R-IN32M3 Module (RY9012A0)

RA Sample Application (uGOAL Edition)

R30AN0398EJ0105
Rev.1.05
May.31.2024

Introduction

This document describes sample application software for RA6M3 and RA6M4 to perform industrial Ethernet communication as host CPU of the R-IN32M3 Module (RY9012A0).

Target Device

RA6M3, RA6M4

R-IN32M3 Module (RY9012A0)

Contents

List of Abbreviations and Acronyms

In this document, the terms below are defined as follows:

Terms	Description
This board	The target boards of the sample program described in this document, EK-RA6M3 or EK-RA6M4 and the adapter boards with R-IN32M3 Module (YCONNECT-IT-I-RJ4501)
This sample	The sample program for the host microcomputer that controls the R-IN32M3 Module in the industrial network sample program for the R-IN32M3 Module.
API	Application Programming Interface
GOAL/uGOAL	Generic Open Abstraction Layer See "R-IN32M3 Module (RY9012A0) User's Manual: Software (R17US0002ED****)"

Related documents

Document Type	Document Title	Document No.
Data Sheet	R-IN32M3 Module Datasheet	R19DS0109ED****
User's Manual	R-IN32M3 Module User's Manual: Hardware	R19UH0122ED****
User's Manual	R-IN32M3 Module User's Manual: Software	R17US0002ED****
Application Note	R-IN32M3 Module Management Tool Instruction Guide	R30AN0390EJ****
Application Note	R-IN32M3 Module Modbus TCP Start-Up Manual	R30AN0406EJ****
User's Manual	Adaptor Board with R-IN32M3 module YCONNECT-IT-I-RJ4501	R12UZ0094EJ****
Application Note	R-IN32M3 Module User's Implementation Guide (uGOAL edition)	R30AN0402EJ****
Quick Start Guide	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 Quick Start Guide	R20QS0011EU***
User's Manual	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 v1 User's Manual	R20UT4623EU****
Quick Start Guide	Evaluation Kit for RA6M4 Microcontroller Group EK-RA6M4 Quick Start Guide	R20QS0016EG***
User's Manual	Evaluation Kit for RA6M4 Microcontroller Group User's Manual	R20UT4836EG****
User's Manual	Renesas Flexible Software Package (FSP) User's Manual	R11UM0155EU****
Application Note	R-IN32M3 Module Software PLC Guide: TwinCAT	R30AN0380EJ****

1. Overview

1.1 Abstract

This document describes the R-IN32M3 module sample software for the EK-RA6M3 and EK-RA6M4.

This sample software can communicate with major industrial Ethernet protocols such as PROFINET, EtherNet/IP, and EtherCAT by running on EK-RA6M4 / EK-RA6M3, which is evaluation board of RA6M4/RA6M3 MCU, connected with R-IN32M3 Module-based adapter board (YCONNECT-IT-I-RJ4501) via Arduino™ connector.

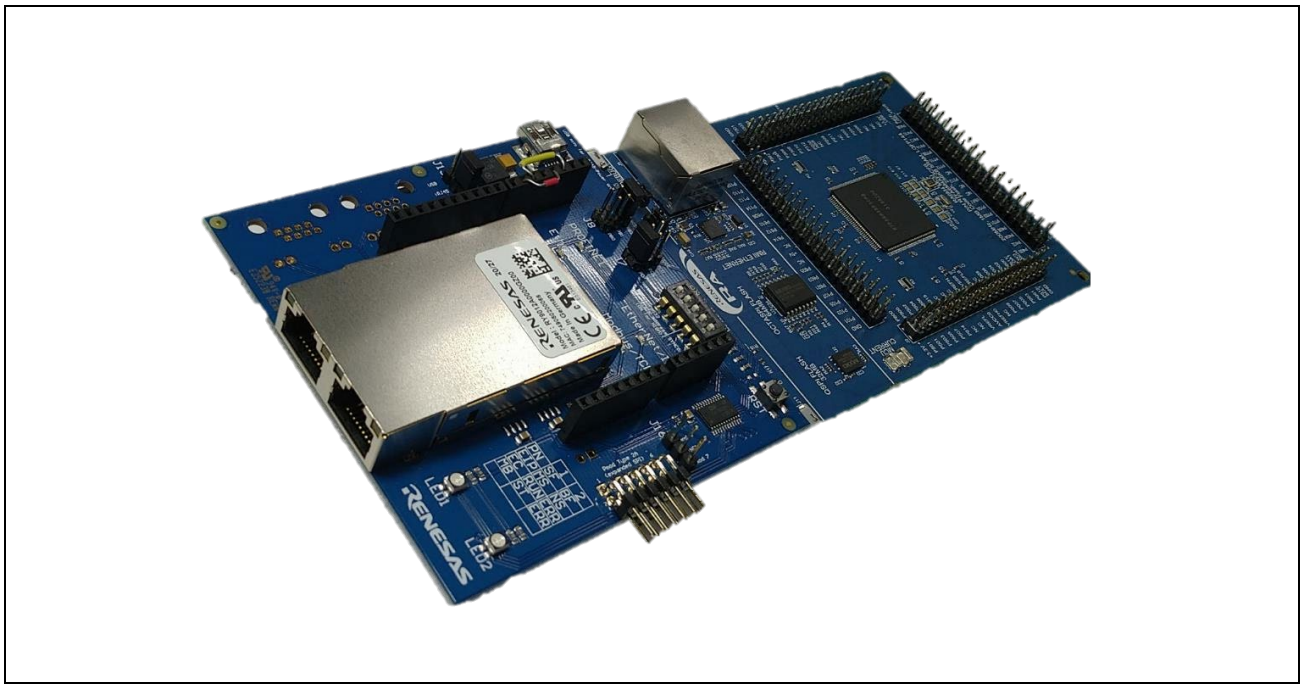

Figure 1-1 R-IN32M3 Module + EK-RA6M4

1.2 Operating environment

1.2.1 Software environment

The operating environment of this sample software is shown inTable 1-1.

Table 1-1 Operating Environments

Category	Name	Version	Link	Remarks
R-IN32M3 module Sample package	Sample package	Rev.1.05	Renesas R-IN32M3 Module Sample Package	https://www.renesas.com/
Integrated development environment	e2studio	2024-04	github.com/renesas	
RA family Flexible Software Package	FSP	V5.3.0	-	Included in installer of e2studio
GNU Arm Embedded Toolchain	GCC Toolchain	13.2.1.arm-13-7	-	Included in installer of e2studio
Management Tool, simple software PLC	ICE	V1.5.1	-	products by port industrial automation GmbH Including with Sample package
Software PLC of EtherCAT	TwinCAT	V3.1	https://www.beckhoff.com/	products by Beckhoff Automation GmbH

1.2.2 Hardware environment

The operation of this sample software is verified with a hardware environment connected to the RA6M4 MCU Group Evaluation Kit (EK-RA6M4) or the RA6M3 MCU Group Evaluation Kit (EK-RA6M3) with an adapter board equipped with an R-IN32M3 module(YCONNECT-IT-I-RJ4501).

If you use EK-RA6M4 board or EK-RA6M3 board, you do not need to prepare the emulator separately for the execution of this sample software by using the default on-board debugging of the debug modes supported by the evaluation board.

Also, this sample software includes multiple applications. Multi-protocol application and Remote I/O application can be executed by connecting the Digilent Pmod™ board in Table 1-2. For details, please refer to Chapter 3.2.2.

Table 1-2 Hardware environments

Name	Type Name	Maker	Link	Note
EK-RA6M4	RTK7EKA6M4S 00001BE	Renesas Electronics Corporation	Evaluation Kit for RA6M4 MCU Group	
EK-RA6M3	RTK7EKA6M3S 00001BU	Renesas Electronics Corporation	RA6M3 MCU Group Evaluation Board	
Adapter Board with R-IN32M3 Module	YCONNECT-IT-I-RJ4501	Renesas Electronics Corporation	R-IN32M3-Module-Solution-Kit	
6-pin Pmod with 4-ch Switch	Pmod SWT (410-083)	Digilent, Inc.	https://reference.digilentinc.com/reference/pmod/pmodswt/start	Multi-protocol application, EtherCAT ID, remote I/O application
6-pin Pmod with 4ch LED	Pmod LED (410-076)	Digilent, Inc.	https://reference.digilentinc.com/reference/pmod/pmodled/start	remote I/O application

2. Hardware configuration

The hardware configuration to run this sample software is described.

2.1 Adaptor Board Configuration

When using this sample software, set J13, J8, and J7 jumper blocks on the adapter board with R-IN32M3 module (YCONNECT-IT-I-RJ4501) as follows.

J13: Connect the Socket pin with the iRJ45 pin

J8: For the CS signal, select PB2

J7: For the RST signal, select PD7

Also, when using EtherCAT DC mode, short-circuit **3pin - 6pin** and **4pin - 7pin** of J10 with bridge wire.

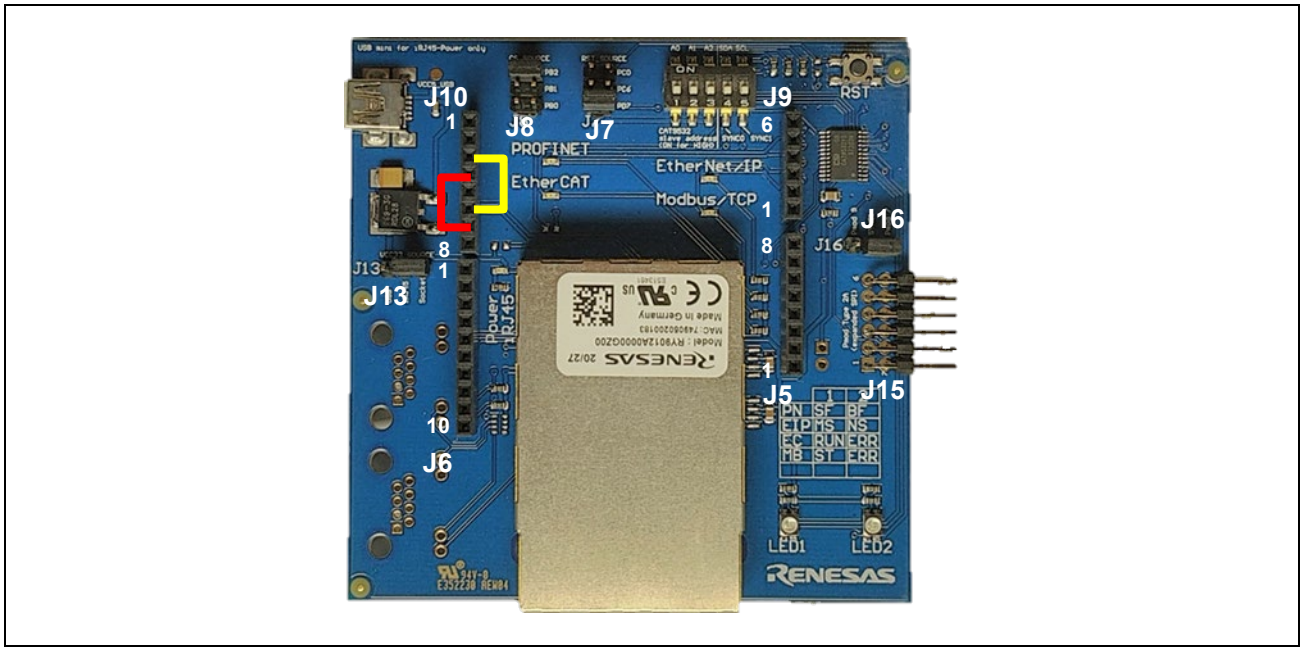

Figure 2-1 Adaptor board with R-IN32M3 module

Plug the male Arduino connector on the back of the adapter board with R-IN32M3 module into the socket of the EK-RA6M4 or EK-RA6M3 board.

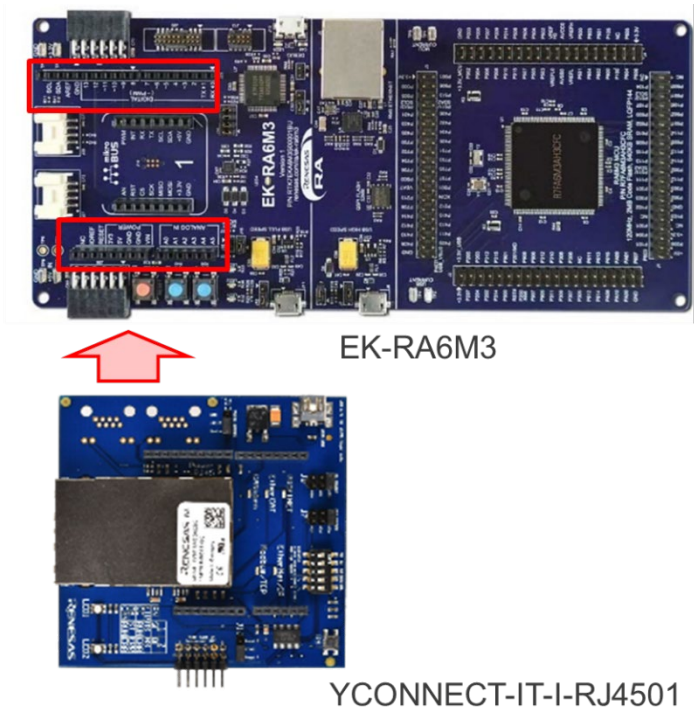

Figure 2-2 Arduino™ Connection

2.2 Multi-protocol application

Multi-protocol (PROFINET, EtherNet/IP, EtherCAT and Modbus TCP) selector input in multi-protocol sample application is confirmed by connecting the Pmod SWT to the lower stage (7-12pin) of the Pmod1 connector on the EK-RA6M4 board or the EK-RA6M3 board.

Table 2-1 Connection of Pmod1 and Pmod SWT for Multi-protocol selector

PMOD1 (J26) Lower	EK-RA6M3	EK-RA6M4	Pmod SWT
7	P004	P008	Selector-SW1
8	P800	P311	Selector-SW2
9	P801	P312	Selector-SW3
10	P802	P313	Selector-SW4
11	GND	GND	GND
12	+3.3V	+3.3V	VCC

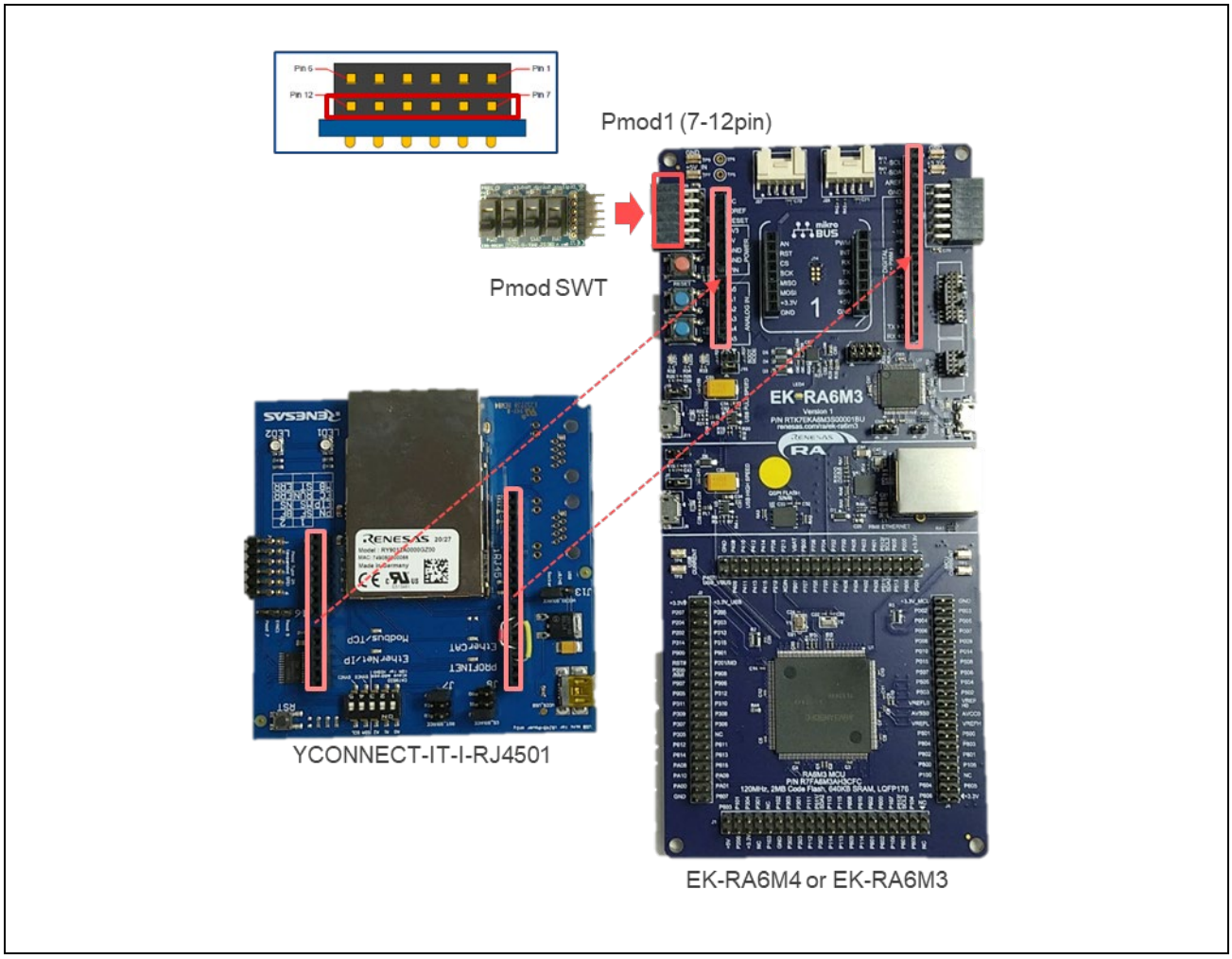

Figure 2-3 Multi-protocol Selector Connection

2.3 Remote I/O application

Sample application software for Remote I/O can be run on the EK-RA6M4 board and the EK-RA6M3 board. It is confirmed in a configuration in which the switch input (Pmod SWT) is connected to the lower of the Pmod1 connector (7-12pin) and the LED output (Pmod LED) is connected to the lower (7-12pin) of the Pmod2 connector. (Figure 2-4)

Table 2-2 Connection of Pmod1 and Pmod SWT

PMOD1 (J26) Lower	EK-RA6M3	EK-RA6M4	Pmod SWT
7	P004	P008	SW1
8	P800	P311	SW2
9	P801	P312	SW3
10	P802	P313	SW4
11	GND	GND	GND
12	+3.3V	+3.3V	VCC

Table 2-3 Connection of Pmod2 and Pmod LED

PMOD2 (J25) Lower	EK-RA6M3	EK-RA6M4	Pmod LED
7	P708	P414	LED1
8	P803	P708	LED2
9	P804	P709	LED3
10	P805	P710	LED4
11	GND	GND	GND
12	+3.3V	+3.3V	VCC

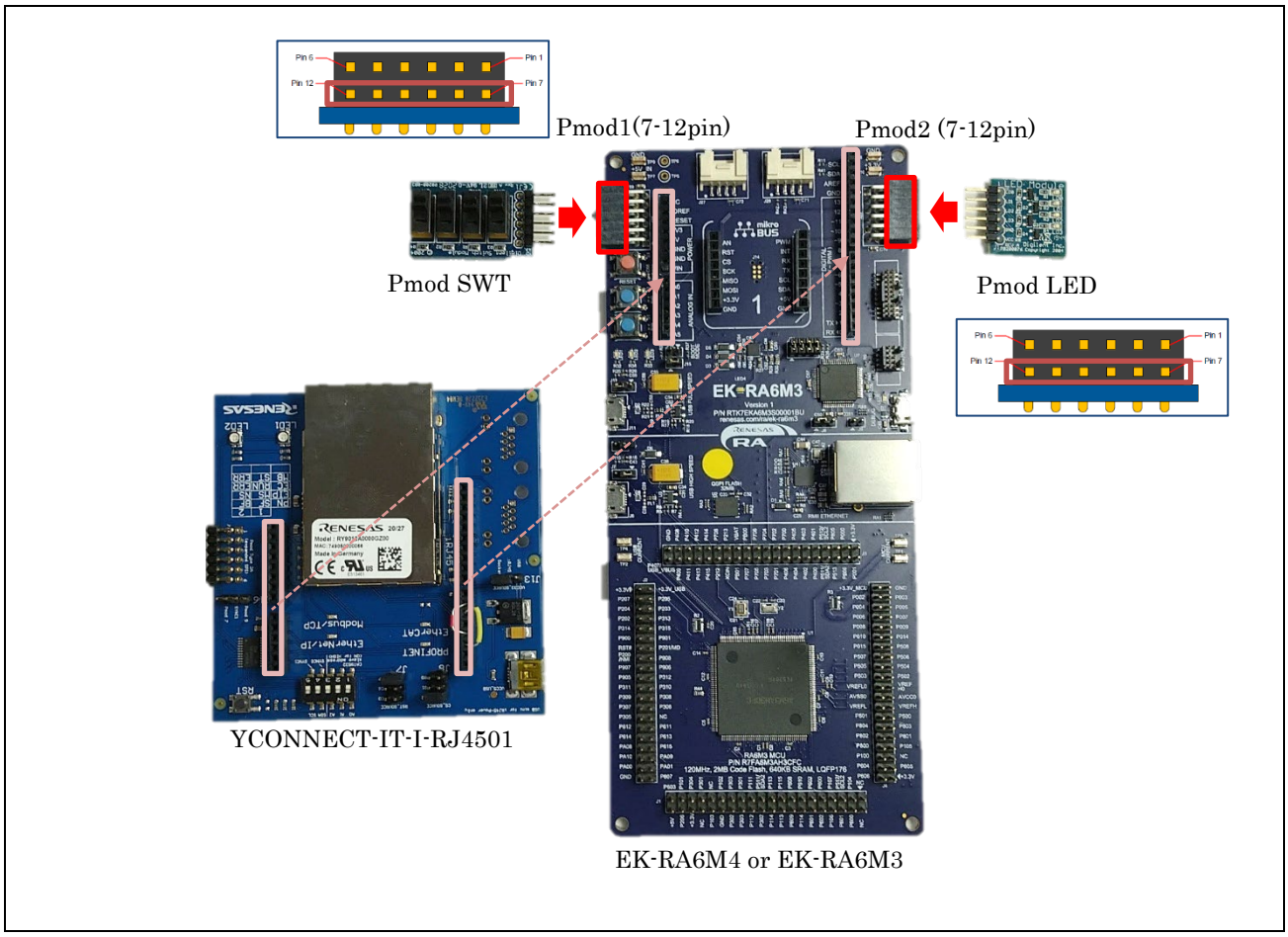

Figure 2-4 Remote I/O Connection

2.4 EtherCAT Explicit Device ID selector

EtherCAT Explicit Device ID selector input in this sample software is confirmed by connecting the Pmod SWT to the upper stage (1-6pin) of the Pmod2 connector on the EK-RA6M4 board or the EK-RA6M3 board.

If you connect the Pmod SWT to the upper of Pmod2 connector for EtherCAT ID selector and the Pmod LED to the lower row of Pmod2 for Remote I/O sample software, it is necessary to separate these two Pmod connection with a kind of branch cable.

Table 2-4 Connection of Pmod2 and Pmod SWT for EtherCAT ID

PMOD2 (J25) Upper	EK-RA6M3	EK-RA6M4	Pmod SWT
1	P107	P413	ECAT-ID1
2	P105	P411	ECAT-ID2
3	P104	P410	ECAT-ID3
4	P106	P412	ECAT-ID4
5	GND	GND	GND
6	+3.3V	+3.3V	VCC

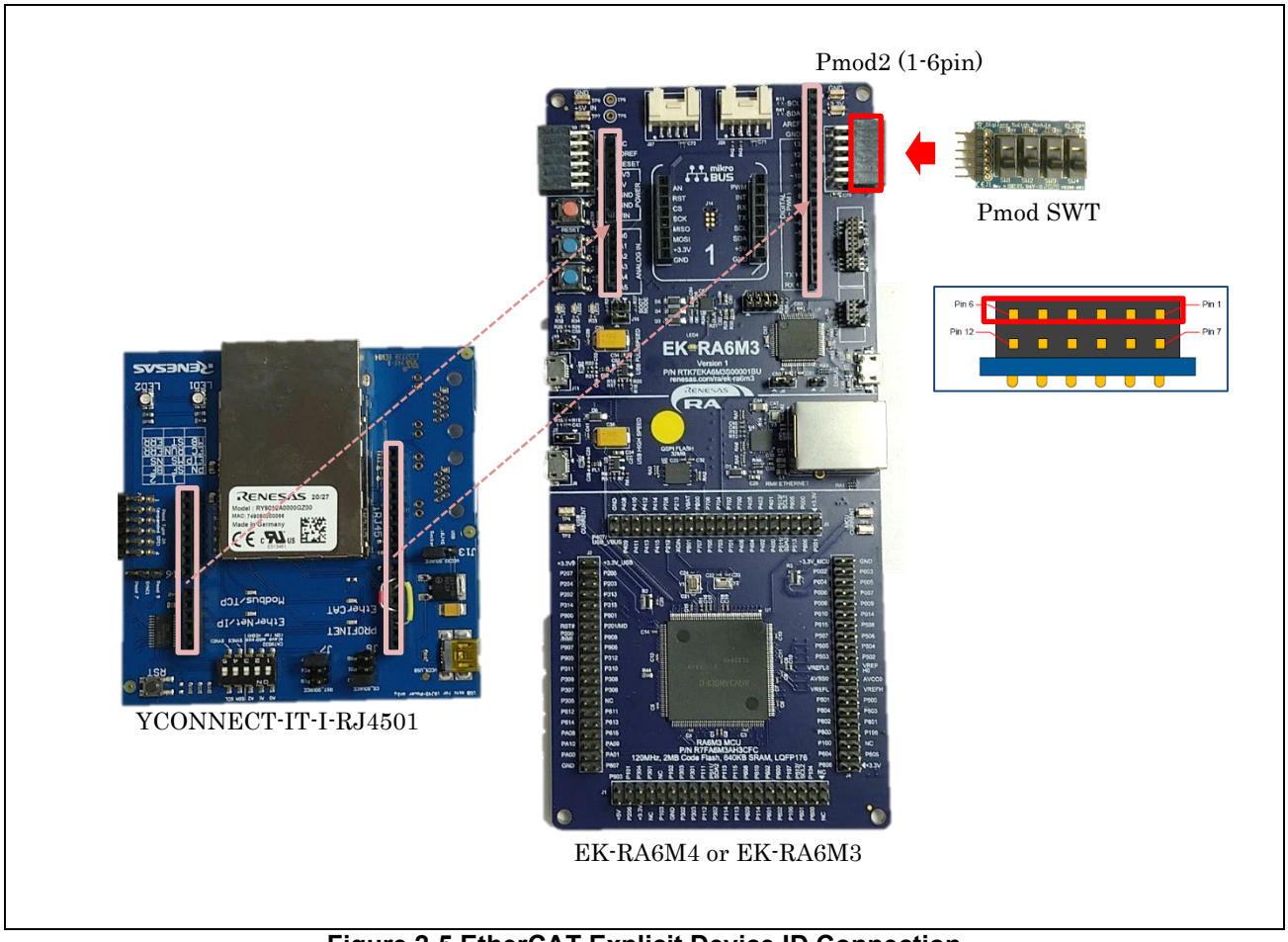

Figure 2-5 EtherCAT Explicit Device ID Connection

EtherCAT Conformance Test tool [Test Case: TF-1201 ESM - Explicit Device Identification] expect this ID set [5].

3. Sample software configuration

3.1 Folder structure

The folder structure of this sample software is shown below.

RA6_uCCM_V***	
├──appl	User application
│ ├──01_pnio	PROFINET sample application
│ ├──02_eip	EtherNet/IP sample application
│ ├──03_ecat	EtherCAT sample application
│ ├──04_pnio_largesize	PROFINET Large data size sample application
│ ├──05_eip_largesize	EtherNet/IP Large data size sample application
│ ├──06_ecat_largesize	EtherCAT Large data size sample application
│ ├──07_modbus_tcp_slave	Modbus TCP sample application
│ ├──10_multi_protocol	multi-protocol [01_pnio, 02_eip, 03_ecat, 07_modbus] sample application
│ ├──11_pnio_http	01_pnio sample Enhanced [web saver and host MCU update function]
│ ├──12_eip_http	02_eip sample Enhanced [web saver and host MCU update function]
│ ├──13_ecat_http	03_ecat sample Enhanced [web saver and host MCU update function]
│ ├──17_fwup_bootloader	Bott Loader for host MCU update function
│ └──18_fwup_spi	FW update function vis SPI
│	
├──plat	HW-dependent components (OS-dependent part, board spec, drivers)
├──projects	Project files corresponding to each user application
│	
└──ugoal	Main part of uGOAL (Generic Open Abstraction Layer *)
├──rpc	Functional parts related to RPC (Remote Procedure Call) including NW protocols and MCTC
├──sapi	Simple API
└──ext	external software component

* For more information about uGOAL, see “R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)”.

3.2 Overview of the project

The protocols (PROFINET, EtherNet/IP and EtherCAT) in this sample software support the following features:

Table 3-1 Protocol and feature

Protocol	Feature
PROFINET	・Conformance : CC-B (RT) ・Netload : I Min Interval : 1ms ・I&M : 1-4
EtherNet/IP	・DLR : Support
EtherCAT	・DC : Support ・Mailbox : CoE / FoE / EoE ・Profile : MDP

The sample software implements two types of data transmission/reception applications as example applications.

- **Remote-IO (LED/Switch):** LED lighting control and Switch status from the evaluation board
- **Mirror:** Sends data received from the master and mirrored back

Project	Protocol	Refer
01_pnio	PROFINET	3.4.1 PROFINET
02_eip	EtherNet/IP	3.4.2 EtherNet/IP
03_ecat	EtherCAT	3.4.3 EtherCAT
04_pnio_largesize	PROFINET	3.4.1 PROFINET
05_eip_largesize	EtherNet/IP	3.4.2 EtherNet/IP
06_ecat_largesize	EtherCAT	3.4.3 EtherCAT
07_mbus_tcp_sever	ModbusTCP	3.4.4 Modbus TCP
10_multi_protocol	PROFINET / EtherNet/IP / EtherCAT / ModbusTCP	3.4.5 multi-protocol
11_pnio_http	PROFINET	3.4.6 Web saver
12_eip_http	EtherNet/IP	
13_ecat_http	EtherCAT	

✓ 04_pnio_largesize, 05_eip_largesize, 06_ecat_largesize project has a sample project for large data transfer using RPC communication. See “User's Implementation Guide (uGOAL Edition) [R30AN0402EJ****]” for details on RPC communication.

✓ This document only describes the data communication function. For details about the firmware update function, please refer to the "R-IN32M3 Module (RY9012A0) Firmware Update Guide (R30AN0401EJ****)".

3.3 Set up of development environment

Please refer to Chapter 1.2 for the operating environment of this sample software.

3.3.1 Install

(1) FSP, e2studio, GCC Toolchain

Download the version listed in Table 1-1 from the following web site and install its on your PC.

In the latest version of FSP, an installer that includes FSP, e2studio, and GCC toolchain in one package is downloaded.

3.3.2 Connection

(1) No additional P-mod connection

After stacking the Adapter board with R-IN32M3 Module on EK-RA6M3 board or EK-RA6M4 board (For details, please refer to Chapter 2.1), connect your PC as follows. Power is supplied to those boards by connecting a USB micro B cable to EK-RA6M3 board or EK-RA6M4 board.

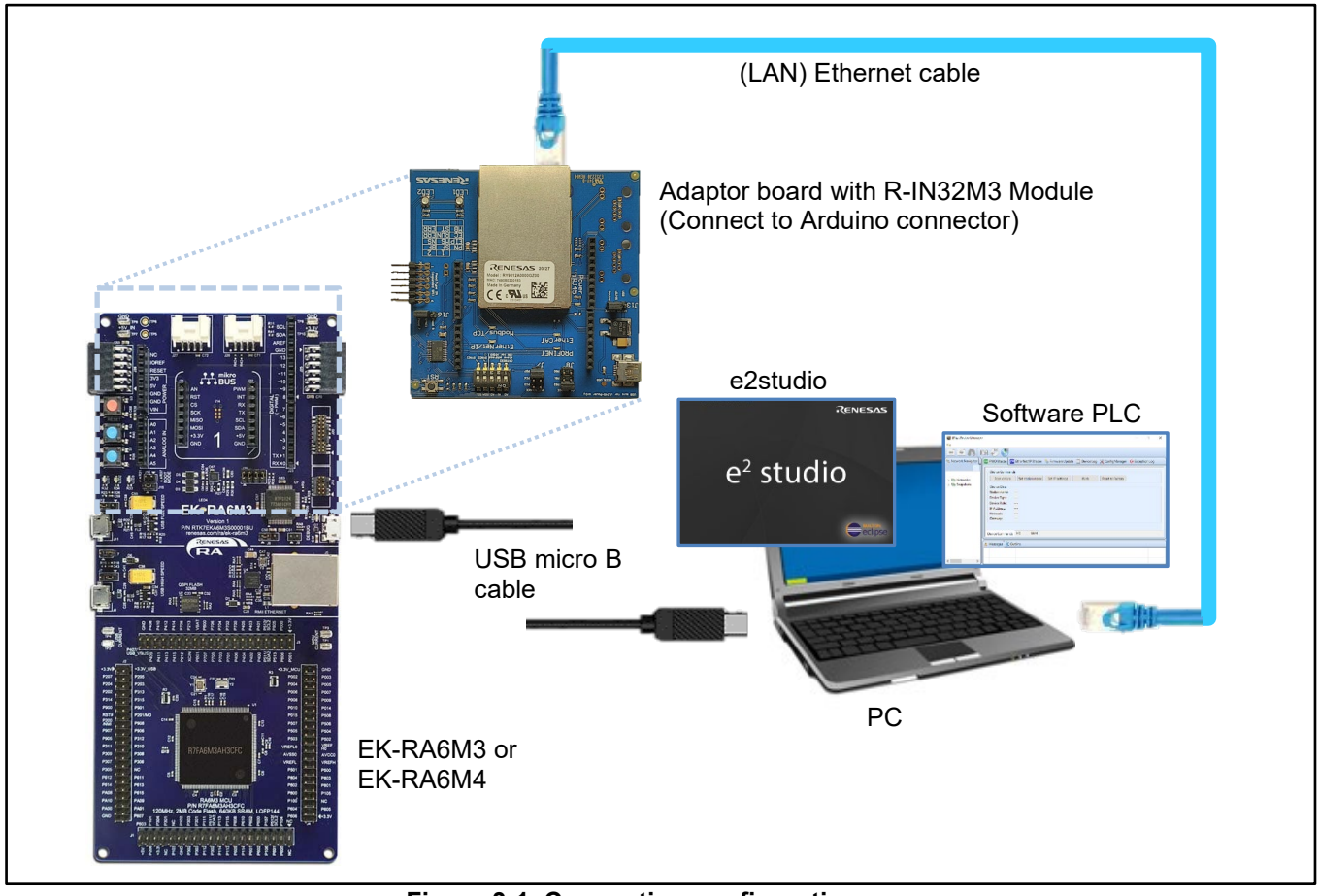

Figure 3-1 Connection configuration

(2) Additional Pmod SWT and Pmod LED connection

Connect Pmod SWT (refer to Table 1-2) to Pmod1 terminal Lower (7-12 pin) and Pmod LED (refer to Table 1-2) to Pmod2 terminal Lower (7-12 pin) on EK-RA6M3 board or EK-RA6M4 board (For details, please refer to Chapter 2.3). Power is supplied to those boards by connecting a USB micro B cable to EK-RA6M3 board or EK-RA6M4 board.

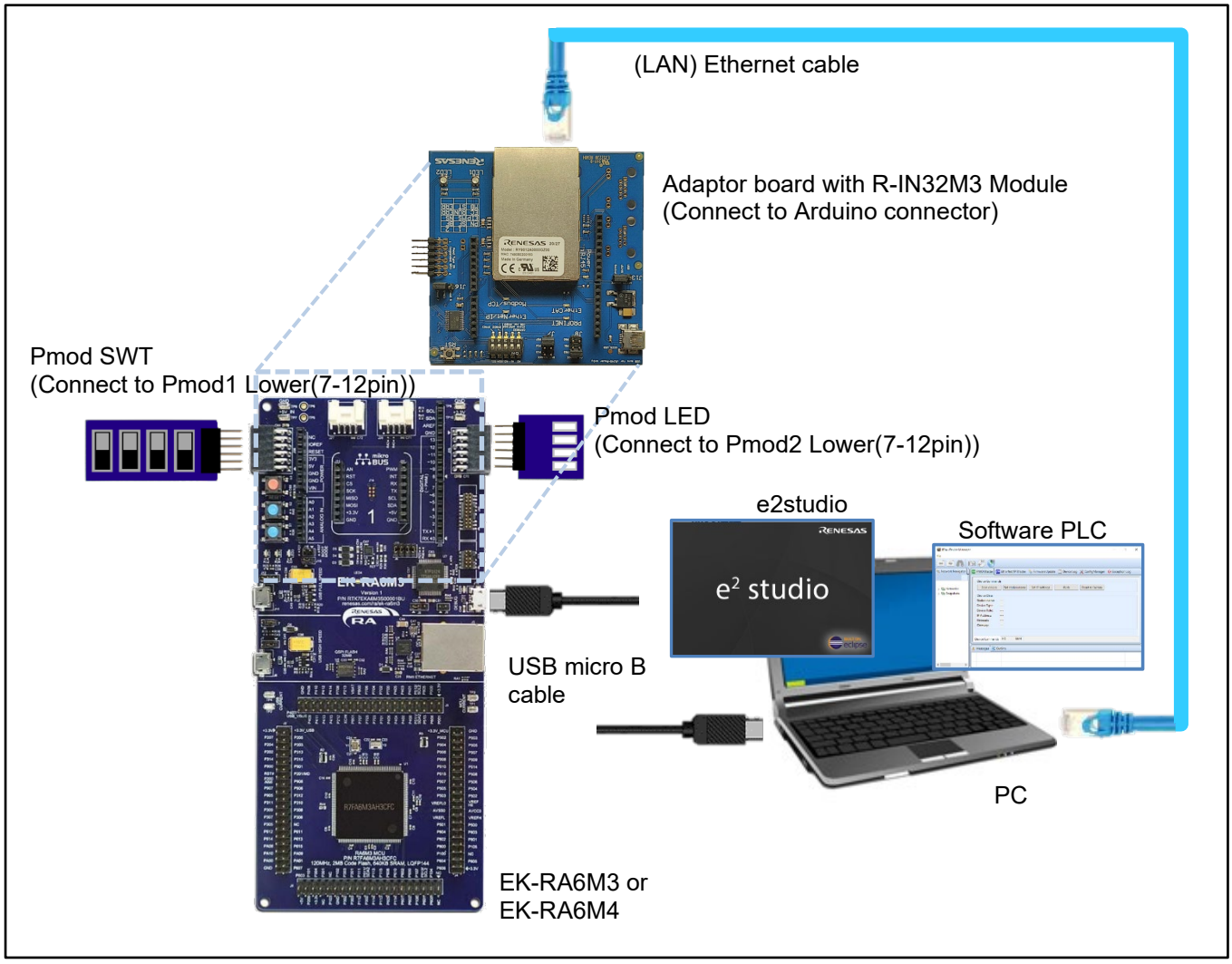

Figure 3-2 Connection configuration (connection with Pmod SWT and Pmod LED)

3.3.3 Import project

(1) Unzip package

First, unzip the archived package of this sample software (RA6_uCCM_V***.zip) and store it in arbitrary folder. Because e2studio cannot recognize project properly if file path is too long in the folder hierarchy, place it in shorter path. Also, do not use multi-byte character, such as Japanese, in the folder path.

(2) Execute e2studio

Execute "e2studio.exe" to start e2studio.

To check the compiler installed above, select [Window] -> [Preferences], and then select [Renesas] -> [Renesas Toolchain Management] in the Settings dialog. In the dialog [Renesas Toolchain Management], it can be seen whether an appropriate compiler has been added to "GNU ARM Embedded".

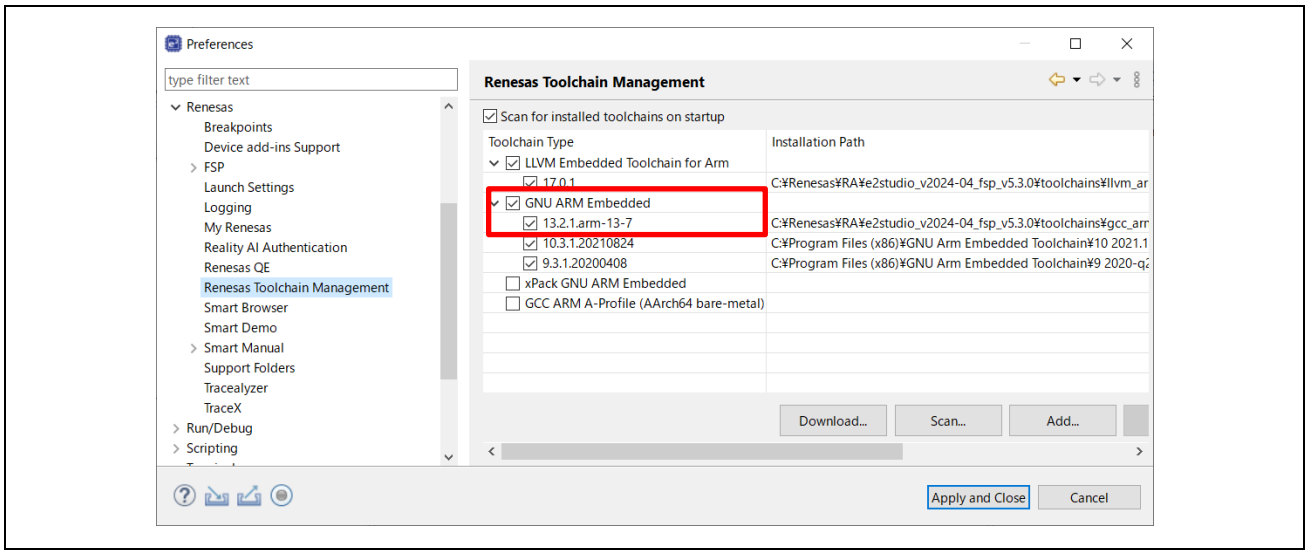

Figure 3.3 Renesas Toolchain Management

(3) Import project

Import the sample project into e2studio from the following steps:

[File] -> [Import…] on the right of the screen.

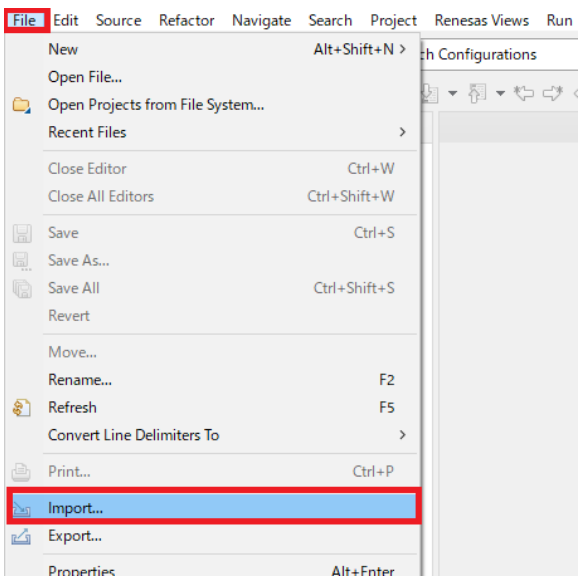

Figure 3.4 Import

In the [Select] dialog, select [General] -> [Existing Project into Workspace], and then select [Next>].

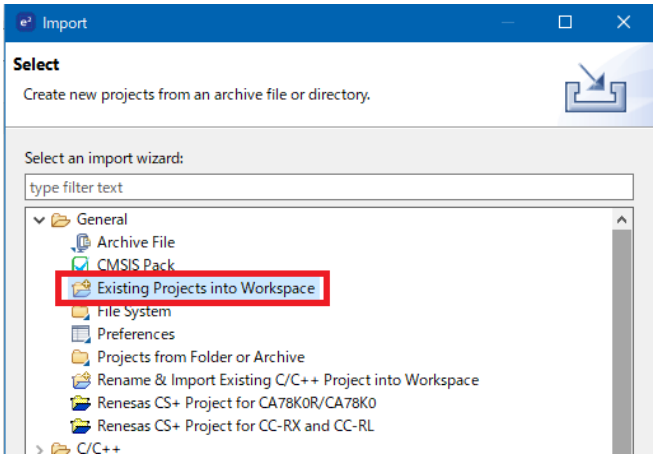

Figure 3.5 Select “Existing Projects into Workspace”

In the [Import Projects] dialog, select [Select root directory] check box, and then select [Browse].
Select the package of this sample software "RA6_uCCM_V***" stored in arbitrary folder at 3.2.3(1) and select [OK].

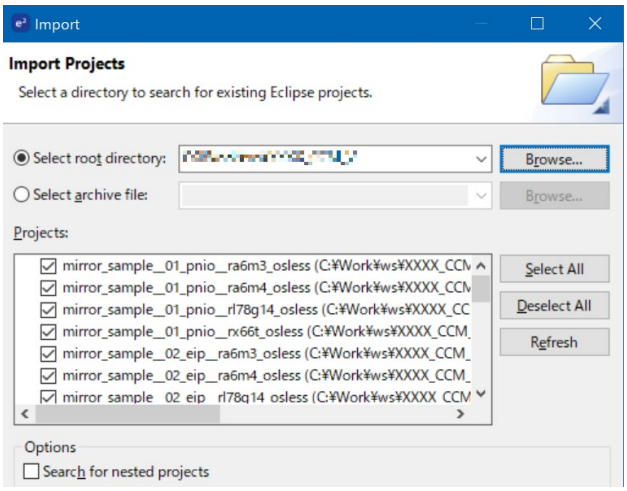

Figure 3.6 Import Projects

After putting a check in the sample project to be used from each sample project listed in [Projects], select [Finish] to import the project.

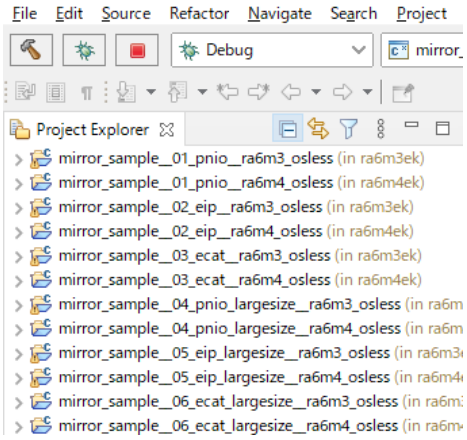

Figure 3.7 Imported projects

3.3.4 FSP code generation

Sample projects in this sample software apply the FSP, so generate the source code of FSP from Configurator in e2studio. Code generation needs to be done on a project-by-project level. For more information, see "Renesas Flexible Software Package (FSP) User's Manual " (R11UM0155EU****).

In the [Project Explorer] on e2studio, expand the sample project and select the configuration file.

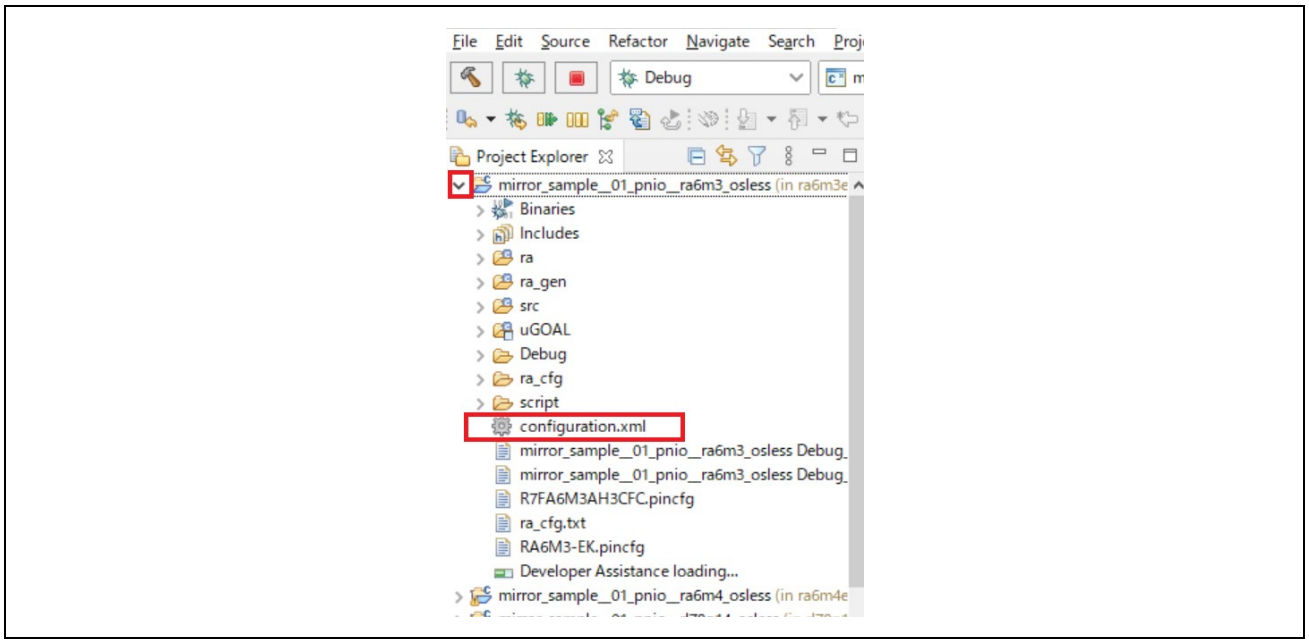

Figure 3.8 Configuration file selection

In configurator perspective screen, select [Generate Project Content] button to generate the required source code.

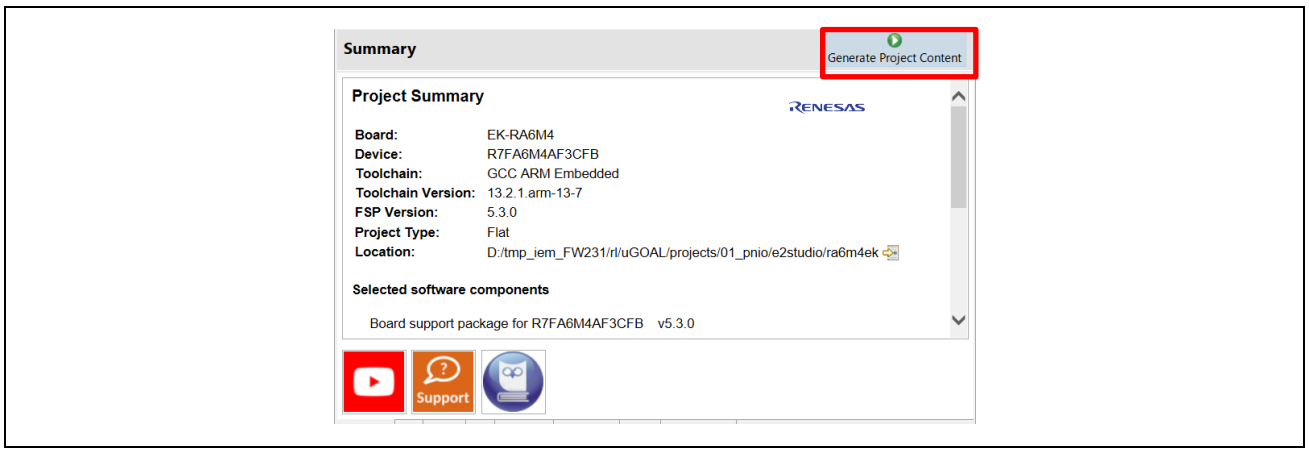

Figure 3.9 Generate code

Now, it is ready to build the project.

3.3.5 Build project

In the [Project Explorer] on e2studio, select the sample project, select the arrow next to the [Build] button (hammer icon), and select [Debug] from the drop-down menu.

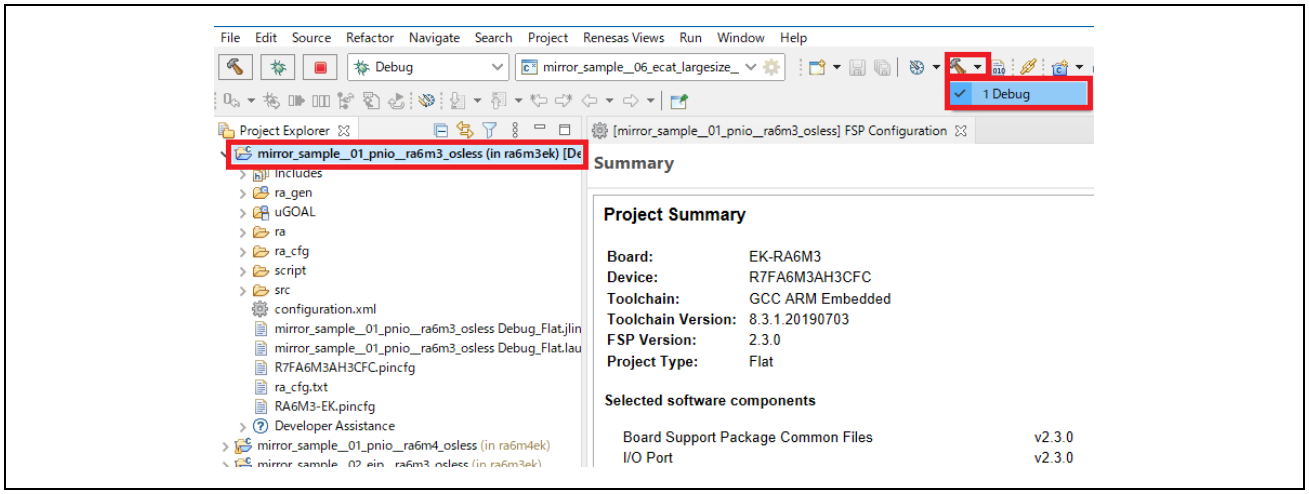

Figure 3.10 Build project

e2studio builds the selected project. When the build is complete, "Build Finished" message can be seen in the [Console] at the bottom of the screen.

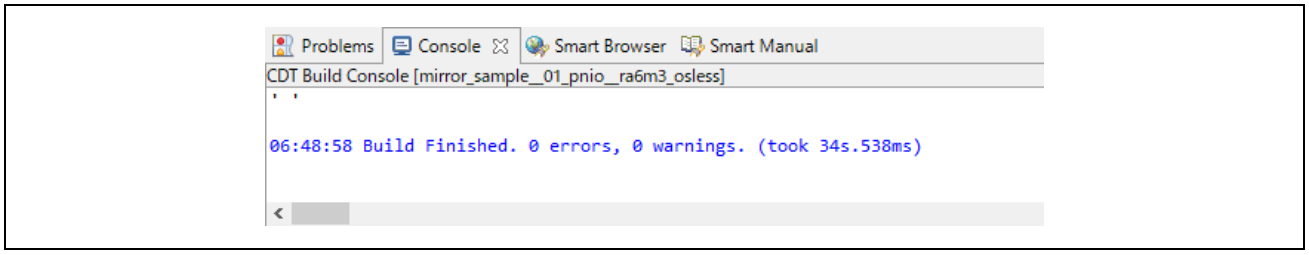

Figure 3.11 Build finished

3.3.6 Debug

Once the build is complete, it is possible to start debugging immediately. Select the arrow next to the [Debug] button (bug icon) and select [Debug Configurations…].

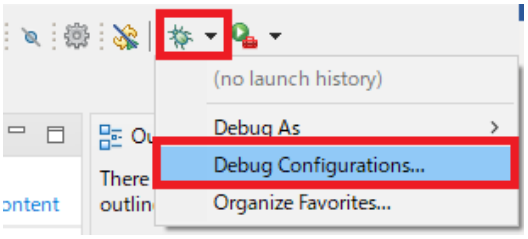

Figure 3.12 Debug Configurations

In the [Debug Configuration] dialog, select the appropriate "xxxx Debug_Flat" from [Renesas GDB Hardware Debugging] and select the [Debug] button to launch the debug screen.

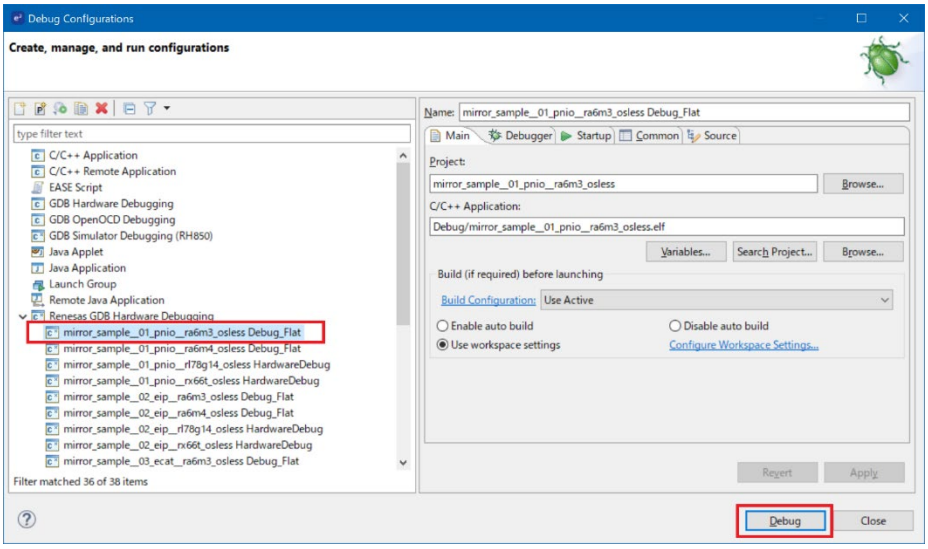

Figure 3.13 Debug start

If a firewall warning for "e2-server-gdb.exe" is shown, check all check boxes, "Domain", "Private" and "Public", and select [Allow access].

If asked to change the perspective in the Confirm Perspective Switch dialog, check the check box of [Always use this setting] and select [Yes].

When the debugger screen is up and the program download is complete, select the [Restart] button to run the program.

3.4 Protocol communication and Application control

This section describes the protocol communication using Management Tool (PROFINET, EtherNet / IP connection) or TwinCAT (EtherCAT connection), and how to control each sample application.

3.4.1 PROFINET

This chapter describes an example of PROFINET communication.
The target sample is below.

Table 3-2 PROFINET Sample software

Sample software	Overview
01_pnio	Cyclic connection sample
04_pnio_largesize	Cyclic and RPC (Large Size data) connection sample
10_multi_protocol	01_pnio, 02_eip, 03_ecat, 07_modbus multi sample
11_pnio_http	01_pnio sample Enhanced [web saver and host MCU update function]

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.5 to 3.3.6. When the sample application is run, the protocol display LED (PROFINET) turn on.

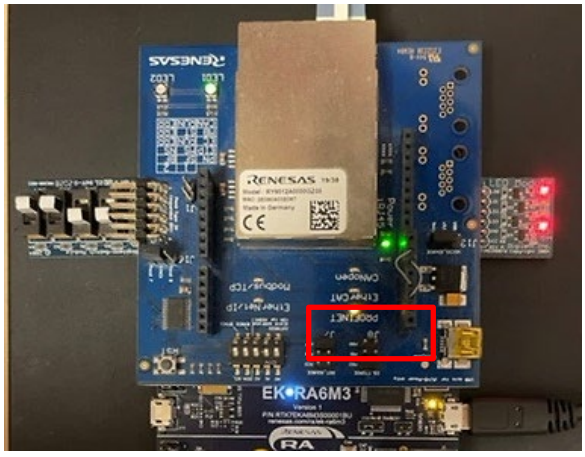

Figure 3-14 Protocol LED: PROFINET

-2. Set IP address

Set Static IP address. Open the [Network Properties] of the network adapter connected to the R-IN32M3 Module and set the static IP (using 192.168.0.1 as an example).

IP address	192.168.0.1
Netmask	255.255.255.0

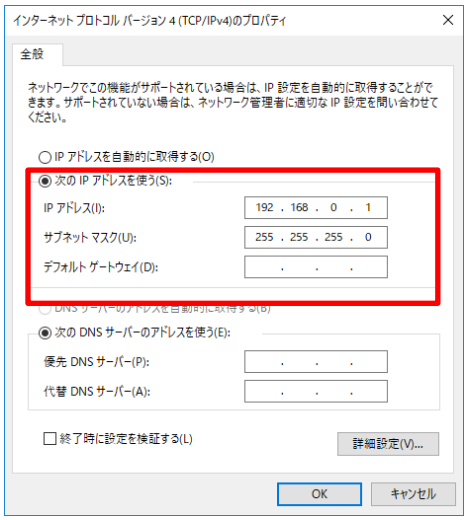

Figure 3-15 Set Static IP address

2. Master connection

Management tool can be used as a PROFINET simple master. It is included with " R-IN32M3 Module (RY9012A0) Sample Package" (R18AN0064EJ****) along with this sample software.

Execute "ice.exe" file in the folder below to start the Management tool. For more information about the Management tool, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

-1. Select network to use in [Network Navigator] panel and select [Scan Network] button.

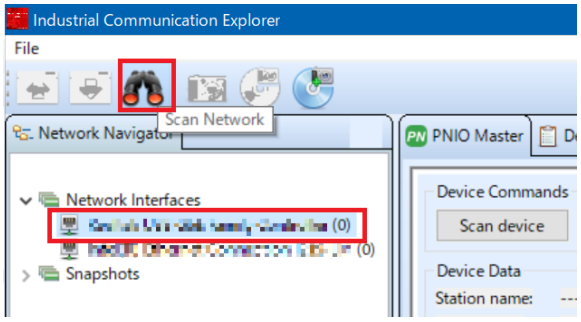

Figure 3-16 Scan network

-2. “Scan complete. found 1 device” message is displayed in [Network Scan] dialog, then select [OK].

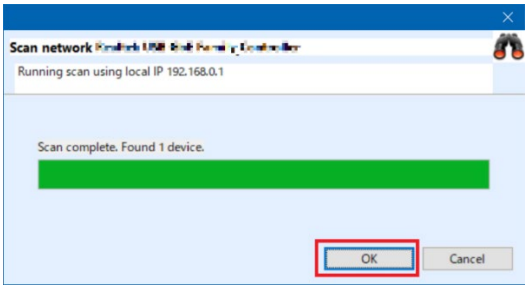

Figure 3-17 Scan completed

-3. In [Network Navigator] panel in the scanned network, “R-IN32M3_Module” is displayed as the new device, so select [R-IN32M3_Module].

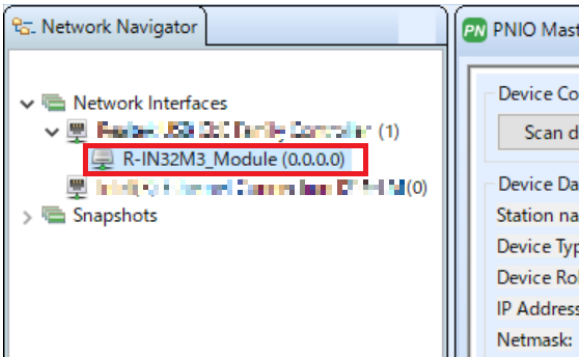

Figure 3-18 Select R-IN32M3 Module

-4. In order to communicate with the R-IN32M3 Module, the IP address of the R-IN32M3 Module must be in the same IP network as the IP address of the PC. Therefore, access the configuration manager variables (volatile memory and non-volatile memory stored configuration variables) of the R-IN32M3 Module to set the IP address and Netmask.
With [R-IN32M3_Module] selected, select [Read Configuration] button while displaying the [ConfigManager] panel.

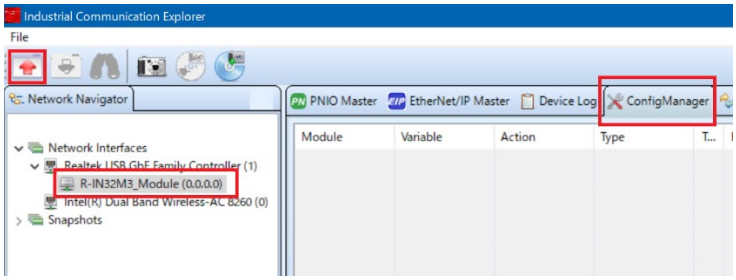

Figure 3-19 ConfigManager

-5. In the configurations displayed in the [ConfigManager] panel, change the following items. Note that it is required to set VALID to 1 due to enable the IP address and Netmask. The changed Value will be highlighted in yellow.

Module	Variable	Value example
GOAL_ID_NET	**IP**	192.168.0.100
GOAL_ID_NET	**NETMASK**	255.255.255.0
GOAL_ID_NET	**VALID**	0x01

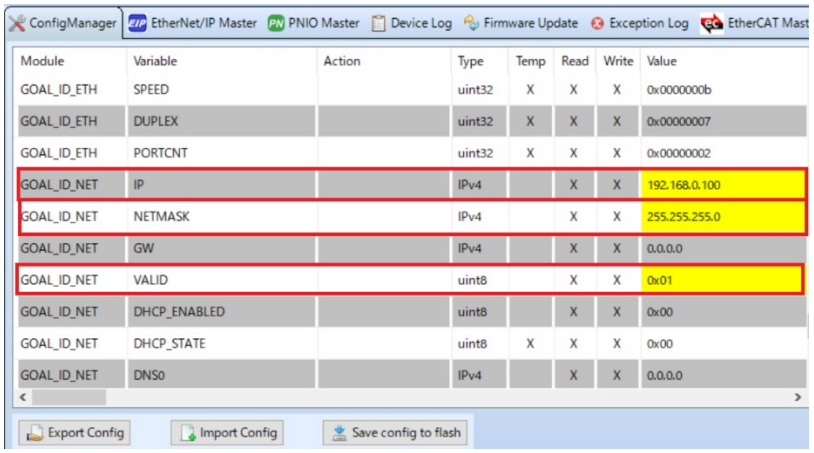

Figure 3-20 Set IP address on R-IN32M3 module

-6. Select [Write Configuration] button to download the changed Configuration Manager variables to the R-IN32M3 Module.

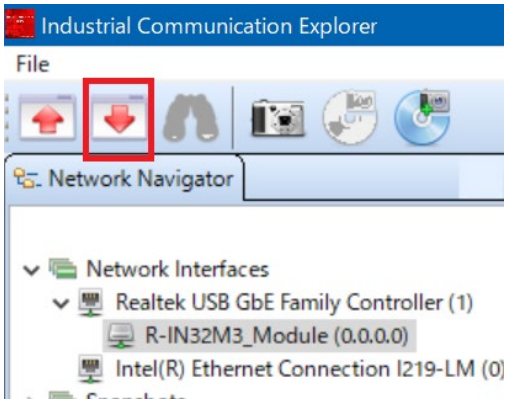

Figure 3-21 Download Config variables

-7. If a change confirmation dialog is displayed, select [Yes]. The changed value is then transferred to the R-IN32M3 Module and changed in RAM only. If change the value of Flash incorporated in the R-IN32M3 Module, use the [Save config to flash]. The changed IP address setting is applied after the system is restarted, so restart this board.

    For details on the IP address setting, refer to Chapter 4.3.

-8. Select [PNIO Master] panel, and then select [Scan device].

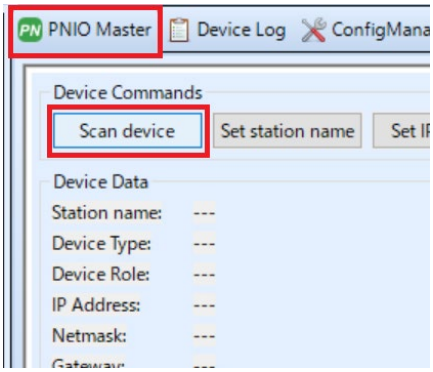

Figure 3-22 PNIO Master

-9. When a PROFINET device is detected, "PNIO: Found 1 device" appears in [Messages] panel at the bottom of the screen, and [Device Data] in the [PNIO Master] panel displays the device information of the R-IN32M3 Module.

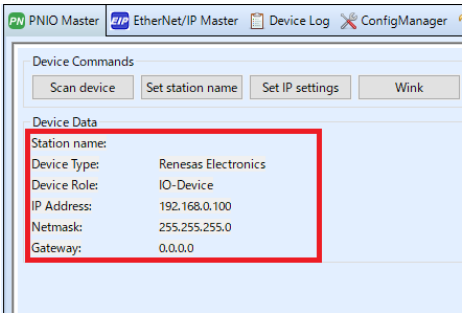

Figure 3-23 Device Data

-10. Open the I/O panel of [PNIO Master] panel and select [Load GSDML file] button to import the GSDML file. GSDML files can be found in the following folder:

Table 3-3 GSDML Files

Sample software	GSDML file
01_pnio	01_pnio\gsdml\GSDML-V2.43-Renesas-irj45-20240130_01_pnio.xml
10_multi_protocol	
11_pnio_http	
04_pnio_largesize	04_pnio_largesize \gsdml\GSDML-V2.43-Renesas-irj45-20240130_04_pnio.xml

Verify that [Slots:] and [Modules] display contents as set in GSDML, select [32] from pull-down of [Device Interval] and then push [Connect] button. If the connection is successful, this button switches to [Disconnect] button. In addition, the protocol status LED lights up.

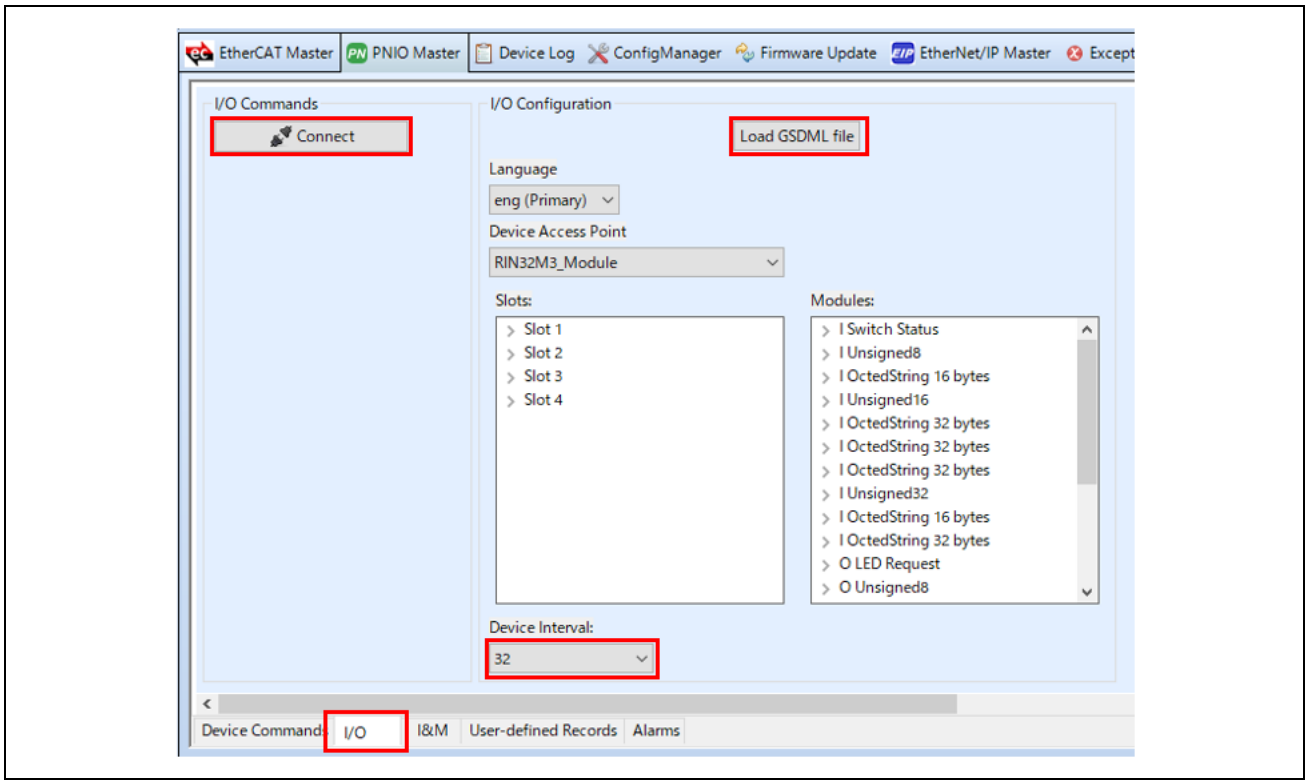

Figure 3-24 GSDML

-11. Data communication for sample applications.
The sample software implements two types of data transmission/reception applications as example applications.

- **Remote-IO (LED/Switch):** LED lighting control and Switch status from the evaluation board

  Target Project : 01_pnio, 04_pnio_largesize, (10_multi_protocol), 11_pnio_http

- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 01_pnio, 04_pnio_largesize, (10_multi_protocol), 11_pnio_http

- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 04_pnio_largesize

Table 3-4 Application defied:

sample		Sample app.	Slot	Size
04_pnio_large	01_pnio	LED Data Reception	Slot 2	1
		Mirror Data Reception	Slot 4	16
		Switch Data Transmission	Slot 1	1
		Mirror Data Transmission	Slot 3	16
	-	Mirror Data Reception_1 (rpc)	Slot 6	32
		Mirror Data Reception_2 (rpc)	Slot 8	32
		Mirror Data Reception_3 (rpc)	Slot 10	32
		Mirror Data Transmission_1 (rpc)	Slot 5	32
		Mirror Data Transmission_2 (rpc)	Slot 7	32
		Mirror Data Transmission_3 (rpc)	Slot 9	32

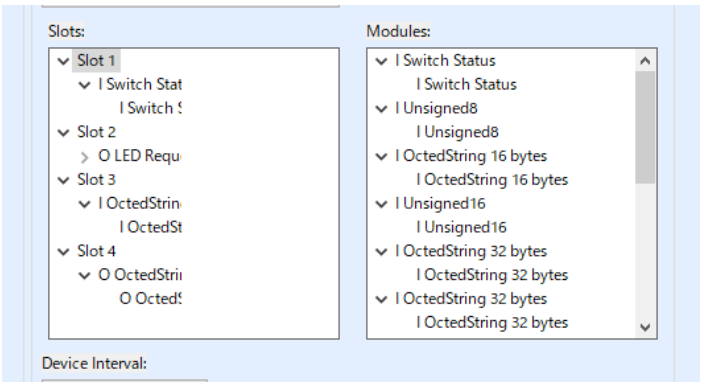

Figure 3-25 Application define (ex. 01_pnio)

Remote-IO (LED/Switch)

Input data corresponding to P-mod switches is registered in Switch, and Output data corresponding to P-mod LED is registered in LED as 1-byte data.

I/O app.		Remote I/O control
Switch (Slot 1)	P-mod Switch	Input Data value changes by operating P-mod switches
LED (Slot 2)	P-mod LED	P-mod LED changes by registering a value to Output Data.

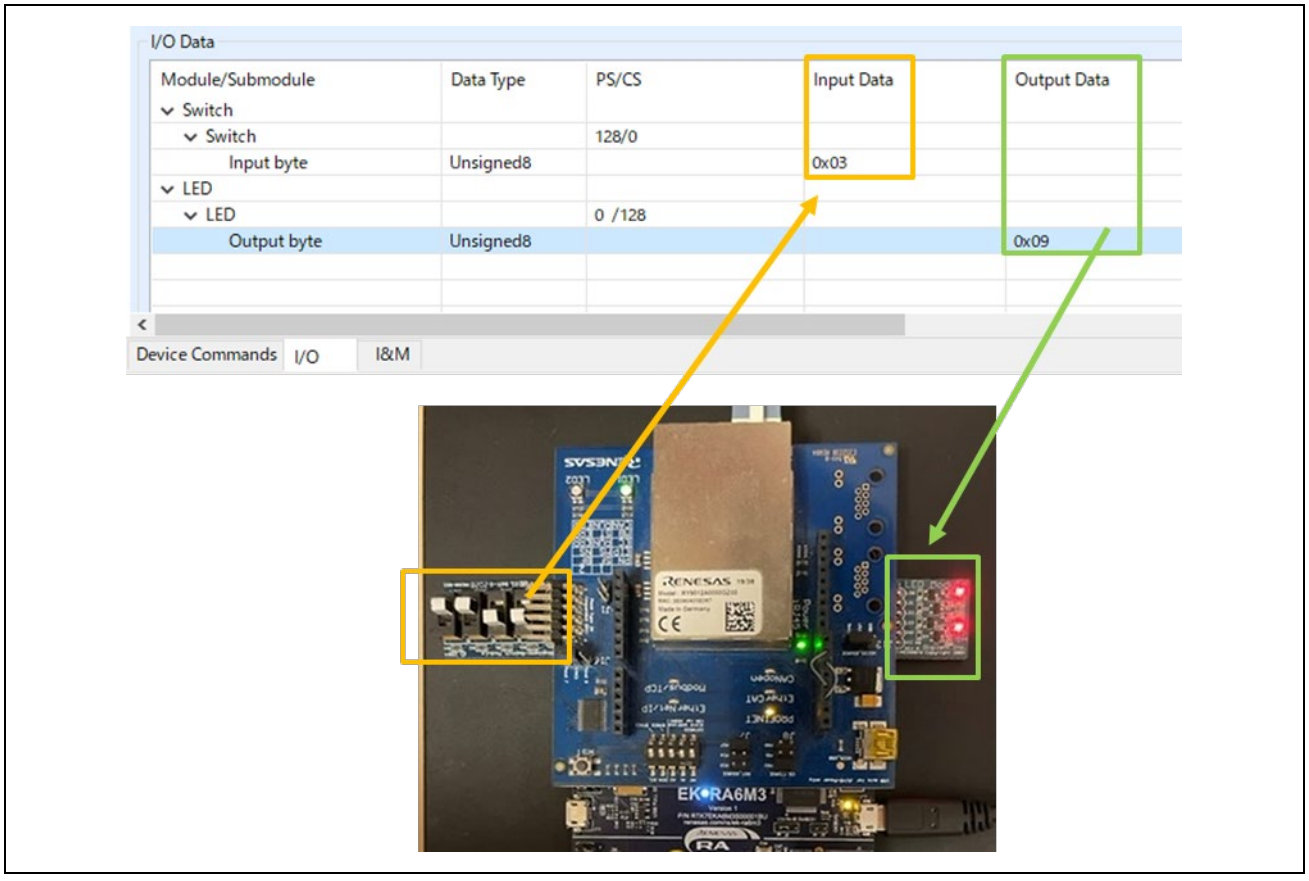

Figure 3-26 Remote I/O control [PROFINET]

Mirror control

When a module receives a value registered in Output Data from the master, the value is mirrored back to the master and reflected in Input Data.

Here is an example of mirror control for the 01_pnio sample.

Mirror app.	Mirror control
Mirror Data Transmission (Slot 3: Input 16Byte)	Values sent from the module under mirror control are reflected in Input Data.
Mirror Data Reception (Slot 4: Output 16Byte)	Module receives values registered in Output Data

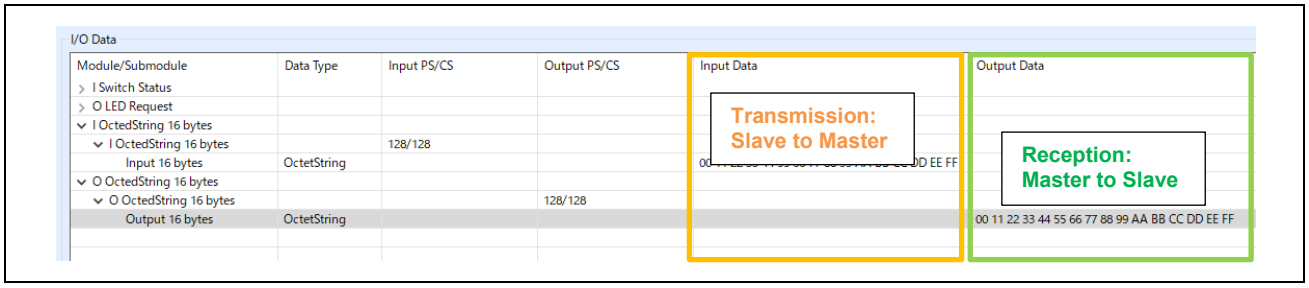

Figure 3-27 Mirror control [PROFINET]

-12. [Disconnect] terminates communication.

3.4.2 EtherNet/IP

This chapter describes an example of EtherNet/IP communication.
The target sample is below.

Table 3-5 EtherNet/IP Sample software

Sample software	Overview
02_eip	Cyclic connection sample
05_eip_largesize	Cyclic and RPC (Large Size data) connection sample
10_multi_protocol	01_pnio, 02_eip, 03_ecat, 07_modbus multi sample
12_eip_http	02_eip sample Enhanced [web saver and host MCU update function]

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.5 to 3.3.6. When the sample application is run, the protocol display LED (EtherNet/IP) turn on.

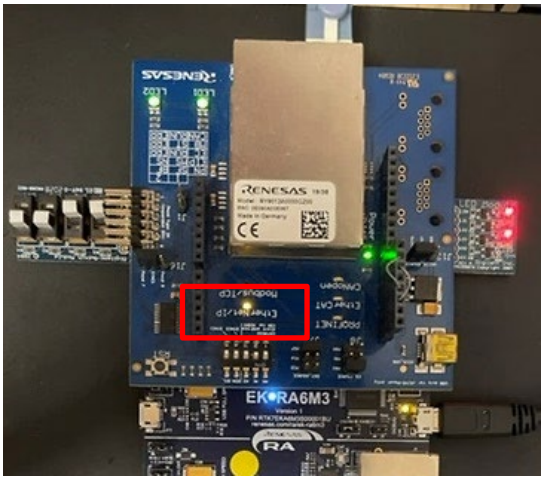

Figure 3-28 Protocol LED: EtherNet/IP

-2. Set IP address

Set Static IP address. Open the [Network Properties] of the network adapter connected to the R-IN32M3 Module and set the static IP (using 192.168.0.1 as an example).

IP address	192.168.0.1
Netmask	255.255.255.0

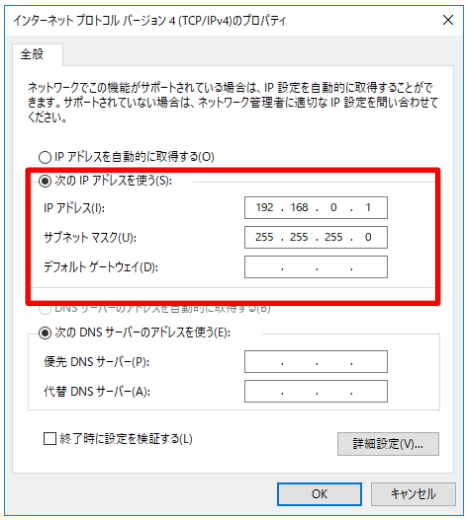

Figure 3-29 Set Static IP address

2. Master connection

Management tool can be used as a EtherNet/IP simple Scanner. It is included with " R-IN32M3 Module (RY9012A0) Sample Package" (R18AN0064EJ****) along with this sample software.

Execute "ice.exe" file in the folder below to start the Management tool. For more information about the Management tool, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

-1. Select network to use in [Network Navigator] panel and select [Scan Network] button.

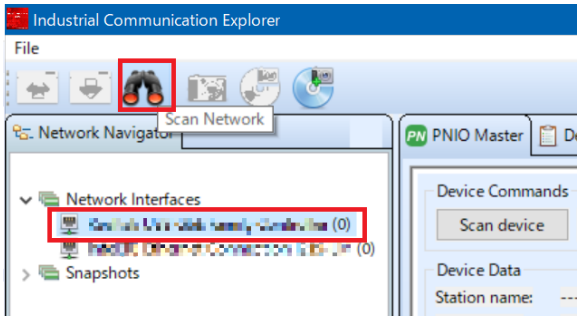

Figure 3-30 Scan network

-2. “Scan complete. found 1 device” message is displayed in [Network Scan] dialog, then select [OK].

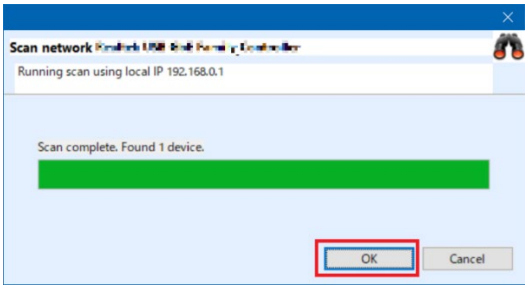

Figure 3-31 Scan completed

-3. In [Network Navigator] panel in the scanned network, “R-IN32M3_Module” is displayed as the new device, so select [R-IN32M3_Module].

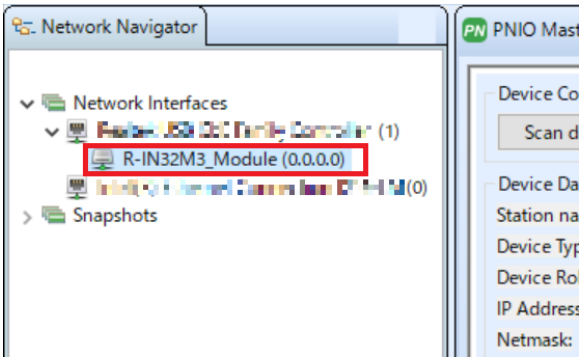

Figure 3-32 Select R-IN32M3 Module

-4. In order to communicate with the R-IN32M3 Module, the IP address of the R-IN32M3 Module must be in the same IP network as the IP address of the PC. Therefore, access the configuration manager variables (volatile memory and non-volatile memory stored configuration variables) of the R-IN32M3 Module to set the IP address and Netmask.
With [R-IN32M3_Module] selected, select [Read Configuration] button while displaying the [ConfigManager] panel.

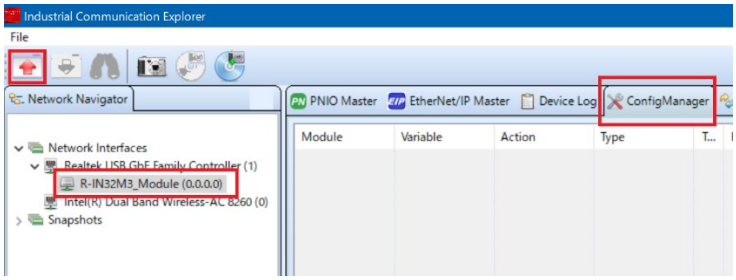

Figure 3-33 ConfigManager

-5. In the configurations displayed in the [ConfigManager] panel, change the following items. Note that it is required to set VALID to 1 due to enable the IP address and Netmask. The changed Value will be highlighted in yellow.

Module	Variable	Value example
GOAL_ID_NET	**IP**	192.168.0.100
GOAL_ID_NET	**NETMASK**	255.255.255.0
GOAL_ID_NET	**VALID**	0x01

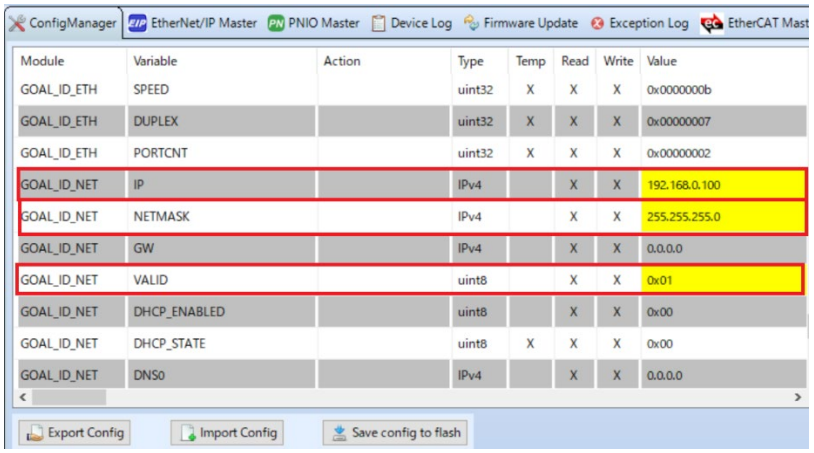

Figure 3-34 Set IP address on R-IN32M3 module

-6. Select [Write Configuration] button to download the changed Configuration Manager variables to the R-IN32M3 Module.

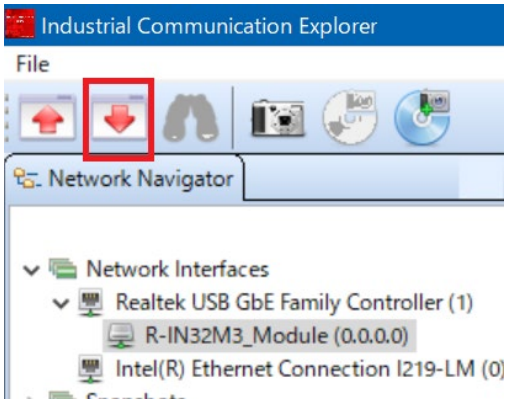

Figure 3-35 Download Config variables

-7. If a change confirmation dialog is displayed, select [Yes]. The changed value is then transferred to the R-IN32M3 Module and changed in RAM only. If change the value of Flash incorporated in the R-IN32M3 Module, use the [Save config to flash]. The changed IP address setting is applied after the system is restarted, so restart this board.

    For details on the IP address setting, refer to Chapter 4.3.

-8. Open [EtherNet/IP Master] panel and select [Scan device] button.

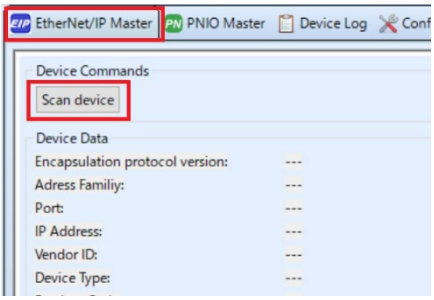

Figure 3-36 Scan device

-9. When an EtherNet/IP device is detected, [Messages panel] at the bottom of the screen displays "EIP: Found 1 device" and [Device Data] in [EtherNet/IP Master] panel displays the device information for the R-IN32M3 Module.

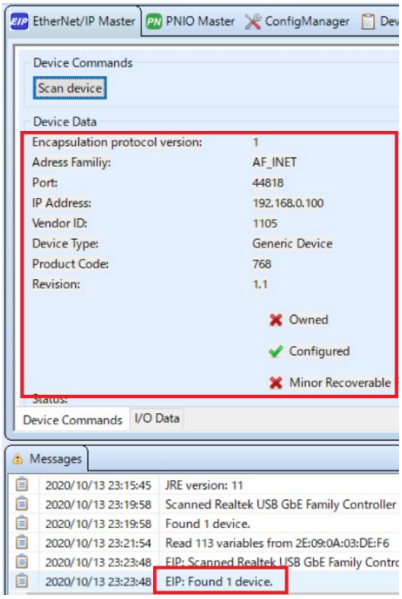

Figure 3-37 Device Data

-10. Open [I/O Data] panel in [EtherNet/IP Master] panel.
The sample software implements two types of data transmission/reception applications as example applications.

- **Remote-IO (LED/Switch):** LED lighting control and Switch status from the evaluation board

  Target Project : 02_eip, 05_eip_largesize, (10_multi_protocol), 12_eip_http

- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 02_eip, 05_eip_largesize, (10_multi_protocol), 12_eip_http

- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 05_eip_largesize

Application defied:

Table 3-6 Data application

sample		Sample app.	Assembly ID	size
05_eip_large	02_eip	LED Data Reception	150	1
		Mirror Data Reception	151	16
		Switch Data Transmission	100	1
		Mirror Data Transmission	101	16
	-	Mirror Data Reception_1 (rpc)	152	32
		Mirror Data Reception_2 (rpc)	153	32
		Mirror Data Reception_3 (rpc)	154	32
		Mirror Data Transmission_1 (rpc)	102	32
		Mirror Data Transmission_2 (rpc)	103	32
		Mirror Data Transmission_3 (rpc)	104	32

Table 3-7 Configuration

sample		Sample app.	Assembly ID	size
05_eip_large	02_eip	Config Data	200	10

Remote-IO (LED/Switch)

Refer to Table 3-6 and Table 3-7 to set the connection parameters.

Packet interval in ms is left at the default value.

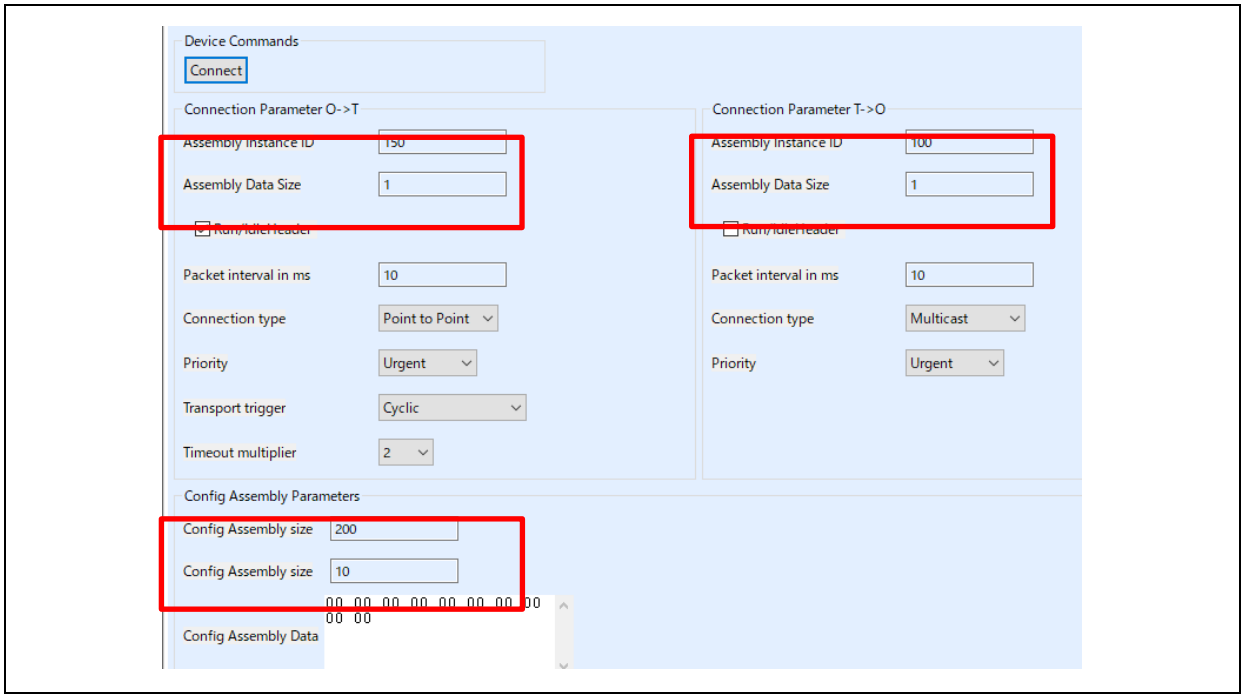

Figure 3-38 Remote-IO application parameter

Mirror control

Refer to Table 3-6 and Table 3-7 to set the connection parameters.

Here is an example of mirror control for the 02_eip sample. Packet interval in ms is left at the default value.

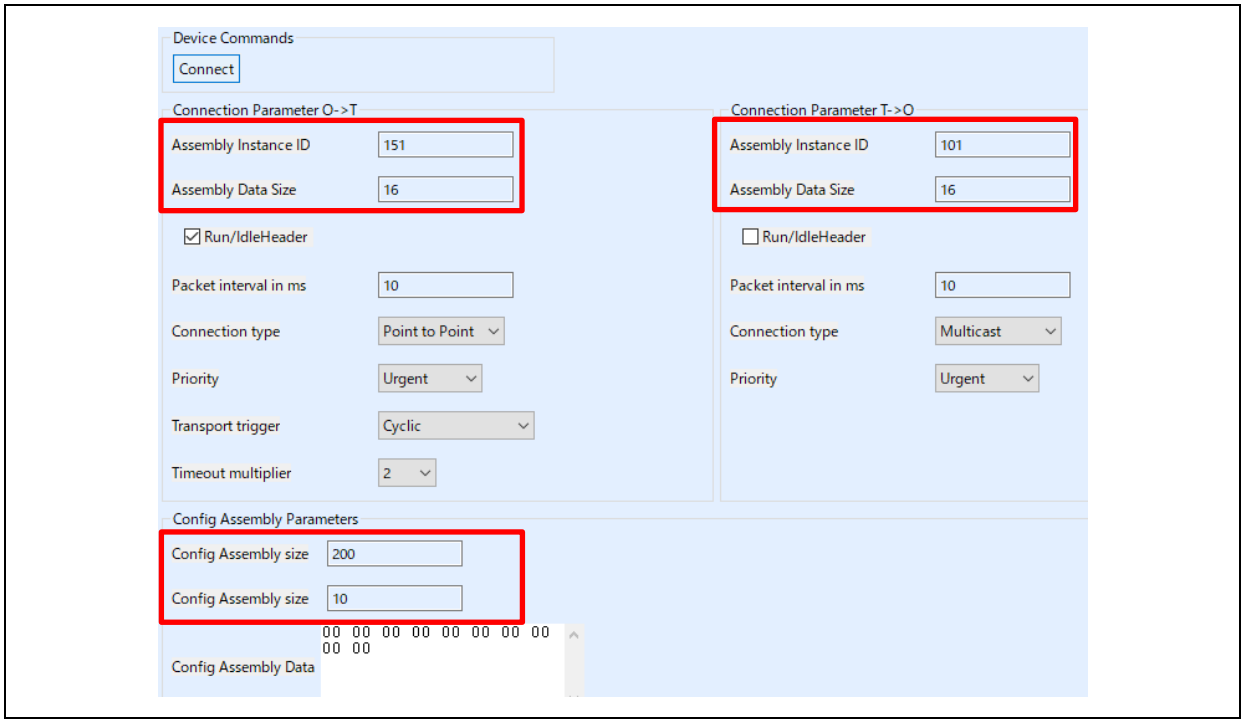

Figure 3-39 Mirror application parameter

Mirror control (RPC)

05_eip_largesize project also provides process data communication using RPC, which is a method of process data communication via RPC data frames in SPI frame (128 bytes) between R-IN32M3 Module and the host MCU. Since RPC frames, which are originally intended for asynchronous data communication, are used, it is possible to send larger data than the method using ordinary Cyclic data frames (more than 69 bytes of process data can be transferred), but the update cycle of the application is restricted. See “R-IN32M3 Module User’s Implementation Guide (R30AN0402EJ****) for details.

Refer to Table 3-6 and Table 3-7 to set the connection parameters. Figure 3-40 shows an example of a communication configuration for Mirror Data Reception_2 (153) and Mirror Data Transmission_2 (103).

The configurable Packet interval in ms setting (so-called RPI setting) affects the data size and the number of connections. Please evaluate carefully before deciding on the configuration values using RPC.

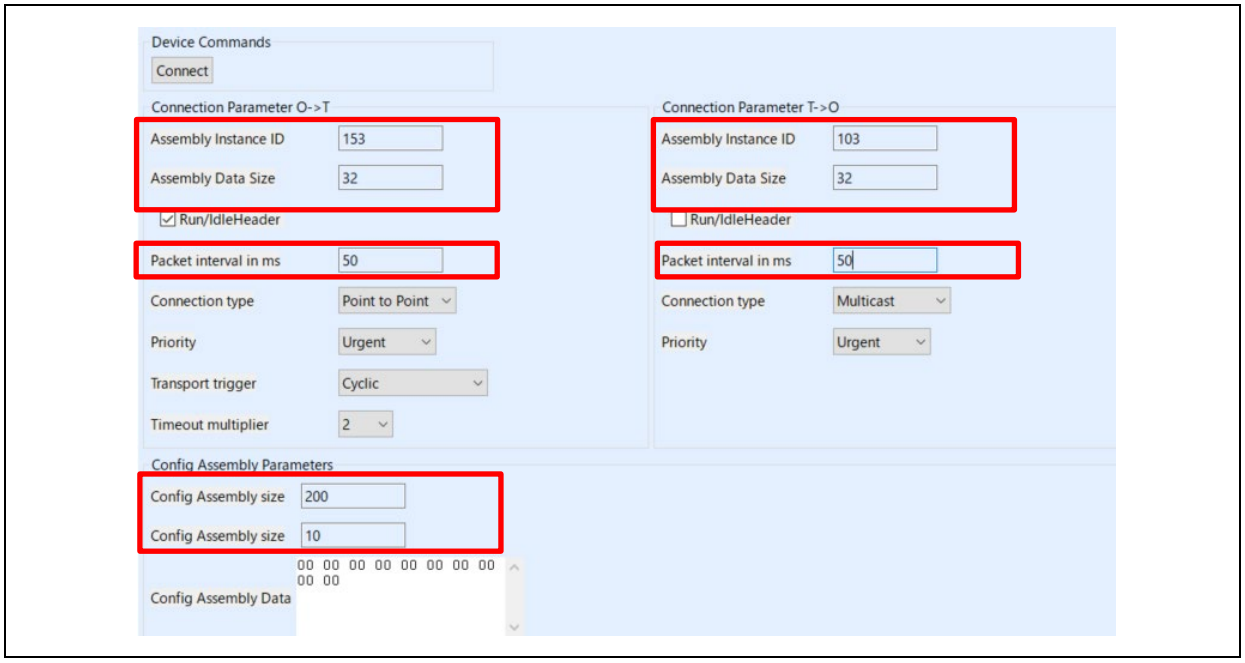

Figure 3-40 Mirror application (RPC) parameter

-11. Select the [Connect] button, which switches to the [Disconnect] button if the connection is successfully established. Also, the protocol status LED on this board will light up.

-12. Check the input/output of the application.

Remote-IO (LED/Switch)

Input data corresponding to general-purpose input switches on the SEMB1320 is registered in Switch, and Output data corresponding to general-purpose output LEDs on the SEMB1320 is registered in LED as 1-byte data.

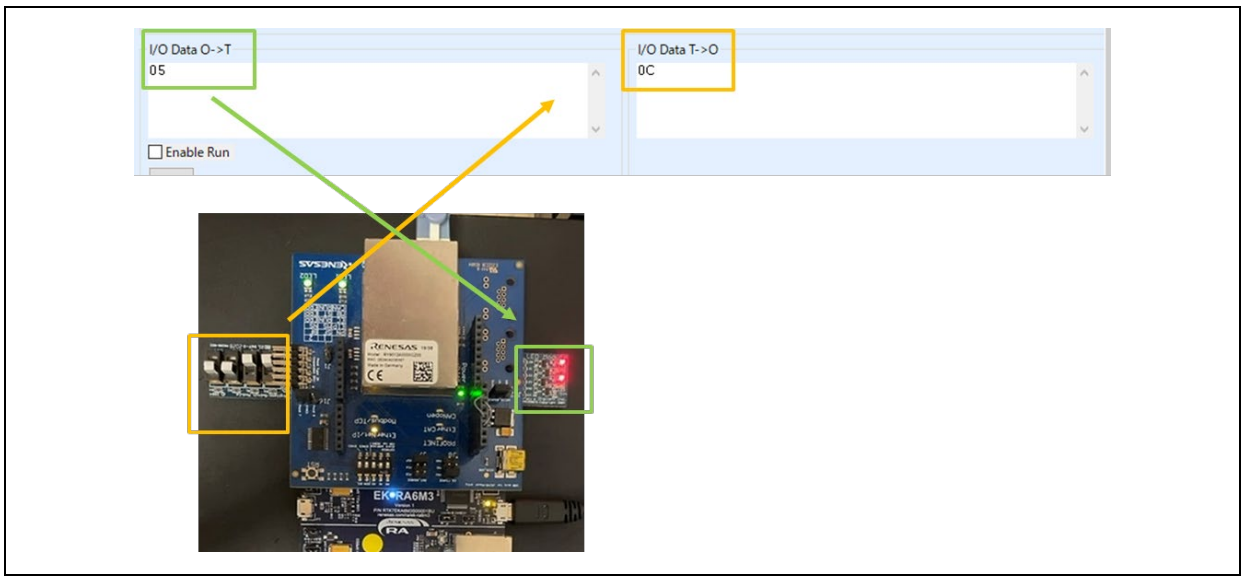

Figure 3-41 Remote-IO (LED/Switch) control [EtherNet/IP]

Mirror control

When a module receives a value registered in I/O Data O->T from the master, the value is mirrored back to the master and reflected in I/O Data T->O.
Here is an example of mirror control for the 02_eip sample.

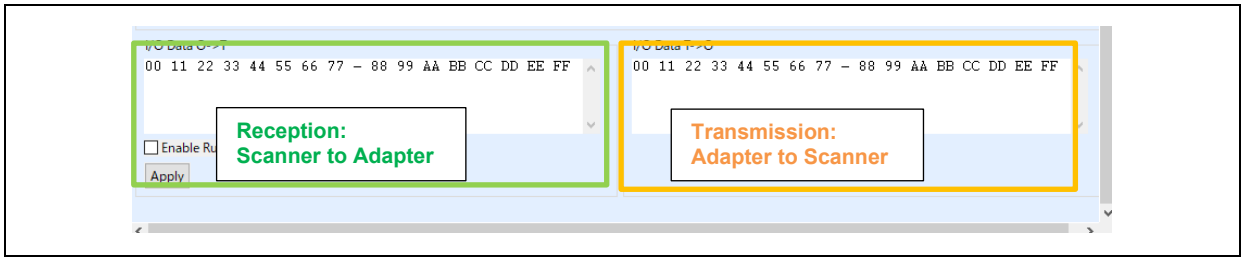

Figure 3-42 Mirror control [EtherNet/IP]

-13. [Disconnect] terminates communication.

3.4.3 EtherCAT

This chapter describes an example of EtherCAT communication.
The target sample is below.

Table 3-8 EtherCAT Sample software

Sample software	Overview
03_ecat	Cyclic connection sample
06_ecat_largesize	Cyclic and RPC (Large Size data) connection sample
10_multi_protocol	01_pnio, 02_eip, 03_ecat, 07_modbus multi sample
13_ecat_http	03_ecat sample Enhanced [web saver and host MCU update function]

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.5 to 3.3.6. When the sample application is run, the protocol display LED (EtherCAT) turn on.

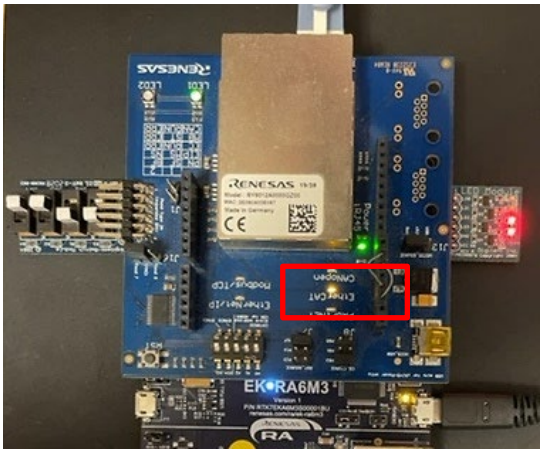

Figure 3-43 Protocol LED: EtherCAT

-2. Set Network Adapter

In order to send and receive EtherCAT frames using TwinCAT 3, the driver must be activated, see "Software PLC Connection Guide TwinCAT (R30AN0380ED****)" for TwinCAT driver installation.

Drivers:

- TwinCAT RT-Ethernet Filter Driver
- TwinCAT Ethernet Protocol for All Network Adapters

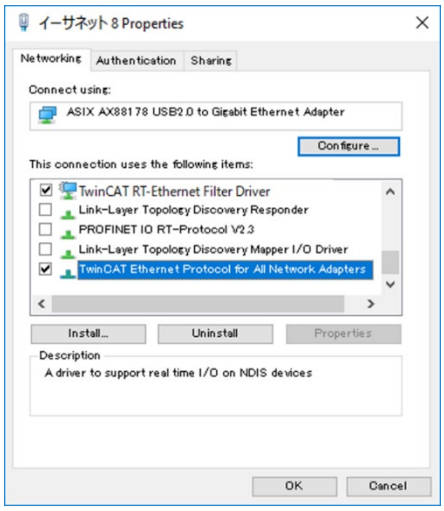

Figure 3-44 Network Adapter: EtherCAT

Note: Depending on the network driver type, the TwinCAT RT-Ethernet Filter Driver may not be installed. In this case, only the **TwinCAT Ethernet Protocol for All Network Adapters** enabled.

-3. ESI file

Before starting TwinCAT 3, an ESI (EtherCAT Slave Information) file must be stored in the TwinCAT folder.

The ESI file is stored in the esi folder of the sample program.

Table 3-9 ESI Files

Sample software	ESI file
03_ecat	03_ecat\esi\Renesas_RINmodule_03ecat.xml
10_multi_protocol	
13_ecat_http	
06_ecat_largesize	06_ecat_largesize \esi\Renesas_RINmodule_06ecat.xml

[Folder for ESI storage]

C:\TwinCAT\3.1\Config\Io\EtherCAT

2. Master connection

TwinCAT from Beckhoff Automation is used as the EtherCAT master. See "Software PLC Connection Guide TwinCAT (R30AN0380ED****)" for TwinCAT connection.

Operate TwinCAT according to the following procedure to check the connection with this sample application and data transmission/reception.

-1. Windows start menu, select [Beckhoff] -> [TwinCAT 3] -> [TwinCAT XAE Shell].

-2. Select [File] -> [New] -> [Project] and create a new project of type [TwinCAT XAE Project].

-3. Select [File] -> [New] -> [Project] and create a new project of type [TwinCAT XAE Project].

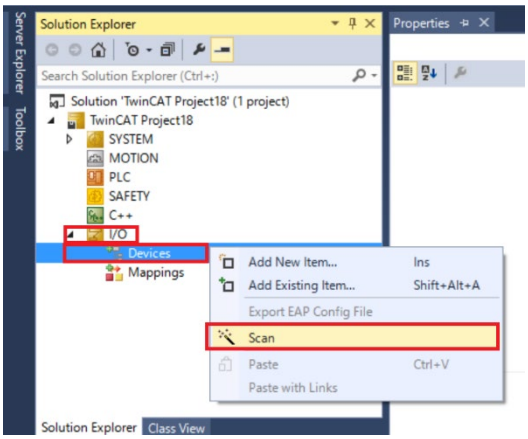

Figure 3-45 Scan network

-4. Click [OK] on [HINT: Not all types of devices can be found automatically] dialog.
Click [OK] on [Init12\IO:Set State…]

-5. When an EtherCAT module is detected, the connected network adapter is displayed with a check mark (☑).

-6. Click [Yes] in [Scan for Boxes] dialog
Click [Yes] in [Active Free Run] dialog

-7. The connection is complete when [Device x] → [Box 1] is added under [I/O] → [Devices].

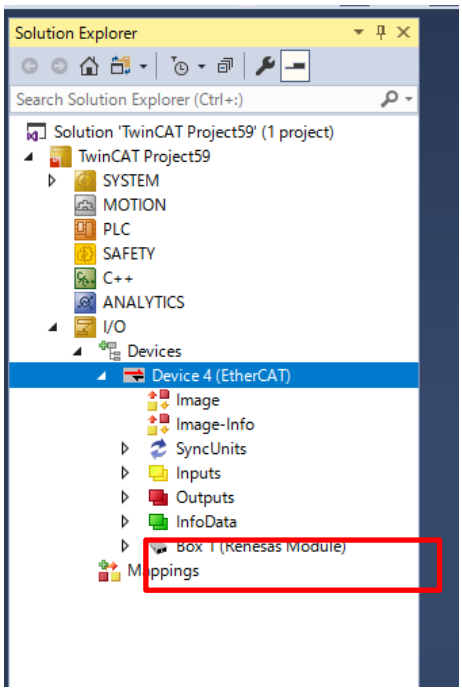

Figure 3-46 TwinCAT connection

If the EEPROM is blank and [Box 1 (PFFFFFFFF RFFFFFFFF)] is displayed, or if the ESI of different sample application is written, it is necessary to write the ESI file of corresponding sample application to the EEPROM. In this case, please refer to "Software PLC Connection Guide TwinCAT (R30AN0380JJ****)" to program SII in EEPROM.

-8. Data communication for sample applications.
The sample software implements two types of data transmission/reception applications as example applications.

- **Remote-IO (LED/Switch):** LED lighting control and Switch status from the evaluation board

  Target Project : 03_ecat, 06_ecat_largesize , (10_multi_protocol), 13_ecat_http

- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 03_ecat, 06_ecat_largesize , (10_multi_protocol), 13_ecat_http

- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 06_ecat_largesize

Table 3-10 Application defied:

sample		Sample app.	Index [sub]	Size
06_ecat_large	03_ecat	LED Output	0x6200 [1]	1
		Mirror Data out 1-16	0x6201 [1]	16
		Switch Data Transmission	0x6000 [1]	1
		Mirror Data in 1-16	0x6001 [1]	16
	-	Mirror Data out (rpc) 1-31	0x6210 [1]	31
		Mirror Data out (rpc) 32-62	0x6210 [2]	31
		Mirror Data out (rpc) 63-93	0x6210 [3]	31
		Mirror Data in (rpc) 1-31	0x6010 [1]	31
		Mirror Data in (rpc) 32-62	0x6010 [2]	31
		Mirror Data in (rpc) 63-93	0x6010 [3]	31

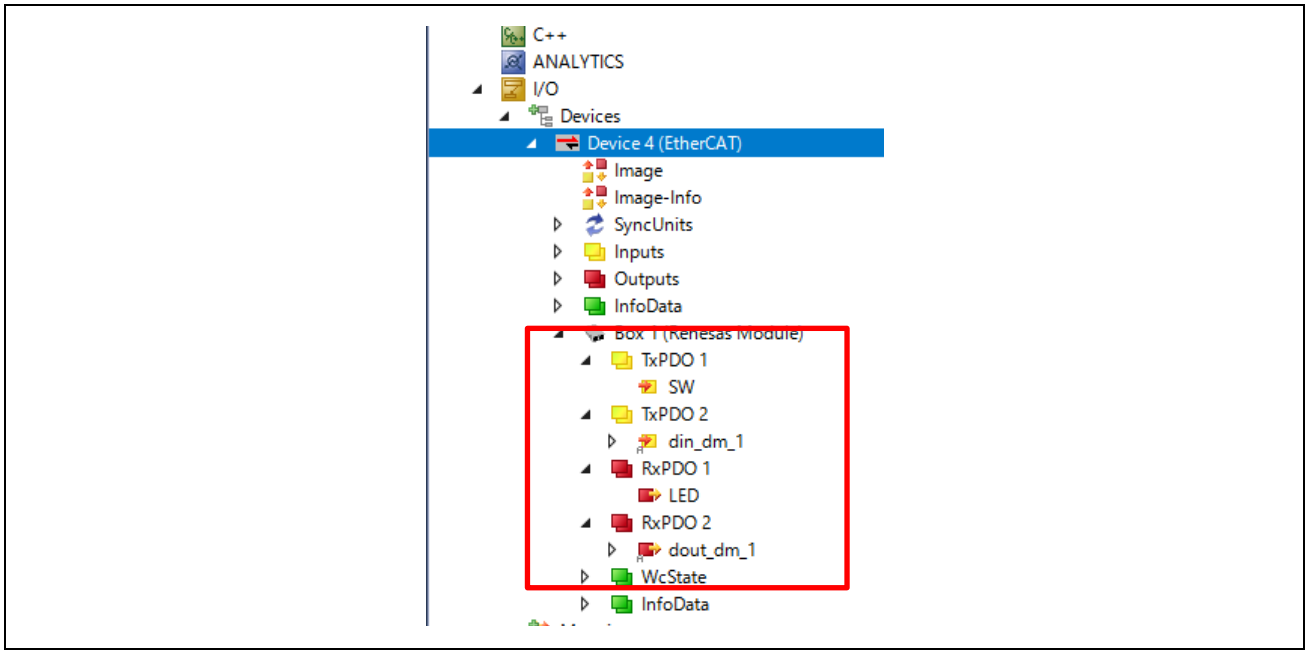

Figure 3-47 Application define (ex. 03_ecat)

Remote-IO (LED/Switch)

Input data corresponding to general-purpose input switches on the SEMB1320 is registered in Switch, and Output data corresponding to general-purpose output LEDs on the SEMB1320 is registered in LED as 1-byte data.

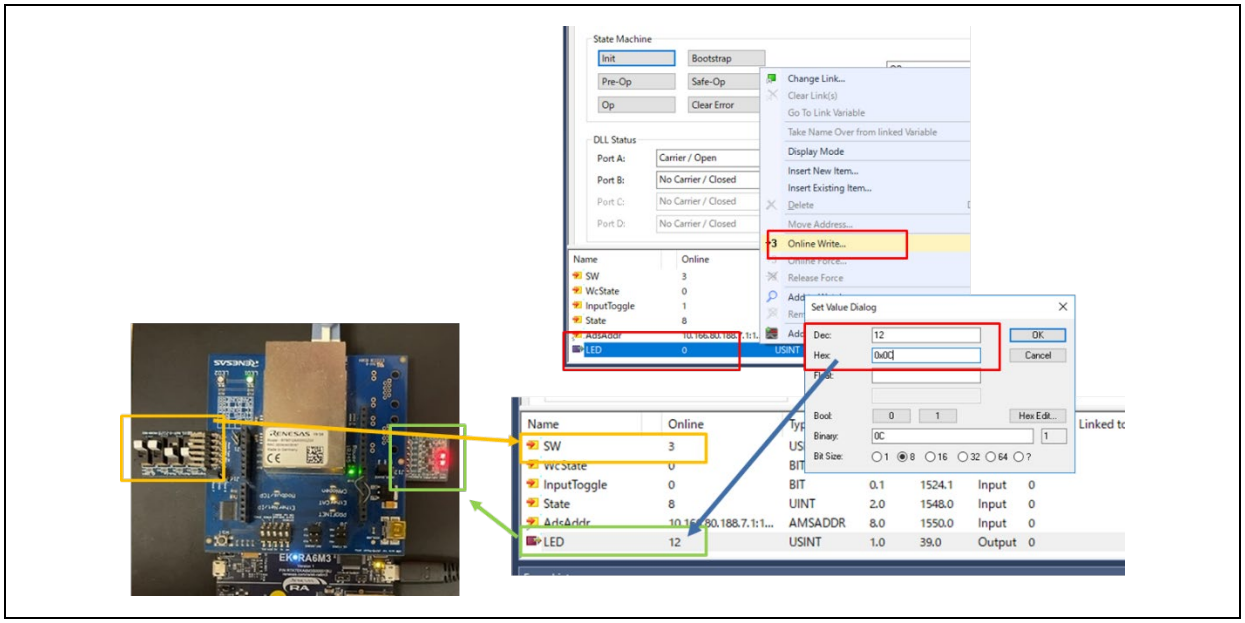

Figure 3-48 Remote I/O control [EtherCAT]

Mirror control

When a module receives a value registered in Output Data from the master, the value is mirrored back to the master and reflected in Input Data.

Here is an example of mirror control for the 03_ecat sample.

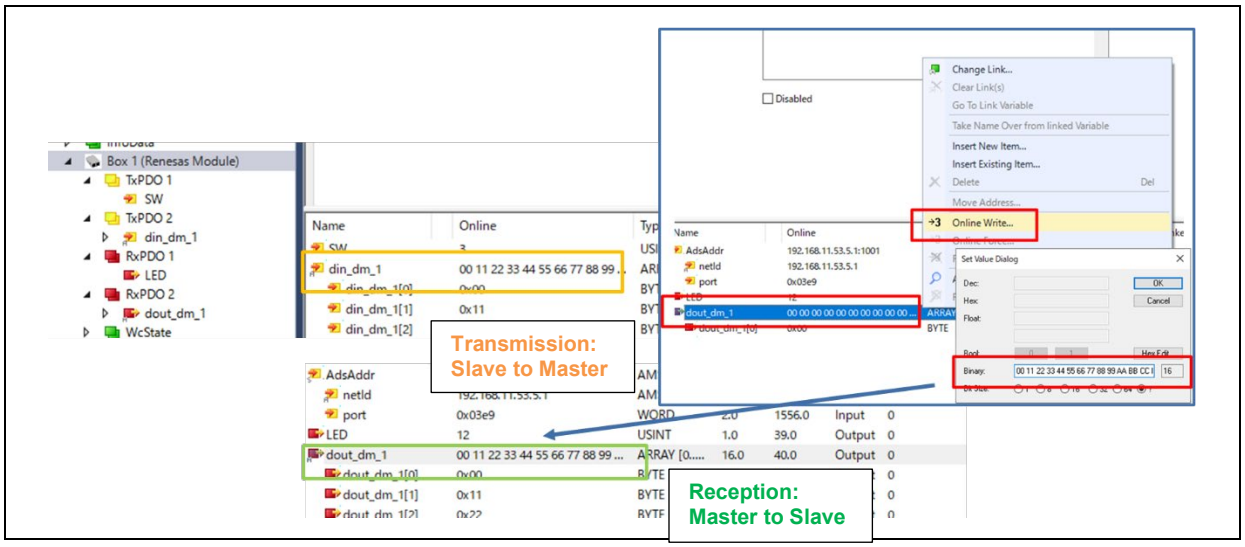

Figure 3-49 Mirror control [EtherCAT]

3.4.4 Modbus TCP

This chapter describes an example of Modbus TCP communication.
For an example of Modbus TCP, see "R-IN32M3 Module Modbus TCP Start-Up Manual (R30AN0406EJ****)".

The target sample is below.

Table 3-11 Modbus Sample software

Sample software	Overview
07_mbus_tcp_server	Modbus TCP sample application

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

3.4.5 multi-protocol

This chapter describes an example of multi-protocol communication (PROFINET, EtherNet/IP, EtherCAT, Modbus TCP).
The target sample is below.

Table 3-12 multi-protocol Sample software

Sample software	Overview
10_multi_protocol	multi-protocol [01_pnio, 02_eip, 03_ecat, 07_modbus] sample application

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

The protocol is executed according to the value of the P-mod, referring to Chapters 2.2.

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to 3.3.5 to 3.3.6.

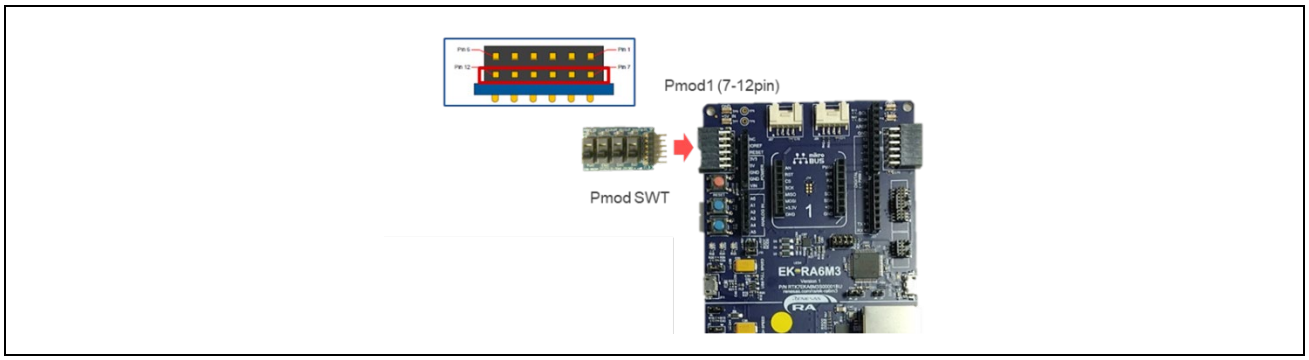

Figure 3-50 General purpose switch (SW9)

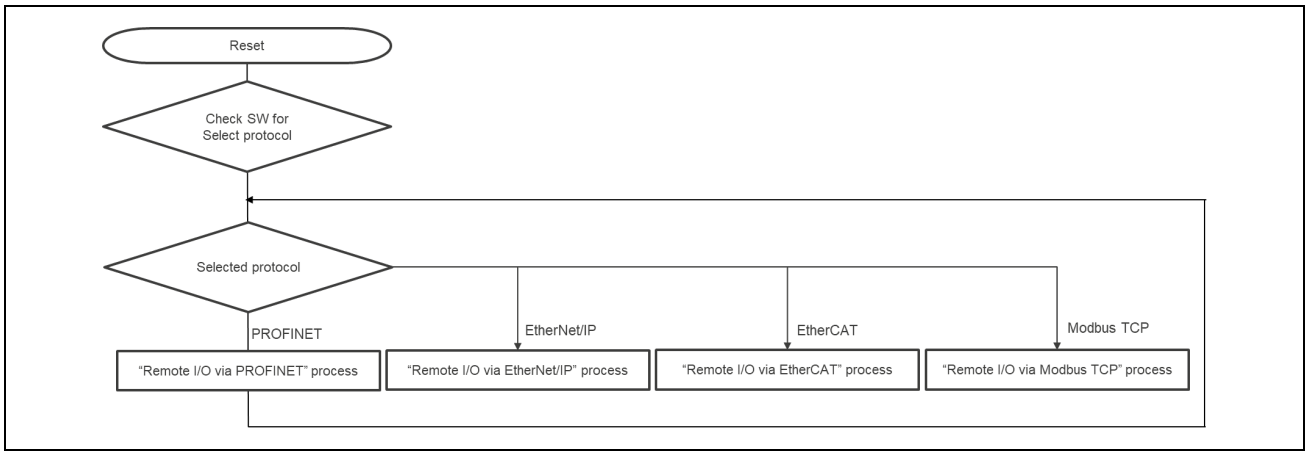

Figure 3-51 Multi-protocol selector flow

Pmod SWT	Protocol
on-off-off-off	PROFINET
off-on-off-off	EtherNet/IP
off-off-on-off	EtherCAT
off-off-off-on	Modbus TCP Server
others	PROFINET

PROFINET：Pmod SWT [on-off-off-off] and others

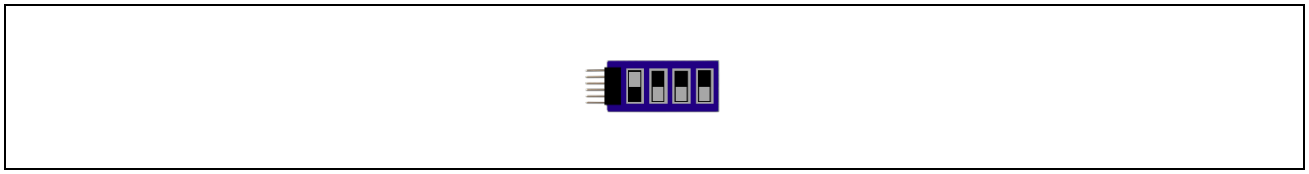

Figure 3.52 PROFINET

EtherNet/IP：Pmod SWT [off-on-off-off]

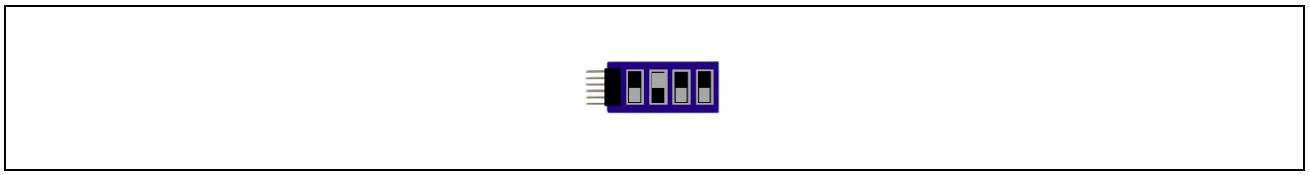

Figure 3.53 EtherNet/IP

EtherCAT：Pmod SWT [off-off-on-off]

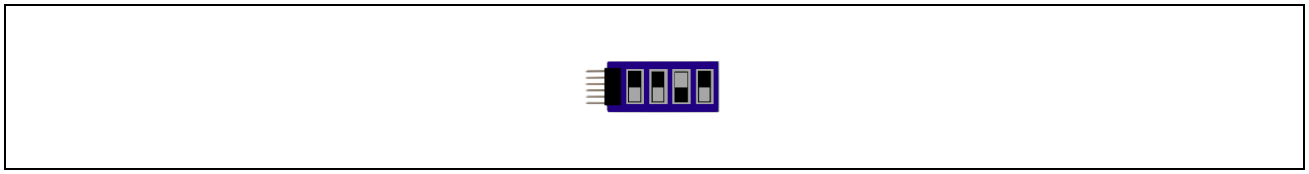

Figure 3.54 EtherCAT

Modbus TCP Server：Pmod SWT [off-off-off-on]

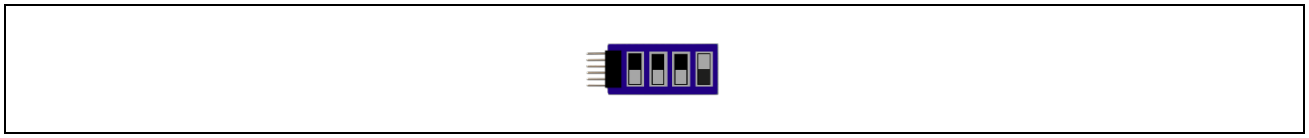

Figure 3.55 Modbus TCP Server

-2. Set Network Adapter

Refer to the network adapter configuration procedures in the respective chapters according to the selected protocol.

Protocol	Refer
PROFINET	3.4.1 PROFINET
EtherNet/IP	3.4.2 EtherNet/IP
EtherCAT	3.4.3 EtherCAT
ModbusTCP	3.4.4 Modbus TCP

2. Master connection

Refer to the Master connect procedures in the respective chapters according to the selected protocol.

Protocol	Refer
PROFINET	3.4.1 PROFINET
EtherNet/IP	3.4.2 EtherNet/IP
EtherCAT	3.4.3 EtherCAT
ModbusTCP	3.4.4 Modbus TCP

3.4.6 Web saver

The web browser access procedure using the sample program with the web server function.
The web content provided as a sample shows protocol status LED in *index.html*. These html data are provided in *goal_http_fs.h*.

The target sample is below.

Table 3-13 Web Server Sample software

Sample software	Overview
11_pnio_http	01_pnio sample Enhanced [web saver and host MCU update function]
12_eip_http	02_eip sample Enhanced [web saver and host MCU update function]
13_ecat_http	03_ecat sample Enhanced [web saver and host MCU update function]

To use this sample application, you need to update the firmware version of the R-IN32M3 Module to 2.1.0.0 or later. For the firmware update method, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)".

Evaluation Environment Setup

Refer to the evaluation environment setup procedures in the respective chapters according to the selected protocol.

Protocol	Refer
PROFINET	3.4.1 PROFINET
EtherNet/IP	3.4.2 EtherNet/IP
EtherCAT	3.4.3 EtherCAT
ModbusTCP	3.4.4 Modbus TCP

The web browser access procedure

PROFINET, EtherNet/IP

The following conditions are used as an example.

PC Network	**[IP]** 192.168.0.1 **[MASK]** 255.255.255.0
R-IN32M3 module	**[IP]** 192.168.0.100 **[MASK]** 255.255.255.0

While the program is running, enter the IP address (192.168.0.100) specified for the R-IN32M3 Module in your web browser to access it, and the web server provided as a sample will be loaded.

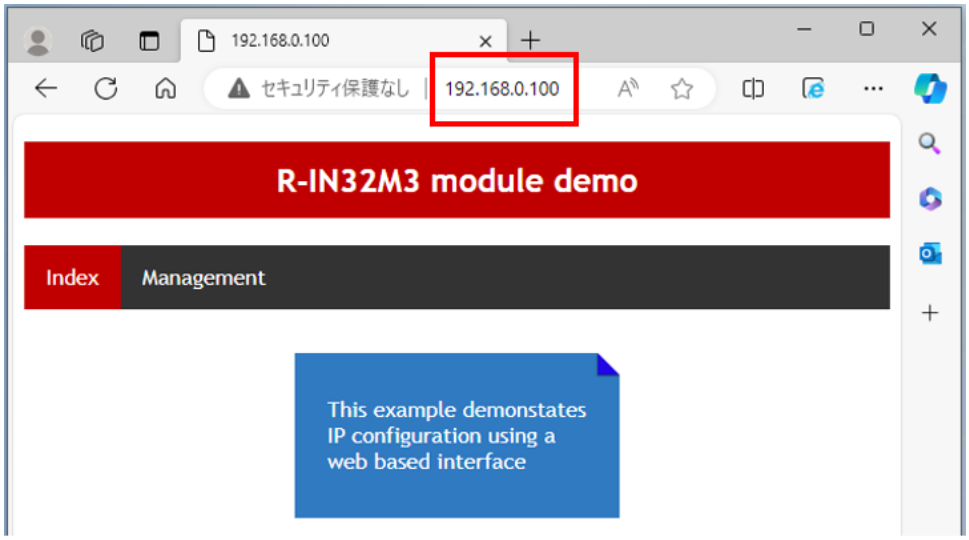

Figure 3.56 web browser access

EtherCAT

The following conditions are used as an example.

The EtherCAT web server uses TwinCAT and the following conditions are explained as an example.

PC Network	**[IP]** 192.168.1.99 **[MASK]** 255.255.255.0
R-IN32M3 module (EtherCAT EoE)	**[IP]** 192.168.1.100 **[MASK]** 255.255.255.0

-1. Enable TwinCAT Driver and Static IP in the network adapter configuration.

- TwinCAT RT-Ethernet Filter Driver
- TwinCAT Ethernet Protocol for All Network Adapters
- Internet Protocol version 4 (TCP/IPv4)

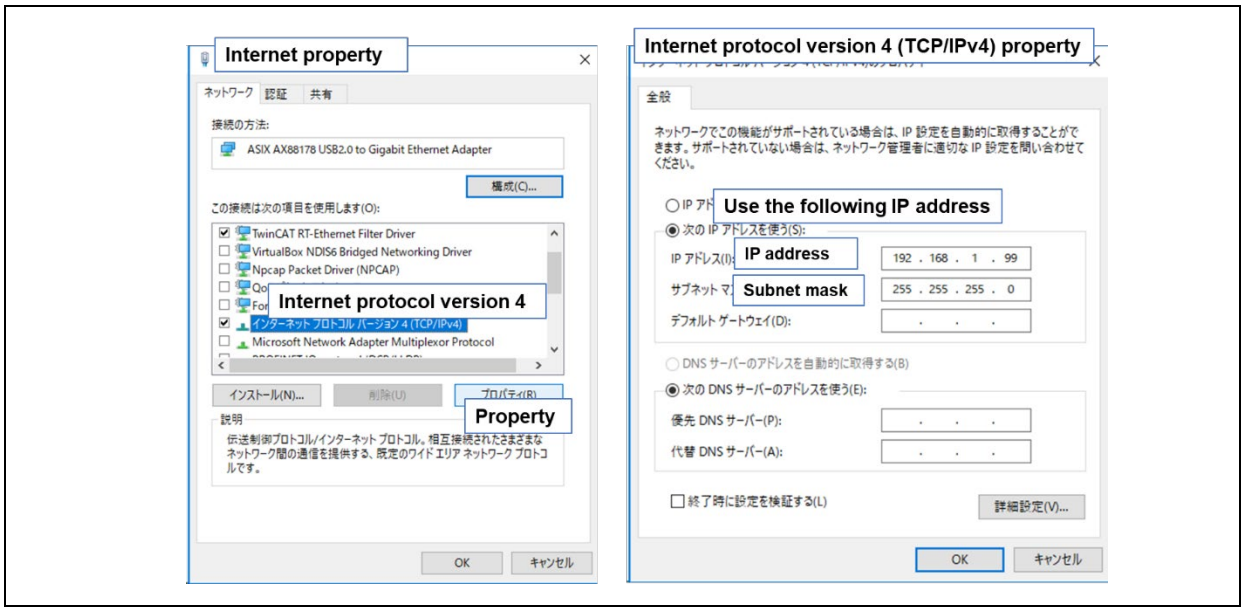

Figure 3.57 EtherCAT web access Network adapter

-2. Connect with TwinCAT and check move to OP state.
For TwinCAT connection procedure, see 3.4.3 EtherCAT.

Select Slave > EtherCAT tab > Advanced Settings...

Mailbox > select “EoE”

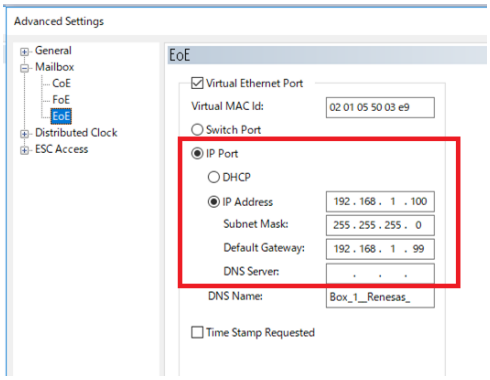

Figure 3.58 TwinCAT Network setting

-3. Execute "Restart TwinCAT (Config Mode)" to reconnect TwinCAT.

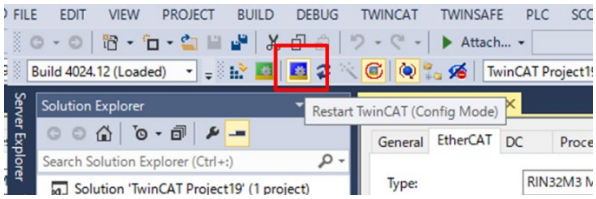

Figure 3.59 re-connect TwinCAT

-4. Accessing R-IN32M3 module IP address [192.168.1.100] in a web browser, the web server content prepared as a sample is loaded.

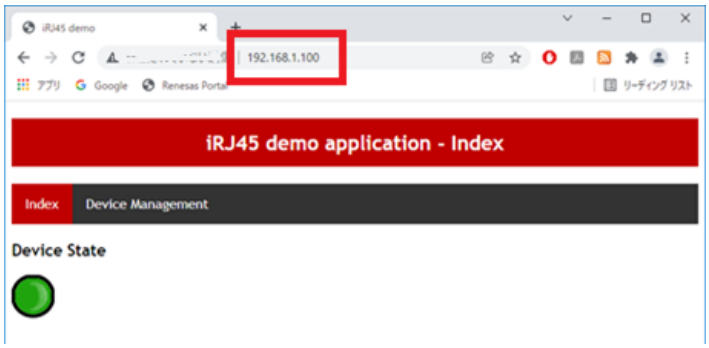

Figure 3.60 web browser access

3.5 Application Implement Guide

This chapter describes the steps to implement unique processing as a user application.

This sample software is equipped with uGOAL middleware and is structured based on its design philosophy. uGOAL provides appl_init(), appl_setup(), and appl_loop() functions for user application-specific processing, with appl_init() and appl_setup() executed in the initial phase of ugoal, followed by periodic appl_loop() in the subsequent loop phases.

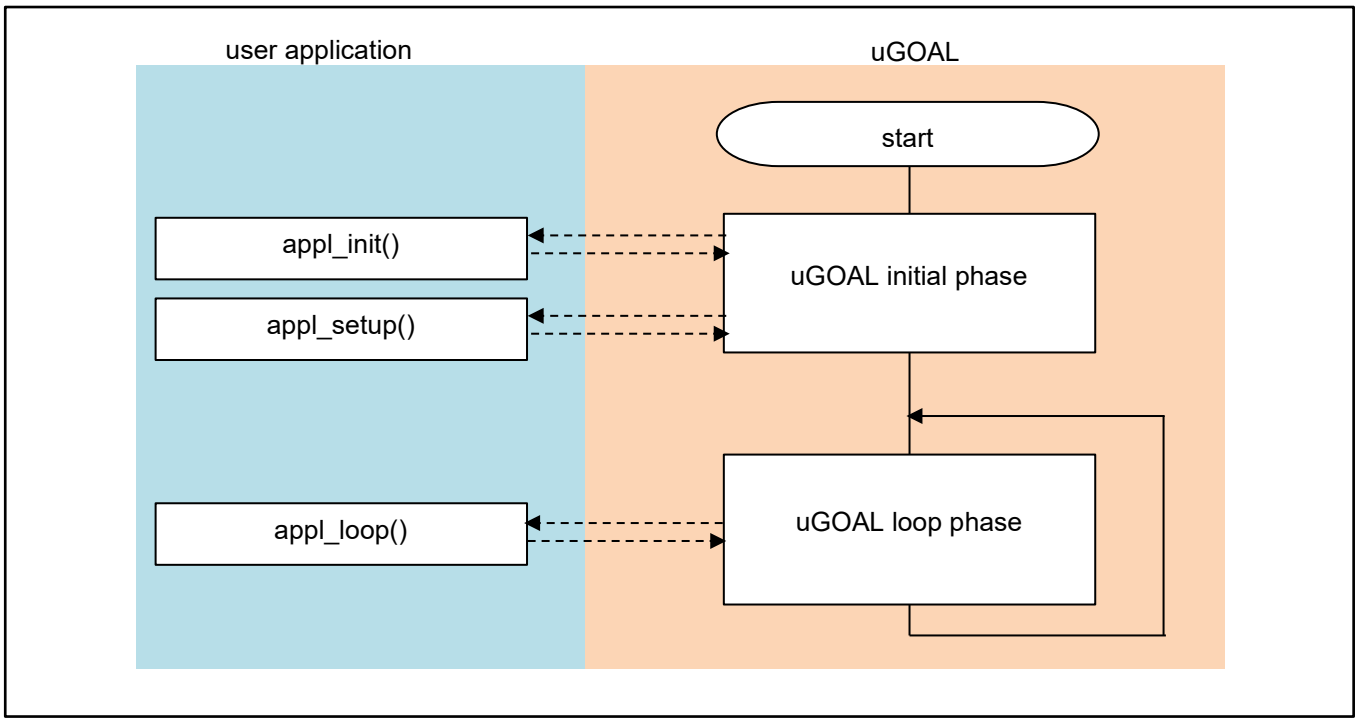

Figure 3-61 Overall flow of the program

The following is an overview of the unique processing of functions in the user application. These are also defined in goal_appl.c, which is the main source code of each sample.

Table 3-14 User applications and unique processing

Use Application	Unique Processing
appl_init()	Perform initialization steps before the uGOAL core part is initialized, such as initialization of each protocol stack, initialization of board-dependent hardware.
appl_setup()	Configure profile settings for each protocol stack, such as vendor ID settings. It also registers callback functions and receives data from the R-IN32M3 Module through each protocol.
appl_loop()	Perform normal operations, including loop control functions.

3.5.1 PROFINET

This chapter describes the implementation of the user application part in the I/O mirror response sample application by PROFINET. For more information about each API, see "R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)".

(1) appl_init

This function includes application-specific initialization steps before the uGOAL core module, etc. is initialized. To enable PROFINET in uGOAL, it is necessary to call goal_pnioInit first and register the uGOAL's PROFINET stack with uGOAL, therefore call the initialization routine for each module, including goal_pnioInit.

```
GOAL_STATUS_T appl_init(
    void
)
{
    GOAL_STATUS_T res;                          /**< result */

    /* initialize ccm RPC interface */
    res = appl_ccmRpcInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of ccm RPC failed");
    }

    res = goal_snmpInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of SNMP failed");
    }

    /* initialize PROFINET */
    res = goal_pnioInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of PROFINET failed");
    }

. . .

    return res;
}
```

① ①

① Initialize each module of GOAL. goal_pnioInit must be called from appl_init.

(2) appl_setup

This function defines static settings for protocols, such as creating instance of PROFINET.

An instance of PROFINET is created in goal_pnioNew and ready for use. Some settings, such as how much slot memory is reserved and which vendor ID to use, must be defined between goal_pnioInit and goal_pnioNew. These settings are set by the API group starting with goal_pnioCfg. After goal_pnioNew, all other APIs, such as creating slots and modules can be used.

```
GOAL_STATUS_T appl_setup(
    void
)
{
・・・

    res = goal_snmpNew(&pInstanceSnmp, APPL_SNMP_ID);                                    ①
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to create SNMP instance”);
        return res;
    }

    /* set SNMP instance id for new PNIO instance */
    res = goal_pnioCfgSnmpIdSet(APPL_SNMP_ID);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to set SNMP instance id”);
        return res;
    }

・・                                                                                      ②

    /* set identification of the slave (vendor name) */
    res = goal_pnioCfgVendorNameSet(APPL_PNIO_VENDOR_NAME);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to set vendor name”);
        return res;
    }

・・・

    /* create new PROFINET instance */
    res = goal_pnioNew(&pPnio, APPL_PNIO_ID, appl_pnioCb);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to create a new PROFINET instance”);                         ③
        return res;
    }

・・・
```

① Create an instance of SNMP.

② Define static settings in the protocol. In this sample, the vendor ID, device ID and else are set.

③ Create an instance of PRFINET and register the main callback (appl_pnioCb). The main callback function describes what to do depending on the state reported by the protocol stack. For information about the reported status, see "R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)".

```
    goal_logInfo(“Initializing device structure”);

    /* create subslots */
    res = goal_pnioSubslotNew(pPnio, APPL_API, APPL_SLOT_1, APPL_SLOT_1_SUB_1, GOAL_PNIO_FLG_AUTO_GEN);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to add subslot”);
        return res;
    }

. . .

    /* create submodules */
    res = goal_pnioSubmodNew(pPnio, APPL_MOD_1, APPL_MOD_1_SUB_1, GOAL_PNIO_MOD_TYPE_INPUT,
                             APPL_SIZE_1_SUB_1_IN, 0, GOAL_PNIO_FLG_AUTO_GEN);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to add submodule”);
        return res;
    }

. . .

    /* plug modules into slots */
    res = goal_pnioSubmodPlug(pPnio, APPL_API, APPL_SLOT_1, APPL_SLOT_1_SUB_1,
                              APPL_MOD_1, APPL_MOD_1_SUB_1);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to plug submodule”);
        return res;
    }

. . .

    /* PROFINET configuration succesful */
    goal_logInfo(“PROFINET ready”);

. . .

    return res;
}
```

④ ⑤

④ Create an instance of a sub-slot.

⑤ Create an instance of the sub-module and associate it with the sub-slot.

(3) appl_loop

Process the data after initialization of uGOAL.

```
void appl_loop(
    void
)
{
    GOAL_STATUS_T res;                          /* result */
    uint8_t iops;                               /* IO producer status */

. . .

    if ((GOAL_TRUE == flgAppReady) && (plat_getElapseTime(tsTout) >= APPL_TIMEOUT_TRIGGER_VAL)) {
        /* read data from output module */
        res = goal_pnioDataOutputGet(pPnio, APPL_API, APPL_SLOT_4, APPL_SLOT_4_SUB_1, dataDm,
                                     APPL_SIZE_13_SUB_1_OUT, &iops);
        if (GOAL_RES_ERR(res)) {
            return;
        }
        /* copy data to input module */
        res = goal_pnioDataInputSet(pPnio, APPL_API, APPL_SLOT_3, APPL_SLOT_3_SUB_1, dataDm,
                                    APPL_SIZE_3_SUB_1_IN , GOAL_PNIO_IOXS_GOOD);
        if (GOAL_RES_ERR(res)) {
            return;
        }

        /* read data from output module */
        res = goal_pnioDataOutputGet(pPnio, APPL_API, APPL_SLOT_2, APPL_SLOT_2_SUB_1, dataDm,
                                     APPL_SIZE_11_SUB_1_OUT, &iops);
        if (GOAL_RES_ERR(res)) {
            return;
        }
        /* copy data to input module */
        res = goal_pnioDataInputSet(pPnio, APPL_API, APPL_SLOT_1, APPL_SLOT_1_SUB_1, dataDm,
                                    APPL_SIZE_1_SUB_1_IN, GOAL_PNIO_IOXS_GOOD);
        if (GOAL_RES_ERR(res)) {
            return;
        }

        /* update base timestamp */
        tsTout = goal_timerTsGet();
    }

. . .

}
```

① Storing the reception data and setting the transmission data as a mirror response at regular intervals.

3.5.2 EtherNet/IP

This chapter describes the implementation of the user application part in the I/O mirror response sample application by EtherNet/IP. For more information about each API, see "R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)".

(1) appl_init

This function includes application-specific initialization steps before the uGOAL core module, etc. is initialized. To enable EtherNet/IP in uGOAL, it is necessary to call goal_eipInit and register the EtherNet/IP stack with uGOAL. Therefore, call the initialization routine for each module, including goal_eipInit.

```
GOAL_STATUS_T appl_init(
    void
)
{
    GOAL_STATUS_T res;                          /**< result */

    /* initialize rpc wrappers */
    res = appl_ccmRpcInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of ccm RPC failed");
    }

    /* initialize EtherNet/IP */
    res = goal_eipInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of EtherNet/IP failed");
    }

 . . .

    return res;
}
```

① ①

① Initialize each module of uGOAL. goal_eipInit must be called from appl_init.

(2) appl_setup

This function defines static settings for protocols, such as creating instance of EtherNet/IP.

Instance of EtherNet/IP is created in goal_eipNew and available for use. Some settings like vendor ID are necessary to be set between goal_eipInit and goal_eipNew. These settings are set by the API group starting with goal_eipCfg. After goal_eipNew, various types of data. are accessible.

```
GOAL_STATUS_T appl_setup(
    void
)
{
・・・

    /* for a real device the serial number should be unique per device */
    res = goal_eipCfgSerialNumSet(123456789);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to set Serial Number”);
        return res;
    }
・・・

    res = goal_eipNew(&pHdlEip, 0, main_eipCallback);
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to create eip instance %"FMT_x32, res);
        return res;
    }

    res = main_eipApplInit(pHdlEip);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to initialize assembly and attribute configuration”);
        return res;
    }

・・・

}
```

① ②

③

① Defines static settings in the protocol. In this sample, the vendor ID, product code, etc. are set.

② Create an instance of EtherNet/IP. Registering the main callback (main_eipCallback). The callback function describes operation depending on the state reported by the protocol stack. For information about the reported status, see “R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)”.

③ Set the created instance of EtherNet/IP to a CIP object.

(3) appl_loop

Process the data after initialization of uGOAL.

```
void appl_loop(
    void
)
{
    GOAL_STATUS_T res;                          /* result */

 . . .

    if ((GOAL_TRUE == flgAppReady) && (plat_getElapseTime(tsTout) >= APPL_TIMEOUT_TRIGGER_VAL)) {
        /* get output data */
        res = goal_eipAssemblyObjectRead(pHdlEip, GOAL_APP_ASM_ID_OUTPUT, &outputData[0],
                                         GOAL_APP_ASM_SIZE_OUTPUT);

        /* mirror output data to input data */
        if (GOAL_RES_OK(res)) {
            GOAL_MEMCPY(&inputData[0], &outputData[0], GOAL_APP_ASM_SIZE_INPUT);

            /* store input data */
            res = goal_eipAssemblyObjectWrite(pHdlEip, GOAL_APP_ASM_ID_INPUT, &inputData[0],
                                              GOAL_APP_ASM_SIZE_INPUT);
        }

        /* update base timestamp */
        tsTout = goal_timerTsGet();
    }
}
```

① Storing the reception data and setting the transmission data as a mirror response at regular intervals.

3.5.3 EtherCAT

This chapter describes the implementation of the user application part in the I/O mirror response sample application by EtherCAT. For more information about each API, see "R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)".

(1) appl_init

This function includes application-specific initialization steps before the uGOAL core module, etc. is initialized. To enable EtherCAT in uGOAL, it is necessary to call goal_ecatInit first and register the EtherCAT stack with uGOAL. Therefore, call the initialization routine for each module, including goal_ecatInit.

```
GOAL_STATUS_T appl_init(
    void
)
{
    GOAL_STATUS_T res;                          /**< result */

    /* initialize ccm RPC interface */
    res = appl_ccmRpcInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of ccm RPC failed");
    }

    /* initialize EtherCAT */
    res = goal_ecatInit();
    if (GOAL_RES_ERR(res)) {
        goal_logErr("Initialization of EtherCAT failed");
    }

    return res;
}
```

① Initialize each module of uGOAL. goal_ecatInit must be called from appl_init.

(2) appl_setup

This function defines static settings for protocols, such as creating instance of EtherCAT.

An instance of EtherCAT is created in goal_ecatNew and ready for use. Also, if necessary, configure EtherCAT protocol before creating instance set by the API group starting with goal_ecatCfg. After creating instance, generate the required object dictionary and set the initial values.

```
GOAL_STATUS_T appl_setup(
    void
)
{

. . .

    /* enable CoE emergency */
    res = goal_ecatCfgEmergencyOn(GOAL_TRUE);
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to enable CoE Emergency support");
        return res;
    }

. . .

#if APPL_ECAT_SII_INIT == 1
    goal_logInfo("initializing EtherCAT SSI data");

    res = appl_ccmCfgSsiVendorId(
        &__03_ecat_slave_eeprom_bin[0],         /* data buffer */
        __03_ecat_slave_eeprom_bin_len,         /* data buffer length */
        APPL_ECAT_VENDOR_ID);
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to configure EEPROM ssi vendor id");
    }

. . .

    /* configure SII in EEPROM before creating the EtherCAT instance */
    res = appl_ccmEcatSsiUpdate(
        &__03_ecat_slave_eeprom_bin[0],         /* data buffer */
        __03_ecat_slave_eeprom_bin_len,         /* data buffer length */
        GOAL_FALSE);                            /* always overwrite ssi data */
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to configure EEPROM ssi data");
    }
#endif
```

① ②

① Setting EtherCAT protocol. goal_ecatNew must be performed before an instance can be created in the application.

② Initialization of SII. (Disabled by default)

```
    res = goal_ecatNew(&pHdlEcat, GOAL_ECAT_INSTANCE_DEFAULT, appl_ecatCallback);
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to create a new EtherCAT instance");
        return res;
    }

    res = appl_ecatCreateObjects(pHdlEcat);
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to initialize object dictionary");
        return res;
    }

    /* set settings for ccm firmware update via FoE */
    res = appl_ccmFoeUpdateSettings(
         "ccm.efw",                          /* filename beginning */
        0,                                  /* 0 -> match all characters */
        0,                                  /* password */
        GOAL_TRUE);                         /* only update in ESM state bootstrap */
    if (GOAL_RES_ERR(res)) {
        goal_logErr("failed to configure FoE firmware update of CC");
        return res;
    }

 . . .

#if GOAL_CONFIG_MEDIA_MA_EVENT == 1
    /* open GPIO ma */
    if (GOAL_RES_OK(res)) {
        res = goal_maEventOpen(GOAL_ID_DEFAULT, &pHdlMaEvent, GOAL_TRUE, appl_gpioDcEvent);
        if (GOAL_RES_OK(res)) {
            goal_logInfo("event generation enabled");
        }
    }
#endif

 . . .

    return res;
}
```

③ ④ ⑤ ⑥

③ Create an instance of EtherCAT and register main callback (main_ecatCallback). The callback function describes operation depending on the state reported by the protocol stack. For information about the reported status, see “R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)”.

④ Generates each object dictionary (OD). OD is added by goal_ecatdynOdObjAdd or else, and end OD generation by goal_ecatdynOdFinish in the end.

⑤ Set up firmware update via FoE.

⑥ Initialize the module for setting the EtherCAT Explicit Device ID. An external Pmod SWT is required to set the ID. For details, please refer to Chapter 2.3.

(3) appl_loop

Process the data after initialization of uGOAL.

```
void appl_loop(
    void
)
{

・・・

    if ((GOAL_TRUE == flgAppReady) && (plat_getElapseTime(tsTout) >= APPL_TIMEOUT_TRIGGER_VAL)) {
        /* map process data */
        read_state8_input1 = write_state8_output1;
        read_state8_input2 = write_state8_output2;

        read_analog16_input1 = write_analog16_output1;
        read_analog16_input2 = write_analog16_output2;

        /* process cyclic process data */
        appl_obj_200d = cntDC0Event;
        appl_obj_200e = cntDC1Event;

        /* update base timestamp */
        tsTout = goal_timerTsGet();
    }

・・・

}
```

① ①

① Storing the reception data and setting the transmission data as a mirror response at regular intervals.

4. Appendix

4.1 uGOAL API

The host microcomputer communicates with the R-IN32M3 Module via an API function to control the R-IN32M3 Module provided by uGOAL. The APIs are categorized by protocol, and for more information, see “R-IN32M3 Module (RY9012A0) User's Manual Software (R17US0002ED****)”.

4.2 Logging

The log message can be outputted for debug in this sample software. There is one way to see the log message.

1. Output log messages to PC terminal software (TeraTerm, etc) via serial communication

This feature is enabled by changing the following compile macros. But this feature is disabled in each sample application by default.

Table 4-1 Log message output way and compile macro

Output way	Compile macro	Default value
1	CONFIG_UGOAL_LOGGING	0

The output method is described below.

4.2.1 Using TeraTerm

This sample software log messages are transferred to the UART communication line on Arduino IF of this board via UART driver implemented in this sample software. By connecting PC and R-IN32M3 Module-based adapter board that is stacked to this board using USB-UART converter cable*, this sample software log messages can be seen using TeraTerm. Note that R-IN32M3 Module log messages cannot be seen using TeraTerm.

The step is shown below.

(*) A USB-UART converter cable is required separately, e.g., “TTL-232R-RPI”.

(1) Connect R-IN32M3 Module-based adapter board and PC using USB-UART converter cable.

- Connect USB-UART converter cable TX line to Pin1 in J10 on R-IN32M3 Module-based adapter board.
- Connect USB-UART converter cable RX line to Pin2 in J10 on R-IN32M3 Module-based adapter board.
- Connect USB-UART converter cable GND line to Pin7 in J6 on R-IN32M3 Module-based adapter board.

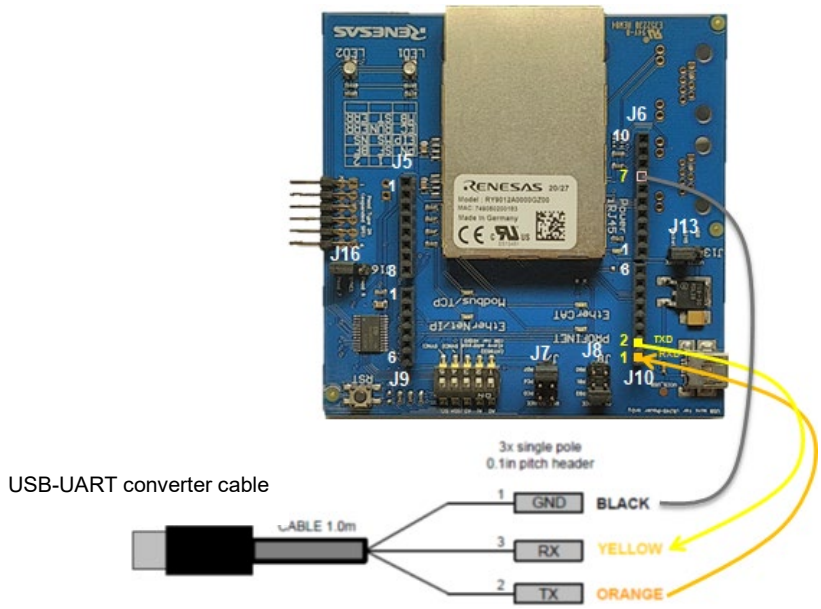

Figure 4-1 Connection of this board and USB-UART converter cable

(2) Launch the TeraTerm on PC and set the serial.

Speed	115200
Data	8 bits
Parity	none
Stop bit	1 bit
Flow control	none

(3) Change the value of “CONFIG_UGOAL_LOGGING” macro from 0 (default) to 1 in goal_config.h that is in the folder for each sample application.

Ex)

File: \appl\ugoal\01_pnio\goal_config.h

```
#define CONFIG_UGOAL_LOGGING (0)
```

(4) Build the project and run the sample application, referring to Chapters 3.2.5 to 3.2.6.

(5) Connect to this sample application and start cyclic communication by operating Management tool referring to Chapter 3.3.

As a result, the following log messages will be displayed on TeraTerm.

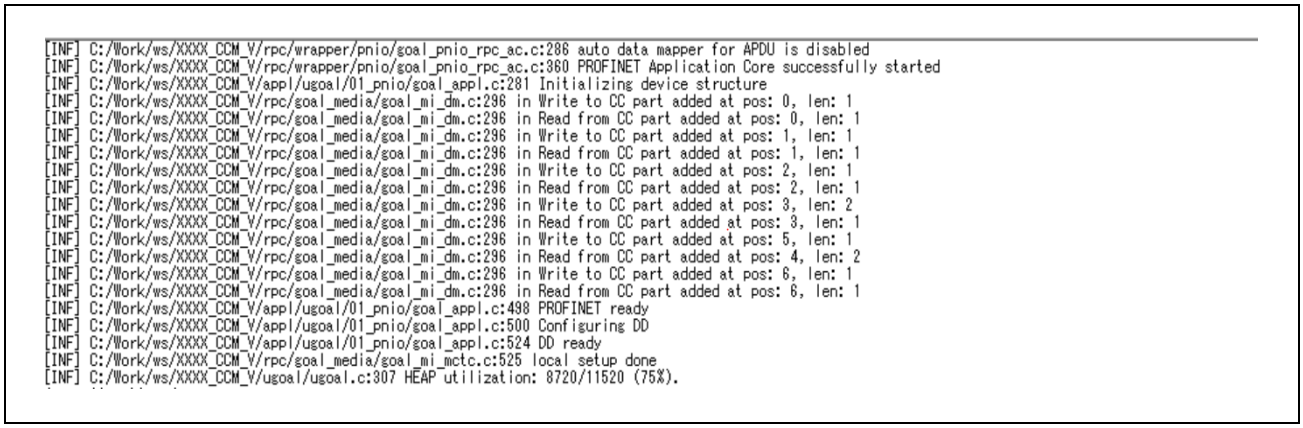

Figure 4-2 Log message on TeraTerm

4.3 IP Address Setting

This chapter describes how to set the IP address of R-IN32M3 Module.

The IP address of the R-IN32M3 Module is set according to the GOAL_ID_NET (12) configuration stored in the internal nonvolatile memory at startup. It is also possible to set the IP address from the host CPU by calling *goal_maNetIpSet()*.
In the default setting in the sample applications of "01_pnio", "02_eip", "04_pnio_large" and "05_eip_large", the IP address is set by the configurations stored inside (Configured IP). Defining the macro of "GOAL_CONFIG_STATIC_IP" in the program enables to set arbitrary IP address (Static IP).

Table 4-2 IP Configuration (GOAL_ID_NET)

Variable Name	Variable ID	Type	Max. Size	Description
IP	0	GOAL_CM_IPV4	4	IP address of first interface
NETMASK	1	GOAL_CM_IPV4	4	NETMASK of first interface
GW	2	GOAL_CM_IPV4	4	GATEWAY of first interface
VALID	3	GOAL_CM_UINT8	1	Validity of IP address: 0, Stored IP address is not valid, interface settings originate from network stack of system 1, Stored IP address is valid, will be applied to interface at start of device
DHCP_ENABLED	4	GOAL_CM_UINT8	1	DHCP enable: 0, DHCP disabled 1, DHCP enabled

Please note that VALID needs to be set "1" to activate IP address configurations stored in nonvolatile memory. By executing the "*goal_maNetIpSet ()"* API, configurations of IP, NETMASK, and GW are stored in the nonvolatile memory, and whether to save the VALID setting can be specified by the last argument, flgTemp. (GOAL_FALSE: Update VALID settings, GOAL_TRUE: not updated)

```
1.  GOAL_STATUS_T goal_maNetIpSet(
2.      GOAL_MA_NET_T *pNetHdl,          /**< pointer to store NET handler */
3.      uint32_t addrIp,                 /**< IP address */
4.      uint32_t addrMask,               /**< subnet mask */
5.      uint32_t addrGw,                  /**< gateway */
6.      GOAL_BOOL_T flgTemp              /**< temporary IP config flag */
7.  );
```

Also, DHCP mode is enabled by setting the "DHCP_ENABLED" in GOAL_ID_NET (12) to 1 or call the API of *goal_eipCfgDhcpOn()* for EtherNet/IP. In the sample software of 02_eip, DHCP is enabled by defining a "GOAL_CONFIG_ENABLE_DHCP" macro as "1" in the program.

Table 4-3 provides a list of how to set up an IP address.

Table 4-3 IP address setting list

Methods	Descriptions
Configured IP	- Use the value held in the non-volatile memory of R-IN32M3 module - The value can be changed using the Management Tool. For more information, see "R-IN32M3 Module (RY9012A0) Management Tool Instruction Guide (R30AN0390EJ****)". - This method is used as the default setting for “01_pnio”, “02_eip”, “04_pnio_large” and “05_eip_large” sample application of this sample.
Static IP	- Mainly used for evaluation. - The changed value is hold in the non-volatile memory of R-IN32M3 Module. - The value can be changed with “01_pnio”, “02_eip”, “04_pnio_large” and “05_eip_large” sample application of this sample. By defining “GOAL_CONFIG_STATIC_IP” macro in the program with 1, any IP address can be set.
DHCP	- It is possible to change enable / disable by using Management Tool. - It is also possible to change using "02_eip" and “05_eip_large” sample application of this sample software, the default value is disable. By defining “GOAL_CONFIG_ENABLE_DHCP” macro in the program with 1, DHCP become enable. - If DHCP is enabled and there is no DHCP server on the network, the value held in the non-volatile memory of R-IN32M3 Module will be used.

Revision History

Rev.	Date	Description	
		Chapter	Summary
1.00	Oct/15/2021	-	First Edition
1.01	Jan/11/2022	3.4	Add Remote I/O sample application
		3.4.4	Add Modbus TCP sample application
1.02	Aug/ 5/2022	3.4.6	Add web saver function sample
1.03	May/31/2023	3.4	Review of description with sample program update
1.04	Dec/15/2023	3.4.2	Added explanation about Mirror (RPC)
		3.4.1	Replaced Figure 3-24 and modified the explanation
		1.2.1	Updated Table 1-1
		-	Updated list of Related documents
		3.4.6	Replaced Figure 3.56
1.05	May/31/2024	1.2.1	Updated Table 1-1
		3.4.1	Updated GSD file name
		3.3.3(2)	Figure 3.3
		3.3.4	Figure 3.9

Trademark

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.