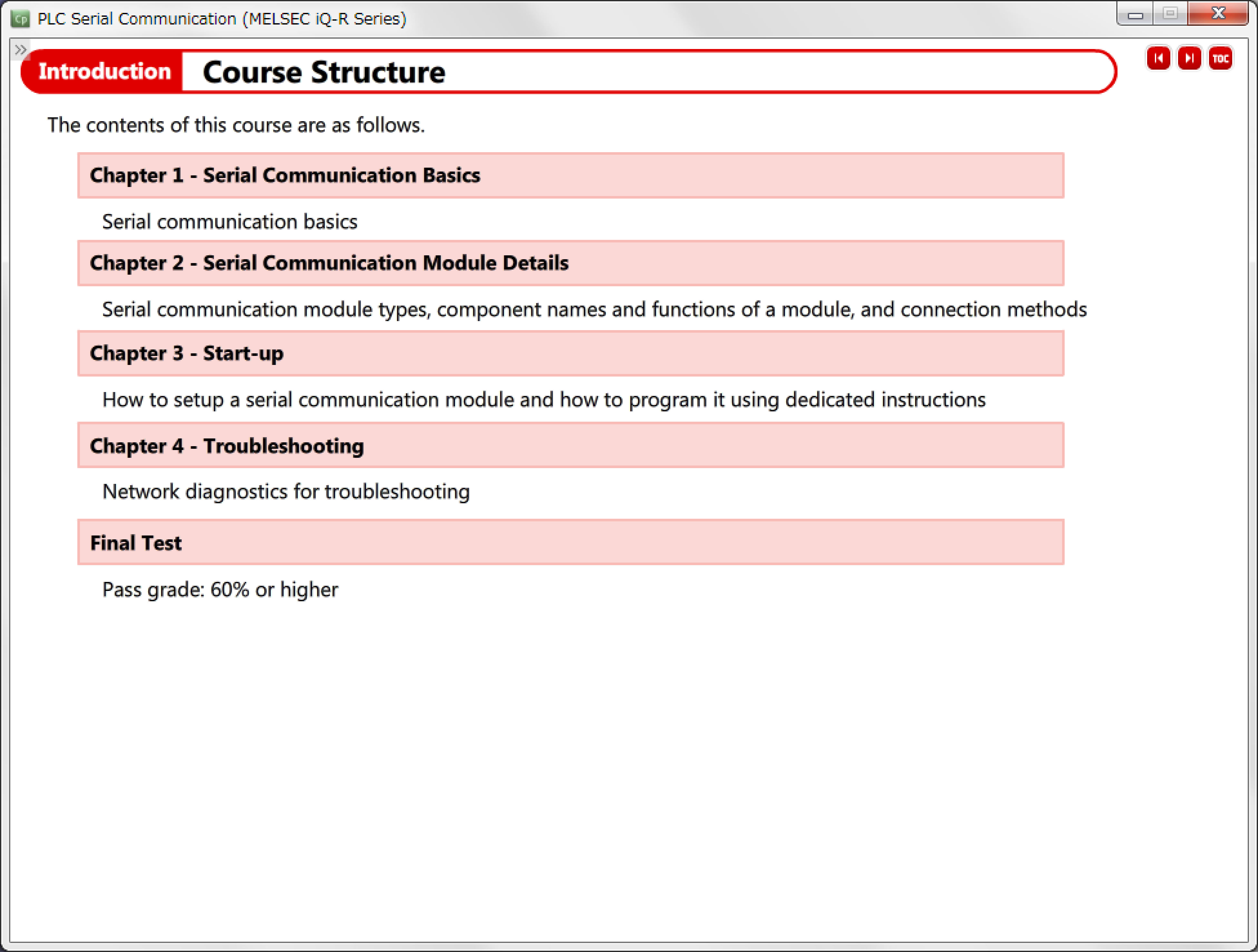
Introduction Purpose of the course

This course explains the basics of a serial communication module that is compatible with the MELSEC iQ-R series programmable controller, and is designed for those who will use the module for the first time.

By taking this course, a participant will understand the data communication mechanism, specifications, settings and the start-up method of the serial communication module.

As prerequisites for this course, you should have already completed the following courses or possess the equivalent knowledge.

- MELSEC iQ-R Series Basic
- Programming Basics



Introduction Course Structure

The contents of this course are as follows.

Chapter 1 - Serial Communication Basics

Serial communication basics

Chapter 2 - Serial Communication Module Details

Serial communication module types, component names and functions of a module, and connection methods

Chapter 3 - Start-up

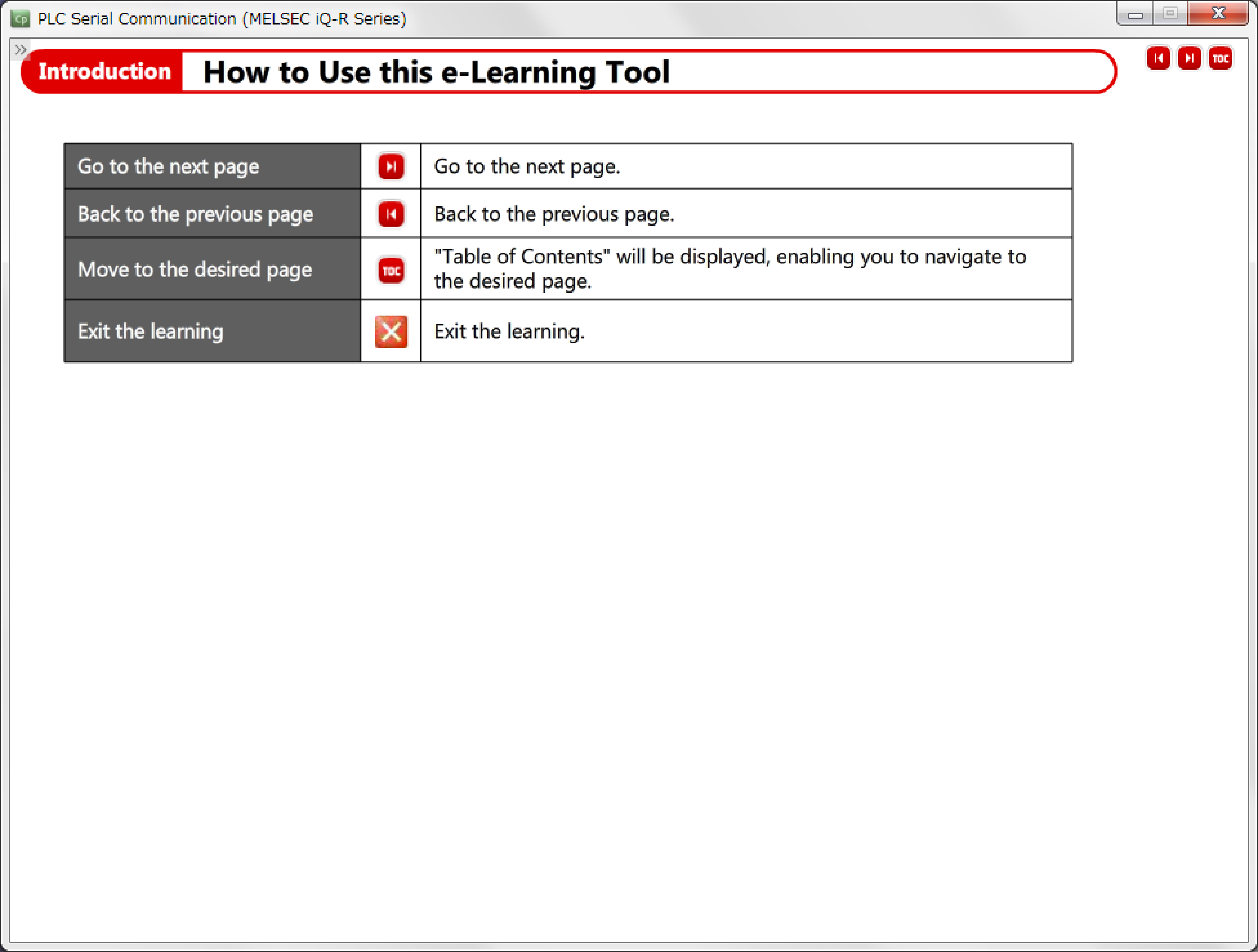
How to setup a serial communication module and how to program it using dedicated instructions

Chapter 4 - Troubleshooting

Network diagnostics for troubleshooting

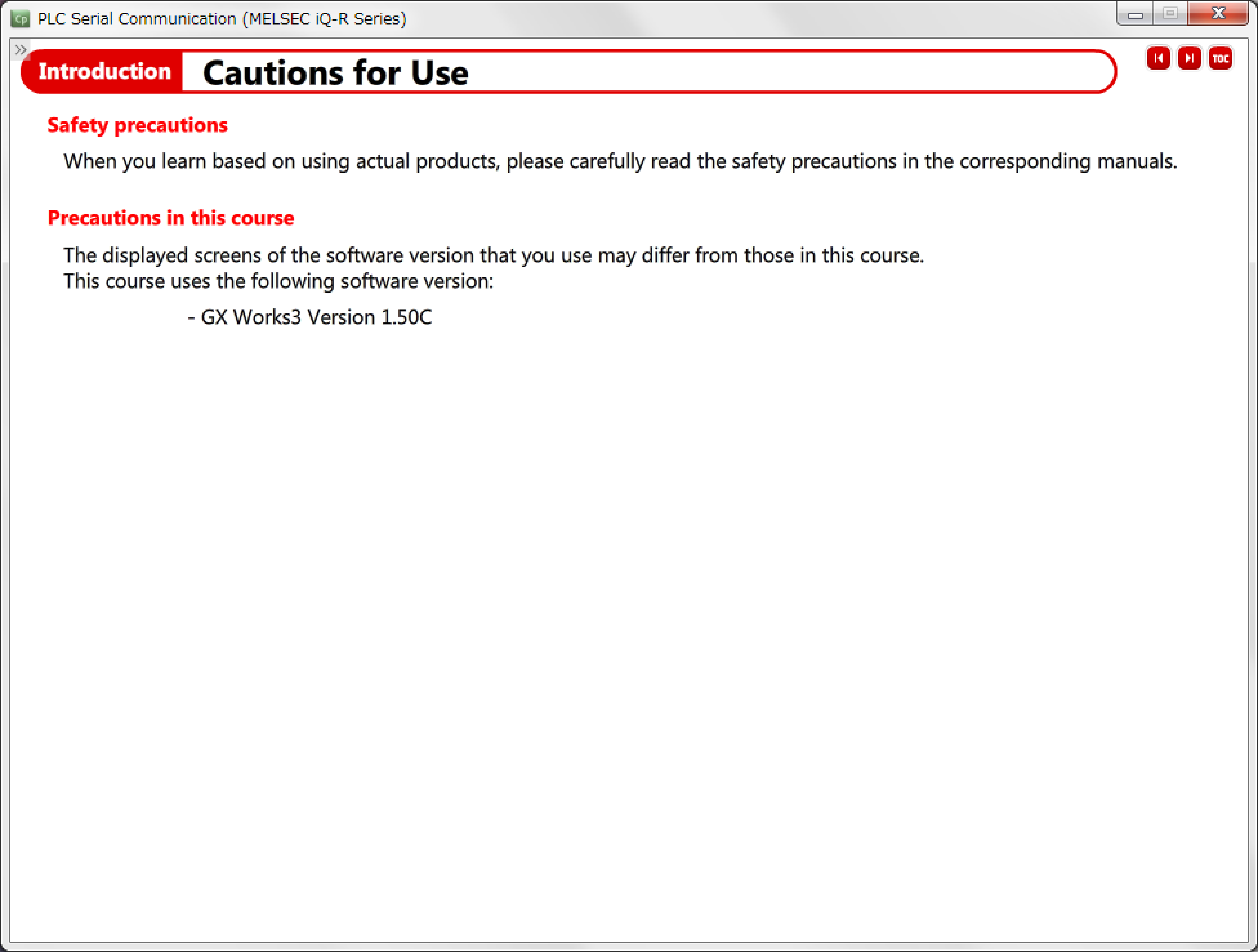
Final Test

Pass grade: 60% or higher



Introduction How to Use this e-Learning Tool

Go to the next page	⏭	Go to the next page.
Back to the previous page	⏮	Back to the previous page.
Move to the desired page	TOC	"Table of Contents" will be displayed, enabling you to navigate to the desired page.
Exit the learning	✕	Exit the learning.



Introduction

Cautions for Use

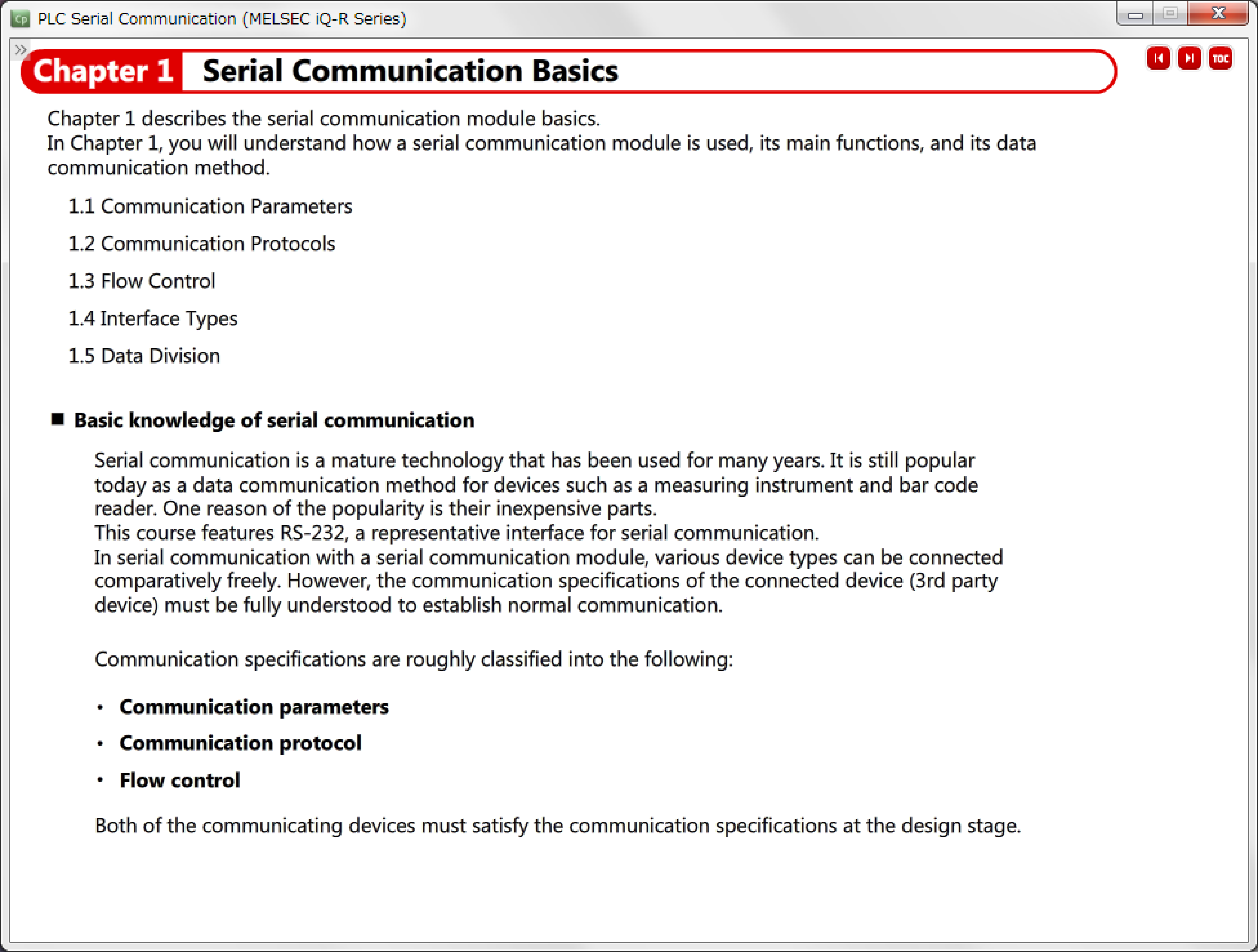
Safety precautions

When you learn based on using actual products, please carefully read the safety precautions in the corresponding manuals.

Precautions in this course

The displayed screens of the software version that you use may differ from those in this course.
This course uses the following software version:

- GX Works3 Version 1.50C



Chapter 1 Serial Communication Basics

Chapter 1 describes the serial communication module basics.
In Chapter 1, you will understand how a serial communication module is used, its main functions, and its data communication method.

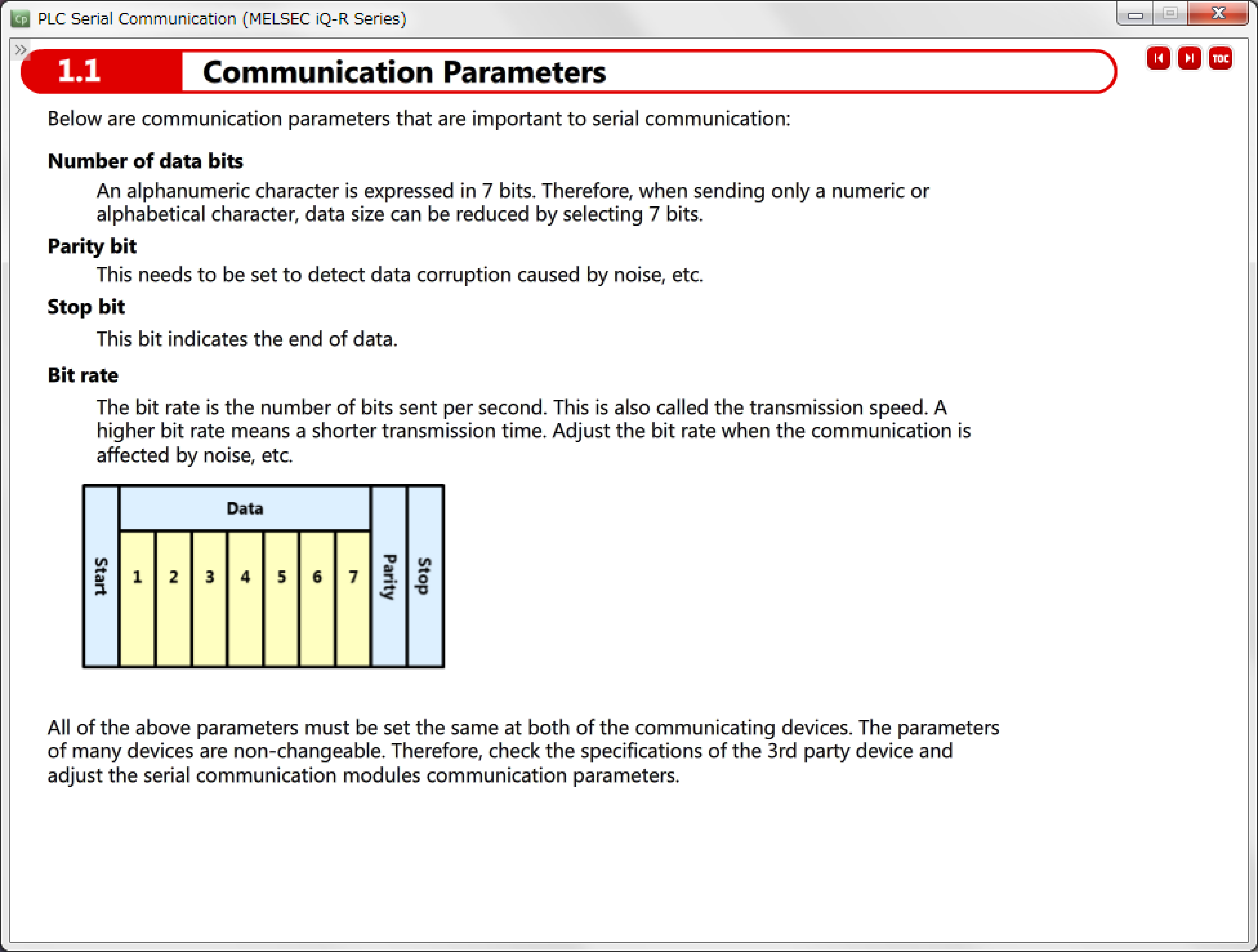
■ Basic knowledge of serial communication

Serial communication is a mature technology that has been used for many years. It is still popular today as a data communication method for devices such as a measuring instrument and bar code reader. One reason of the popularity is their inexpensive parts.
This course features RS-232, a representative interface for serial communication.
In serial communication with a serial communication module, various device types can be connected comparatively freely. However, the communication specifications of the connected device (3rd party device) must be fully understood to establish normal communication.

Communication specifications are roughly classified into the following:

- **Communication parameters**
- **Communication protocol**
- **Flow control**

Both of the communicating devices must satisfy the communication specifications at the design stage.



1.1 Communication Parameters

Below are communication parameters that are important to serial communication:

Number of data bits

An alphanumeric character is expressed in 7 bits. Therefore, when sending only a numeric or alphabetical character, data size can be reduced by selecting 7 bits.

Parity bit

This needs to be set to detect data corruption caused by noise, etc.

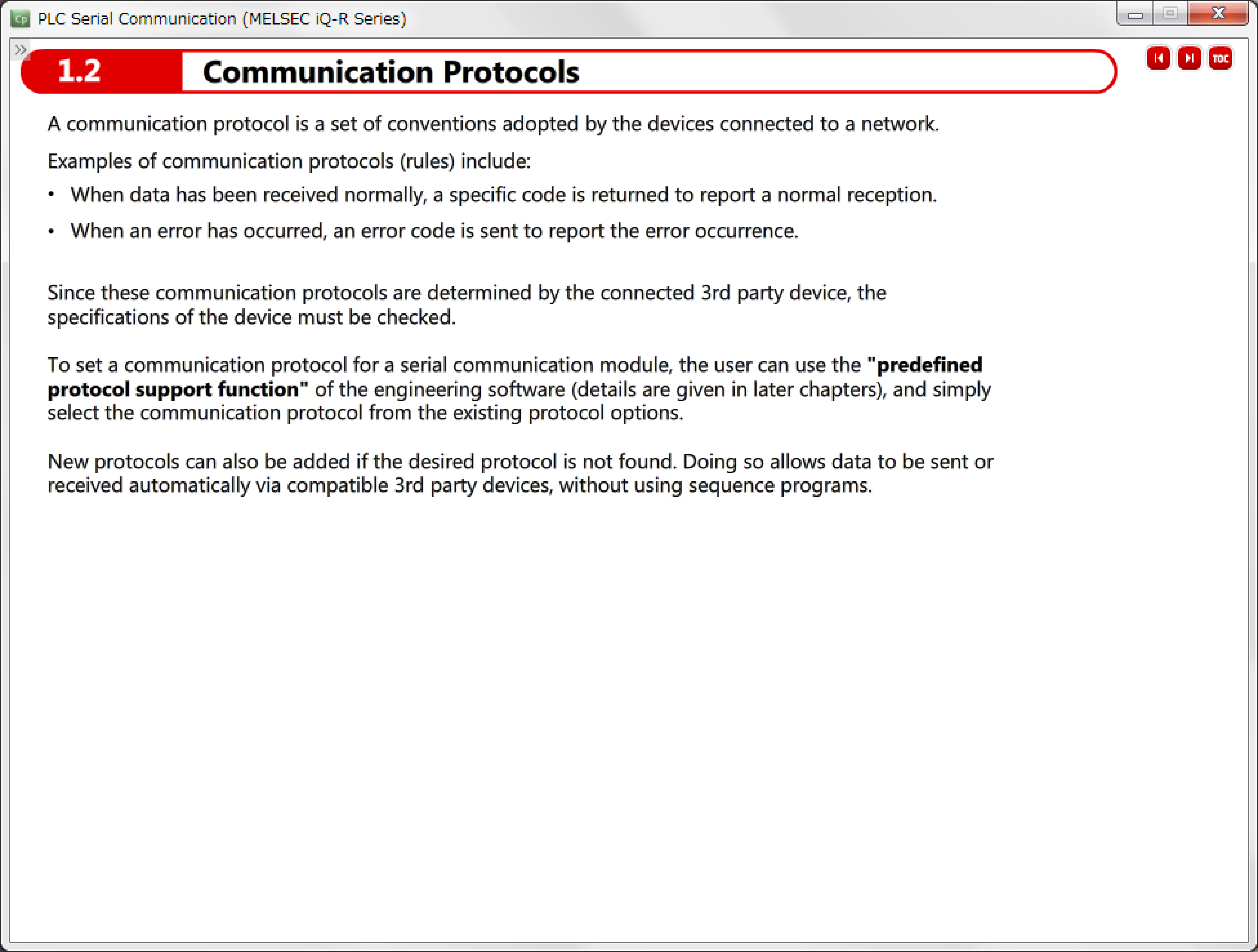
Stop bit

This bit indicates the end of data.

Bit rate

The bit rate is the number of bits sent per second. This is also called the transmission speed. A higher bit rate means a shorter transmission time. Adjust the bit rate when the communication is affected by noise, etc.

All of the above parameters must be set the same at both of the communicating devices. The parameters of many devices are non-changeable. Therefore, check the specifications of the 3rd party device and adjust the serial communication modules communication parameters.



1.2 Communication Protocols

A communication protocol is a set of conventions adopted by the devices connected to a network.

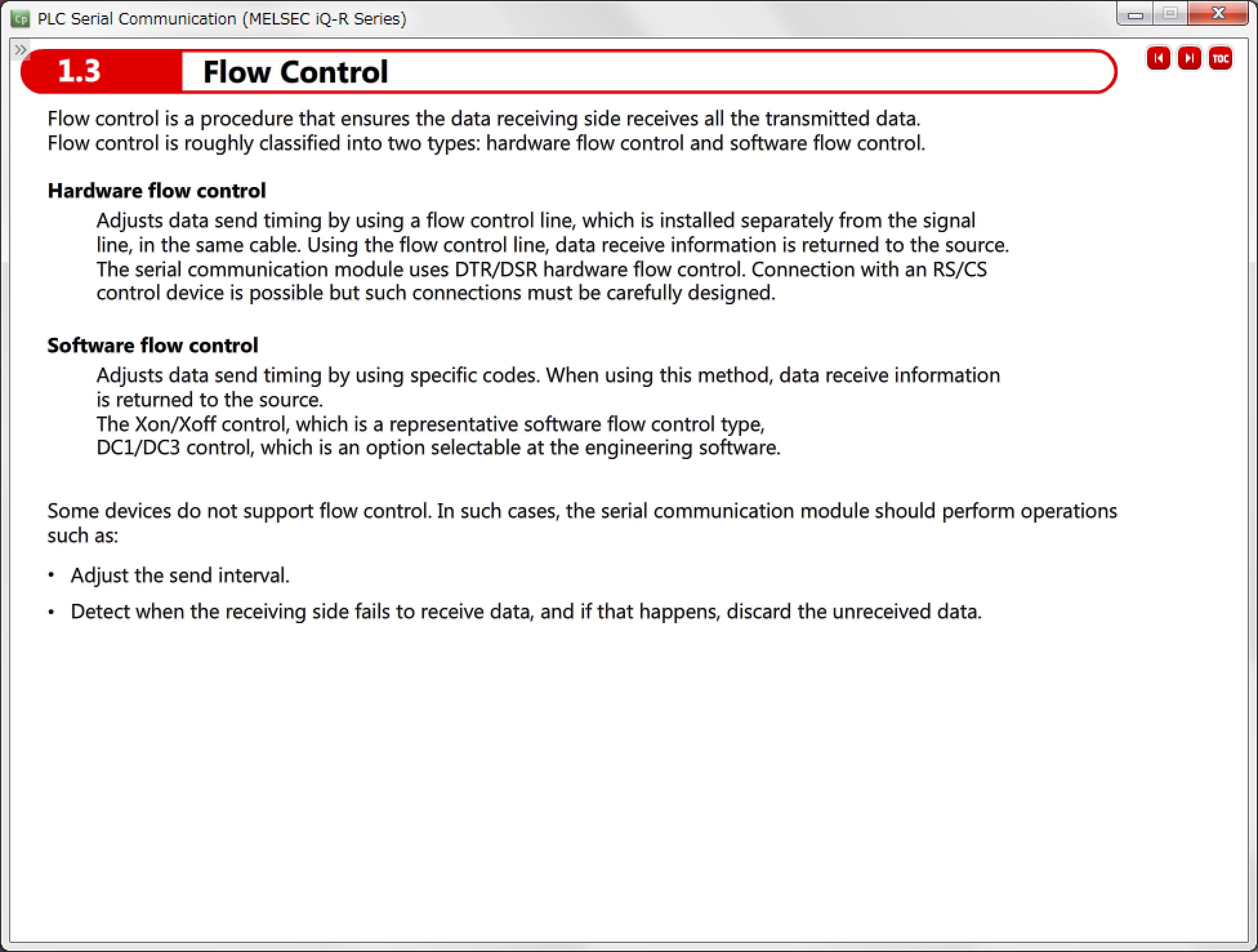
Examples of communication protocols (rules) include:

- When data has been received normally, a specific code is returned to report a normal reception.
- When an error has occurred, an error code is sent to report the error occurrence.

Since these communication protocols are determined by the connected 3rd party device, the specifications of the device must be checked.

To set a communication protocol for a serial communication module, the user can use the **"predefined protocol support function"** of the engineering software (details are given in later chapters), and simply select the communication protocol from the existing protocol options.

New protocols can also be added if the desired protocol is not found. Doing so allows data to be sent or received automatically via compatible 3rd party devices, without using sequence programs.



1.3 Flow Control

Flow control is a procedure that ensures the data receiving side receives all the transmitted data.
Flow control is roughly classified into two types: hardware flow control and software flow control.

Hardware flow control

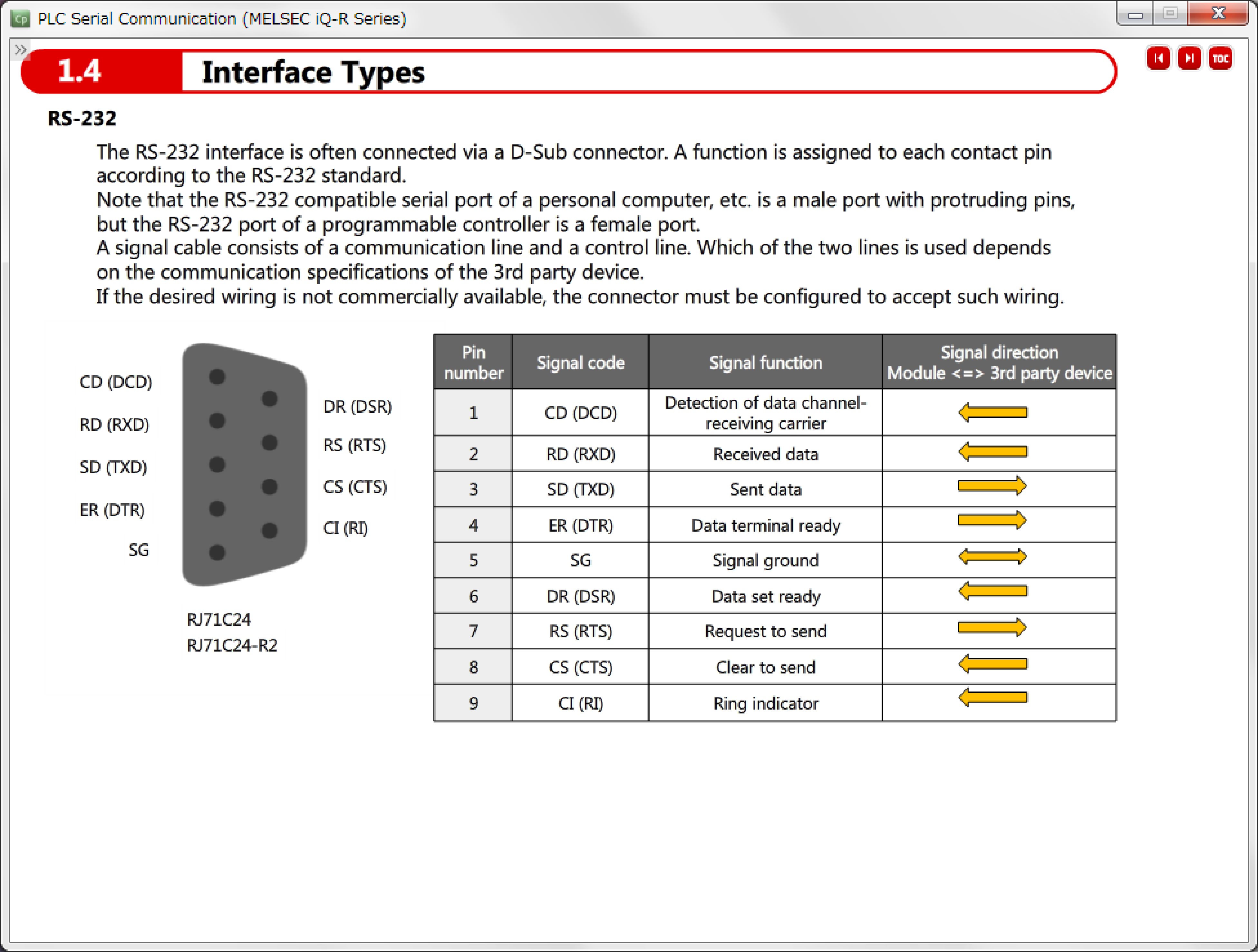
Adjusts data send timing by using a flow control line, which is installed separately from the signal line, in the same cable. Using the flow control line, data receive information is returned to the source.
The serial communication module uses DTR/DSR hardware flow control. Connection with an RS/CS control device is possible but such connections must be carefully designed.

Software flow control

Adjusts data send timing by using specific codes. When using this method, data receive information is returned to the source.
The Xon/Xoff control, which is a representative software flow control type,
DC1/DC3 control, which is an option selectable at the engineering software.

Some devices do not support flow control. In such cases, the serial communication module should perform operations such as:

- Adjust the send interval.
- Detect when the receiving side fails to receive data, and if that happens, discard the unreceived data.



PLC Serial Communication (MELSEC iQ-R Series)

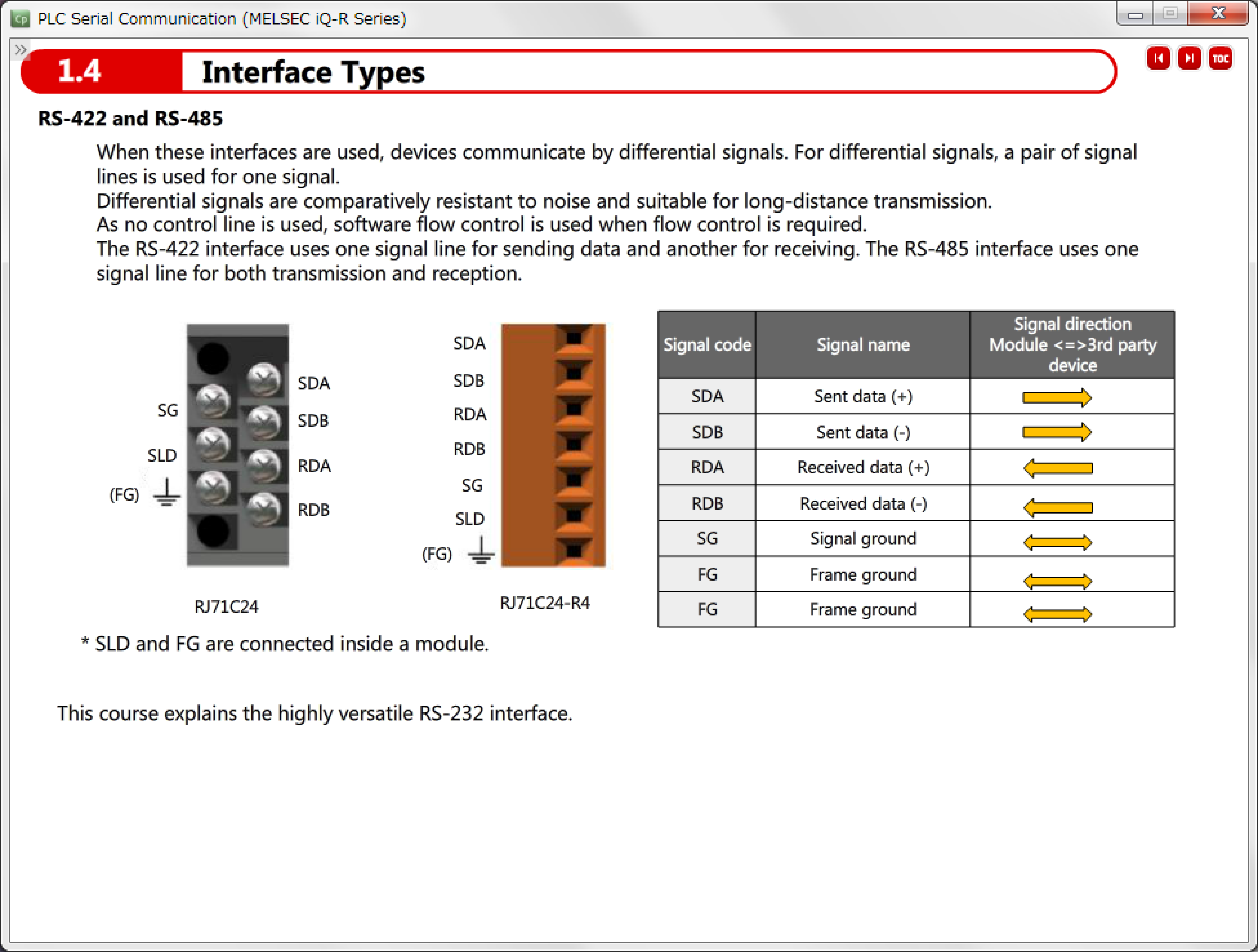
1.4 Interface Types

RS-232

The RS-232 interface is often connected via a D-Sub connector. A function is assigned to each contact pin according to the RS-232 standard.
Note that the RS-232 compatible serial port of a personal computer, etc. is a male port with protruding pins, but the RS-232 port of a programmable controller is a female port.
A signal cable consists of a communication line and a control line. Which of the two lines is used depends on the communication specifications of the 3rd party device.
If the desired wiring is not commercially available, the connector must be configured to accept such wiring.

RJ71C24
RJ71C24-R2

Pin number	Signal code	Signal function	Signal direction Module <=> 3rd party device
1	CD (DCD)	Detection of data channel-receiving carrier	←
2	RD (RXD)	Received data	←
3	SD (TXD)	Sent data	→
4	ER (DTR)	Data terminal ready	→
5	SG	Signal ground	↔
6	DR (DSR)	Data set ready	←
7	RS (RTS)	Request to send	→
8	CS (CTS)	Clear to send	←
9	CI (RI)	Ring indicator	←



1.4 Interface Types

RS-422 and RS-485

When these interfaces are used, devices communicate by differential signals. For differential signals, a pair of signal lines is used for one signal.
Differential signals are comparatively resistant to noise and suitable for long-distance transmission.
As no control line is used, software flow control is used when flow control is required.
The RS-422 interface uses one signal line for sending data and another for receiving. The RS-485 interface uses one signal line for both transmission and reception.

RJ71C24

RJ71C24-R4

Signal code	Signal name	Signal direction Module <=>3rd party device
SDA	Sent data (+)	→
SDB	Sent data (-)	→
RDA	Received data (+)	←
RDB	Received data (-)	←
SG	Signal ground	↔
FG	Frame ground	↔
FG	Frame ground	↔

* SLD and FG are connected inside a module.

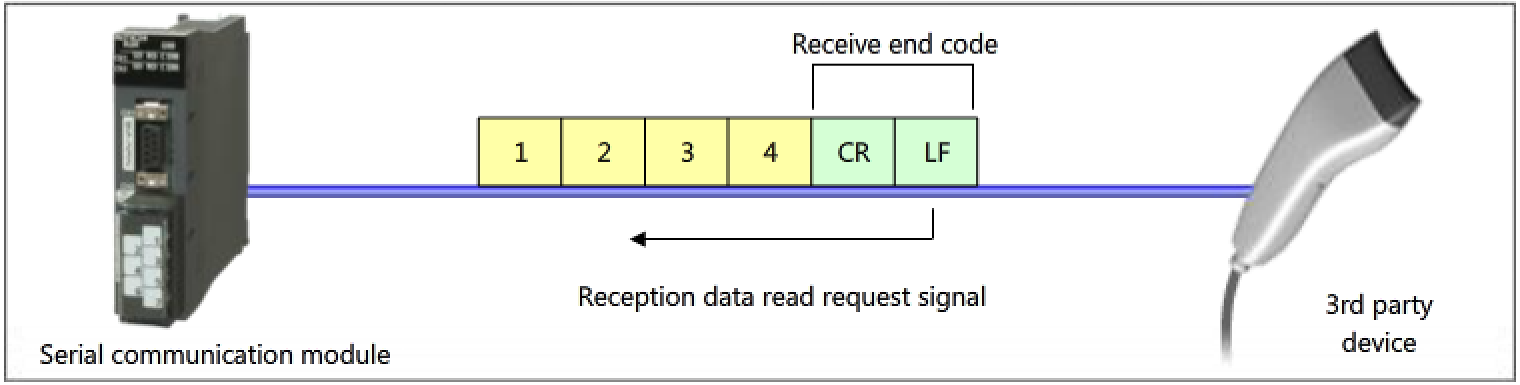
This course explains the highly versatile RS-232 interface.

1.5 Data Division

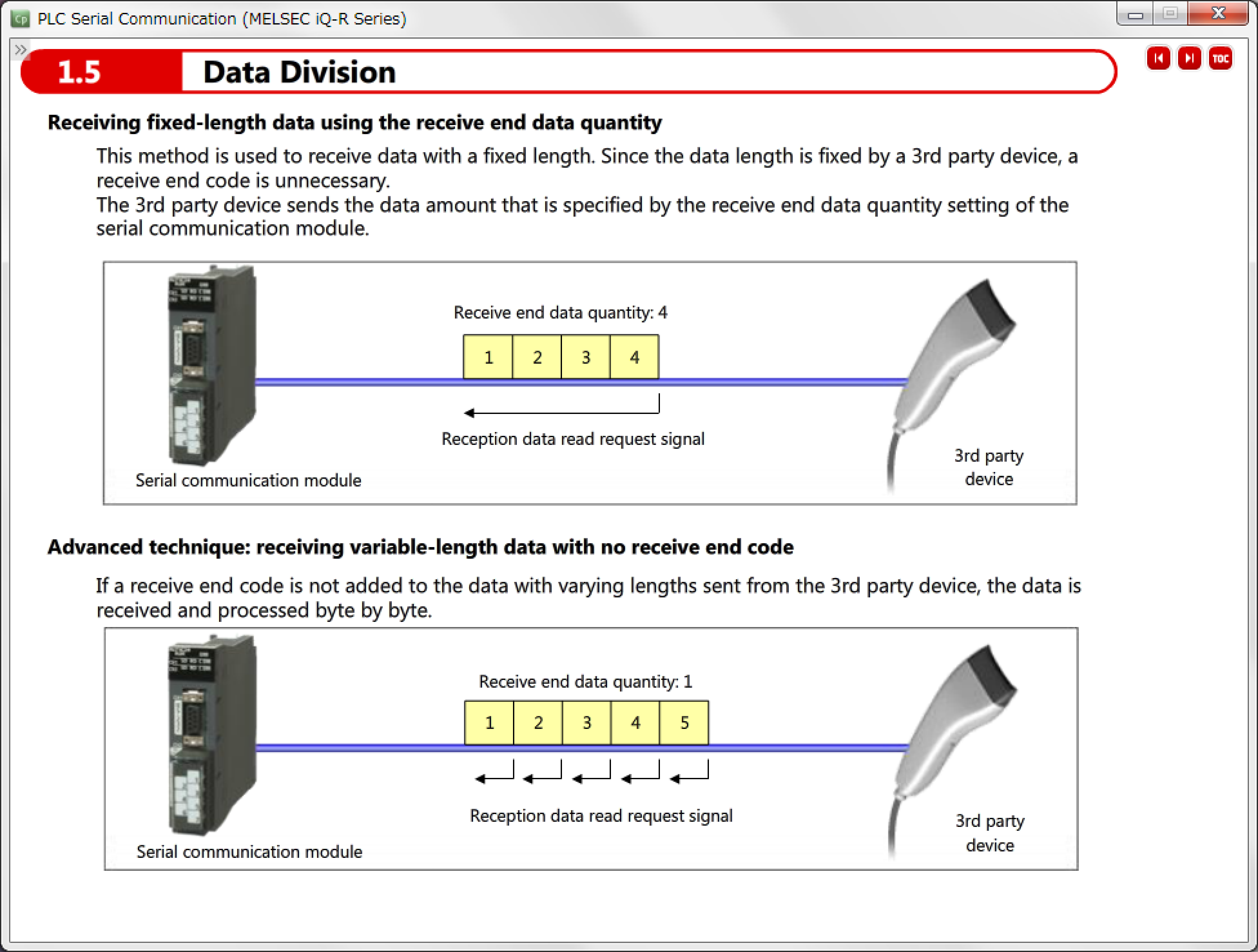
When data is received, it is usually divided into parts of a certain length.
There are two data division methods: division by the number of data and division by a receive end code.
Each method depends on the communication specifications of the 3rd party device, therefore make sure to confirm the specifications.
If necessary, a receive end code and the receive end data quantity can be changed from their default settings.

Receiving variable-length data using a receive end code

This method is used to receive data with varying lengths from a 3rd party device. Before data is sent from the 3rd party device, a receive end code (CR+LF or one-byte data), which is specified by the serial communication module, is added to the end of the message.


Receive end code
1
2
3
4
CR
LF
Reception data read request signal
Serial communication module
3rd party device

This course explains **how the system explained in this course receives data using a receive end code**.



1.5 Data Division

Receiving fixed-length data using the receive end data quantity

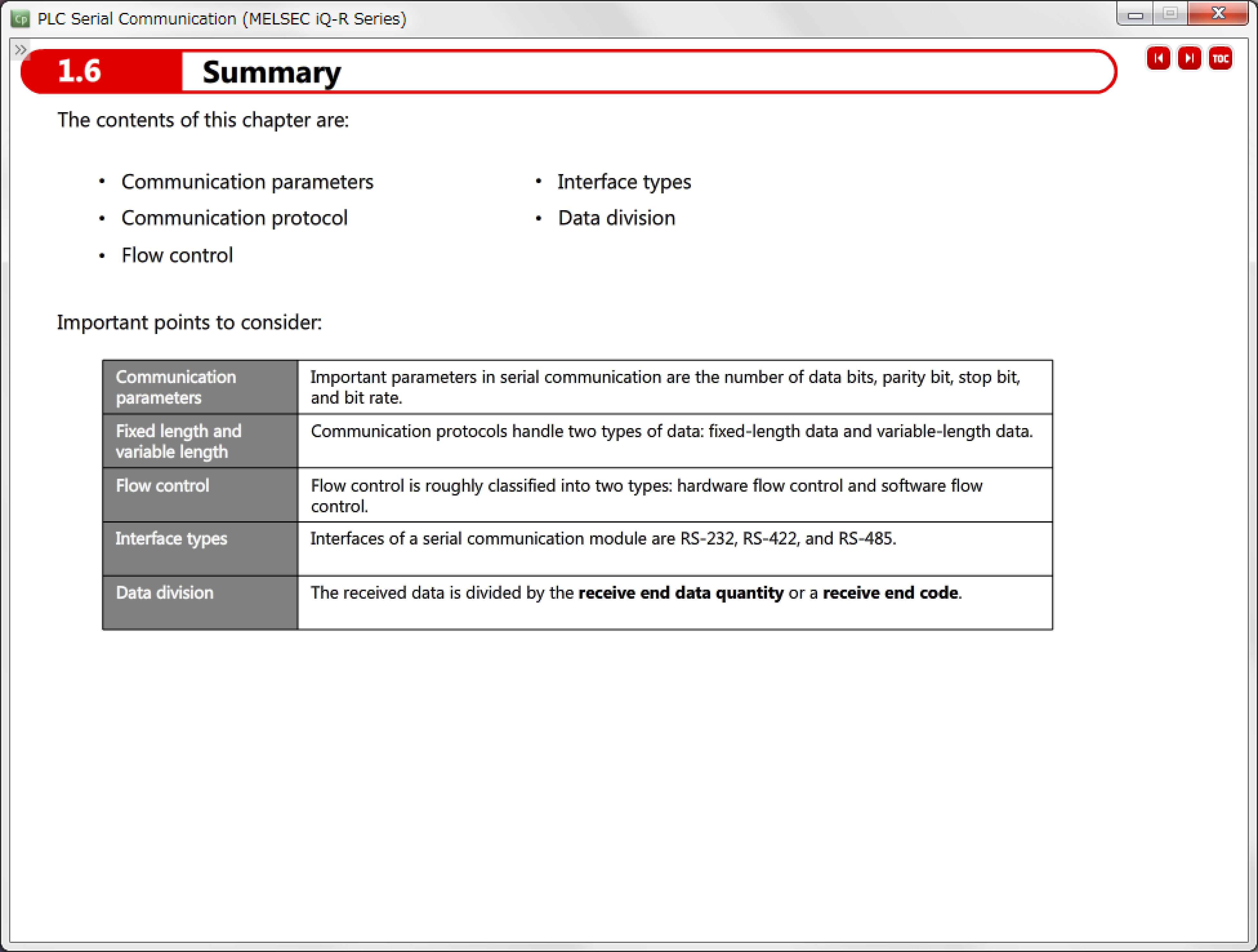
This method is used to receive data with a fixed length. Since the data length is fixed by a 3rd party device, a receive end code is unnecessary.
The 3rd party device sends the data amount that is specified by the receive end data quantity setting of the serial communication module.

Receive end data quantity: 4
1
2
3
4
Reception data read request signal
Serial communication module
3rd party device

Advanced technique: receiving variable-length data with no receive end code

If a receive end code is not added to the data with varying lengths sent from the 3rd party device, the data is received and processed byte by byte.

Receive end data quantity: 1
1
2
3
4
5
Reception data read request signal
Serial communication module
3rd party device



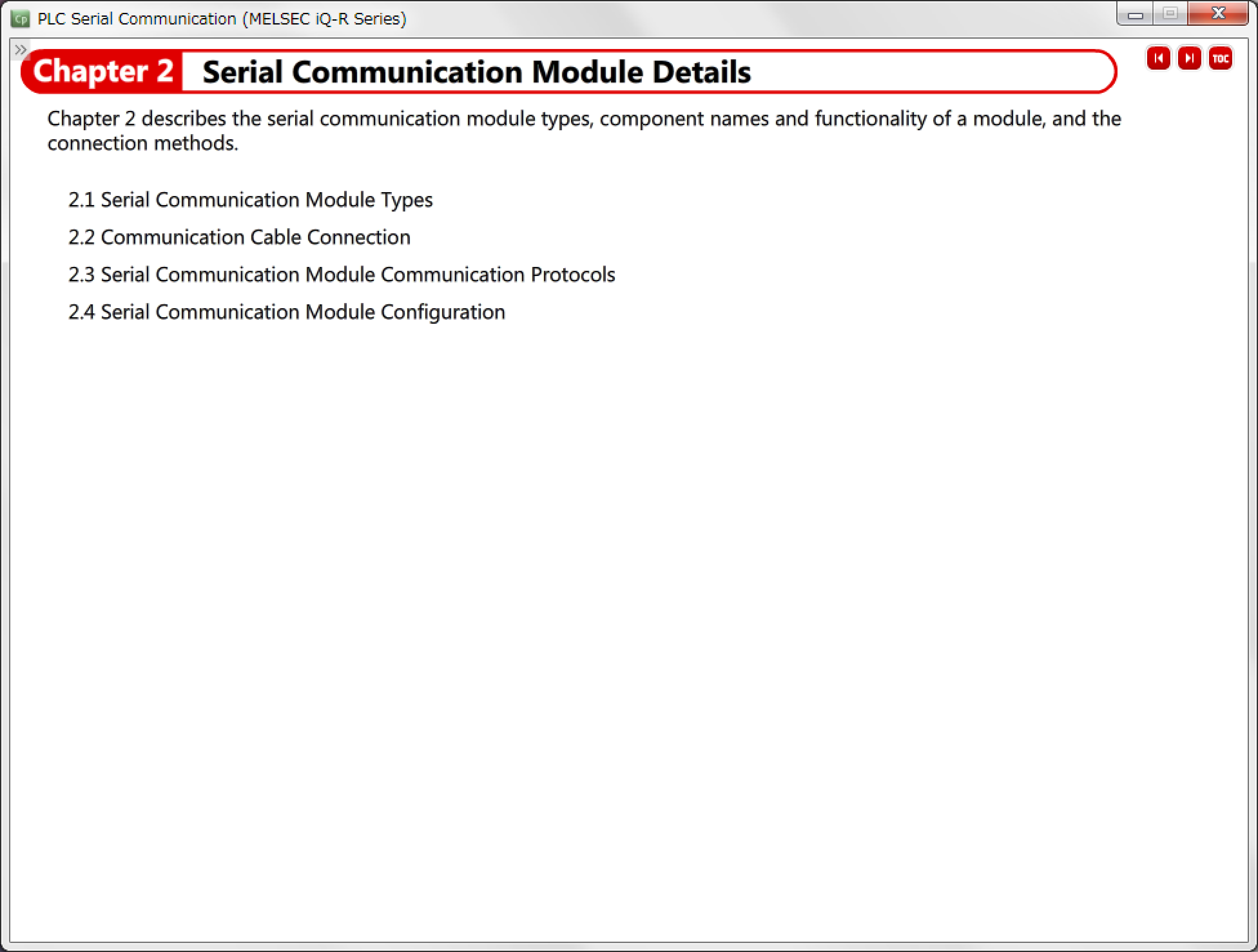
1.6 Summary

The contents of this chapter are:

- Communication parameters
- Communication protocol
- Flow control
- Interface types
- Data division

Important points to consider:

Communication parameters	Important parameters in serial communication are the number of data bits, parity bit, stop bit, and bit rate.
Fixed length and variable length	Communication protocols handle two types of data: fixed-length data and variable-length data.
Flow control	Flow control is roughly classified into two types: hardware flow control and software flow control.
Interface types	Interfaces of a serial communication module are RS-232, RS-422, and RS-485.
Data division	The received data is divided by the **receive end data quantity** or a **receive end code**.



Chapter 2 Serial Communication Module Details

Chapter 2 describes the serial communication module types, component names and functionality of a module, and the connection methods.

2.1 Serial Communication Module Types

This section describes the serial communication module types, the component names of a module, and its LED indicators.

Serial communication module

A serial communication module is an intelligent function module. A serial communication module connects an external device such as a measuring instrument and bar code reader, to a MELSEC iQ-R series CPU module through its RS-232 or RS-422/485 interface, which are typical serial communication interfaces, to enable data communication between the connected devices.
Each module provides two communication channels that can be simultaneously used. Three module types, with different combinations of interfaces, are available.

RJ71C24

RS-232: 1 channel
RS-422/485: 1 channel

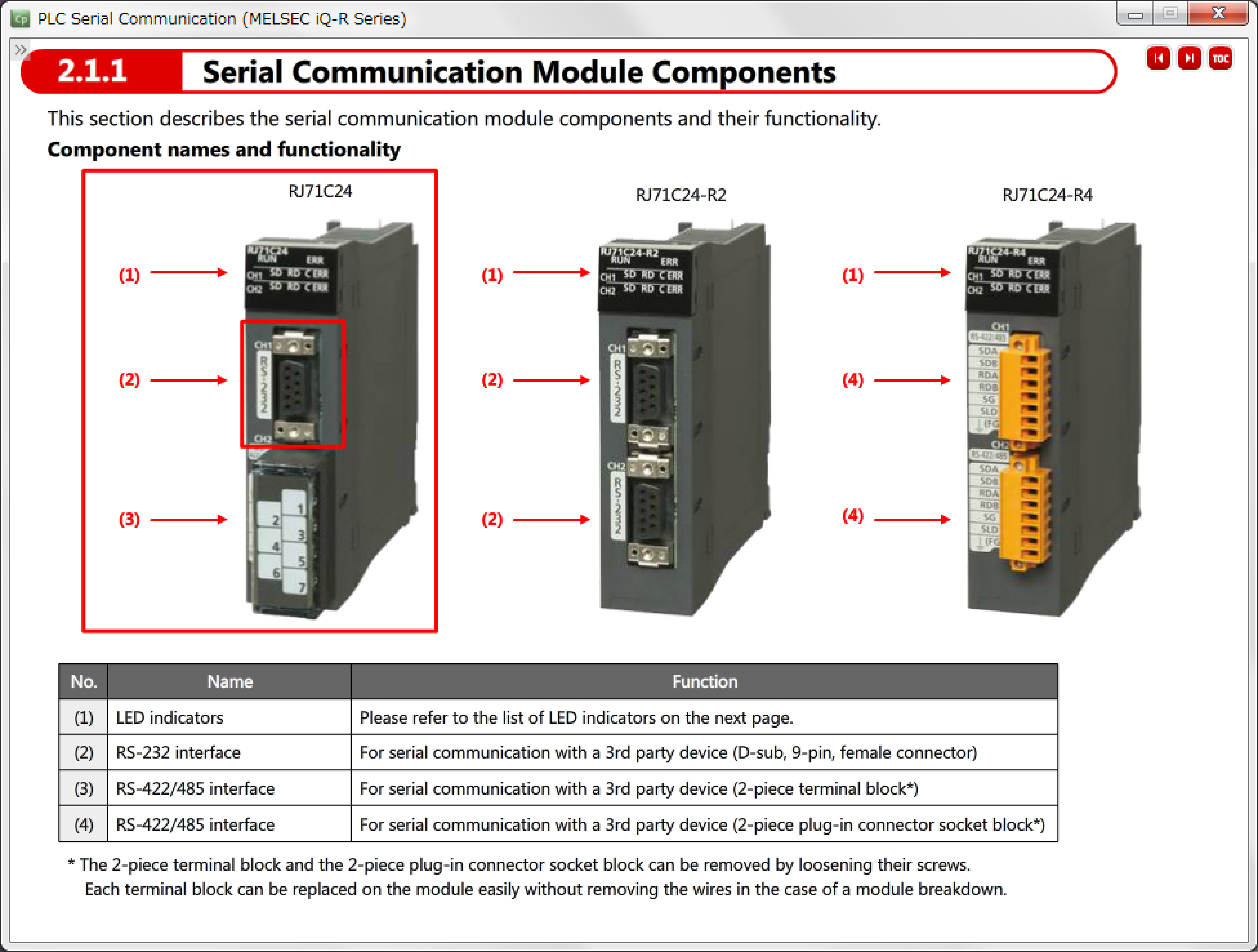
RJ71C24-R2

RS-232: 2 channels

RJ71C24-R4

RS-422/485: 2 channels

This course uses the RJ71C24 single channel RS-232 interface as an example.



2.1.1 Serial Communication Module Components

This section describes the serial communication module components and their functionality.

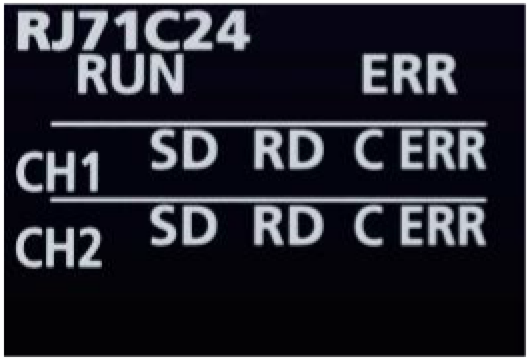
Component names and functionality

No.	Name	Function
(1)	LED indicators	Please refer to the list of LED indicators on the next page.
(2)	RS-232 interface	For serial communication with a 3rd party device (D-sub, 9-pin, female connector)
(3)	RS-422/485 interface	For serial communication with a 3rd party device (2-piece terminal block*)
(4)	RS-422/485 interface	For serial communication with a 3rd party device (2-piece plug-in connector socket block*)

* The 2-piece terminal block and the 2-piece plug-in connector socket block can be removed by loosening their screws.
Each terminal block can be replaced on the module easily without removing the wires in the case of a module breakdown.

2.1.2 LED Indicators and their Functions

This section describes the functionality of the LED indicators that are on a serial communication module.



LED indicators

CH	LED indicator name	Function	Description		
			On	Flashing	Off
-	RUN	Operating status	Normal	-	Major error
	ERR	Error status of a module	Hardware or data communication error	Parameter error	Normal
CH1/2	SD	Data send status	Sending data		Not sending data
	RD	Data reception status	Receiving data		Not receiving data
	C ERR	Communication error status	Communication error	-	Normal

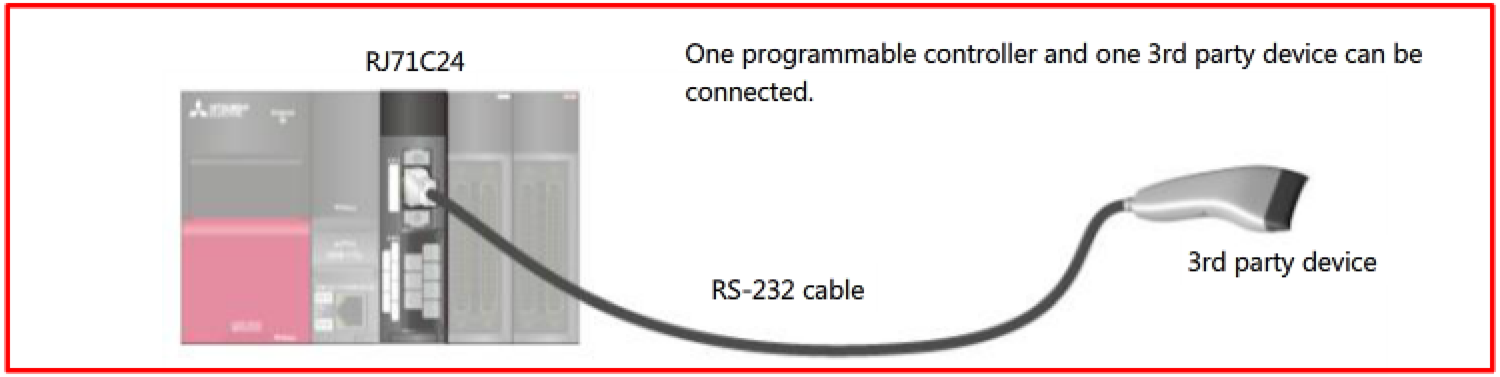
2.2 Communication Cable Connection

This section shows the connections of the serial communication modules.

2.2.1 Connecting the RS-232 interface to a device

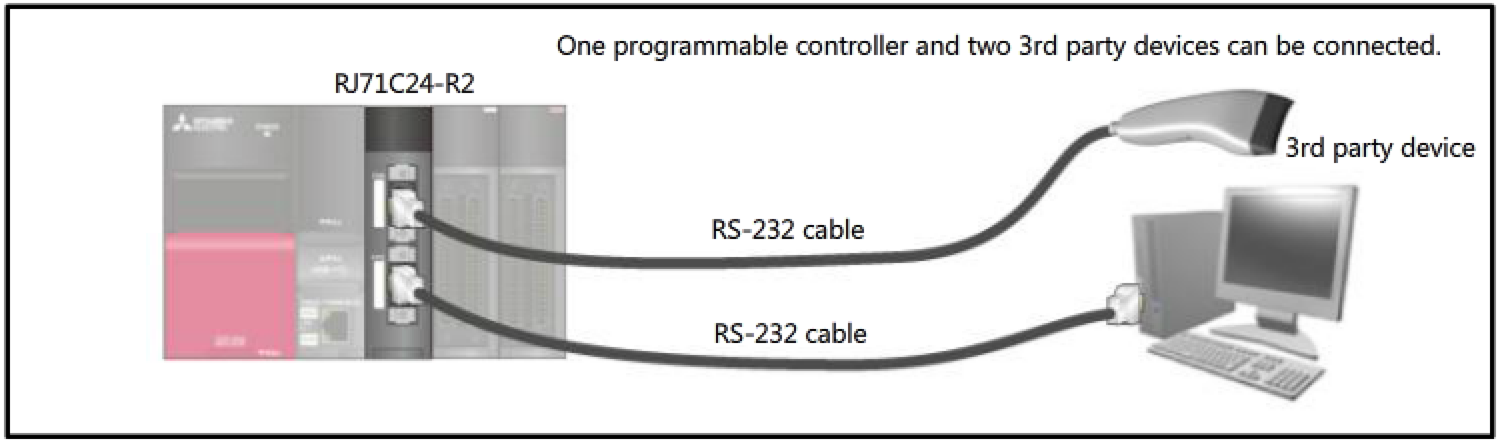
Below are the connection examples of the RS-232 interface, its 3rd party device, and the RJ71C24 and the RJ71C24-R2.

When RJ71C24 is used

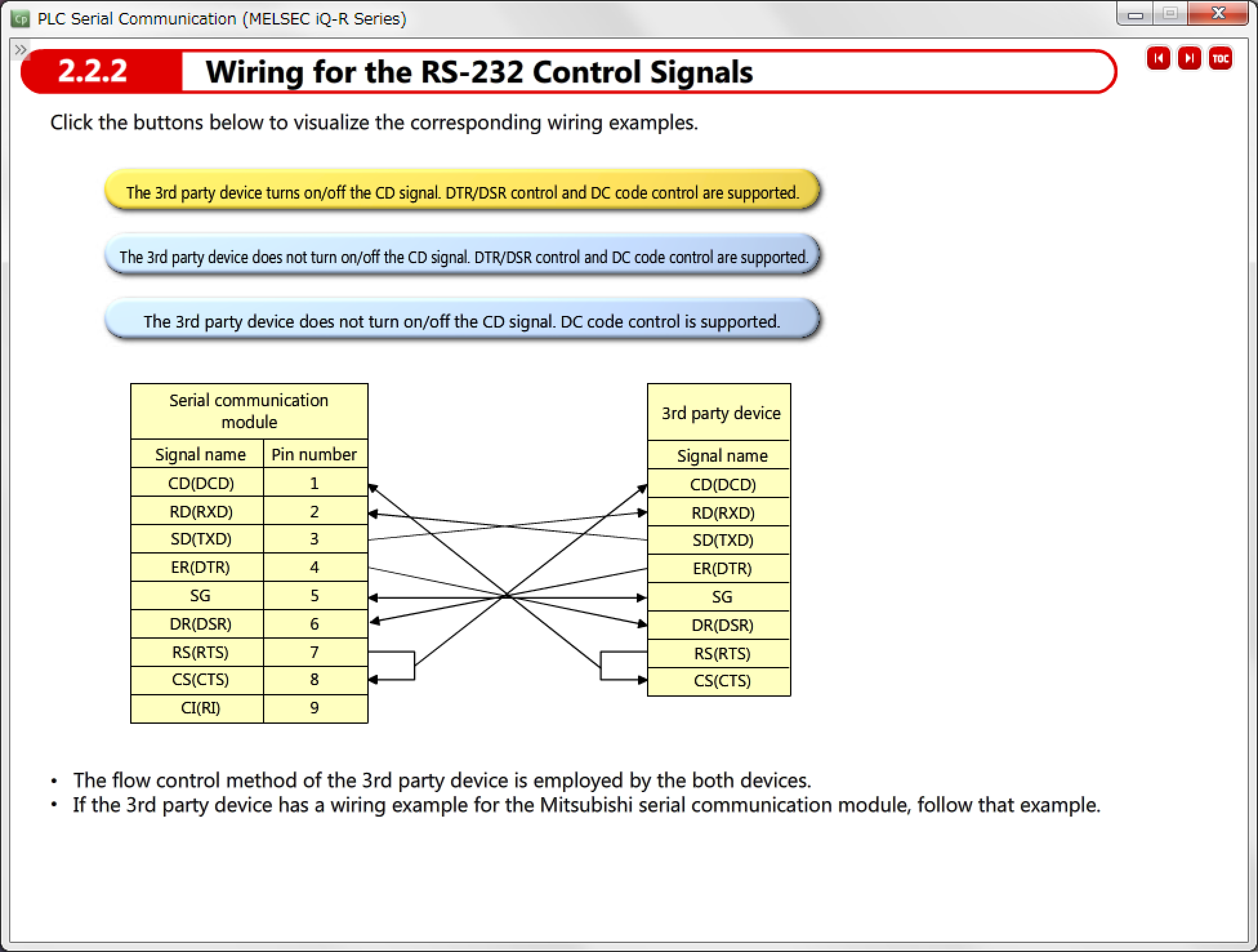


One programmable controller and one 3rd party device can be connected.

When RJ71C24-R2 is used



One programmable controller and two 3rd party devices can be connected.



2.2.2 Wiring for the RS-232 Control Signals

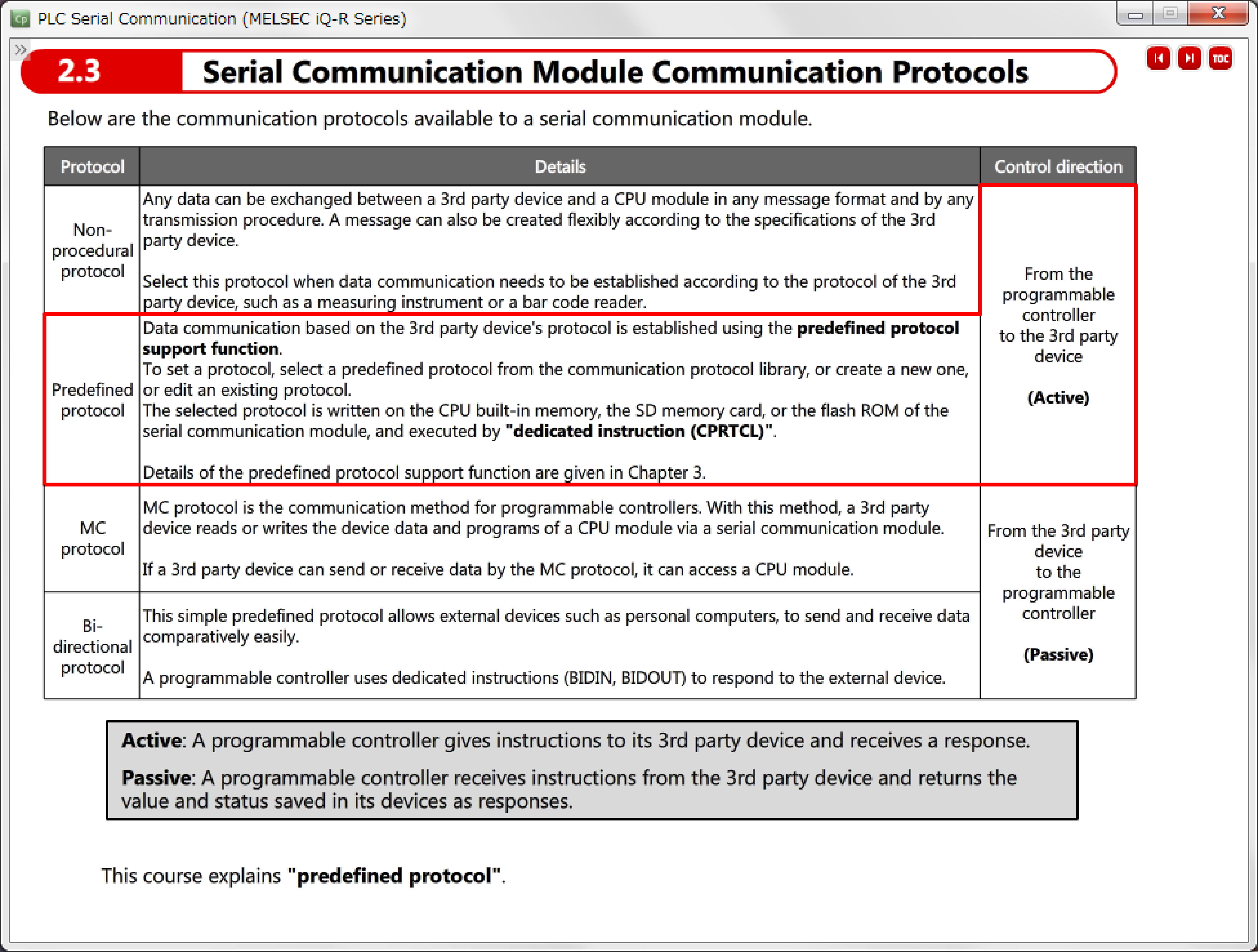
Click the buttons below to visualize the corresponding wiring examples.

- The 3rd party device turns on/off the CD signal. DTR/DSR control and DC code control are supported.
- The 3rd party device does not turn on/off the CD signal. DTR/DSR control and DC code control are supported.
- The 3rd party device does not turn on/off the CD signal. DC code control is supported.

Serial communication module	
Signal name	Pin number
CD(DCD)	1
RD(RXD)	2
SD(TXD)	3
ER(DTR)	4
SG	5
DR(DSR)	6
RS(RTS)	7
CS(CTS)	8
CI(RI)	9

3rd party device
Signal name
CD(DCD)
RD(RXD)
SD(TXD)
ER(DTR)
SG
DR(DSR)
RS(RTS)
CS(CTS)

- The flow control method of the 3rd party device is employed by the both devices.
- If the 3rd party device has a wiring example for the Mitsubishi serial communication module, follow that example.



2.3 Serial Communication Module Communication Protocols

Below are the communication protocols available to a serial communication module.

Protocol	Details	Control direction
Non-procedural protocol	Any data can be exchanged between a 3rd party device and a CPU module in any message format and by any transmission procedure. A message can also be created flexibly according to the specifications of the 3rd party device. Select this protocol when data communication needs to be established according to the protocol of the 3rd party device, such as a measuring instrument or a bar code reader.	From the programmable controller to the 3rd party device **(Active)**
Predefined protocol	Data communication based on the 3rd party device's protocol is established using the **predefined protocol support function**. To set a protocol, select a predefined protocol from the communication protocol library, or create a new one, or edit an existing protocol. The selected protocol is written on the CPU built-in memory, the SD memory card, or the flash ROM of the serial communication module, and executed by **"dedicated instruction (CPRTCL)"**. Details of the predefined protocol support function are given in Chapter 3.	
MC protocol	MC protocol is the communication method for programmable controllers. With this method, a 3rd party device reads or writes the device data and programs of a CPU module via a serial communication module. If a 3rd party device can send or receive data by the MC protocol, it can access a CPU module.	From the 3rd party device to the programmable controller **(Passive)**
Bi-directional protocol	This simple predefined protocol allows external devices such as personal computers, to send and receive data comparatively easily. A programmable controller uses dedicated instructions (BIDIN, BIDOUT) to respond to the external device.	

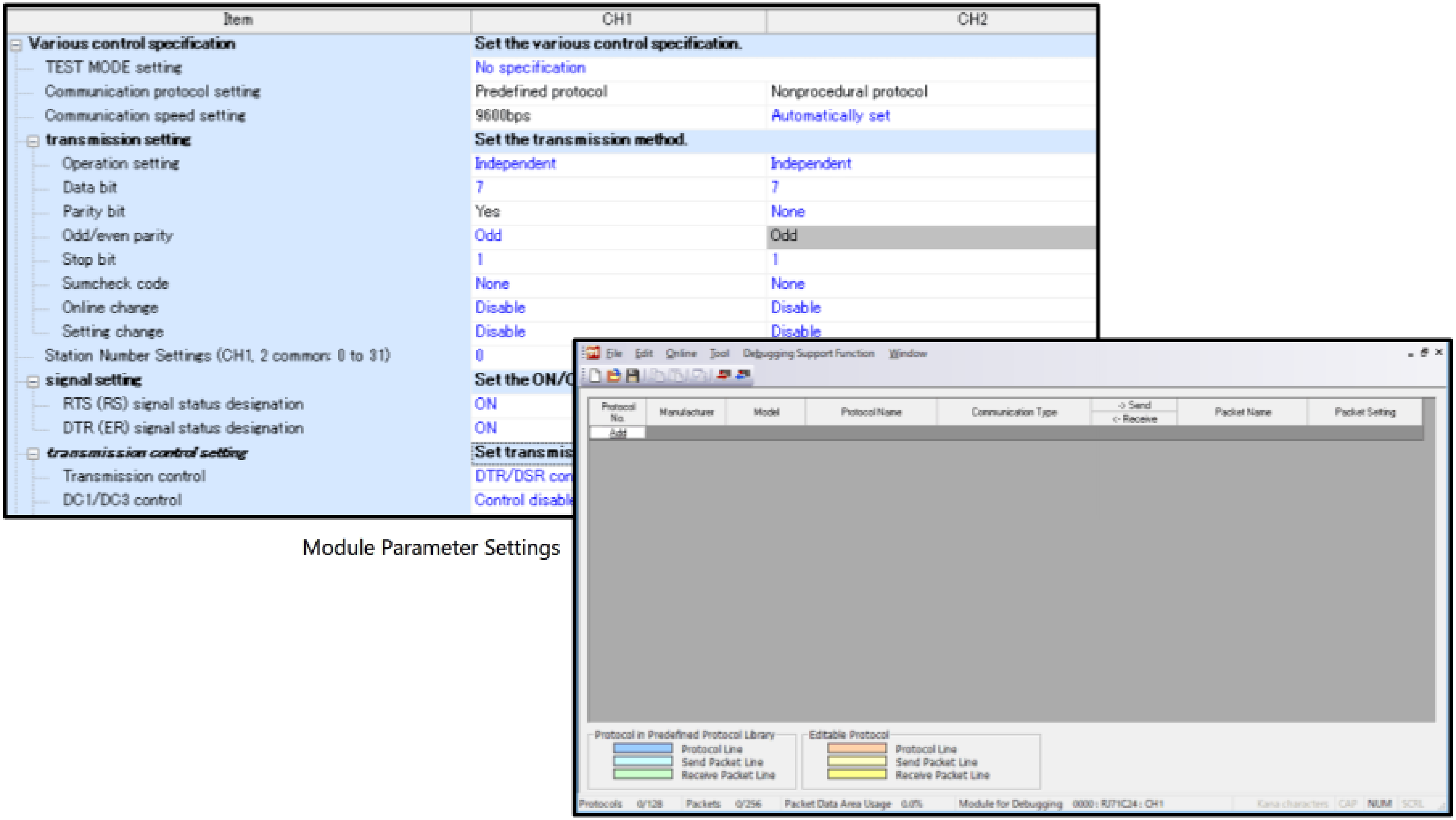
Active: A programmable controller gives instructions to its 3rd party device and receives a response.

Passive: A programmable controller receives instructions from the 3rd party device and returns the value and status saved in its devices as responses.

This course explains **"predefined protocol"**.

2.4 Serial Communication Module Configuration

Engineering software, GX Works3, is useful in configuring initial settings and registering predefined protocols (predefined protocol support function) to the serial communication modules. Please refer to Chapter 3 for details.



Item	CH1	CH2
Various control specification	Set the various control specification.	
TEST MODE setting	No specification	
Communication protocol setting	Predefined protocol	Nonprocedural protocol
Communication speed setting	9600bps	Automatically set
transmission setting	Set the transmission method.	
Operation setting	Independent	Independent
Data bit	7	7
Parity bit	Yes	None
Odd/even parity	Odd	Odd
Stop bit	1	1
Sumcheck code	None	None
Online change	Disable	Disable
Setting change	Disable	Disable
Station Number Settings (CH1, 2 common: 0 to 31)	0	
signal setting	Set the ON/O	
RTS (RS) signal status designation	ON	
DTR (ER) signal status designation	ON	
transmission control setting	Set transmis	
Transmission control	DTR/DSR con	
DC1/DC3 control	Control disabl	

Module Parameter Settings

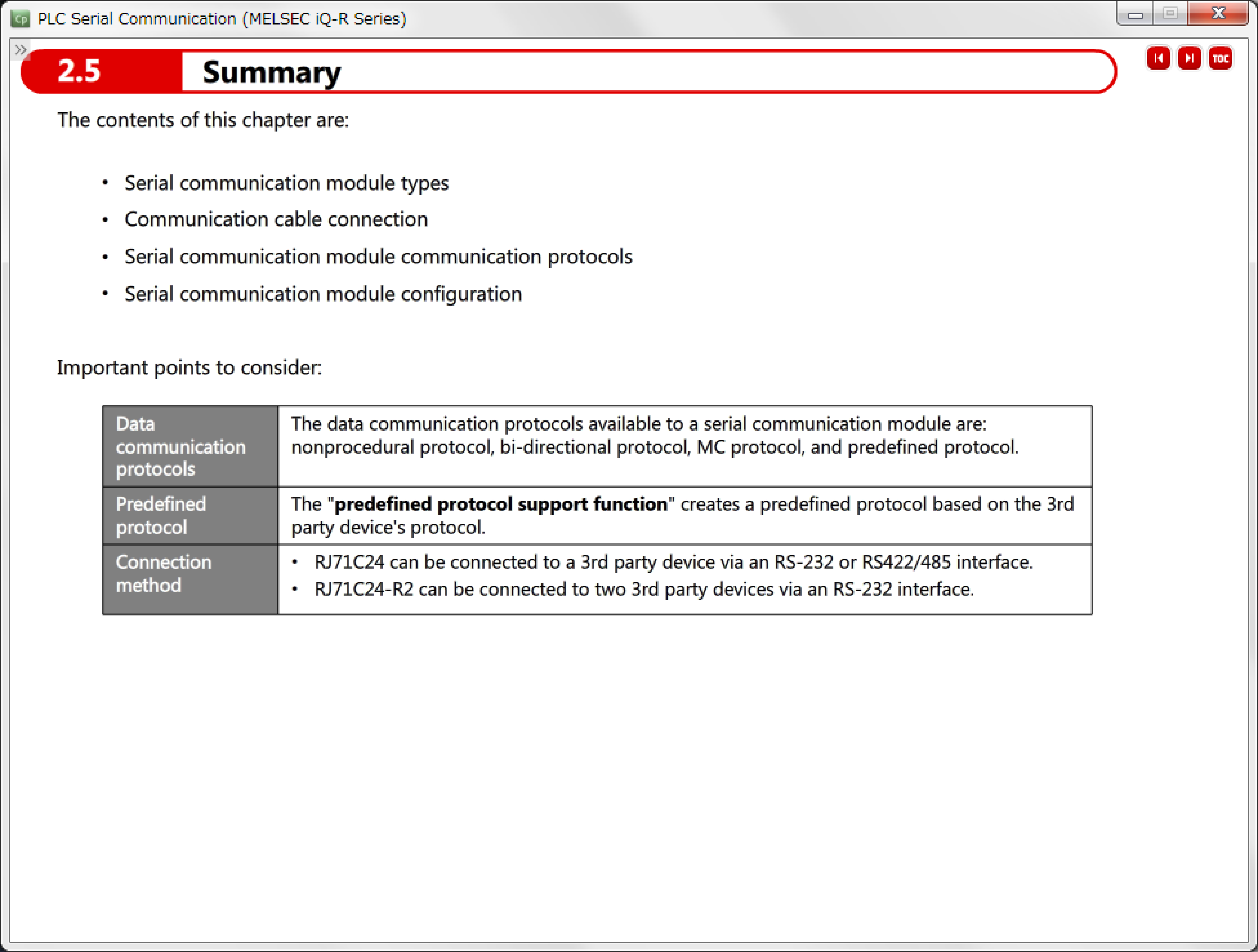
Protocol No.	Manufacturer	Model	Protocol Name	Communication Type	-> Send <- Receive	Packet Name	Packet Setting
Add							

Protocol in Predefined Protocol Library
- 🟦 Protocol Line
- 🟦 Send Packet Line
- 🟩 Receive Packet Line

Editable Protocol
- 🟧 Protocol Line
- 🟨 Send Packet Line
- 🟨 Receive Packet Line

Protocols 0/128 Packets 0/256 Packet Data Area Usage 0.0% Module for Debugging 0000 : RJ71C24 : CH1 Kana characters CAP NUM SCRL

Predefined Protocol Support Function



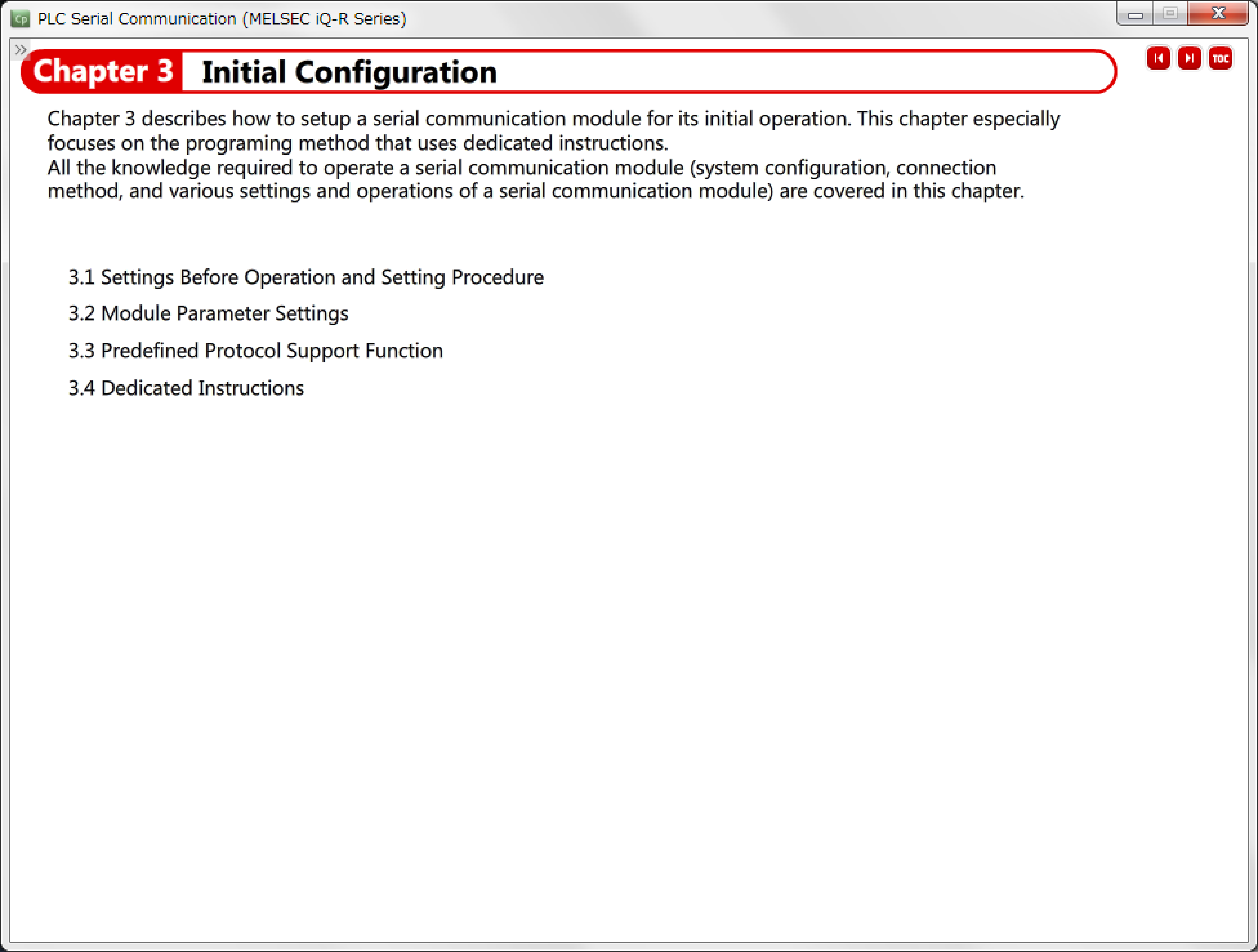
2.5 Summary

The contents of this chapter are:

- Serial communication module types
- Communication cable connection
- Serial communication module communication protocols
- Serial communication module configuration

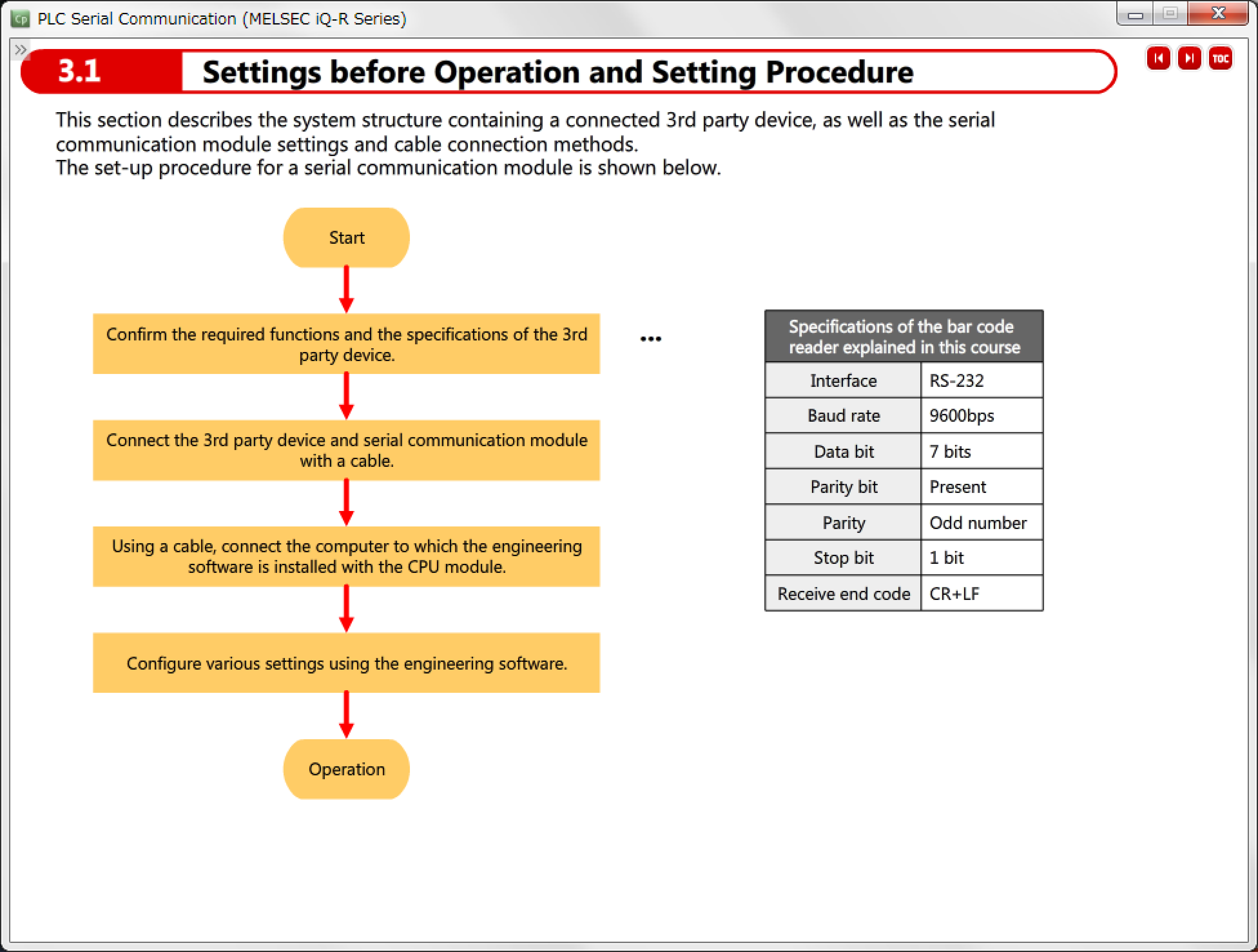
Important points to consider:

Data communication protocols	The data communication protocols available to a serial communication module are: nonprocedural protocol, bi-directional protocol, MC protocol, and predefined protocol.
Predefined protocol	The "**predefined protocol support function**" creates a predefined protocol based on the 3rd party device's protocol.
Connection method	• RJ71C24 can be connected to a 3rd party device via an RS-232 or RS422/485 interface. • RJ71C24-R2 can be connected to two 3rd party devices via an RS-232 interface.



Chapter 3 Initial Configuration

Chapter 3 describes how to setup a serial communication module for its initial operation. This chapter especially focuses on the programing method that uses dedicated instructions.
All the knowledge required to operate a serial communication module (system configuration, connection method, and various settings and operations of a serial communication module) are covered in this chapter.



3.1 Settings before Operation and Setting Procedure

This section describes the system structure containing a connected 3rd party device, as well as the serial communication module settings and cable connection methods.
The set-up procedure for a serial communication module is shown below.

Start

Confirm the required functions and the specifications of the 3rd party device. …

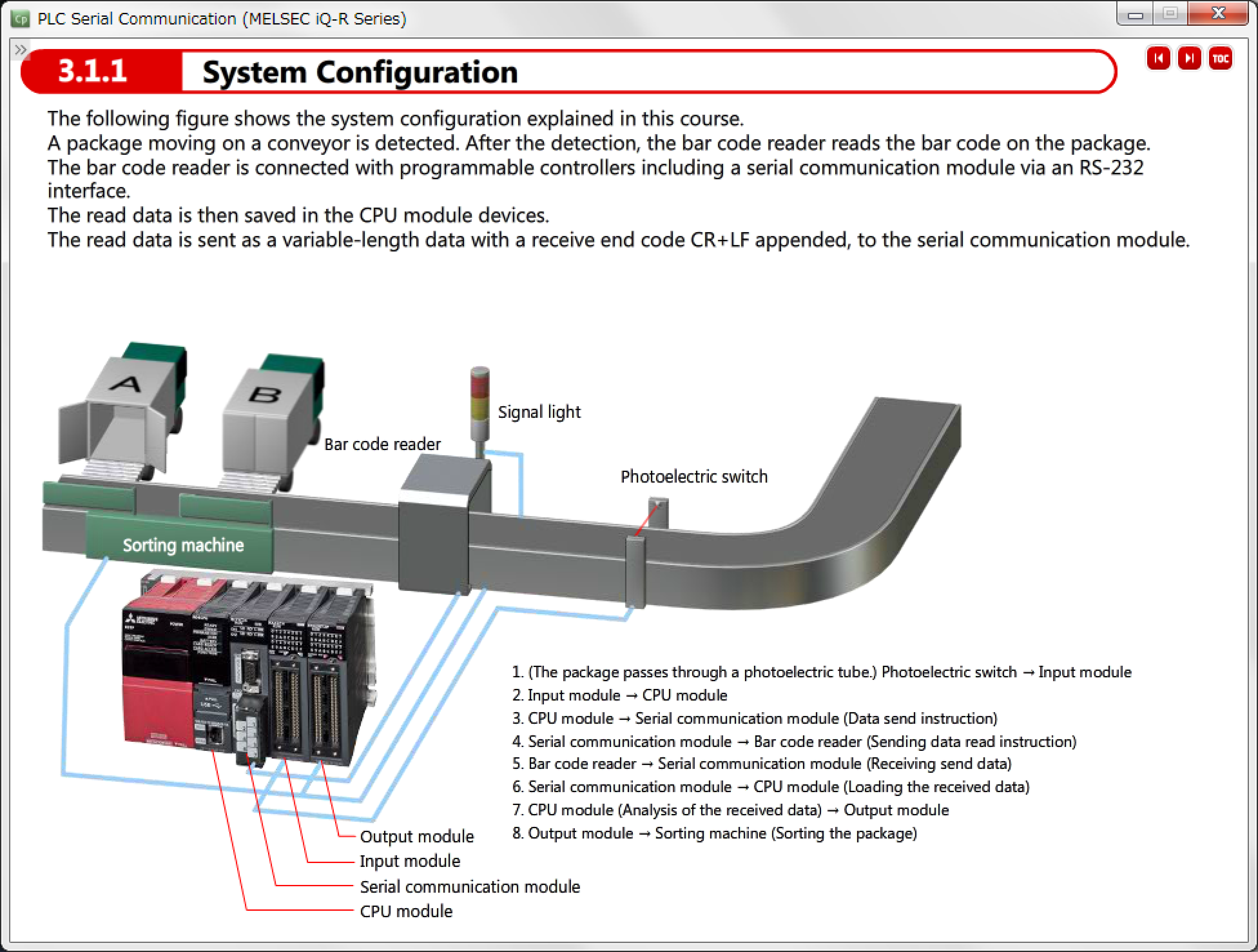
Connect the 3rd party device and serial communication module with a cable.

Using a cable, connect the computer to which the engineering software is installed with the CPU module.

Configure various settings using the engineering software.

Operation

Specifications of the bar code reader explained in this course	
Interface	RS-232
Baud rate	9600bps
Data bit	7 bits
Parity bit	Present
Parity	Odd number
Stop bit	1 bit
Receive end code	CR+LF



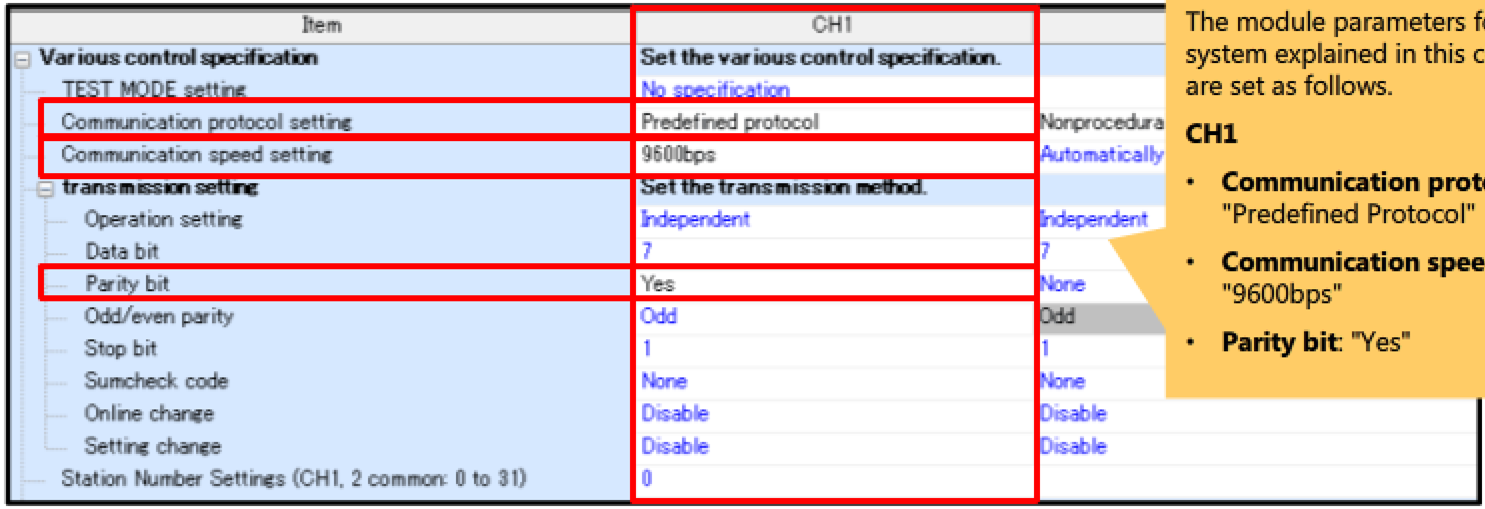
3.1.1 System Configuration

The following figure shows the system configuration explained in this course.
A package moving on a conveyor is detected. After the detection, the bar code reader reads the bar code on the package.
The bar code reader is connected with programmable controllers including a serial communication module via an RS-232 interface.
The read data is then saved in the CPU module devices.
The read data is sent as a variable-length data with a receive end code CR+LF appended, to the serial communication module.

1. (The package passes through a photoelectric tube.) Photoelectric switch → Input module
2. Input module → CPU module
3. CPU module → Serial communication module (Data send instruction)
4. Serial communication module → Bar code reader (Sending data read instruction)
5. Bar code reader → Serial communication module (Receiving send data)
6. Serial communication module → CPU module (Loading the received data)
7. CPU module (Analysis of the received data) → Output module
8. Output module → Sorting machine (Sorting the package)

3.2 Module Parameter Settings

This section describes the parameter settings required for data communications with a 3rd party device.
On the Project view of the Navigation window of GX Works3, select "Parameters" → "Module Information" →"RJ71C24" to open the "Module Parameter" window. On the "Module Parameter" window, set the necessary parameters, such as "Communication protocol setting", Communication speed setting", and "Parity bit", to communicate with the 3rd party device for each channel.



Item	CH1	
Various control specification	Set the various control specification.	
TEST MODE setting	No specification	
Communication protocol setting	Predefined protocol	Nonprocedura
Communication speed setting	9600bps	Automatically
transmission setting	Set the transmission method.	
Operation setting	Independent	Independent
Data bit	7	7
Parity bit	Yes	None
Odd/even parity	Odd	Odd
Stop bit	1	1
Sumcheck code	None	None
Online change	Disable	Disable
Setting change	Disable	Disable
Station Number Settings (CH1, 2 common: 0 to 31)	0	

The module parameters for the system explained in this course are set as follows.

CH1

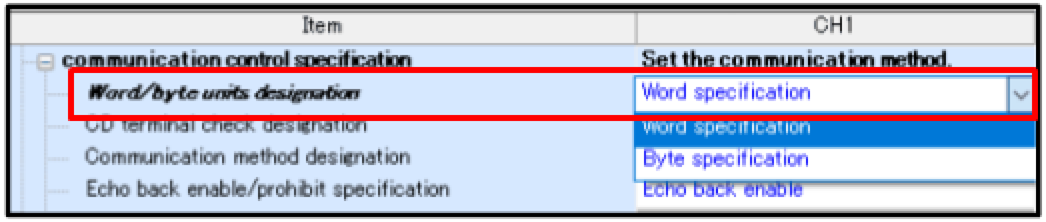
- **Communication protocol**: "Predefined Protocol"
- **Communication speed**: "9600bps"
- **Parity bit**: "Yes"

Item		Item setting details
Communication protocol setting		Set the details of communication with the 3rd party device.
Communication rate setting		Set the speed of communication with the 3rd party device.
Transmission Setting	Operation setting	Set whether two channels are used separately or linked for data communication.
	Data bit	Set the bit length of one character in the communication data.
	Parity bit	Set whether to add a parity bit to the communication data.
	Even/odd parity	Set whether to add an odd or even parity bit.
	Stop bit	Set the stop bit length of the data exchanged with the 3rd party device.
	Sum check code	Set whether to add a sum check code to sent and received messages.
	Online change	Set whether to write while the CPU module is in the "RUN" state.
	Setting modifications	Set whether to permit changes to the settings after the module has started up.
Station number setting (0 to 31)		Set the station number set by the 3rd party device when MC protocol is used.

3.2 Module Parameter Settings

Word/byte units designation

Set the unit of send/receive data.
The unit can be specified in **word** or **byte**.
The default value is specified in word unit. When the send/receive data is handled in byte unit, the setting needs to be changed.



Item	CH1
communication control specification	Set the communication method.
Word/byte units designation	Word specification
CD terminal check designation	
Communication method designation	
Echo back enable/prohibit specification	Echo back enable

The system explained in this course uses the default value, **word unit**.

3.2 Module Parameter Settings

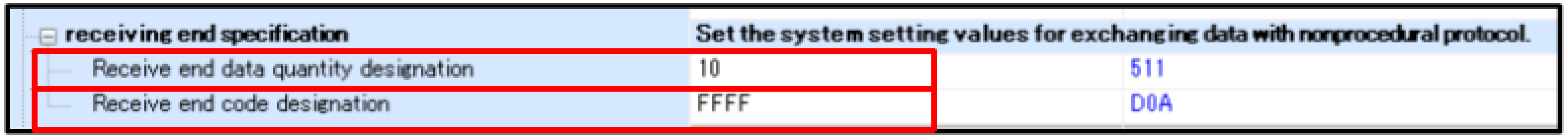
Receive end data quantity and the receive end code settings

The default values for the receive end data quantity and receive end code in the system explained in this course are not changed. The settings for data communications using the nonprocedural protocol are described as a reference.

The following table shows the settings to specify the codes used for the number of received data (size) and the data receive end.

Reception method	Receive end data quantity Default value: 511 (1FFH) words	Receive end code Default value: CR+LF
Variable length	To receive data equal to or smaller than the default value, **use this setting as it is**. If the receive end data quantity (size) exceeds the default value, the data is divided to be received. When data reception is completed at a time, **change of the setting is required**. For details, please refer to the corresponding manual of the serial communication module.	To use a receive end code other than the default value, **change this setting**.
Fixed length	**Change the setting** according to the length of the received data.	**Change to** "Not specified (FFFFH)".

The following table shows the settings when the receive end code is not specified and the received data is set to fixed length (10 words).



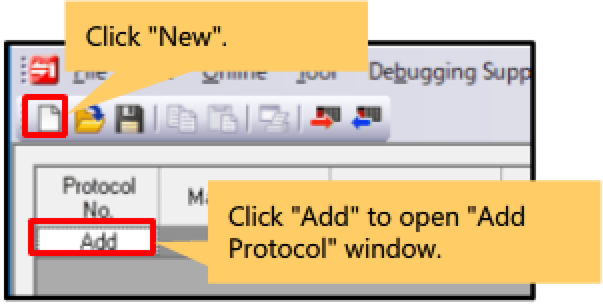
⊟ receiving end specification	Set the system setting values for exchanging data with nonprocedural protocol.	
Receive end data quantity designation	10	511
Receive end code designation	FFFF	D0A

We have covered how to set the module parameters so far.
Now, we are moving onto how to write the module parameters to a CPU module and reset the CPU module.

3.3 Predefined Protocol Support Function

The **"predefined protocol support function"** enables protocol communication with a 3rd party device using a simple sequence program containing the dedicated instructions. The predefined protocol support function reduces the program size and program-creation time as compared to using individual sequence programs.

Select "Predefined Protocol Support Function" from "Tool" of GX Works3 and select "Serial Communication Module" from "Module Type".
The "Predefined Protocol Support Function" window is opened.



Click "New".

Click "Add" to open "Add Protocol" window.

"Predefined Protocol Support Function" window

Some predefined protocols are already in the engineering software, but if the protocol of the 3rd party device is not found, new protocol can be created.

(1) When the predefined protocol is already in the engineering software

Select the manufacturer, model, and protocol name in the "Add Protocol" window.

(2) When the predefined protocol is not found in the engineering software

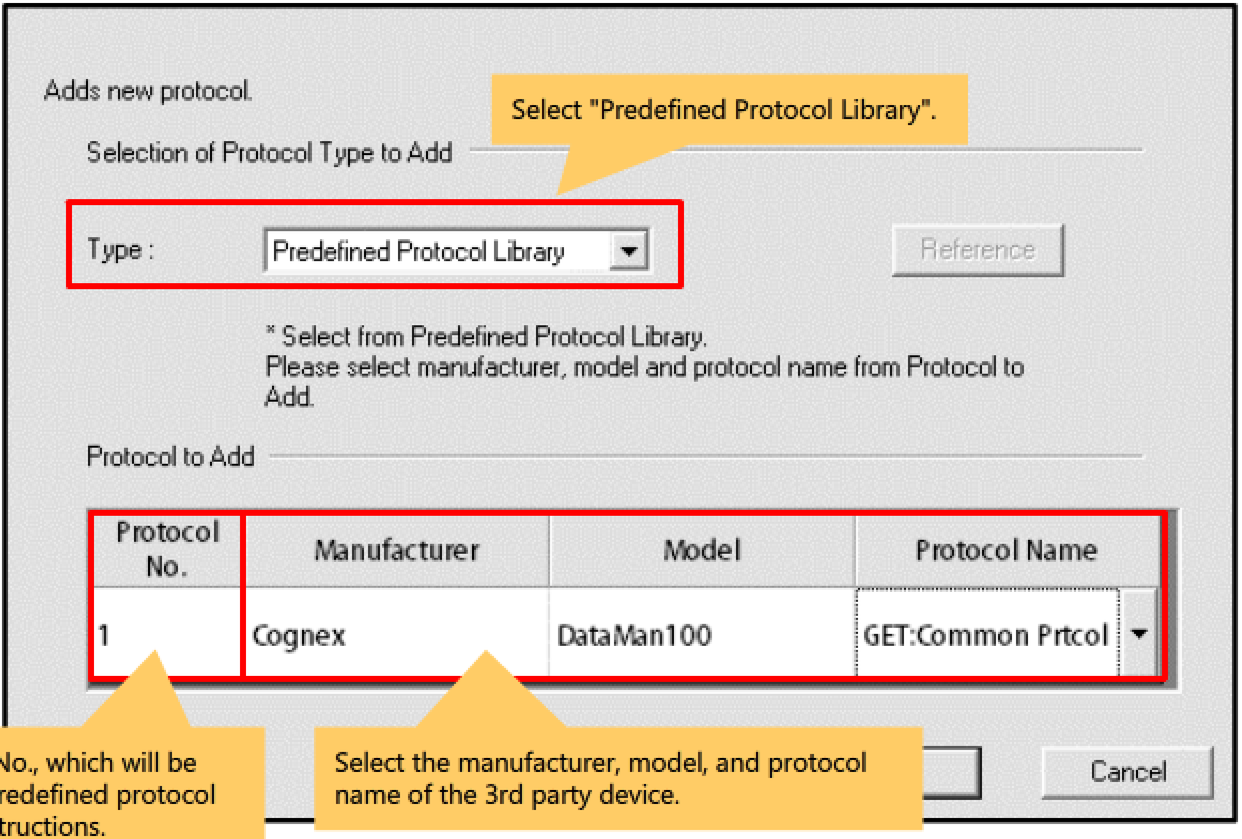
Create a new predefined protocol.

In this course, how to create a new predefined protocol according to the 3rd party device is explained. ((2) in this slide)

3.3.1 Adding a Protocol

(1) When the predefined protocol is already in the engineering software

When the desired predefined protocol already exists, select the manufacturer and model in the "Add Protocol" window to register it.



Adds new protocol.

Selection of Protocol Type to Add

Type : Predefined Protocol Library

* Select from Predefined Protocol Library.
Please select manufacturer, model and protocol name from Protocol to Add.

Protocol to Add

Protocol No.	Manufacturer	Model	Protocol Name
1	Cognex	DataMan100	GET:Common Prtcol

Select "Predefined Protocol Library".

Set Protocol No., which will be specified in predefined protocol dedicated instructions.

The number can be selected from 1 to 128.

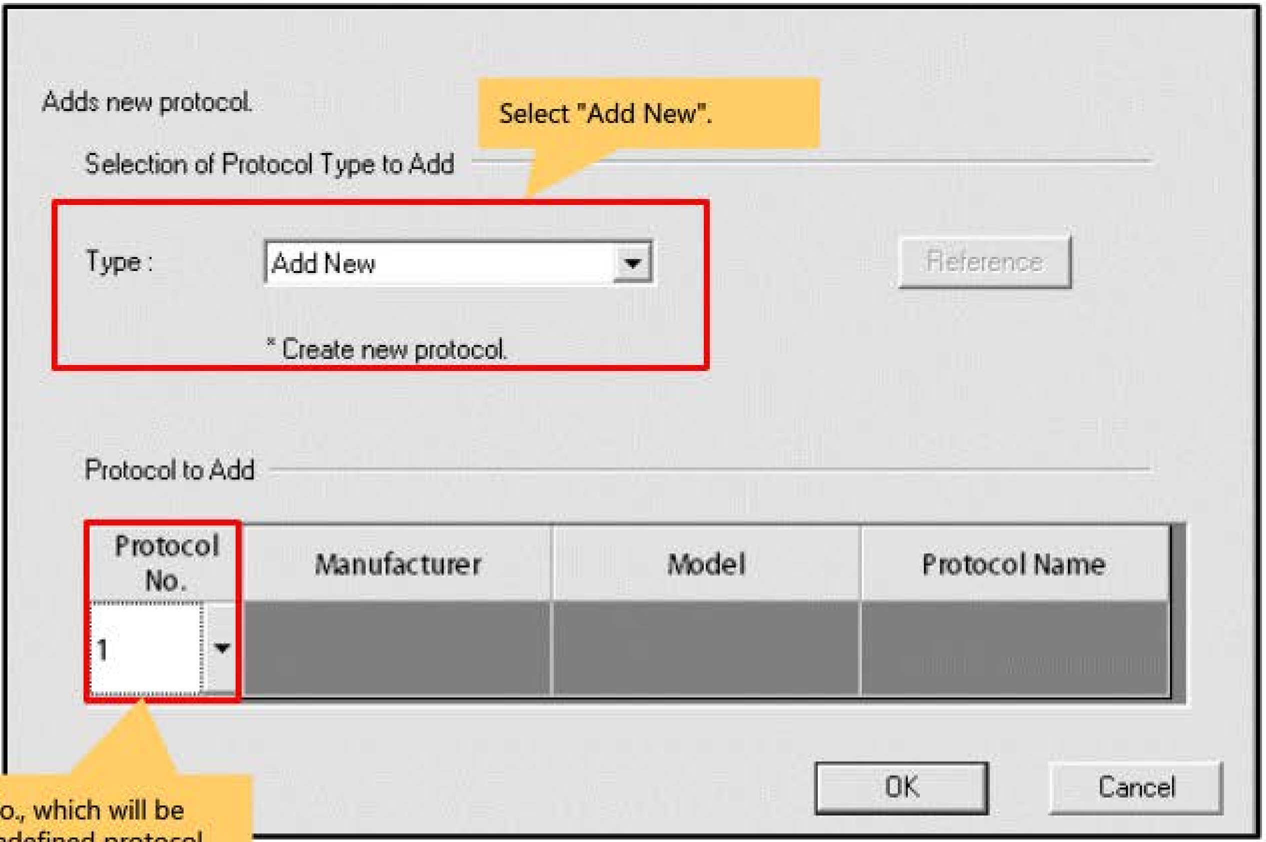
Select the manufacturer, model, and protocol name of the 3rd party device.

"Add Protocol" window

3.3.1 Adding a Protocol

(2) When the predefined protocol is not found in the engineering software

On the "Add Protocol" window, select "Add New" at "Type".



Adds new protocol.

Selection of Protocol Type to Add

Type : Add New

* Create new protocol.

Protocol to Add

Protocol No.	Manufacturer	Model	Protocol Name
1			

Select "Add New".

Set Protocol No., which will be specified in predefined protocol dedicated instructions.

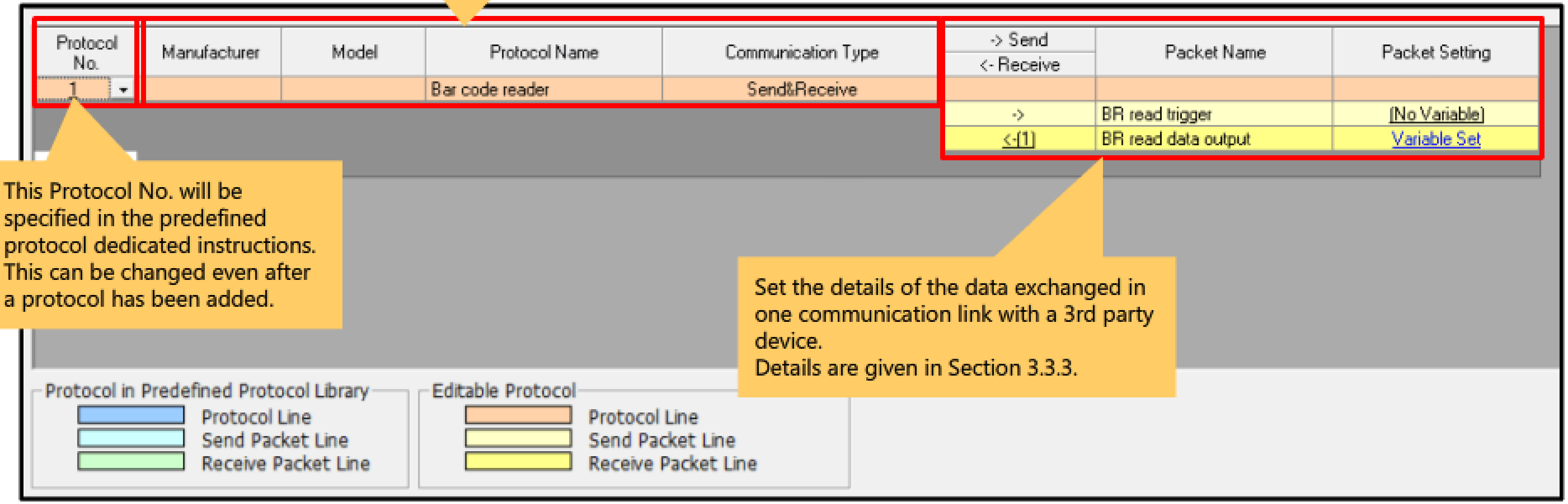
The number can be selected from 1 to 128.

"Add Protocol" window

3.3.2 Protocol Setting

Set the information of the newly added predefined protocol and the details of the communication data.

Set the information about the 3rd party device and the newly added protocol.
Double-click this area to open "Protocol Detailed Setting" window.
Please refer to the next page for details.



Protocol No.	Manufacturer	Model	Protocol Name	Communication Type	-> Send <- Receive	Packet Name	Packet Setting
1			Bar code reader	Send&Receive			
					->	BR read trigger	[No Variable]
					<-[1]	BR read data output	Variable Set

This Protocol No. will be specified in the predefined protocol dedicated instructions. This can be changed even after a protocol has been added.

Set the details of the data exchanged in one communication link with a 3rd party device.
Details are given in Section 3.3.3.

Protocol in Predefined Protocol Library
- 🟦 Protocol Line
- 🟦 Send Packet Line
- 🟩 Receive Packet Line

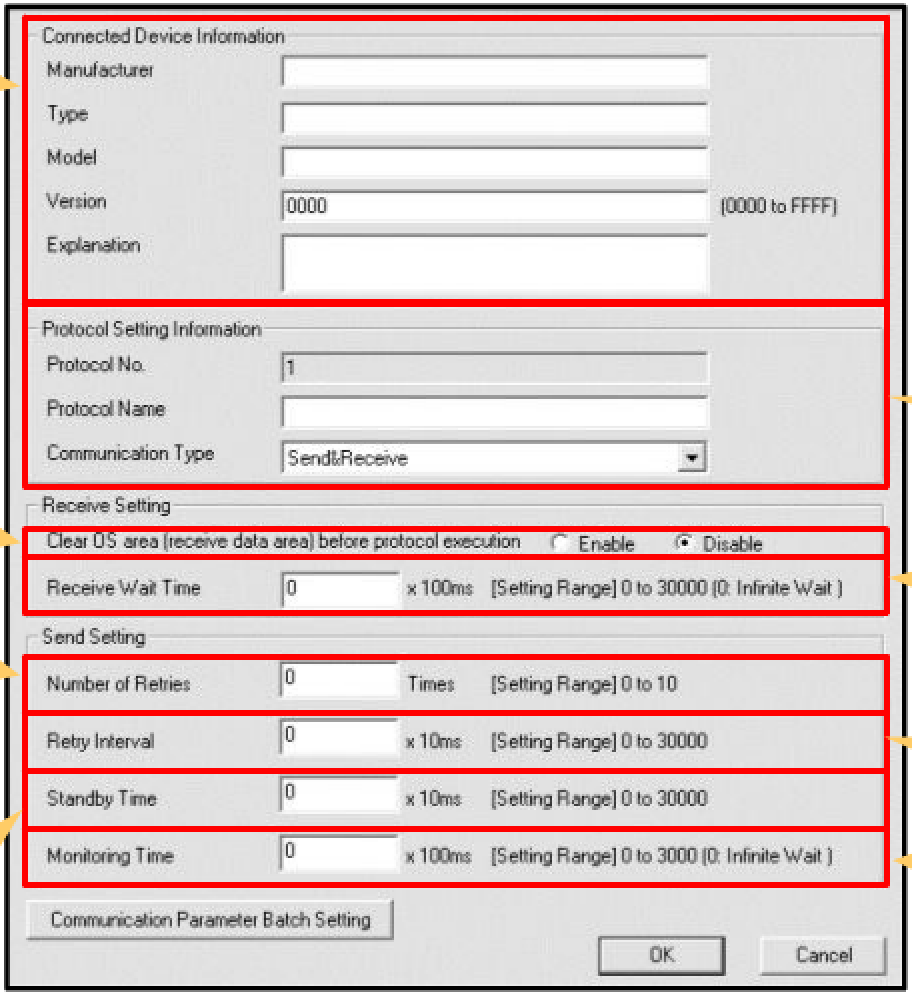
Editable Protocol
- 🟧 Protocol Line
- 🟨 Send Packet Line
- 🟨 Receive Packet Line

"Predefined Protocol Support Function" window

3.3.2 Protocol Setting

Detailed protocol settings

Set the information of the connected device, protocol, and data communication.



Connected Device Information

Manufacturer		
Type		
Model		
Version	0000	(0000 to FFFF)
Explanation		

Protocol Setting Information

Protocol No.	1
Protocol Name	
Communication Type	Send&Receive

Receive Setting

Clear OS area (receive data area) before protocol execution ○ Enable ● Disable

Receive Wait Time	0	x 100ms	[Setting Range] 0 to 30000 (0: Infinite Wait)

Send Setting

Number of Retries	0	Times	[Setting Range] 0 to 10
Retry Interval	0	x 10ms	[Setting Range] 0 to 30000
Standby Time	0	x 10ms	[Setting Range] 0 to 30000
Monitoring Time	0	x 100ms	[Setting Range] 0 to 3000 (0: Infinite Wait)

"Protocol Detailed Setting" window

Set the information about the connected device.

Set the protocol information.

Select whether to clear the module's OS area (received data area) before executing a program by the protocol.

Set the data reception waiting time period of the serial communication module.

Set the number of retries when the transmission from the module is not completed within the "monitoring time".

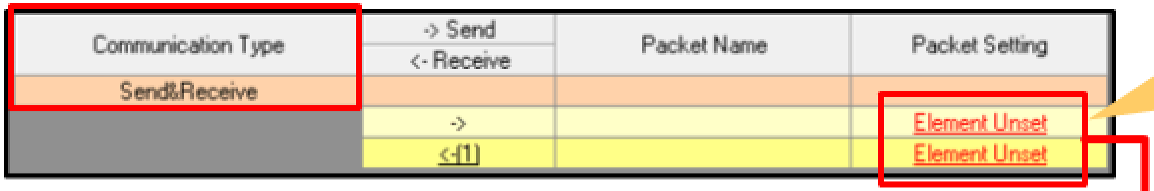
Set the time until the next retry.

Set the time period for which the module waits before transmitting the data instructed by the predefined protocol.

Set the time period from the module goes into the "Sending" state until transmission is completed.

3.3.3 Packet Setting

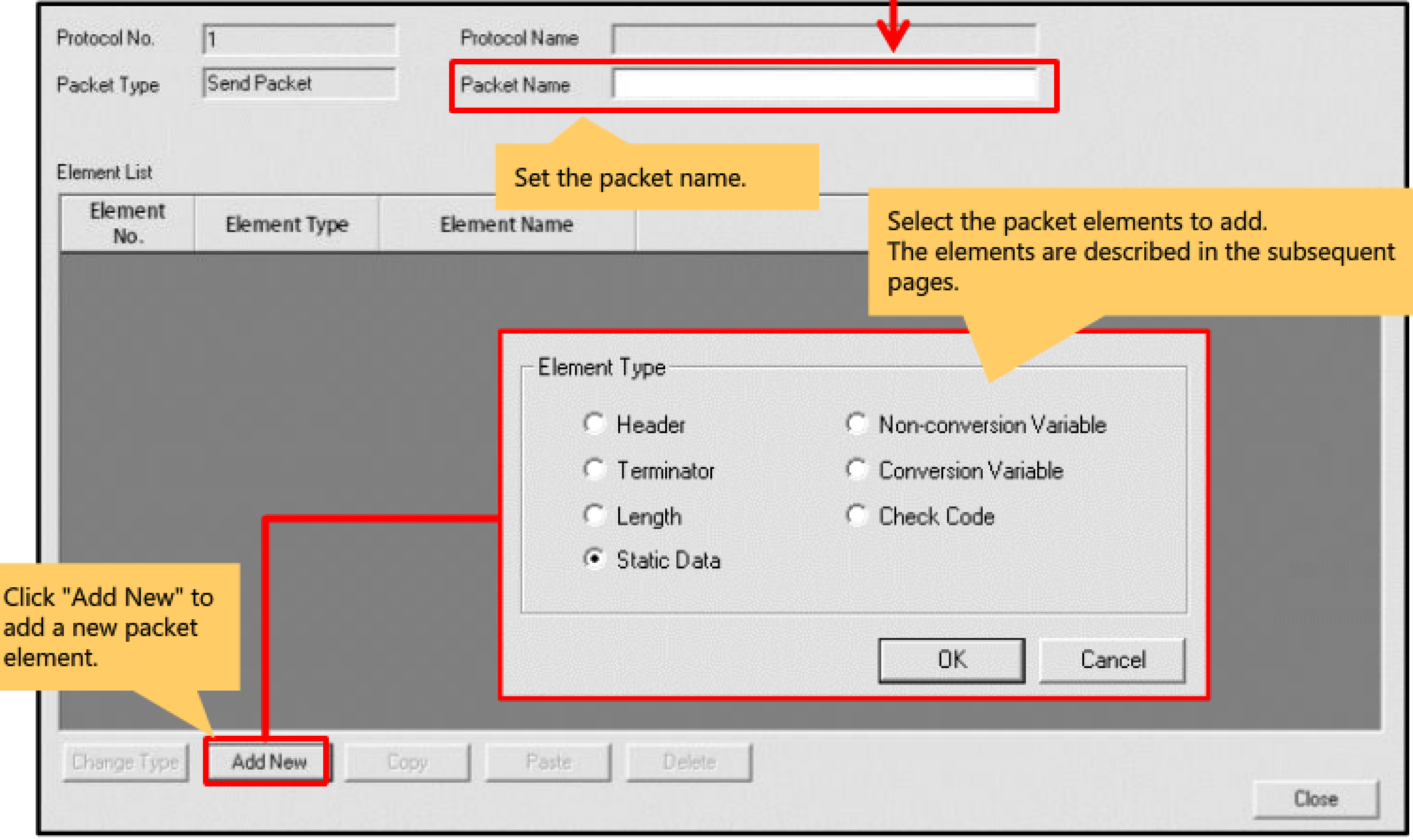
The data that is exchanged in one communication link with the 3rd party device is called a "packet", and a packet consists of different elements. The packet configuration can be set in "Packet Setting".



Communication Type	-> Send <- Receive	Packet Name	Packet Setting
Send&Receive			
	->		Element Unset
	<-(1)		Element Unset

Click "Element Unset" to display the "Packet Setting" window.
When the communication type is "->Send <-Receive", set the packet for sending and receiving.

"Predefined Protocol Support Function" window



Protocol No. 1

Protocol Name

Packet Type Send Packet

Packet Name

Set the packet name.

Element List

Element No.	Element Type	Element Name

Select the packet elements to add.
The elements are described in the subsequent pages.

Element Type

- ○ Header
- ○ Terminator
- ○ Length
- ● Static Data
- ○ Non-conversion Variable
- ○ Conversion Variable
- ○ Check Code

Click "Add New" to add a new packet element.

"Packet Setting" window

3.3.4 Packet Element Type

Header

A specific code or character string can be added to the head of a packet.

- When transmitted: The specified code or character string is sent.
- When received: The header is verified against the received data.

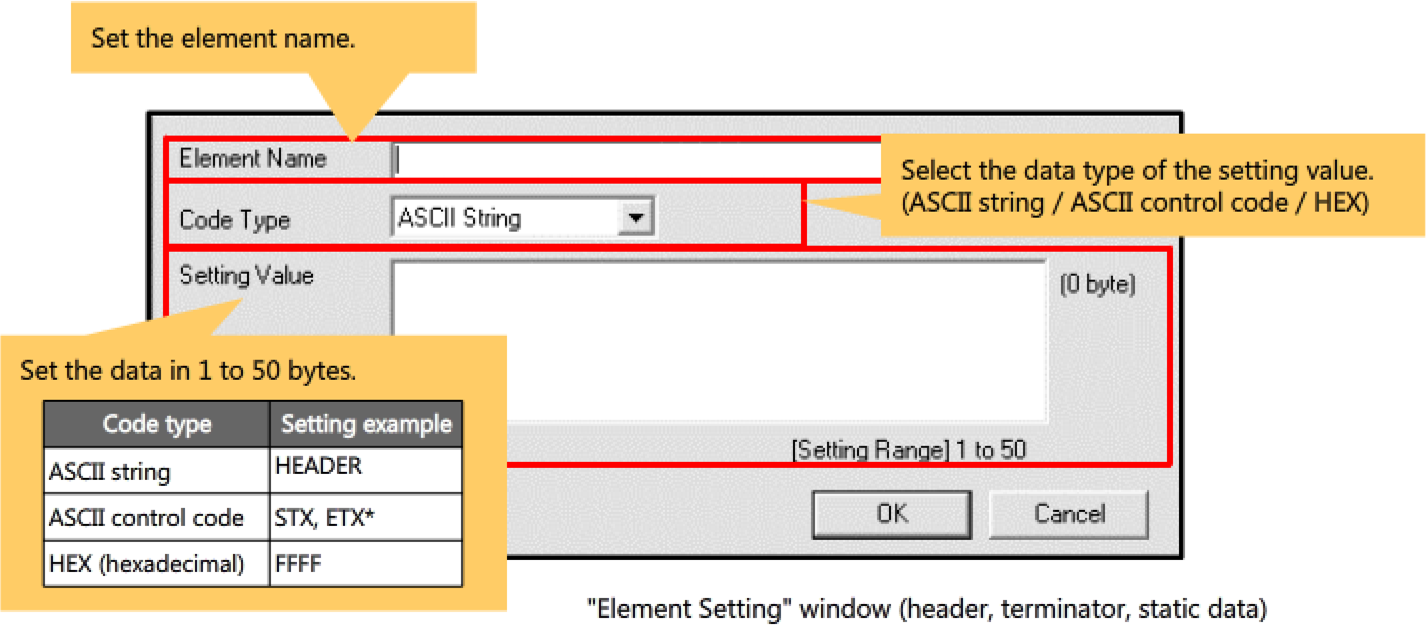
Terminator

A code or character string can be added to indicate the end of a packet.

Static data

A specific code or character string, such as a command, can be included in a packet.

- When transmitted: The specified code or character string is sent.
- When received: Received data is verified.



Set the element name.

Select the data type of the setting value.
(ASCII string / ASCII control code / HEX)

Set the data in 1 to 50 bytes.

Code type	Setting example
ASCII string	HEADER
ASCII control code	STX, ETX*
HEX (hexadecimal)	FFFF

"Element Setting" window (header, terminator, static data)

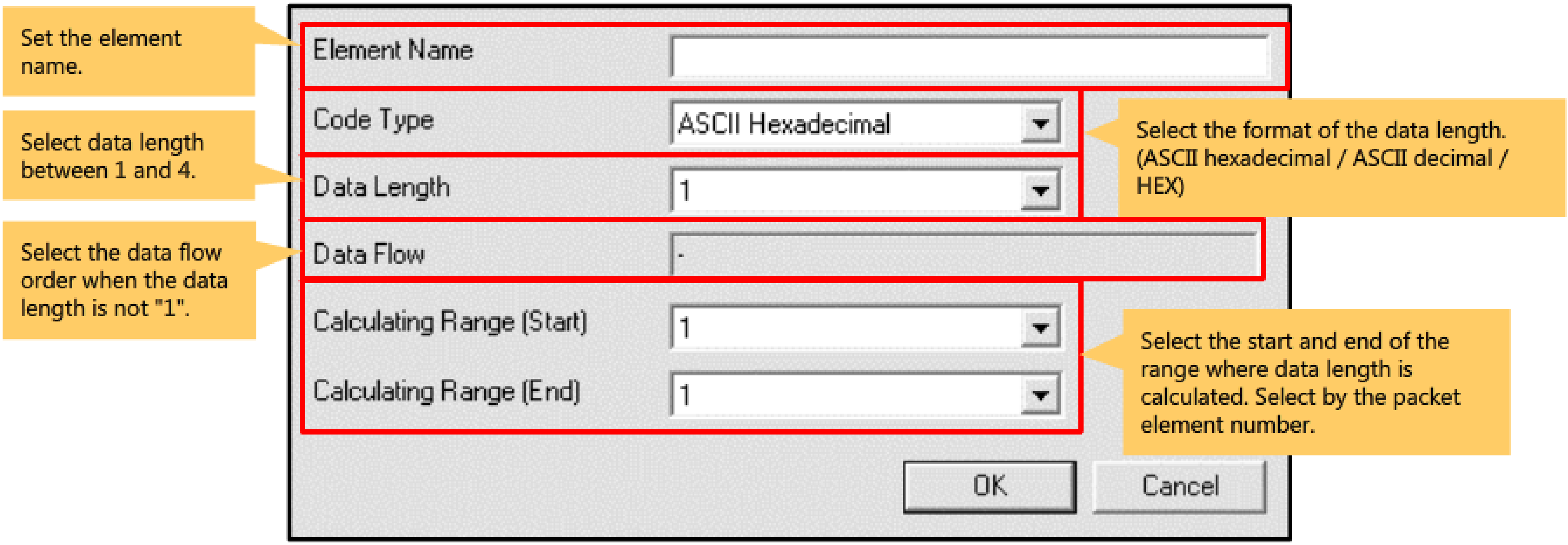
* STX: Start of text, ETX: End of text

3.3.4 Packet Element Type

Length

An element indicating the data length can be included in a packet.

- When transmitted: Data length of the specified range is automatically calculated, added to the packet, and sent.
- When received: The received data is checked against the data length information (value) contained in the received data.



Element Name:

Code Type: ASCII Hexadecimal

Data Length: 1

Data Flow: -

Calculating Range (Start): 1

Calculating Range (End): 1

Set the element name.

Select data length between 1 and 4.

Select the data flow order when the data length is not "1".

Select the format of the data length. (ASCII hexadecimal / ASCII decimal / HEX)

Select the start and end of the range where data length is calculated. Select by the packet element number.

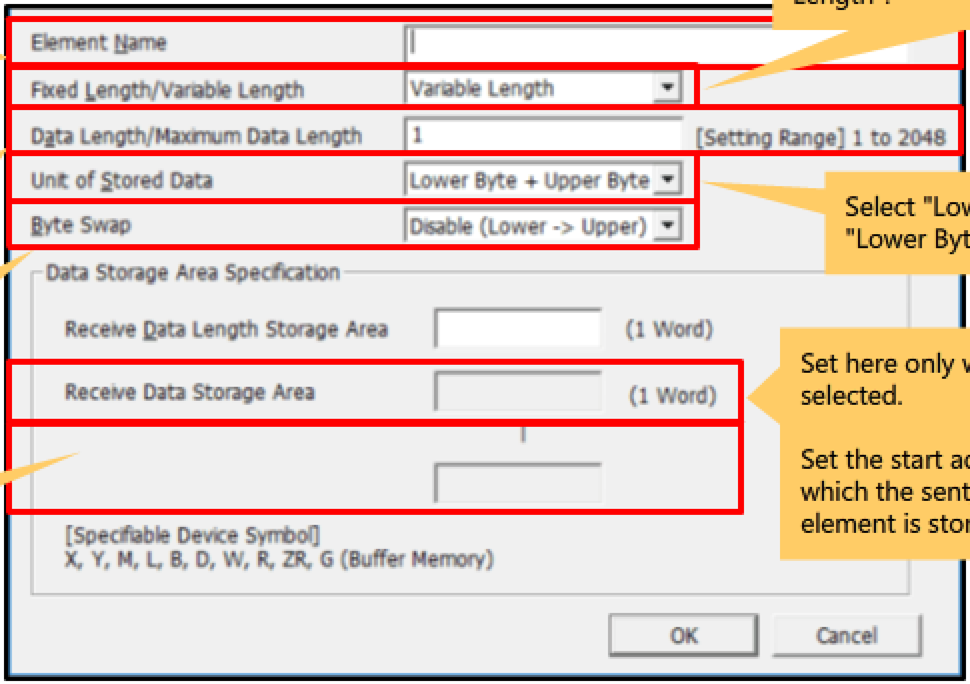
"Element Setting" window (length)

3.3.4 Packet Element Type

Non-conversion variable

Use a non-conversion variable when:

- Data in a device or the buffer memory is sent as it is without data conversion.
- Part of a received packet is stored in a device or buffer memory without data conversion.



Element Name: |

Fixed Length/Variable Length: Variable Length

Data Length/Maximum Data Length: 1 [Setting Range] 1 to 2048

Unit of Stored Data: Lower Byte + Upper Byte

Byte Swap: Disable (Lower -> Upper)

Data Storage Area Specification

Receive Data Length Storage Area (1 Word)

Receive Data Storage Area (1 Word)

[Specifiable Device Symbol]
X, Y, M, L, B, D, W, R, ZR, G (Buffer Memory)

Set the name of an element that specifies the data storage area.

Select "Fixed Length" or "Variable Length".

Set the data length. When the data length is varied, set the maximum data length.

Select "Lower Byte + Upper Byte" or "Lower Byte Only".

Select whether to conduct the byte swap.

Set here only when "Variable Length" is selected.

Set the start address of the devices in which the sent/received data length of the element is stored.

- When the data length is fixed, set the start address of the device in which a variable is stored. The end address is set automatically.
- When the data length is varied, this area is set automatically according to the setting at Send Data Storage Area.

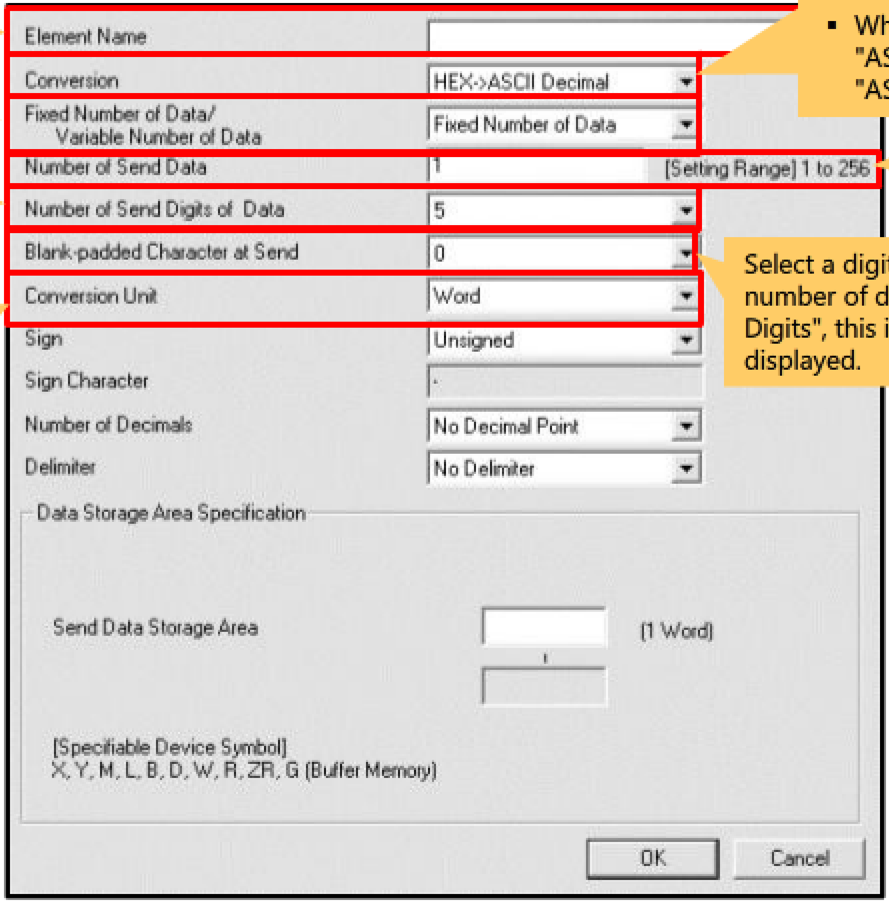
"Element Setting" window (non-conversion variable)

3.3.4 Packet Element Type

Conversion variable

The data in the device or buffer memory is sent after being converted, and received data is converted and then stored in the device or buffer memory. This data conversion process does not require a sequence program and reduces the total program size and programming time.

(Continued on the next page)



Element Name	
Conversion	HEX->ASCII Decimal
Fixed Number of Data/ Variable Number of Data	Fixed Number of Data
Number of Send Data	1 [Setting Range] 1 to 256
Number of Send Digits of Data	5
Blank-padded Character at Send	0
Conversion Unit	Word
Sign	Unsigned
Sign Character	-
Number of Decimals	No Decimal Point
Delimiter	No Delimiter

Data Storage Area Specification

Send Data Storage Area (1 Word)

[Specifiable Device Symbol]
X, Y, M, L, B, D, W, R, ZR, G (Buffer Memory)

Set the name of an element that specifies the data storage area.

Select "Fixed Number of Data" or "Variable Number of Data".

Select the number of digits "1 to 10" or "Variable Number of Digits".

Determine how many words of the data in the data storage area are handled as one set of data. "Word" or "Double word"

- When data is sent

  "HEX -> ASCII hexadecimal"
  "HEX -> ASCII decimal"
- When data is received
  "ASCII hexadecimal -> HEX"
  "ASCII decimal -> HEX"

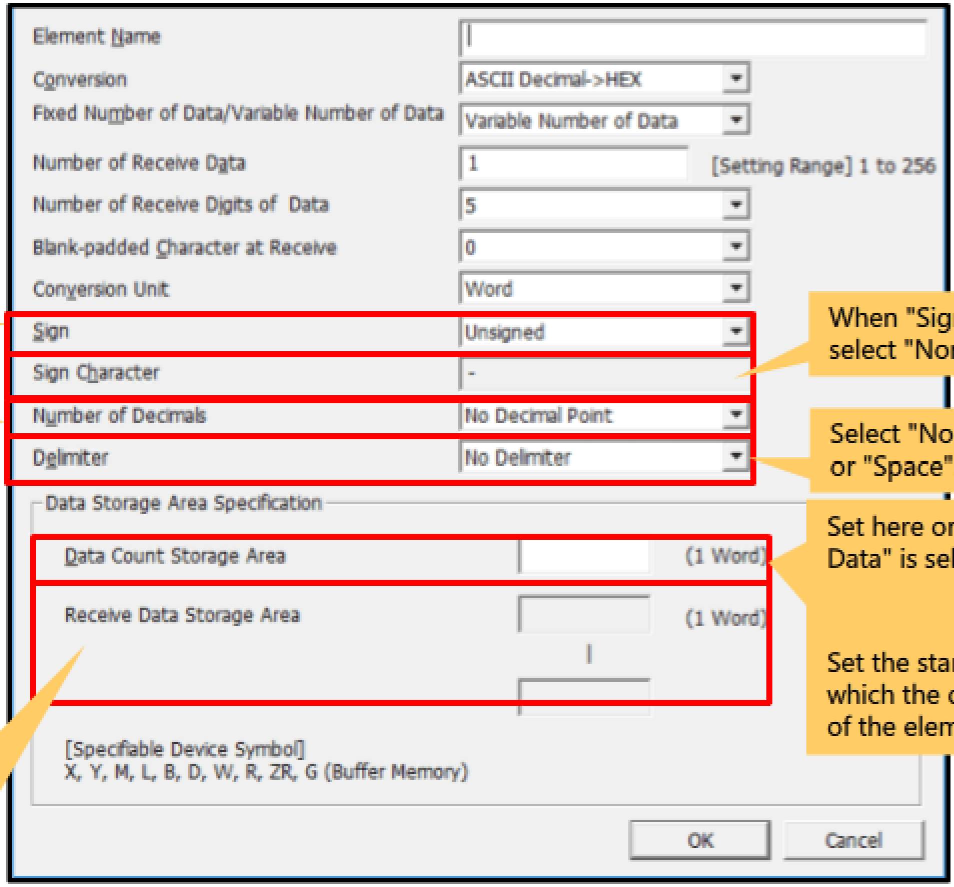
Set the data quantity (1 to 256).

Select a digit character "-" or "0". When the number of digits is "Variable Number of Digits", this item is disabled and "-" is displayed.

"Element Setting" window (conversion variable)

3.3.4 Packet Element Type

(Continued from the previous page)



Element Name		
Conversion	ASCII Decimal->HEX	
Fixed Number of Data/Variable Number of Data	Variable Number of Data	
Number of Receive Data	1	[Setting Range] 1 to 256
Number of Receive Digits of Data	5	
Blank-padded Character at Receive	0	
Conversion Unit	Word	
Sign	Unsigned	
Sign Character	-	
Number of Decimals	No Decimal Point	
Delimiter	No Delimiter	

Data Storage Area Specification

Data Count Storage Area		(1 Word)
Receive Data Storage Area	| 	(1 Word)

[Specifiable Device Symbol]
X, Y, M, L, B, D, W, R, ZR, G (Buffer Memory)

Select "Unsigned" or "Signed".

When "Signed" is selected at "Sign", select "None", "+", "0", or "-".*

Select "No Decimal Point", "1 to 9", or "Variable Point".

Select "No Delimiter", "One-byte Comma", or "Space".

Set here only when "Variable Number of Data" is selected.

Set the start address of the devices in which the quantity of sent/received data of the element is stored.

- When the data length is fixed, set the start address of the device in which a variable is stored. The end address is set automatically.
- When the data length is varied, this area is set automatically according to the setting at Send Data Storage Area.

"Element Setting" window (conversion variable)

* Select "+".
Negative values always need the "-" symbol.

3.3.4 Packet Element Type

Check code

An element that checks for incorrect data can be included in a packet.
The check code can be added to a transmitting packet or used against a reception packet. The check code calculation is automatically performed at data reception/transmission.

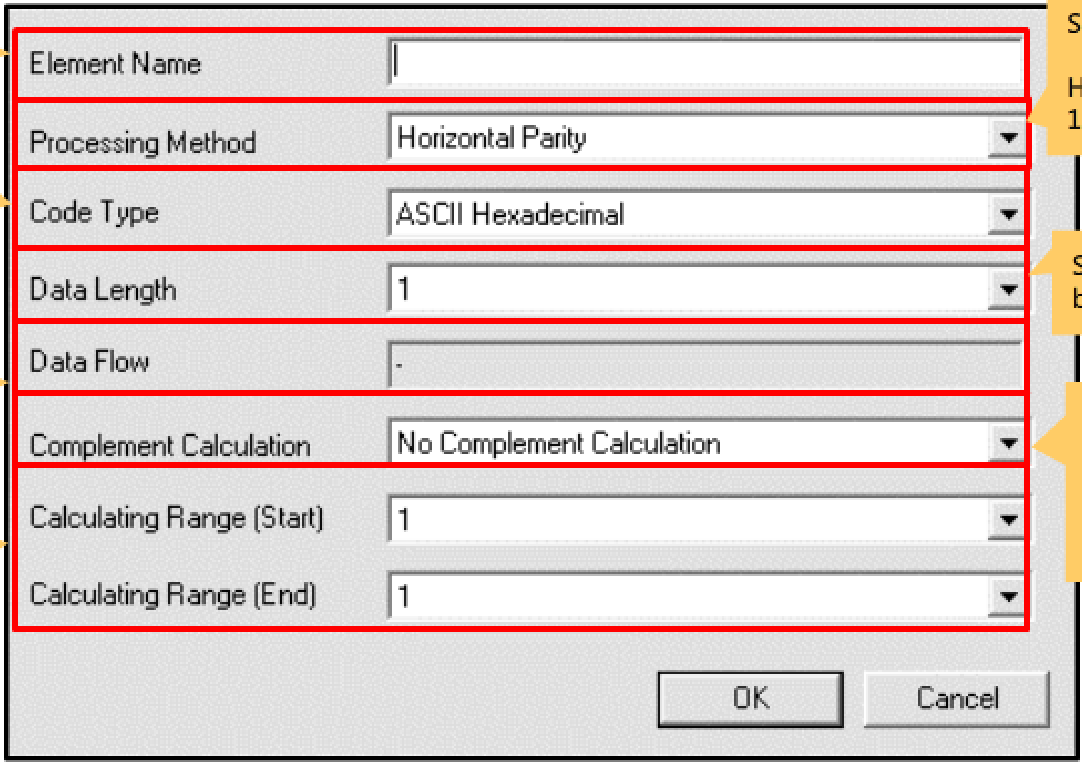
Set "Element name".

Select the send/receive format.

(ASCII Hexadecimal / ASCII Decimal / HEX)

Select the data flow order when the data length is not "1".

Select the start and end of the calculation range. Set by the packet element number.



Element Name	
Processing Method	Horizontal Parity
Code Type	ASCII Hexadecimal
Data Length	1
Data Flow	-
Complement Calculation	No Complement Calculation
Calculating Range (Start)	1
Calculating Range (End)	1

Select the calculation method.

Horizontal Parity / Sum Check / 16-bit CRC (for MODBUS)

Set the data length between 1 and 4.

"No Complement Calculation"

"One's Complement"

"Two's Complement"

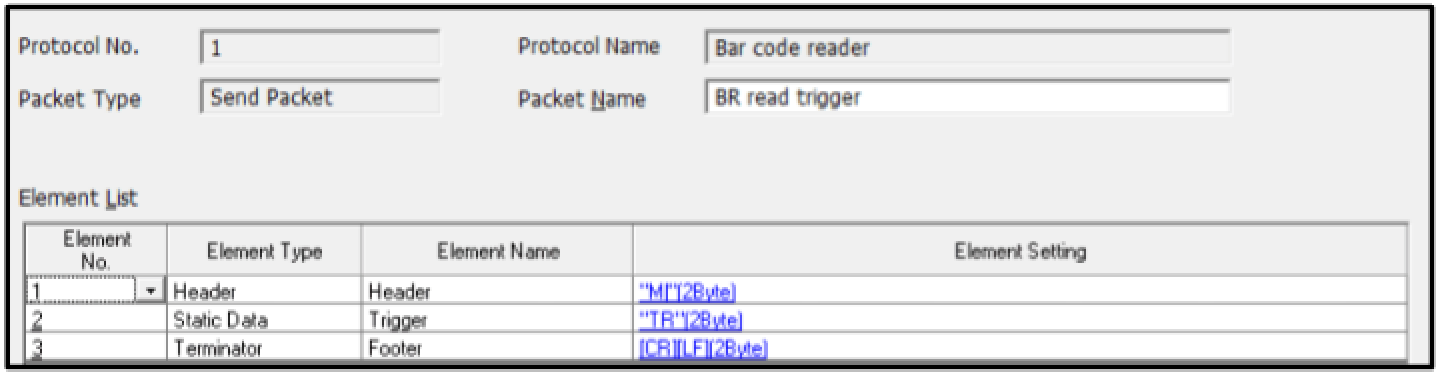
"Element Setting" window (check code)

3.3.5 Protocol Setting of the System

This section describes the packets sent/received by the predefined protocol in the system explained in this course.

(1) Send packet

The send packet contains the command character string for instructing a bar code read.
The send packet is composed of the header character string **"M"** (header, ASCII character), command character string **"TR"** (static data, ASCII character), and packet end code **"CR+LF"** (terminator, ASCII control code).



Protocol No.	1	Protocol Name	Bar code reader
Packet Type	Send Packet	Packet Name	BR read trigger

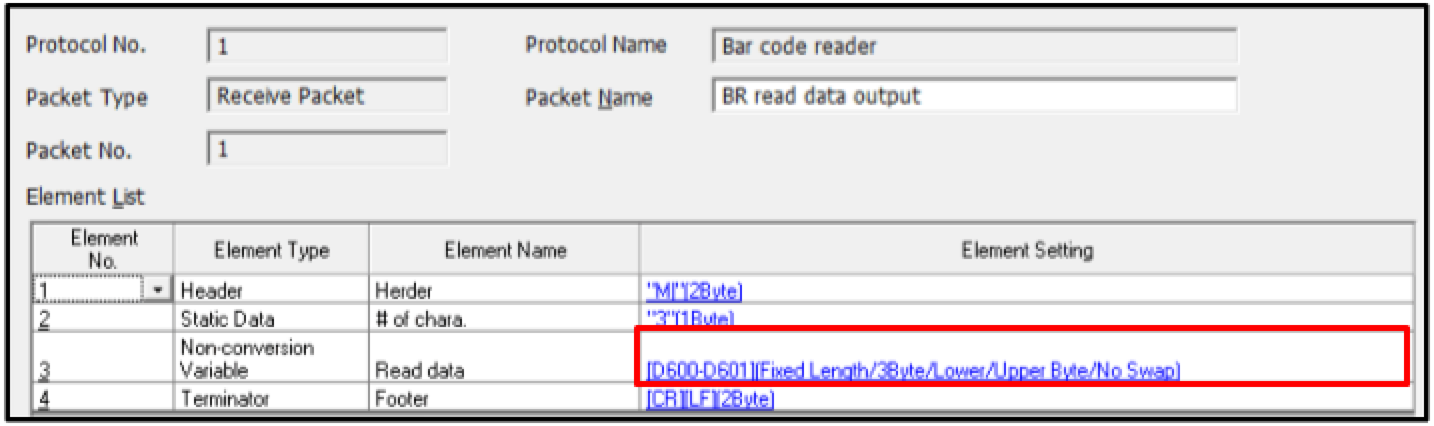
Element List

Element No.	Element Type	Element Name	Element Setting
1	Header	Header	"M"(2Byte)
2	Static Data	Trigger	"TR"(2Byte)
3	Terminator	Footer	[CR][LF](2Byte)

"Packet Setting" window (send packet)

(2) Receive packet

The receive packet contains the country ID code (JPN/USA) that has been read by the bar code reader.
The receive packet is composed of the header character string **"M"** (header, ASCII character), the number of country ID code characters **"3"** (static data, ASCII character), the country ID code (non-conversion variable, ASCII character), and a packet end code **"CR+LF"** (terminator, ASCII control code). After the packet is received, the country ID code is stored in the devices **"D600"** and **"D601"**.



Protocol No.	1	Protocol Name	Bar code reader
Packet Type	Receive Packet	Packet Name	BR read data output
Packet No.	1		

Element List

Element No.	Element Type	Element Name	Element Setting
1	Header	Herder	"M"(2Byte)
2	Static Data	# of chara.	"3"(1Byte)
3	Non-conversion Variable	Read data	[D600-D601](Fixed Length/3Byte/Lower/Upper Byte/No Swap)
4	Terminator	Footer	[CR][LF](2Byte)

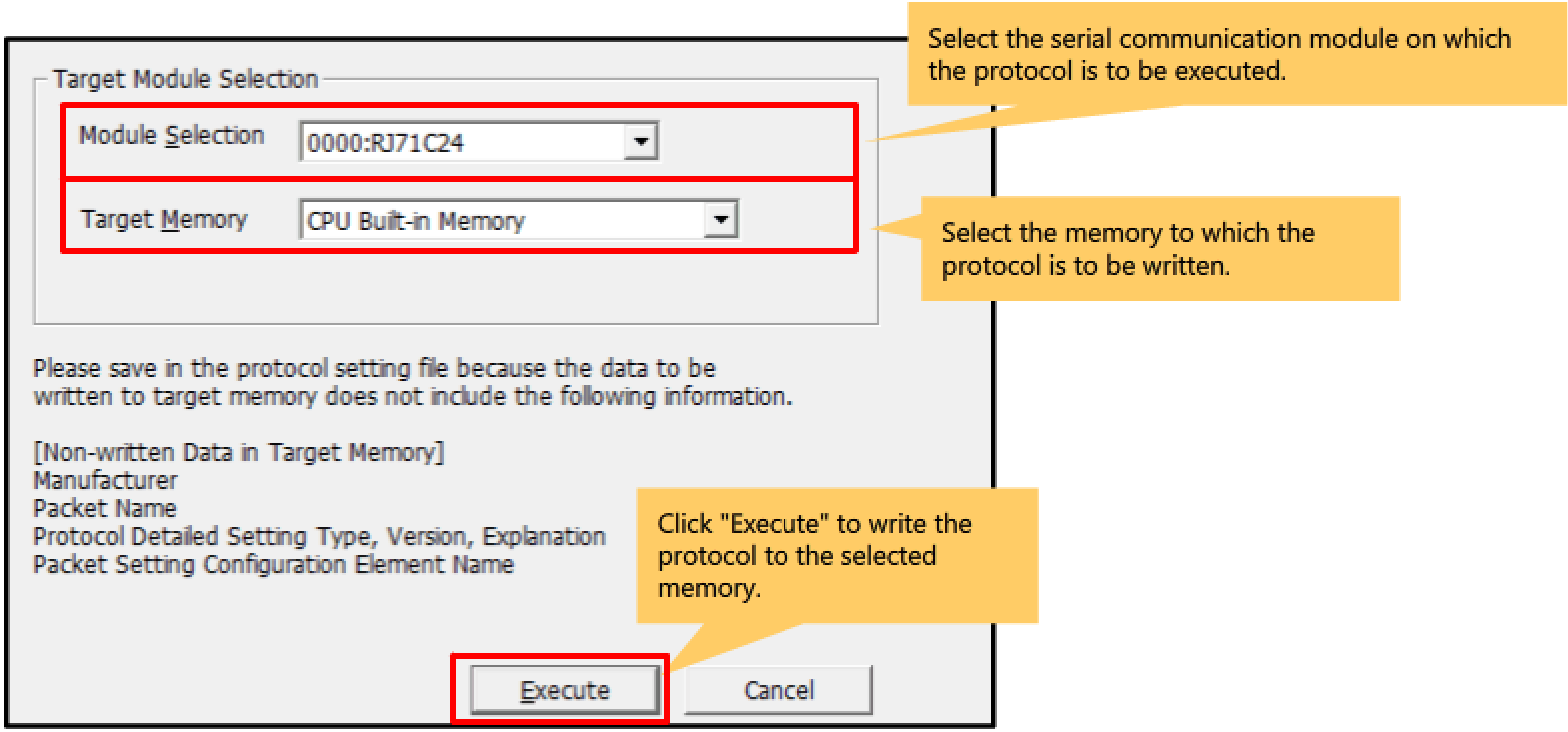
"Packet Setting" window (receive packet)

3.3.6 Saving and Writing Created Protocols

To save the created protocol in a protocol setting file, select "File" → "Save as" in the "Predefined Protocol Support Function" window.

The created protocol is written on the CPU built-in memory, the SD memory card, or the serial communication module. Once the protocol is written on the CPU built-in memory, rewriting the protocol is not required even after replacing the serial communication module.

Select "Write to Module" from "Online" on the "Predefined Protocol Support Function" window to write the protocol.



Target Module Selection

Module Selection: 0000:RJ71C24

Target Memory: CPU Built-in Memory

Please save in the protocol setting file because the data to be written to target memory does not include the following information.

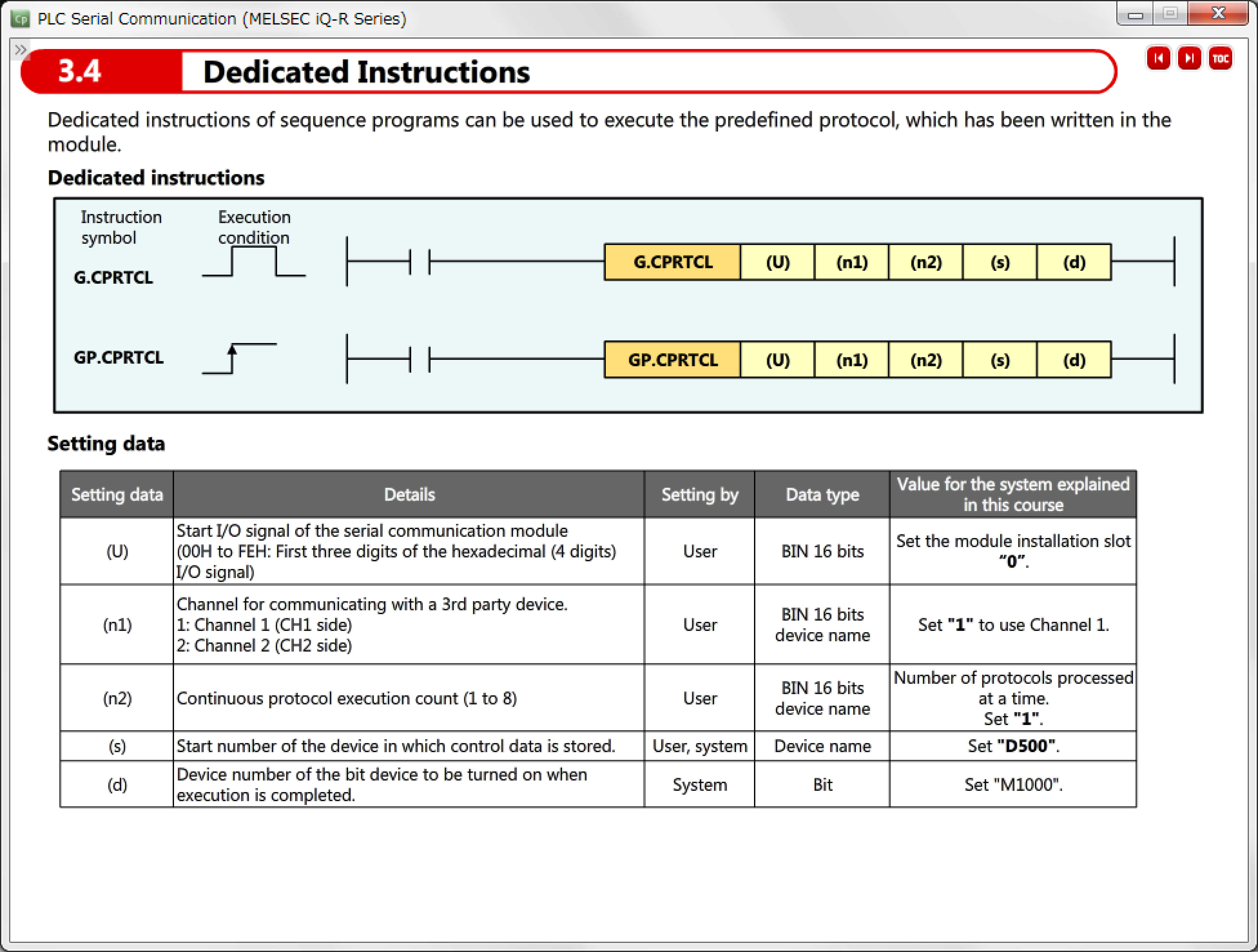
[Non-written Data in Target Memory]
Manufacturer
Packet Name
Protocol Detailed Setting Type, Version, Explanation
Packet Setting Configuration Element Name

Select the serial communication module on which the protocol is to be executed.

Select the memory to which the protocol is to be written.

Click "Execute" to write the protocol to the selected memory.

"Module Write" window



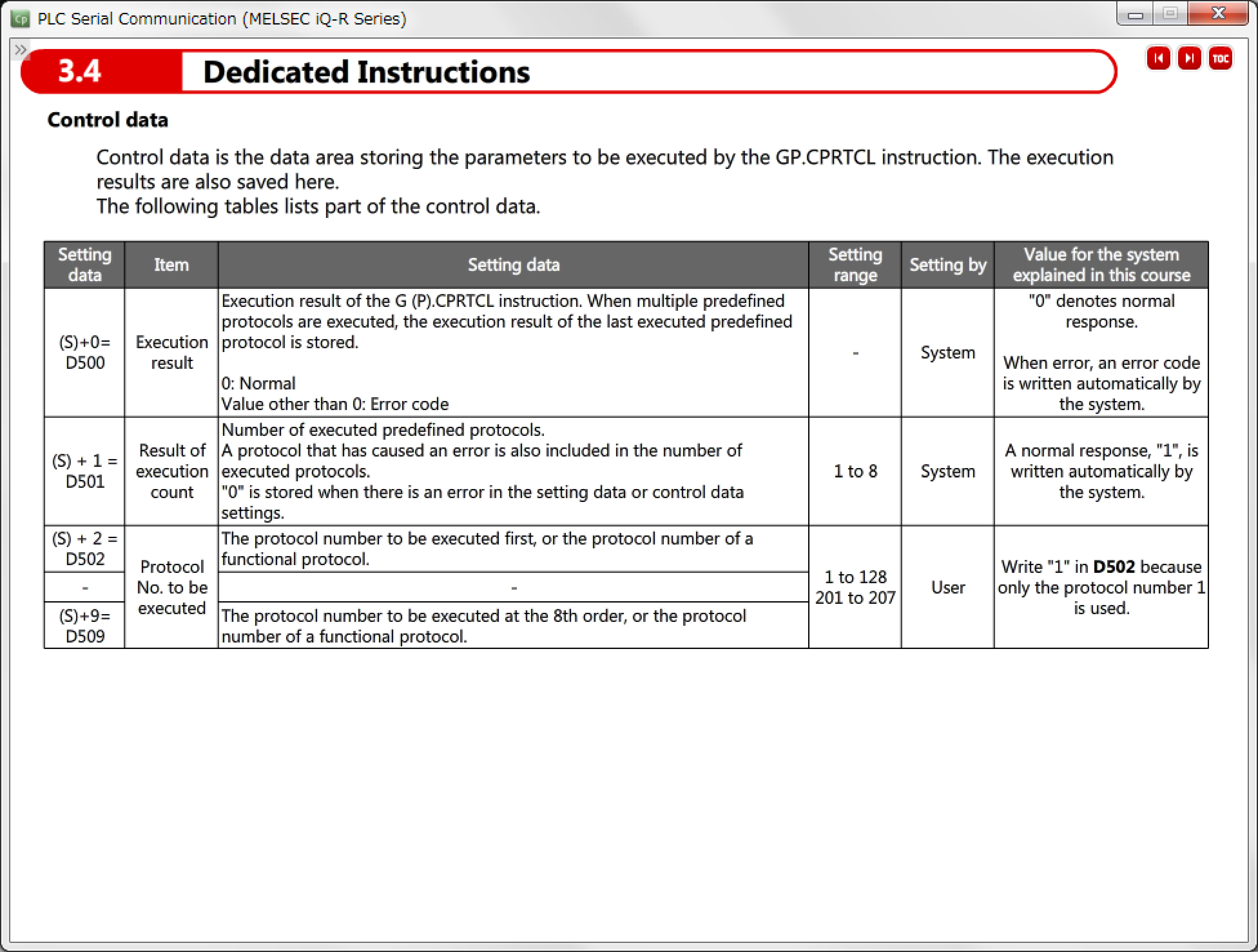
3.4 Dedicated Instructions

Dedicated instructions of sequence programs can be used to execute the predefined protocol, which has been written in the module.

Dedicated instructions

Setting data

Setting data	Details	Setting by	Data type	Value for the system explained in this course
(U)	Start I/O signal of the serial communication module (00H to FEH: First three digits of the hexadecimal (4 digits) I/O signal)	User	BIN 16 bits	Set the module installation slot **"0"**.
(n1)	Channel for communicating with a 3rd party device. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	User	BIN 16 bits device name	Set **"1"** to use Channel 1.
(n2)	Continuous protocol execution count (1 to 8)	User	BIN 16 bits device name	Number of protocols processed at a time. Set **"1"**.
(s)	Start number of the device in which control data is stored.	User, system	Device name	Set **"D500"**.
(d)	Device number of the bit device to be turned on when execution is completed.	System	Bit	Set "M1000".



3.4 Dedicated Instructions

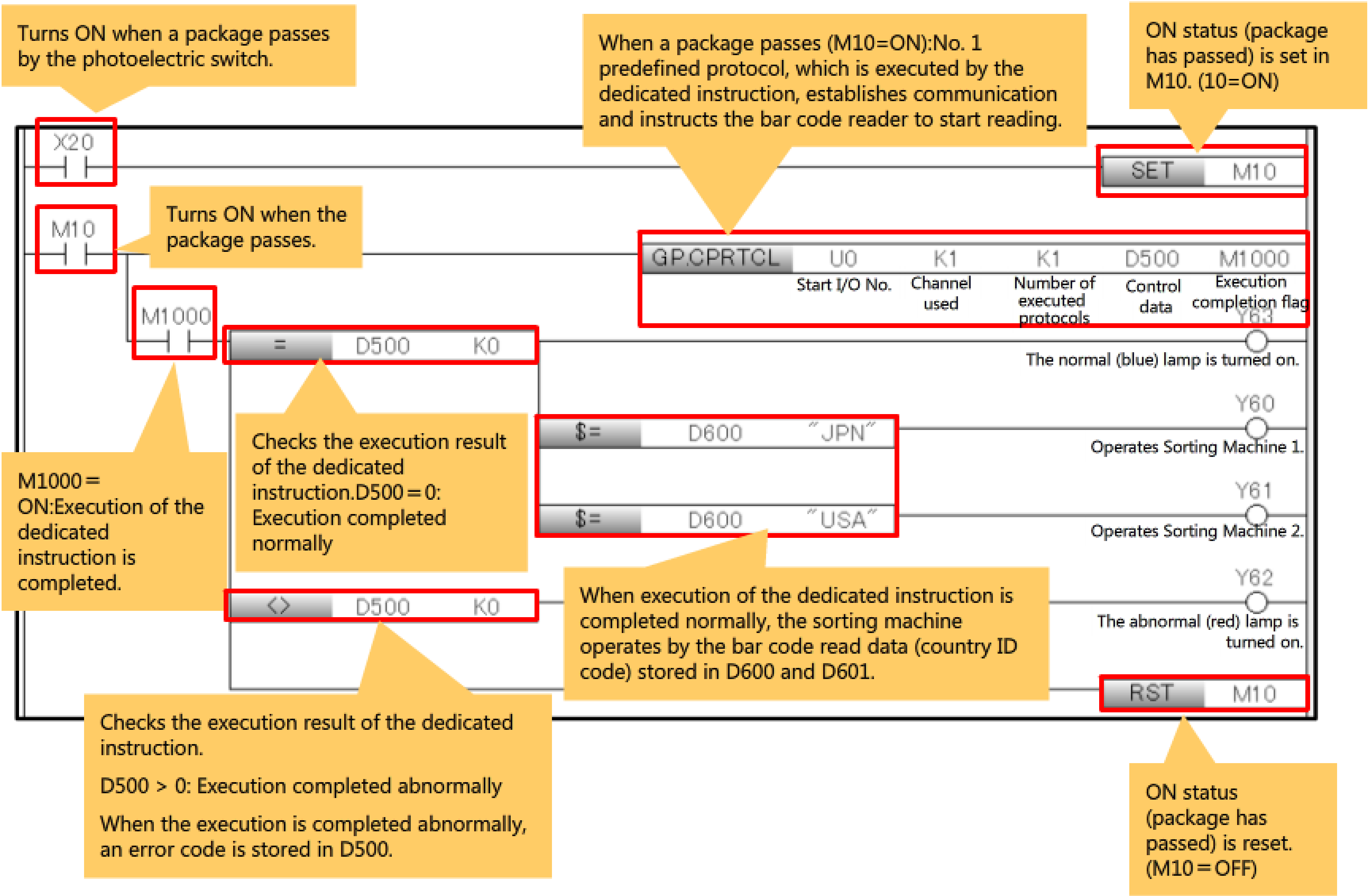
Control data

Control data is the data area storing the parameters to be executed by the GP.CPRTCL instruction. The execution results are also saved here.
The following tables lists part of the control data.

Setting data	Item	Setting data	Setting range	Setting by	Value for the system explained in this course
(S)+0= D500	Execution result	Execution result of the G (P).CPRTCL instruction. When multiple predefined protocols are executed, the execution result of the last executed predefined protocol is stored. 0: Normal Value other than 0: Error code	-	System	"0" denotes normal response. When error, an error code is written automatically by the system.
(S) + 1 = D501	Result of execution count	Number of executed predefined protocols. A protocol that has caused an error is also included in the number of executed protocols. "0" is stored when there is an error in the setting data or control data settings.	1 to 8	System	A normal response, "1", is written automatically by the system.
(S) + 2 = D502	Protocol No. to be executed	The protocol number to be executed first, or the protocol number of a functional protocol.	1 to 128 201 to 207	User	Write "1" in **D502** because only the protocol number 1 is used.
-		-			
(S)+9= D509		The protocol number to be executed at the 8th order, or the protocol number of a functional protocol.			

3.4.1 Sequence Program

The following chart shows a sequence program using dedicated instructions.
When a package passes by the photoelectric switch, the predefined protocol setting that instructs the bar code reader to start reading is executed.


X20
SET M10
M10
GP.CPRTCL U0 K1 K1 D500 M1000
Start I/O No.
Channel used
Number of executed protocols
Control data
Execution completion flag
M1000
= D500 K0
Y63
The normal (blue) lamp is turned on.
$= D600 "JPN"
Y60
Operates Sorting Machine 1.
$= D600 "USA"
Y61
Operates Sorting Machine 2.
<> D500 K0
Y62
The abnormal (red) lamp is turned on.
RST M10

Turns ON when a package passes by the photoelectric switch.

When a package passes (M10=ON):No. 1 predefined protocol, which is executed by the dedicated instruction, establishes communication and instructs the bar code reader to start reading.

ON status (package has passed) is set in M10. (10=ON)

Turns ON when the package passes.

M1000＝ON:Execution of the dedicated instruction is completed.

Checks the execution result of the dedicated instruction.D500＝0: Execution completed normally

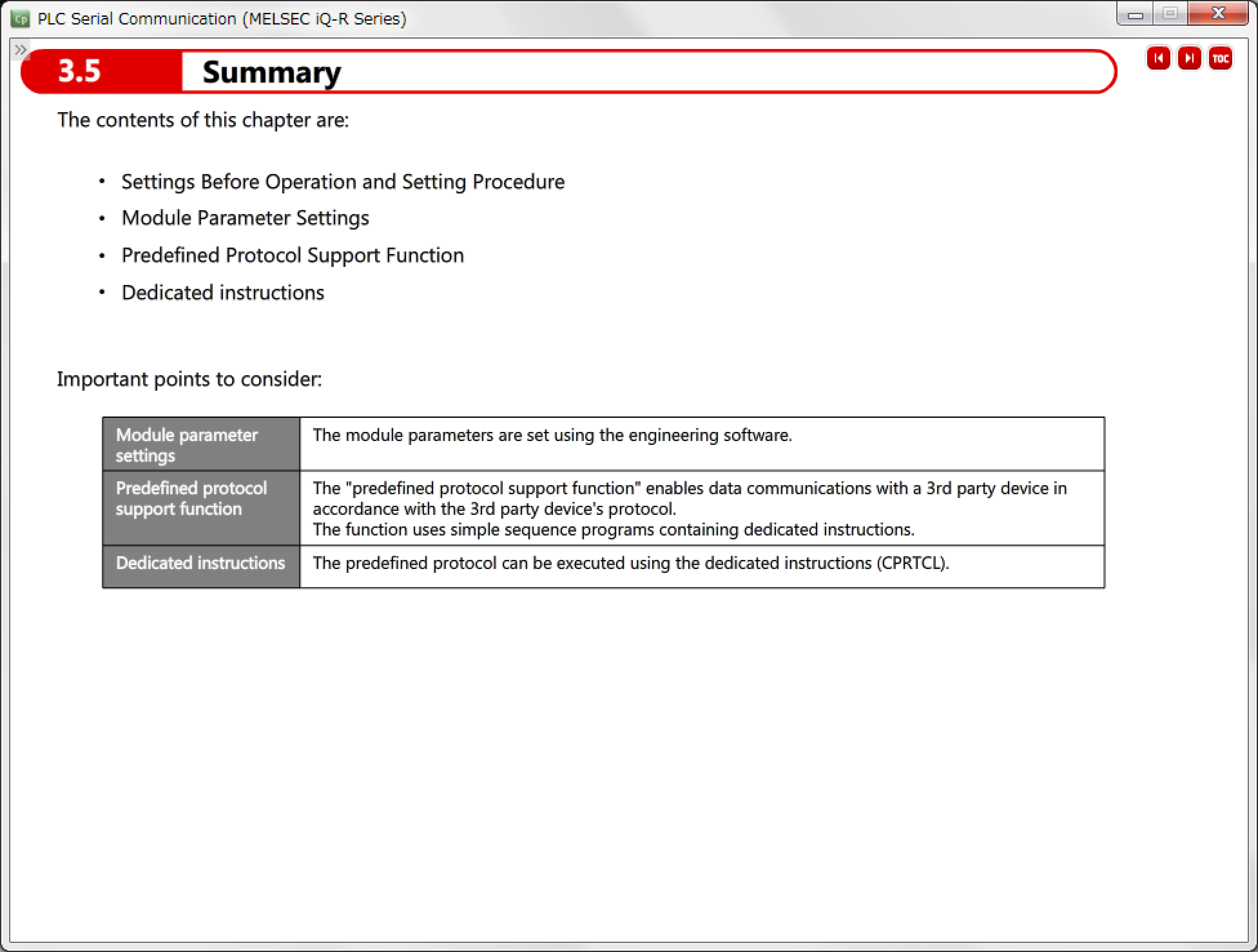
When execution of the dedicated instruction is completed normally, the sorting machine operates by the bar code read data (country ID code) stored in D600 and D601.

Checks the execution result of the dedicated instruction.

D500 > 0: Execution completed abnormally

When the execution is completed abnormally, an error code is stored in D500.

ON status (package has passed) is reset. (M10＝OFF)



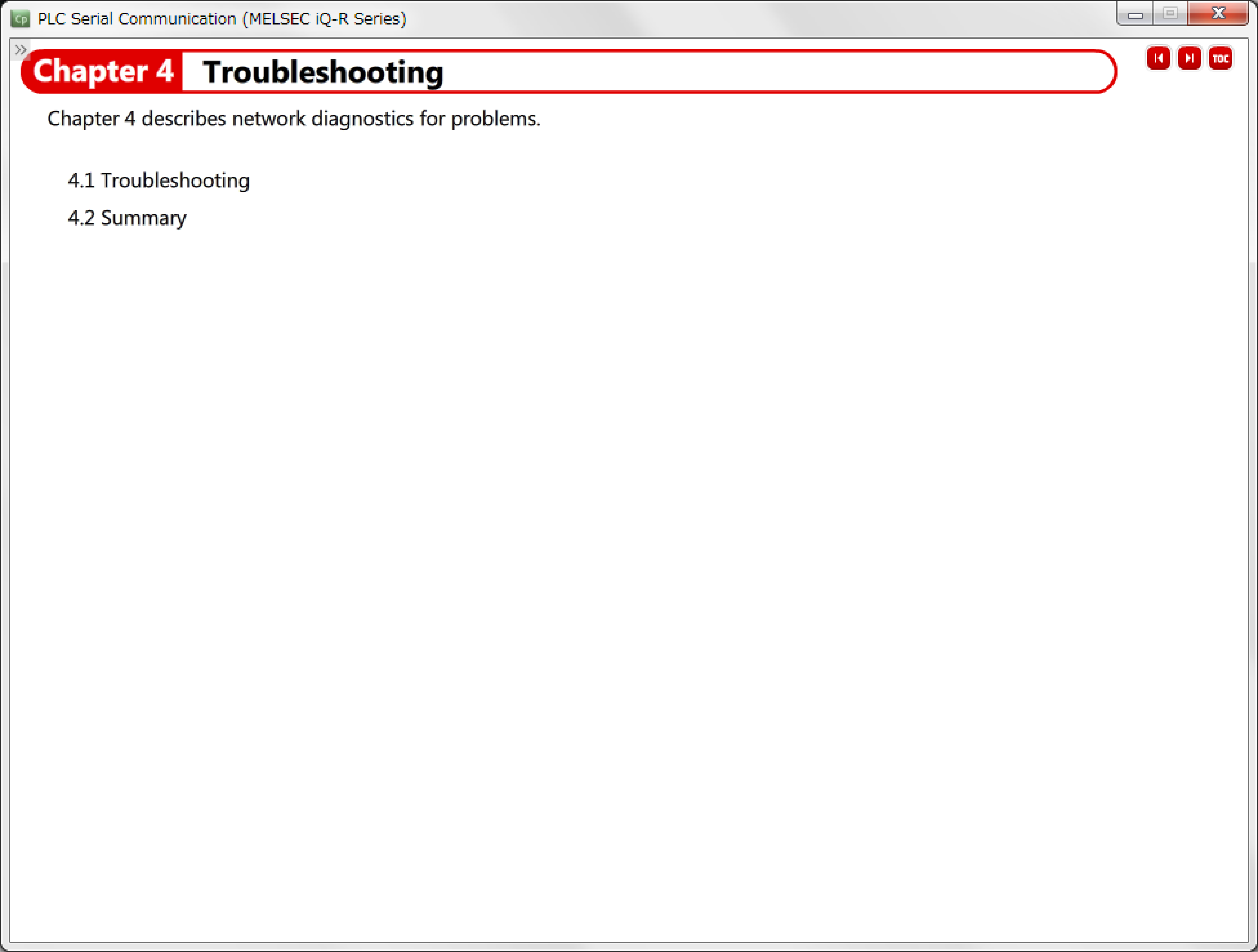
3.5 Summary

The contents of this chapter are:

- Settings Before Operation and Setting Procedure
- Module Parameter Settings
- Predefined Protocol Support Function
- Dedicated instructions

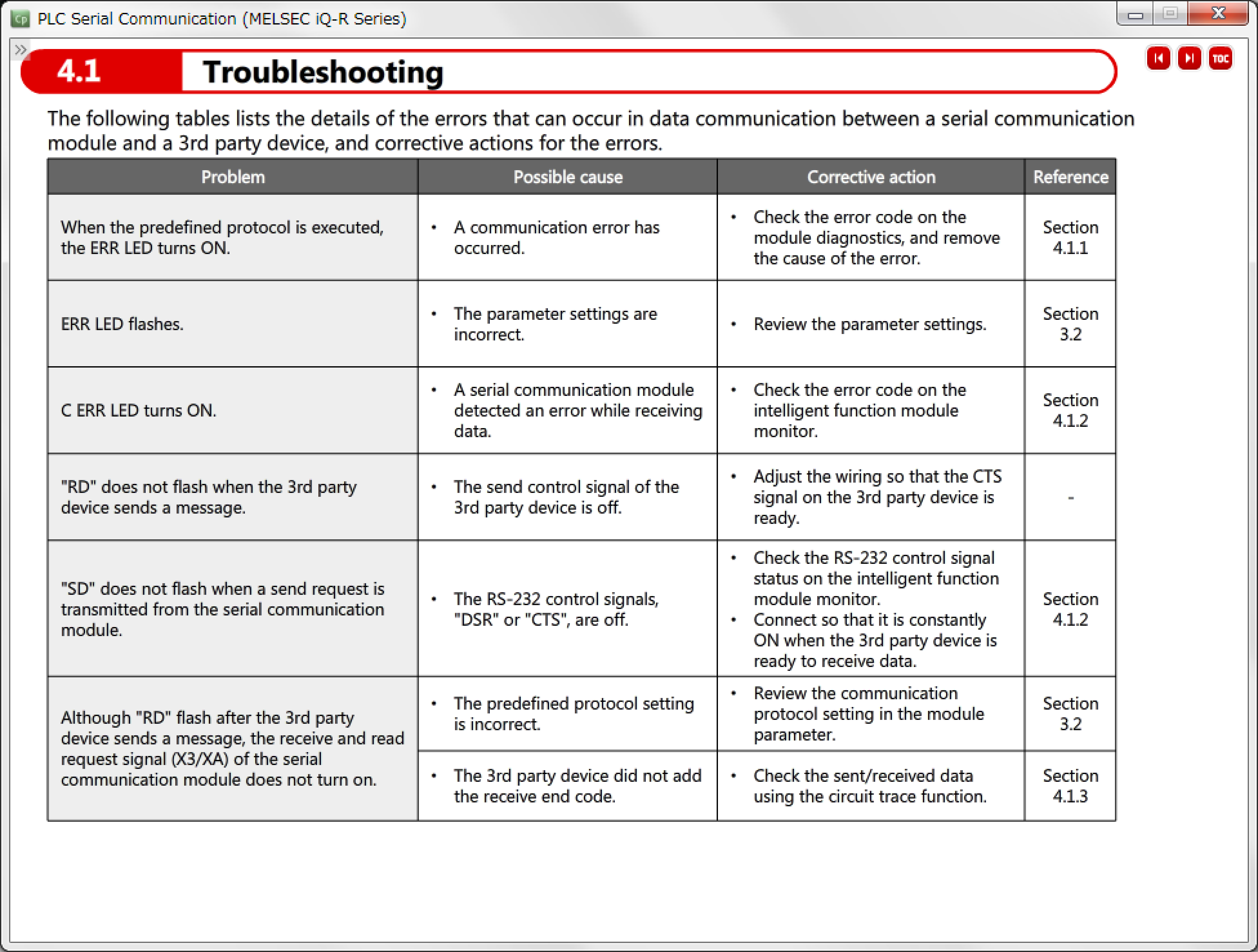
Important points to consider:

Module parameter settings	The module parameters are set using the engineering software.
Predefined protocol support function	The "predefined protocol support function" enables data communications with a 3rd party device in accordance with the 3rd party device's protocol. The function uses simple sequence programs containing dedicated instructions.
Dedicated instructions	The predefined protocol can be executed using the dedicated instructions (CPRTCL).



Chapter 4 Troubleshooting

Chapter 4 describes network diagnostics for problems.



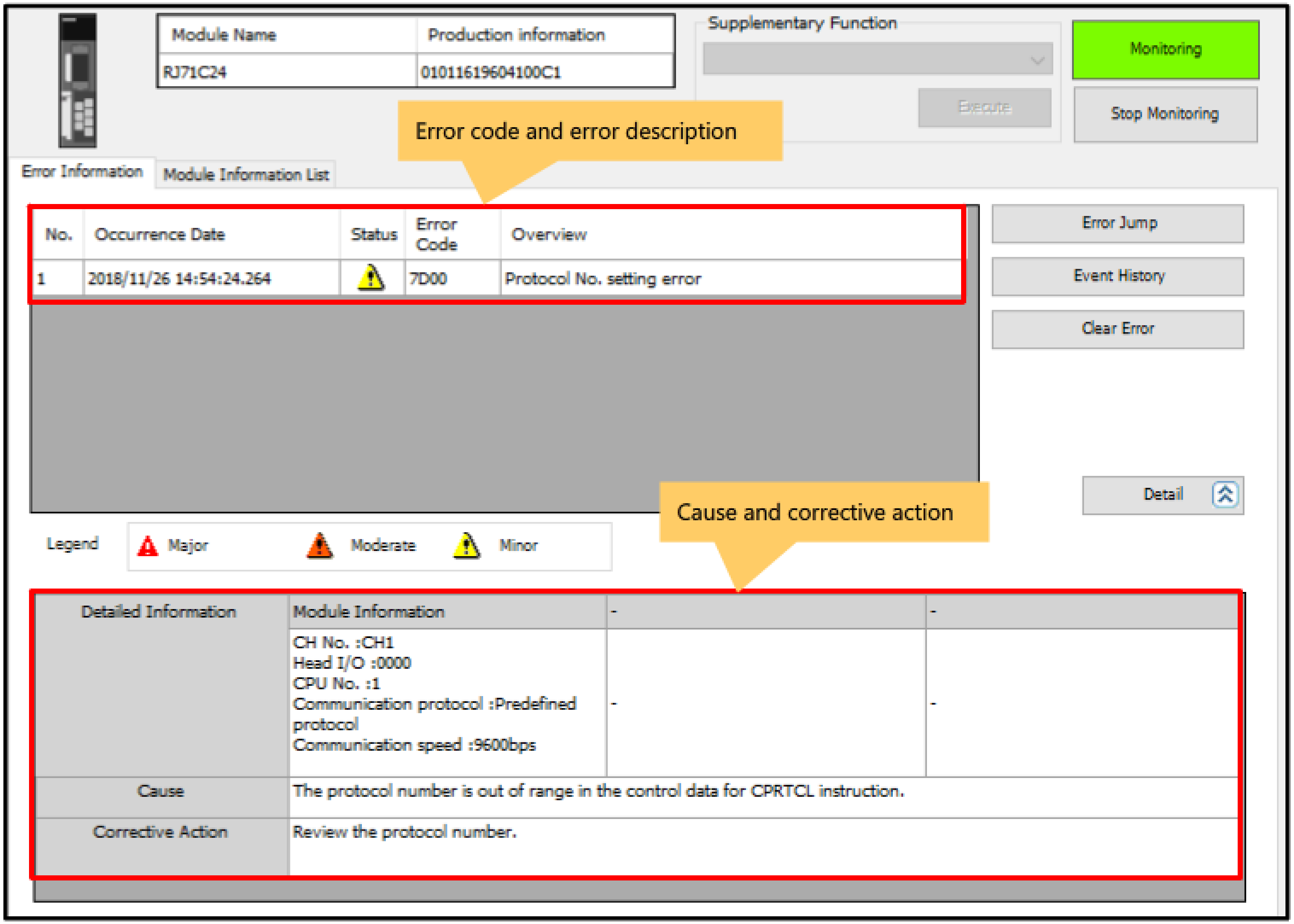
4.1 Troubleshooting

The following tables lists the details of the errors that can occur in data communication between a serial communication module and a 3rd party device, and corrective actions for the errors.

Problem	Possible cause	Corrective action	Reference
When the predefined protocol is executed, the ERR LED turns ON.	• A communication error has occurred.	• Check the error code on the module diagnostics, and remove the cause of the error.	Section 4.1.1
ERR LED flashes.	• The parameter settings are incorrect.	• Review the parameter settings.	Section 3.2
C ERR LED turns ON.	• A serial communication module detected an error while receiving data.	• Check the error code on the intelligent function module monitor.	Section 4.1.2
"RD" does not flash when the 3rd party device sends a message.	• The send control signal of the 3rd party device is off.	• Adjust the wiring so that the CTS signal on the 3rd party device is ready.	-
"SD" does not flash when a send request is transmitted from the serial communication module.	• The RS-232 control signals, "DSR" or "CTS", are off.	• Check the RS-232 control signal status on the intelligent function module monitor. • Connect so that it is constantly ON when the 3rd party device is ready to receive data.	Section 4.1.2
Although "RD" flash after the 3rd party device sends a message, the receive and read request signal (X3/XA) of the serial communication module does not turn on.	• The predefined protocol setting is incorrect.	• Review the communication protocol setting in the module parameter.	Section 3.2
	• The 3rd party device did not add the receive end code.	• Check the sent/received data using the circuit trace function.	Section 4.1.3

4.1.1 Checking for Errors by the Module Diagnostics

Details, causes, and corrective actions of errors occurred can be checked using the module diagnostic function of GX Works3.
To open the "Module Diagnostics" window on GX Works3, select "System Monitor" from "Diagnostics".



Module Name	Production information
RJ71C24	01011619604100C1

Error code and error description

No.	Occurrence Date	Status	Error Code	Overview
1	2018/11/26 14:54:24.264	⚠	7D00	Protocol No. setting error

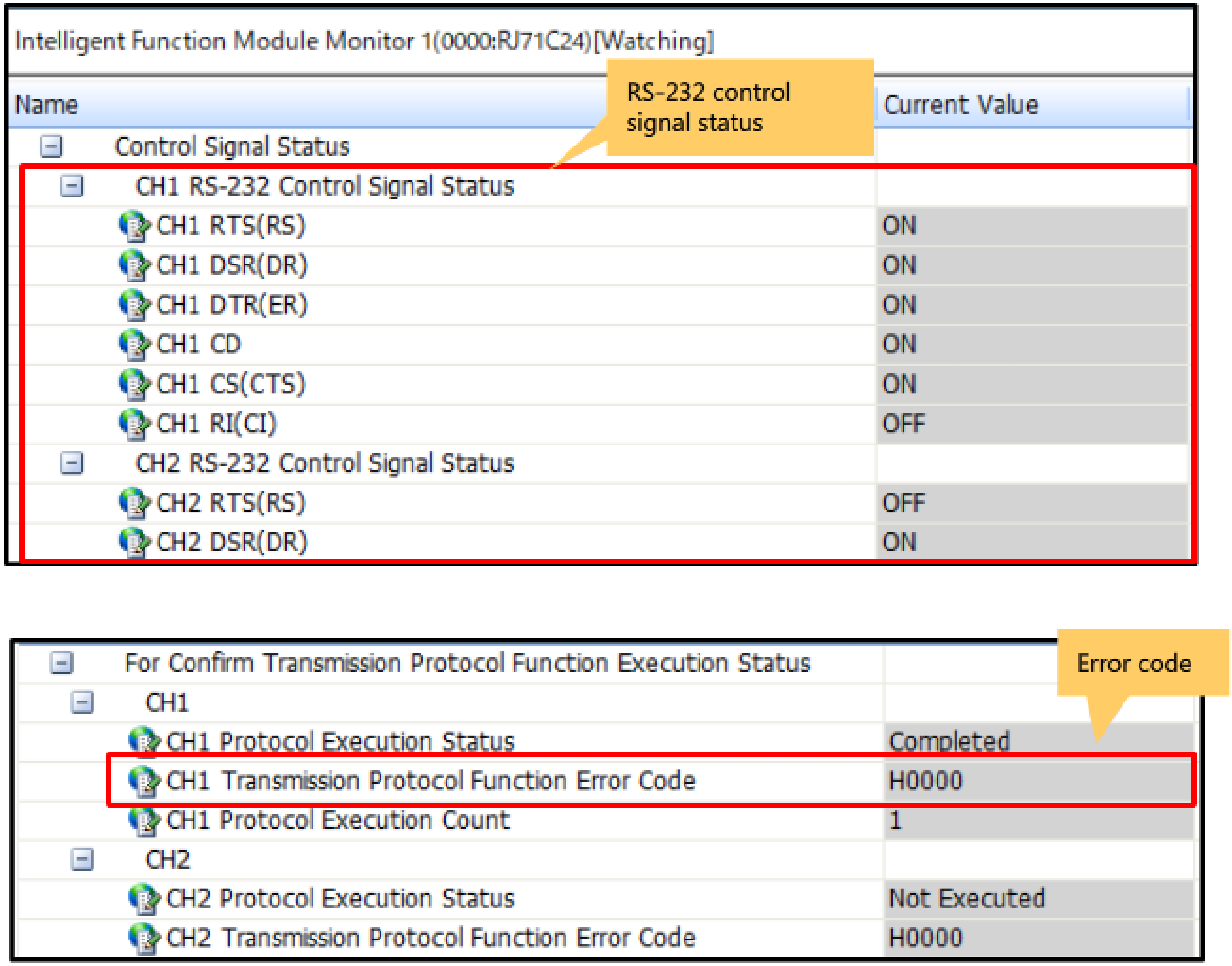
Cause and corrective action

Detailed Information	Module Information	-	-
	CH No. :CH1 Head I/O :0000 CPU No. :1 Communication protocol :Predefined protocol Communication speed :9600bps	-	-
Cause	The protocol number is out of range in the control data for CPRTCL instruction.		
Corrective Action	Review the protocol number.		

"Module Diagnostics" window

4.1.2 Intelligent Function Module Monitor

The serial communication module status, including the RS-232 control signal status and error codes, can be checked on the the intelligent function module monitor.
To execute this function with GX Works3, register the serial communication module to be monitored on the "Intelligent Function Module Monitor" window.



Intelligent Function Module Monitor 1(0000:RJ71C24)[Watching]

Name	Current Value
⊟ Control Signal Status	
⊟ CH1 RS-232 Control Signal Status	
CH1 RTS(RS)	ON
CH1 DSR(DR)	ON
CH1 DTR(ER)	ON
CH1 CD	ON
CH1 CS(CTS)	ON
CH1 RI(CI)	OFF
⊟ CH2 RS-232 Control Signal Status	
CH2 RTS(RS)	OFF
CH2 DSR(DR)	ON

RS-232 control signal status

⊟ For Confirm Transmission Protocol Function Execution Status	
⊟ CH1	
CH1 Protocol Execution Status	Completed
CH1 Transmission Protocol Function Error Code	H0000
CH1 Protocol Execution Count	1
⊟ CH2	
CH2 Protocol Execution Status	Not Executed
CH2 Transmission Protocol Function Error Code	H0000

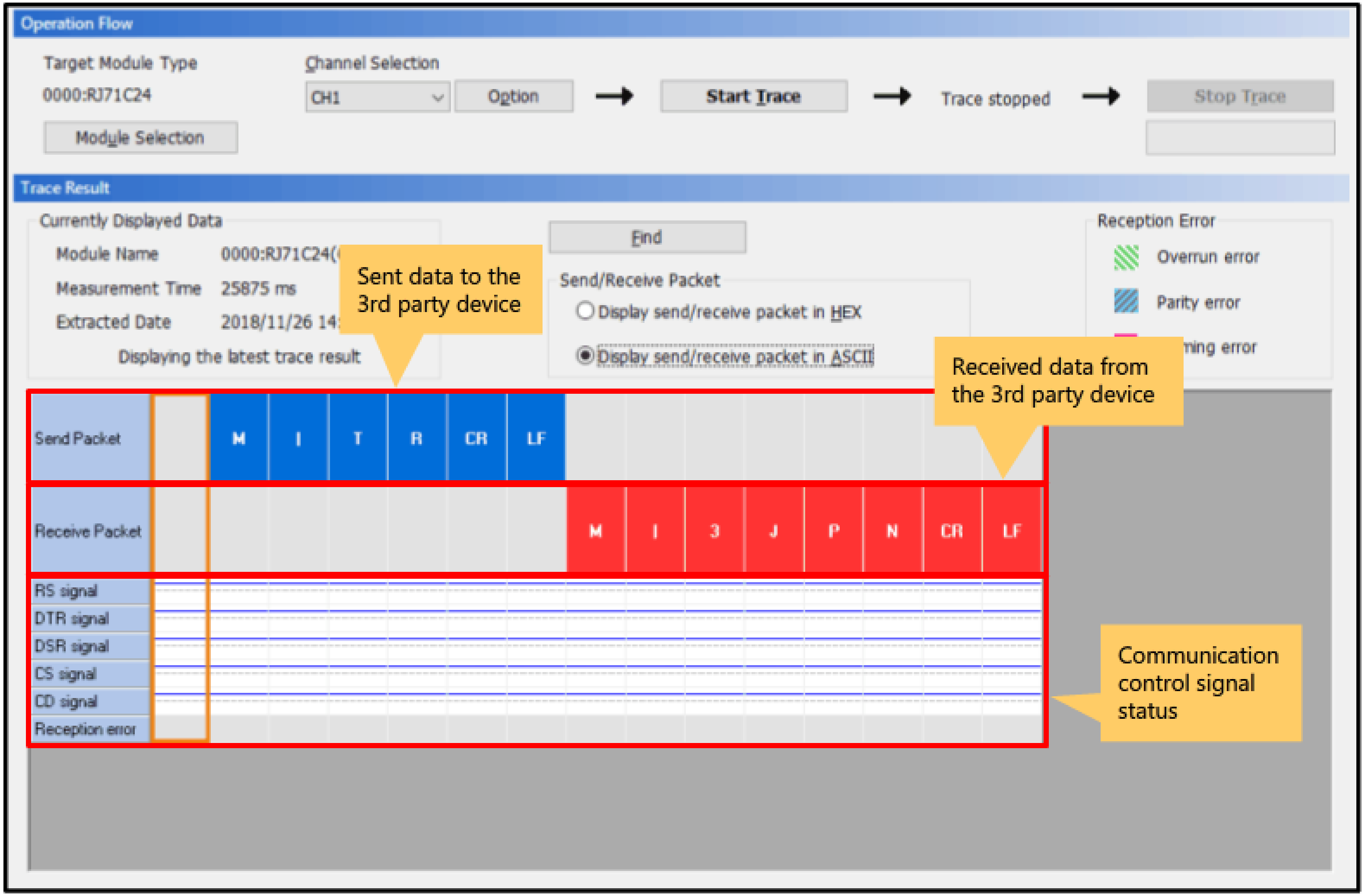
Error code

Intelligent Function Module Monitor

4.1.3 Checking the Sent/Received Data using the Circuit Trace

The circuit trace function enables to check whether data communications between a serial communication module and a 3rd party device have been performed as intended by temporarily recording the sent/received data and communication control signal statuses.

To execute this function, select "Circuit Trace" from "Tool" and open the "Circuit Trace" window on GX Works3.



Operation Flow

Target Module Type: 0000:RJ71C24

Channel Selection: CH1

Trace Result

Currently Displayed Data

Module Name: 0000:RJ71C24(

Measurement Time: 25875 ms

Extracted Date: 2018/11/26 14:

Displaying the latest trace result

Send/Receive Packet

○ Display send/receive packet in HEX

● Display send/receive packet in ASCII

Reception Error

▨ Overrun error

▨ Parity error

▨ ...ming error

Send Packet		M	I	T	R	CR	LF									
Receive Packet								M	I	3	J	P	N	CR	LF	

RS signal
DTR signal
DSR signal
CS signal
CD signal
Reception error

Sent data to the 3rd party device

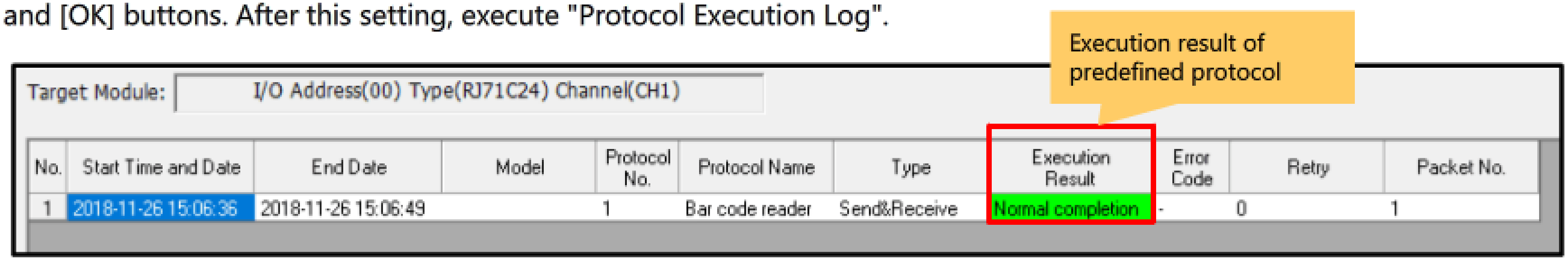
Received data from the 3rd party device

Communication control signal status

"Circuit Trace" window

4.1.4 Protocol Execution Log

The detailed predefined protocol execution status and results can be checked on the "Protocol Execution Log" window of GX Works3.
To execute this function, open the" Predefined Protocol Support Function" window and select "Debugging Support Function" and "Module Selection". On the "Module Selection" window, select a module to be debugged and click the [Set] and [OK] buttons. After this setting, execute "Protocol Execution Log".



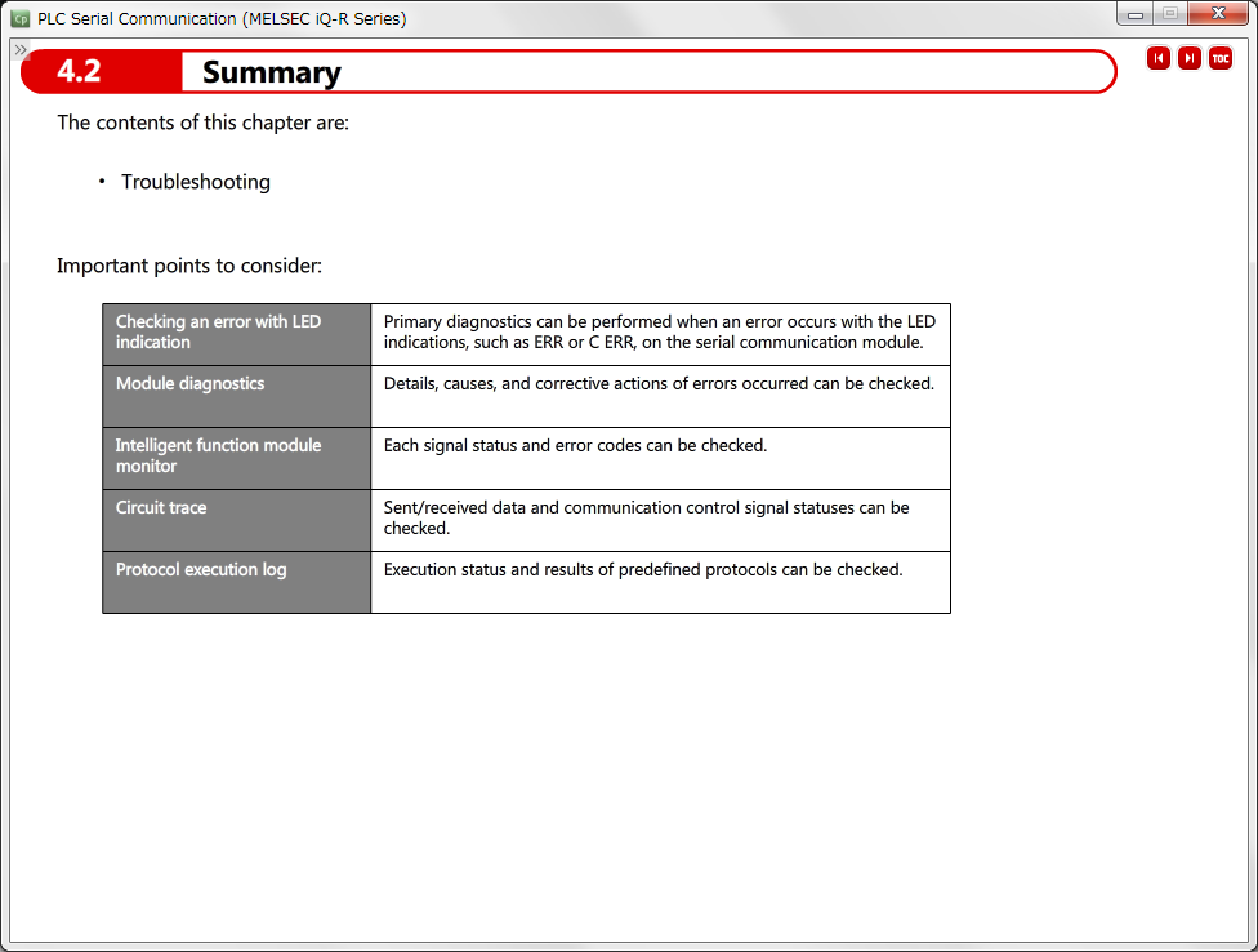
Execution result of predefined protocol

Target Module: I/O Address(00) Type(RJ71C24) Channel(CH1)

No.	Start Time and Date	End Date	Model	Protocol No.	Protocol Name	Type	Execution Result	Error Code	Retry	Packet No.
1	2018-11-26 15:06:36	2018-11-26 15:06:49		1	Bar code reader	Send&Receive	Normal completion	-	0	1

"Protocol Execution Log" window

Protocol execution log is displayed only when the protocol execution is completed with an error at the initial status.
To display the execution statuses and execution logs of all the protocols with GX Works3, on the Project view of the Navigation window, select "Parameter" → "Module Information" → "RJ71C24" to open the "Module Parameter" window. On the "Module Parameter" window, set "Protocol execution history specification option" to "1: All protocol execution status and execution history" in "Basic Settings".



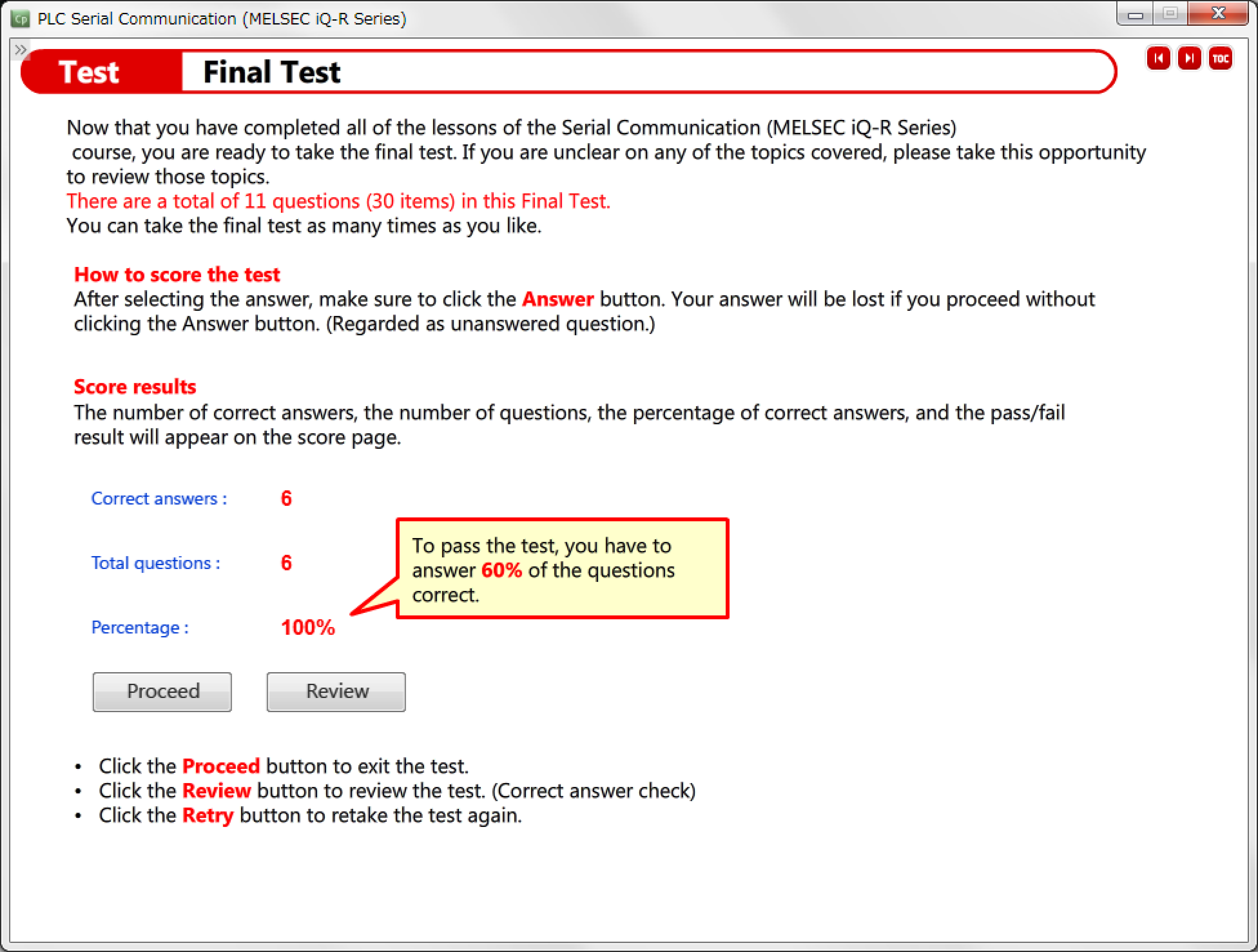
4.2 Summary

The contents of this chapter are:

- Troubleshooting

Important points to consider:

Checking an error with LED indication	Primary diagnostics can be performed when an error occurs with the LED indications, such as ERR or C ERR, on the serial communication module.
Module diagnostics	Details, causes, and corrective actions of errors occurred can be checked.
Intelligent function module monitor	Each signal status and error codes can be checked.
Circuit trace	Sent/received data and communication control signal statuses can be checked.
Protocol execution log	Execution status and results of predefined protocols can be checked.



Test Final Test

Now that you have completed all of the lessons of the Serial Communication (MELSEC iQ-R Series) course, you are ready to take the final test. If you are unclear on any of the topics covered, please take this opportunity to review those topics.
There are a total of 11 questions (30 items) in this Final Test.
You can take the final test as many times as you like.

How to score the test

After selecting the answer, make sure to click the **Answer** button. Your answer will be lost if you proceed without clicking the Answer button. (Regarded as unanswered question.)

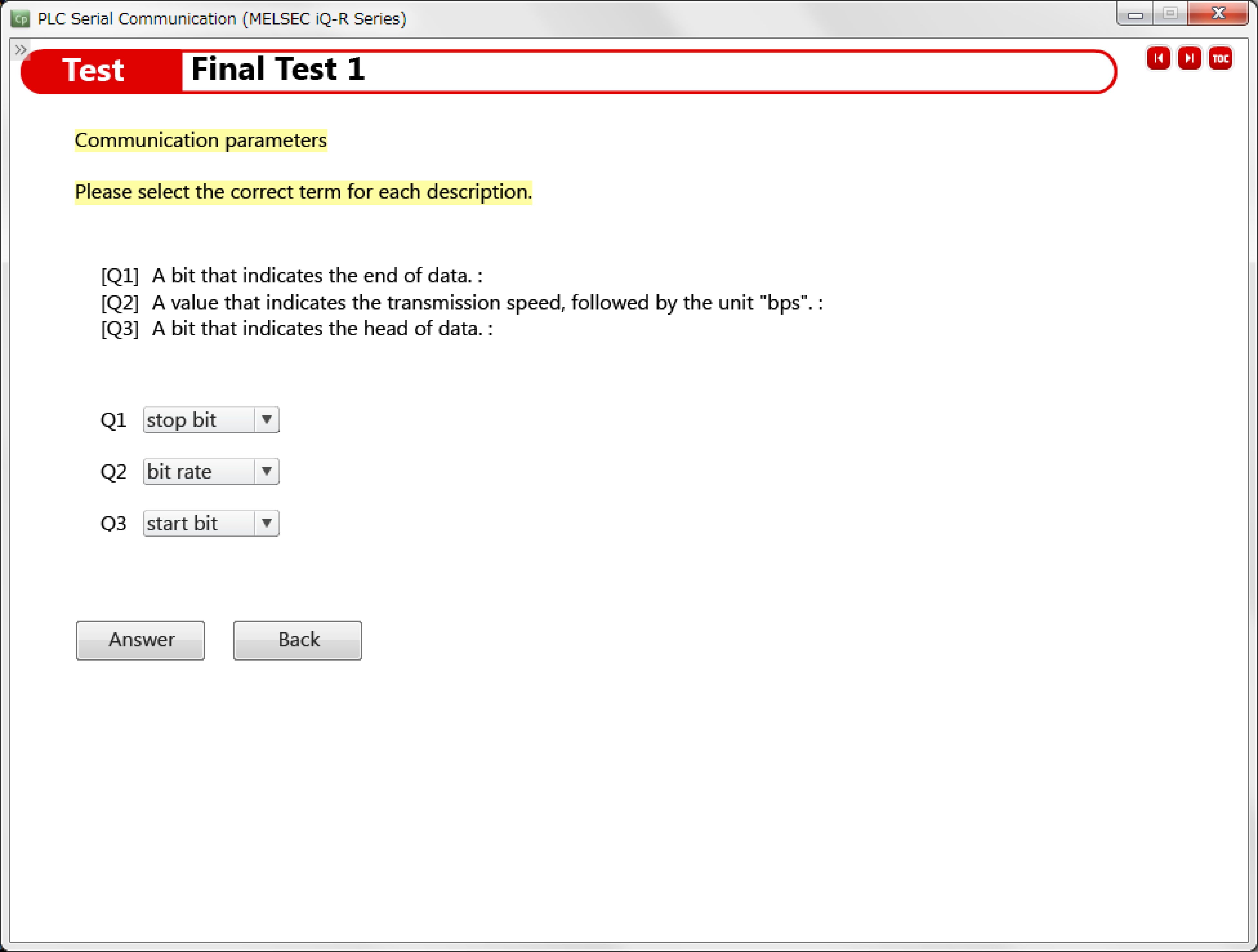
Score results

The number of correct answers, the number of questions, the percentage of correct answers, and the pass/fail result will appear on the score page.

Correct answers :	**6**
Total questions :	**6**
Percentage :	**100%**

To pass the test, you have to answer **60%** of the questions correct.

- Click the **Proceed** button to exit the test.
- Click the **Review** button to review the test. (Correct answer check)
- Click the **Retry** button to retake the test again.



Test Final Test 1

Communication parameters

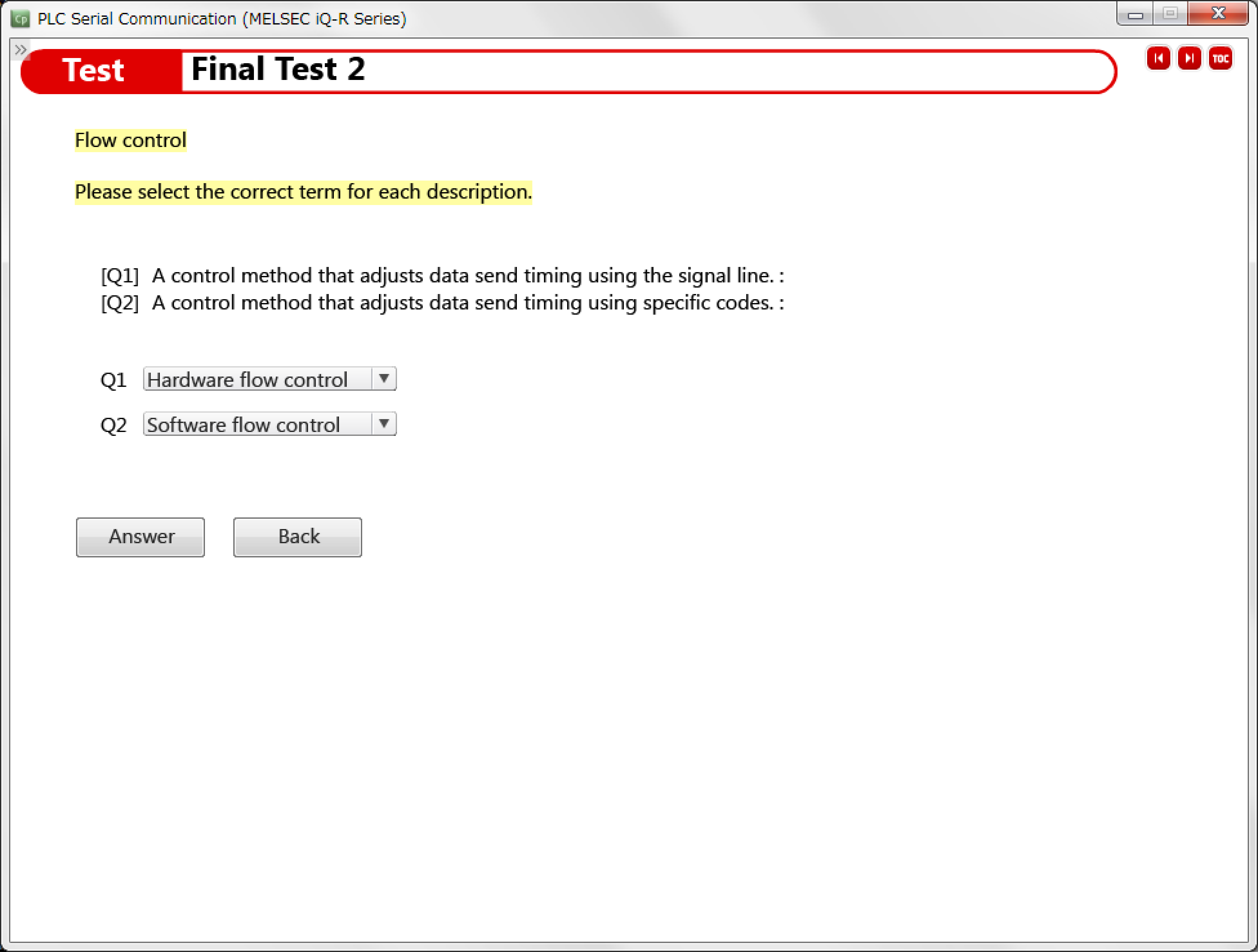
Please select the correct term for each description.

[Q1] A bit that indicates the end of data. :
[Q2] A value that indicates the transmission speed, followed by the unit "bps". :
[Q3] A bit that indicates the head of data. :

Q1 stop bit ▼

Q2 bit rate ▼

Q3 start bit ▼



Test

Final Test 2

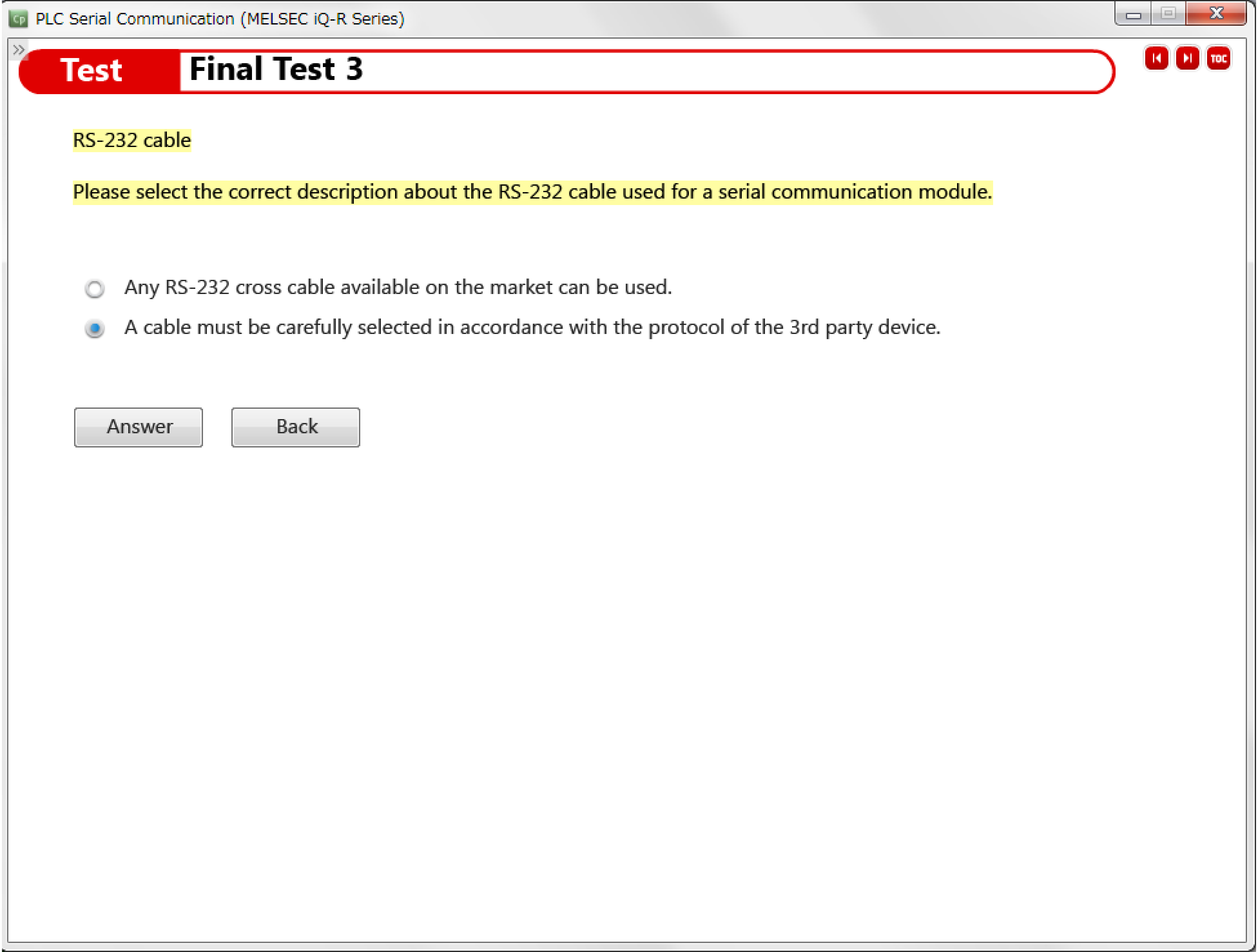
Flow control

Please select the correct term for each description.

[Q1] A control method that adjusts data send timing using the signal line. :
[Q2] A control method that adjusts data send timing using specific codes. :

Q1 Hardware flow control ▼

Q2 Software flow control ▼



Test

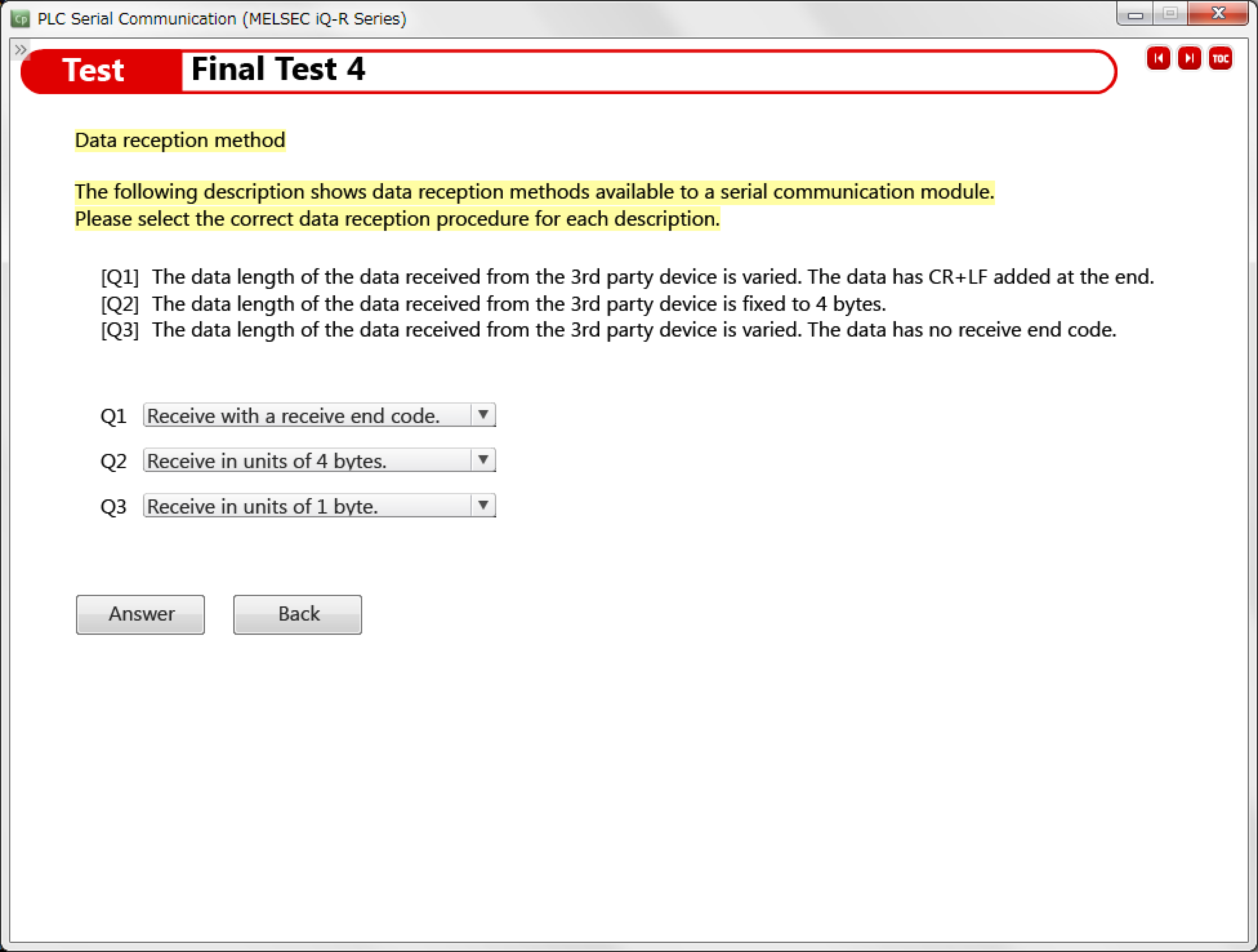
Final Test 3

RS-232 cable

Please select the correct description about the RS-232 cable used for a serial communication module.

○ Any RS-232 cross cable available on the market can be used.

● A cable must be carefully selected in accordance with the protocol of the 3rd party device.



Test

Final Test 4

Data reception method

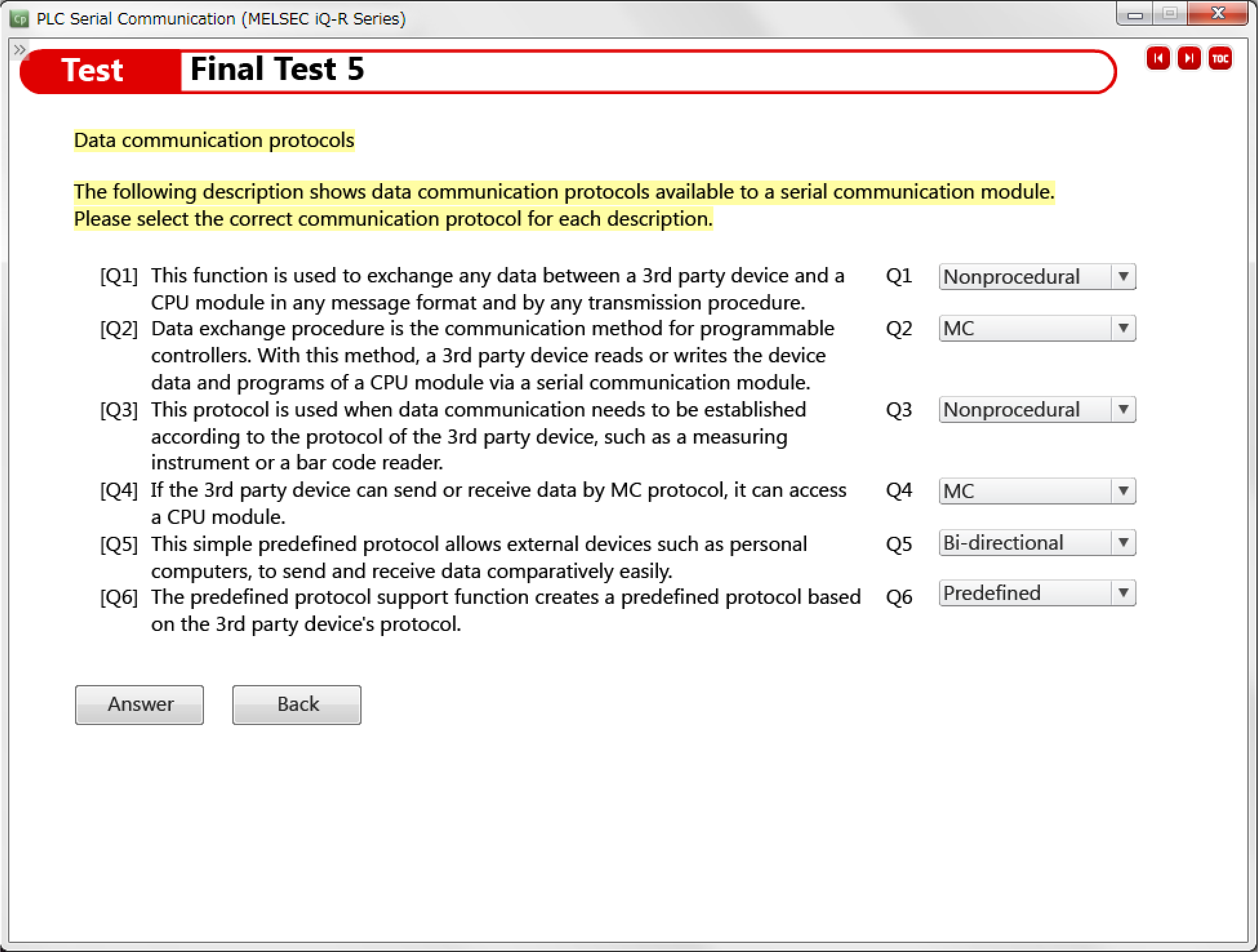
The following description shows data reception methods available to a serial communication module.
Please select the correct data reception procedure for each description.

[Q1] The data length of the data received from the 3rd party device is varied. The data has CR+LF added at the end.
[Q2] The data length of the data received from the 3rd party device is fixed to 4 bytes.
[Q3] The data length of the data received from the 3rd party device is varied. The data has no receive end code.

Q1 Receive with a receive end code. ▼

Q2 Receive in units of 4 bytes. ▼

Q3 Receive in units of 1 byte. ▼



Test Final Test 5

Data communication protocols

The following description shows data communication protocols available to a serial communication module.
Please select the correct communication protocol for each description.

[Q1] This function is used to exchange any data between a 3rd party device and a CPU module in any message format and by any transmission procedure.

[Q2] Data exchange procedure is the communication method for programmable controllers. With this method, a 3rd party device reads or writes the device data and programs of a CPU module via a serial communication module.

[Q3] This protocol is used when data communication needs to be established according to the protocol of the 3rd party device, such as a measuring instrument or a bar code reader.

[Q4] If the 3rd party device can send or receive data by MC protocol, it can access a CPU module.

[Q5] This simple predefined protocol allows external devices such as personal computers, to send and receive data comparatively easily.

[Q6] The predefined protocol support function creates a predefined protocol based on the 3rd party device's protocol.

Q1 Nonprocedural ▼

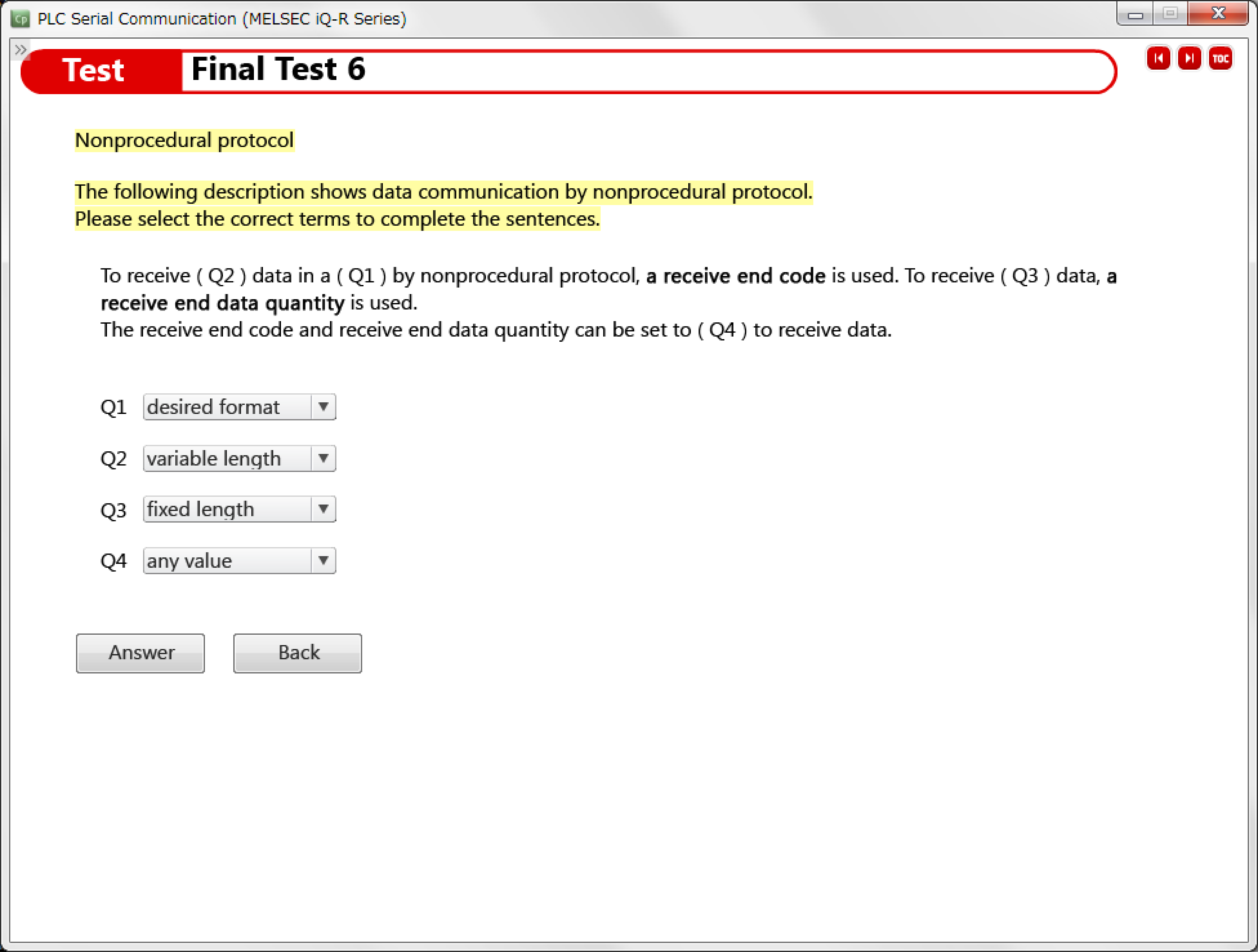
Q2 MC ▼

Q3 Nonprocedural ▼

Q4 MC ▼

Q5 Bi-directional ▼

Q6 Predefined ▼



Test Final Test 6

Nonprocedural protocol

The following description shows data communication by nonprocedural protocol.
Please select the correct terms to complete the sentences.

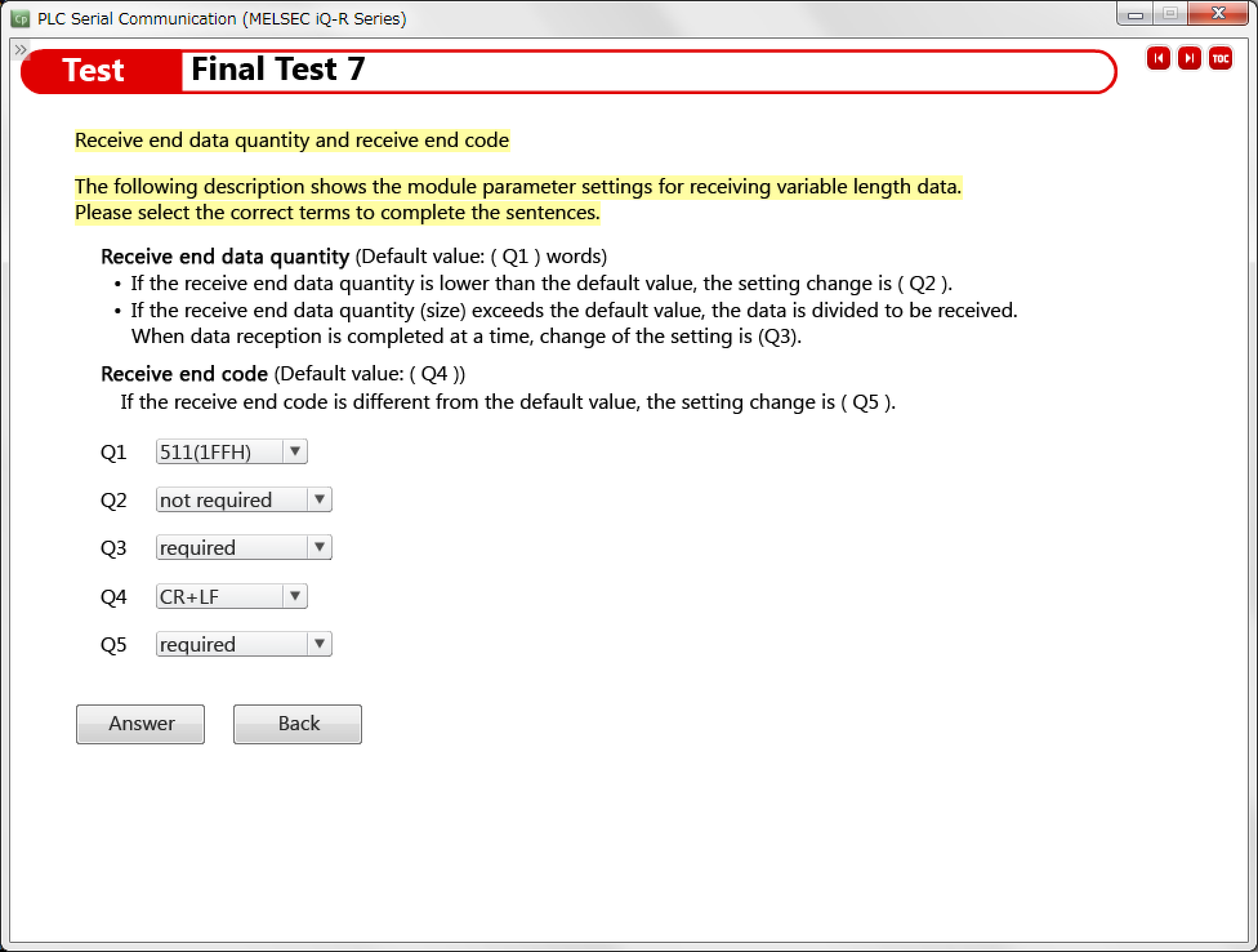
To receive (Q2) data in a (Q1) by nonprocedural protocol, **a receive end code** is used. To receive (Q3) data, **a receive end data quantity** is used.
The receive end code and receive end data quantity can be set to (Q4) to receive data.

Q1 desired format ▼

Q2 variable length ▼

Q3 fixed length ▼

Q4 any value ▼



Test Final Test 7

Receive end data quantity and receive end code

The following description shows the module parameter settings for receiving variable length data.
Please select the correct terms to complete the sentences.

Receive end data quantity (Default value: (Q1) words)

- If the receive end data quantity is lower than the default value, the setting change is (Q2).
- If the receive end data quantity (size) exceeds the default value, the data is divided to be received. When data reception is completed at a time, change of the setting is (Q3).

Receive end code (Default value: (Q4))

If the receive end code is different from the default value, the setting change is (Q5).

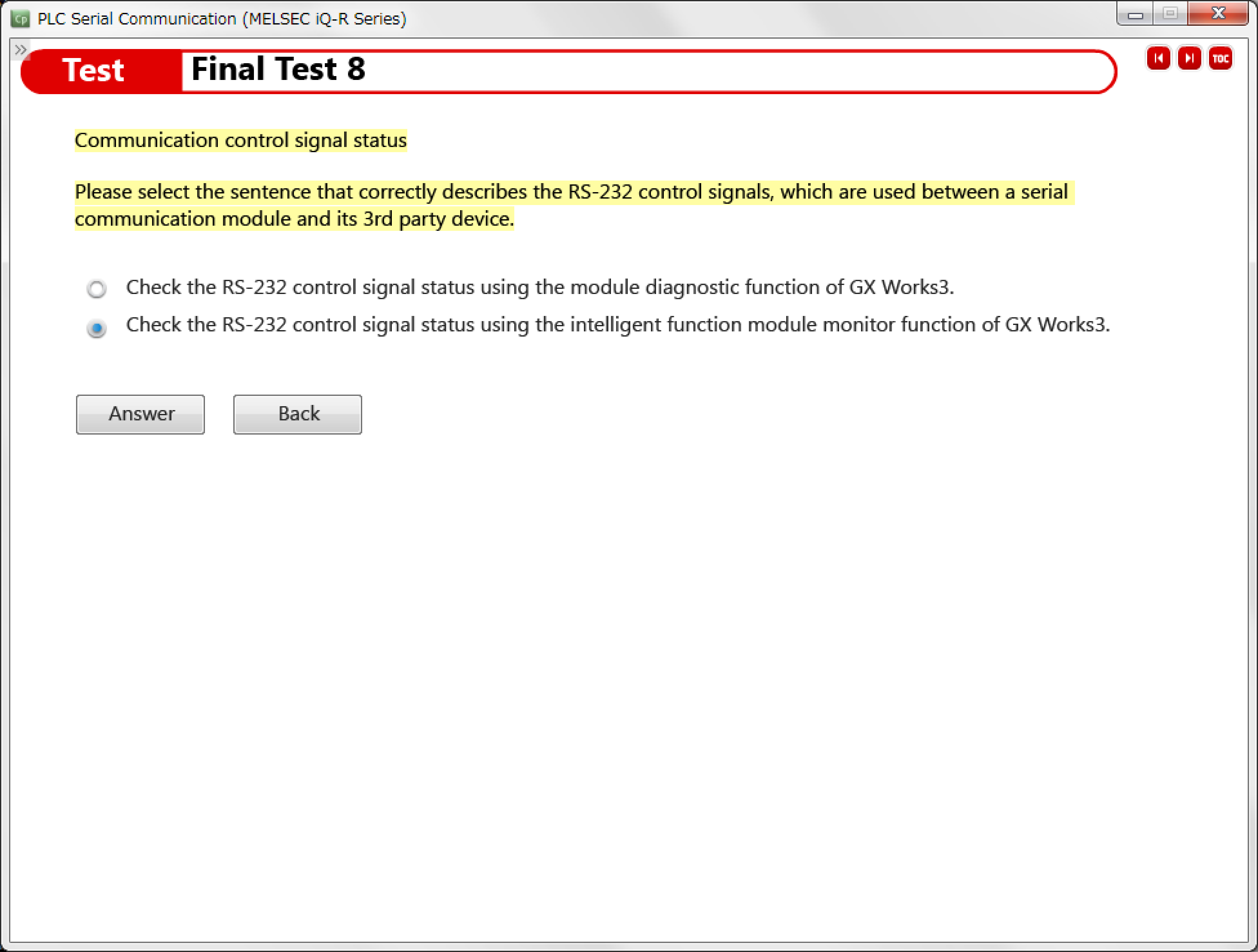
Q1 511(1FFH) ▼

Q2 not required ▼

Q3 required ▼

Q4 CR+LF ▼

Q5 required ▼



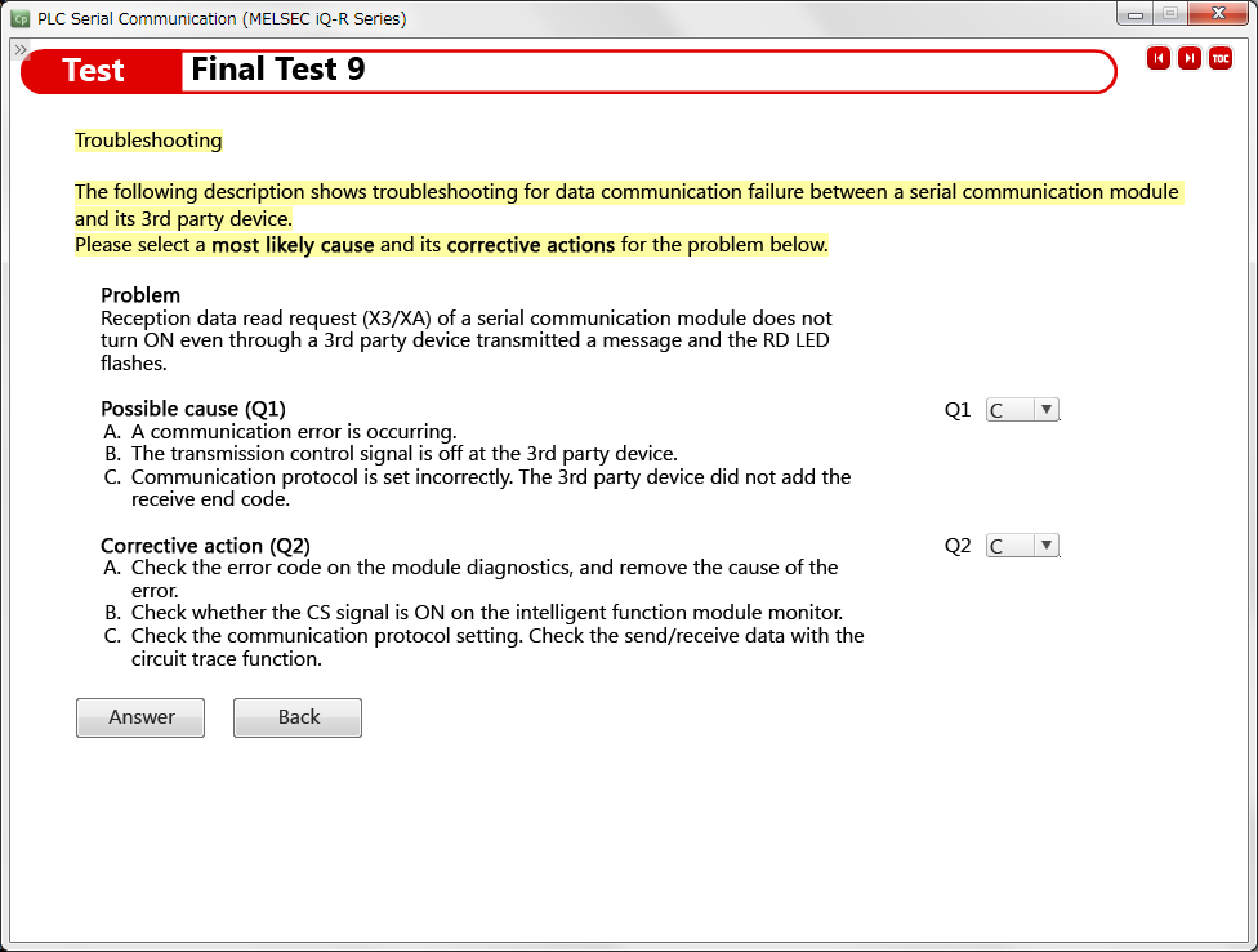
Final Test 8

Communication control signal status

Please select the sentence that correctly describes the RS-232 control signals, which are used between a serial communication module and its 3rd party device.

○ Check the RS-232 control signal status using the module diagnostic function of GX Works3.

● Check the RS-232 control signal status using the intelligent function module monitor function of GX Works3.



Test Final Test 9

Troubleshooting

The following description shows troubleshooting for data communication failure between a serial communication module and its 3rd party device.
Please select a **most likely cause** and its **corrective actions** for the problem below.

Problem
Reception data read request (X3/XA) of a serial communication module does not turn ON even through a 3rd party device transmitted a message and the RD LED flashes.

Possible cause (Q1)

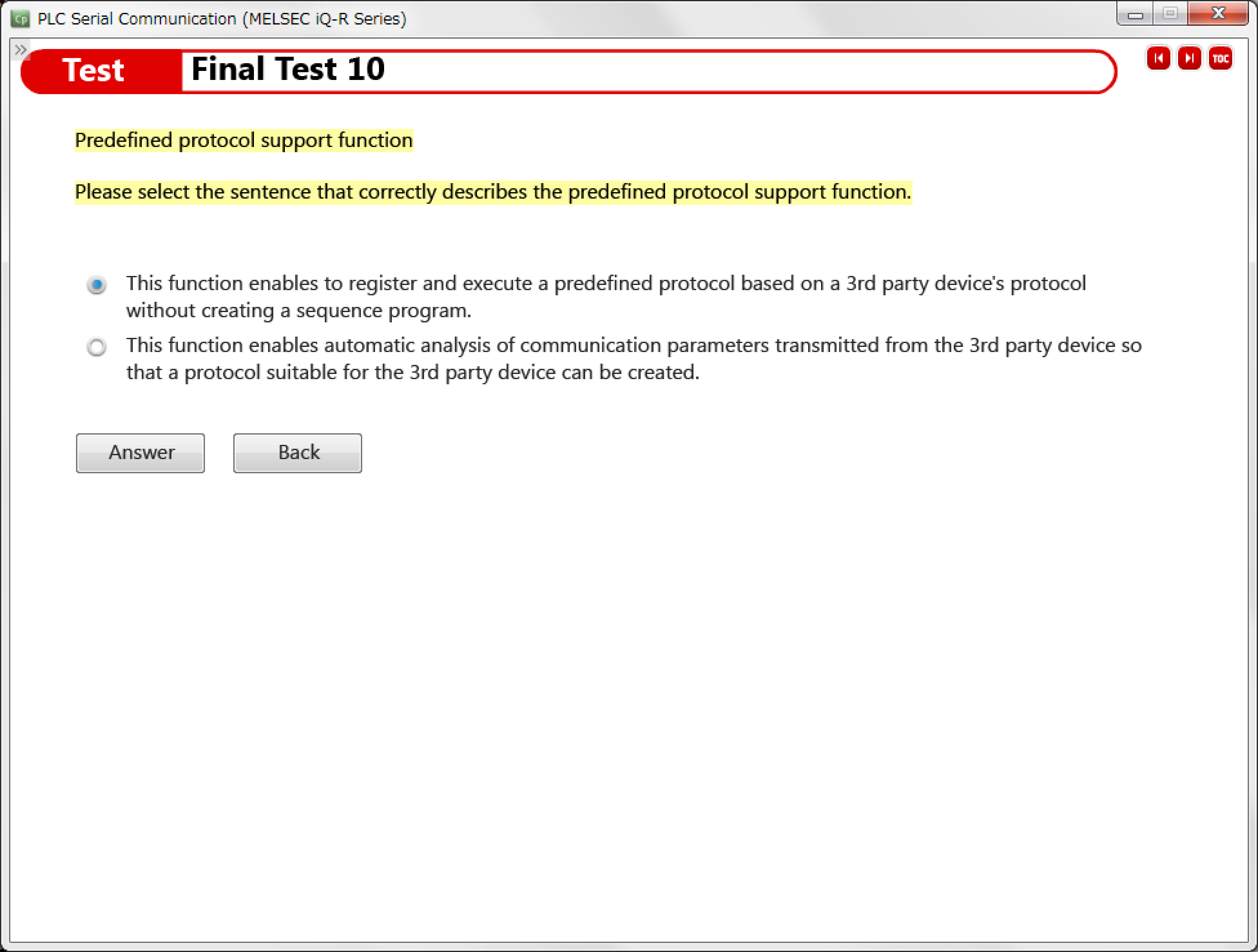
A. A communication error is occurring.
B. The transmission control signal is off at the 3rd party device.
C. Communication protocol is set incorrectly. The 3rd party device did not add the receive end code.

Q1 C ▼

Corrective action (Q2)

A. Check the error code on the module diagnostics, and remove the cause of the error.
B. Check whether the CS signal is ON on the intelligent function module monitor.
C. Check the communication protocol setting. Check the send/receive data with the circuit trace function.

Q2 C ▼



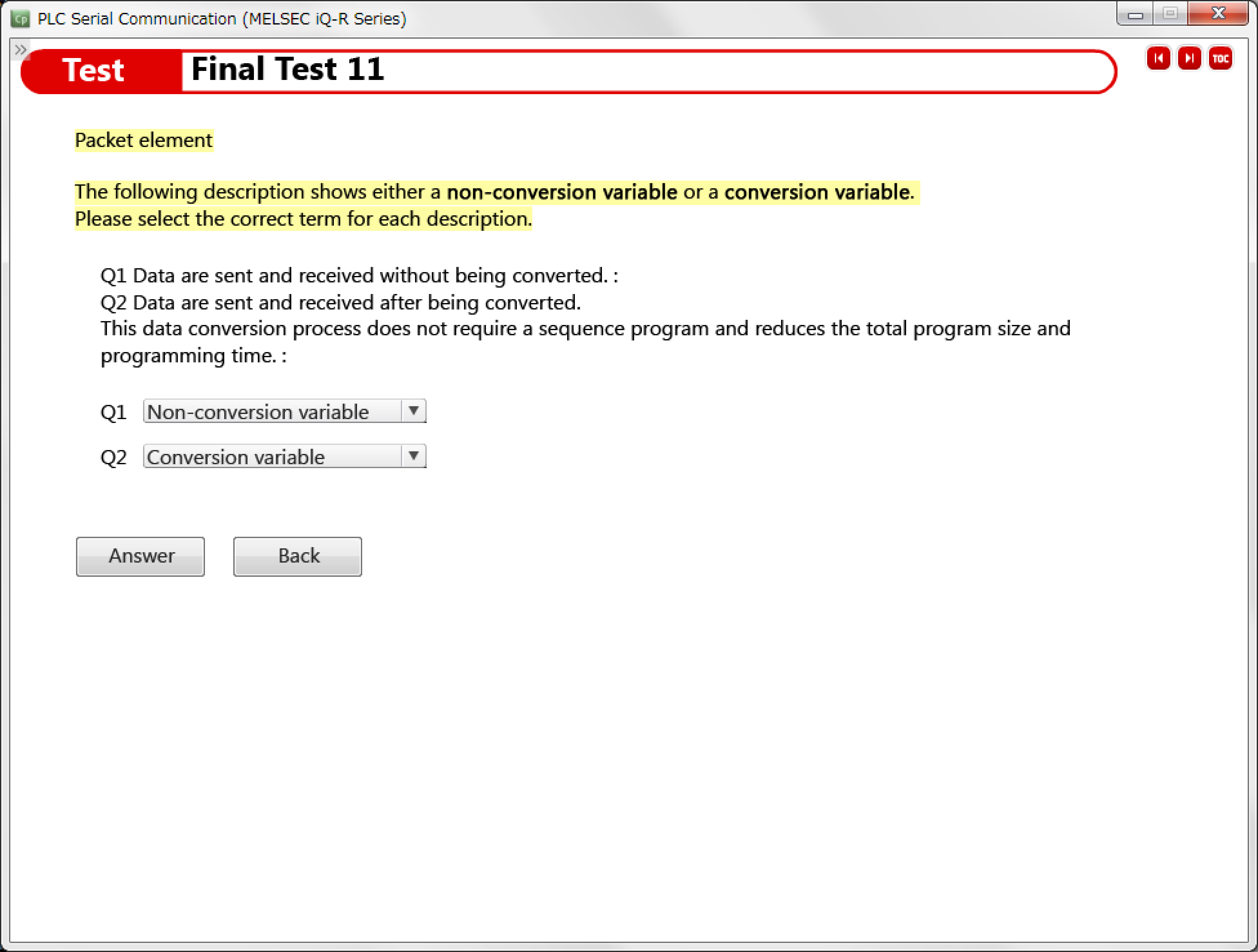
Test Final Test 10

Predefined protocol support function

Please select the sentence that correctly describes the predefined protocol support function.

● This function enables to register and execute a predefined protocol based on a 3rd party device's protocol without creating a sequence program.

○ This function enables automatic analysis of communication parameters transmitted from the 3rd party device so that a protocol suitable for the 3rd party device can be created.



Test Final Test 11

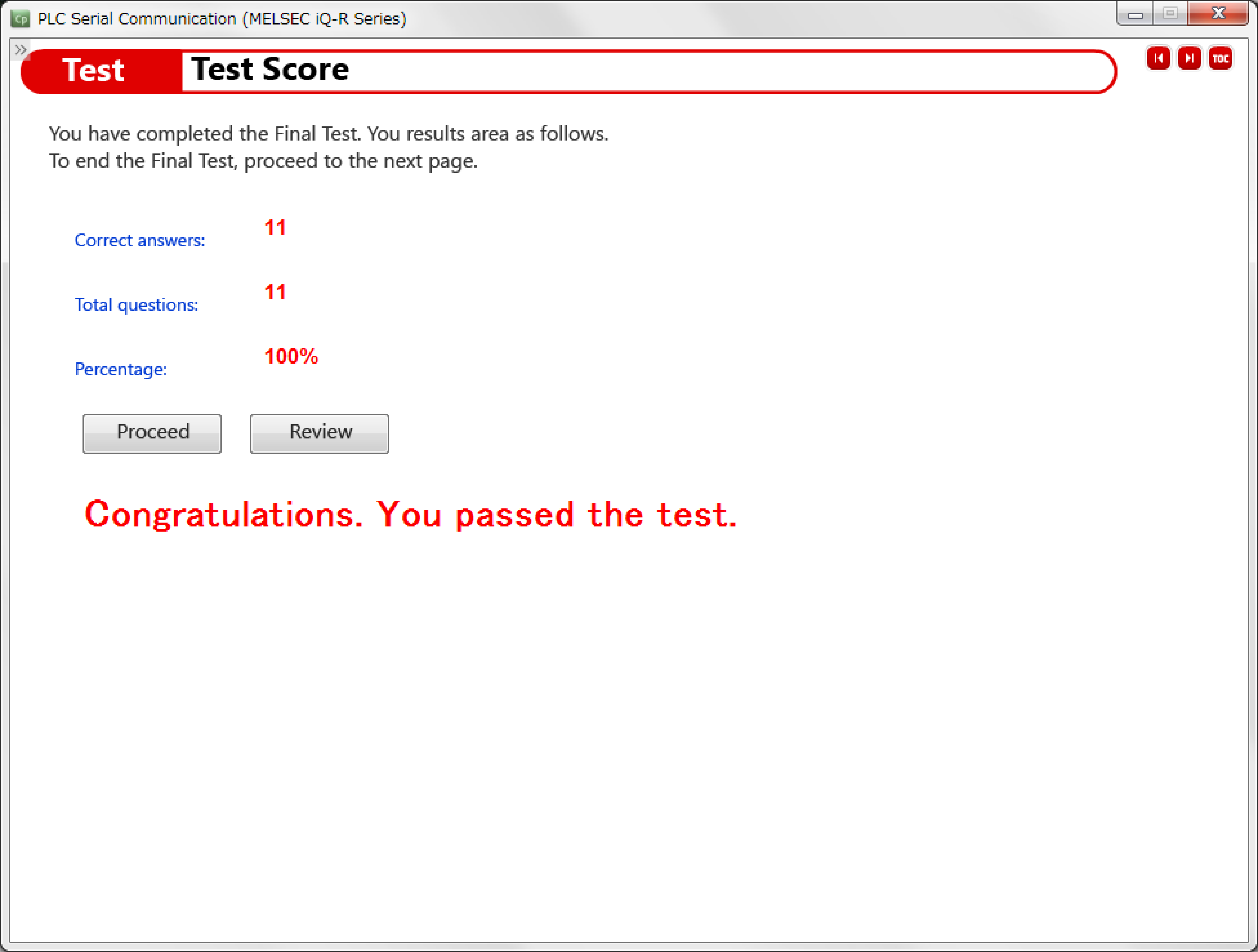
Packet element

The following description shows either a **non-conversion variable** or a **conversion variable**.
Please select the correct term for each description.

Q1 Data are sent and received without being converted. :
Q2 Data are sent and received after being converted.
This data conversion process does not require a sequence program and reduces the total program size and programming time. :

Q1 Non-conversion variable ▼

Q2 Conversion variable ▼

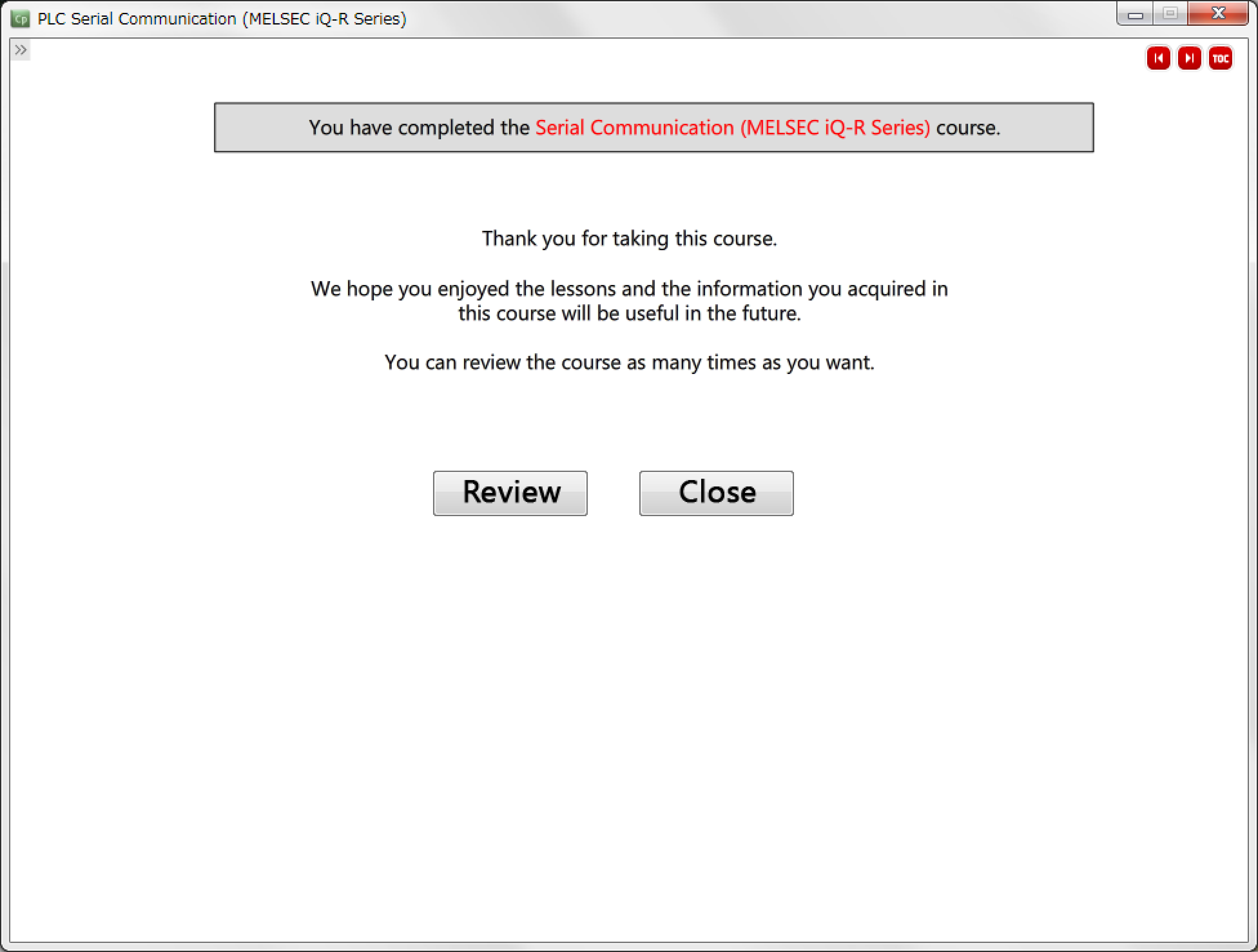


Test Test Score

You have completed the Final Test. You results area as follows.
To end the Final Test, proceed to the next page.

Correct answers:	11
Total questions:	11
Percentage:	100%

Congratulations. You passed the test.



You have completed the Serial Communication (MELSEC iQ-R Series) course.

Thank you for taking this course.

We hope you enjoyed the lessons and the information you acquired in this course will be useful in the future.

You can review the course as many times as you want.