R-IN32M3 Module (RY9012A0)

RL78/G14 Sample Application (uGOAL Edition)

R30AN0400EJ0105
Rev.1.05
May.31.2024

Introduction

This document describes sample application software for RL78/G14 to perform industrial Ethernet communication as host CPU of the R-IN32M3 Module (RY9012A0).

Target Device

RL78/G14

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, RL78/G14 Fast Prototype Board 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	RL78/G14 Fast Prototyping Board Quick Start Guide	R20UT4571EJ****
User’s Manual	RL78/G14 Fast Prototyping Board User’s Manual	R20UT4573EJ****
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 RL78/G14 Fast Prototyping Board.

This sample software can communicate with major industrial Ethernet protocols such as PROFINET, EtherNet/IP, and EtherCAT by running on RL78/G14 Fast Prototyping Board, which is evaluation board of RL78/G14 MCU, connected with R-IN32M3 Module-based adapter board (YCONNECT-IT-I-RJ4501) via Arduino^{TM} connector.

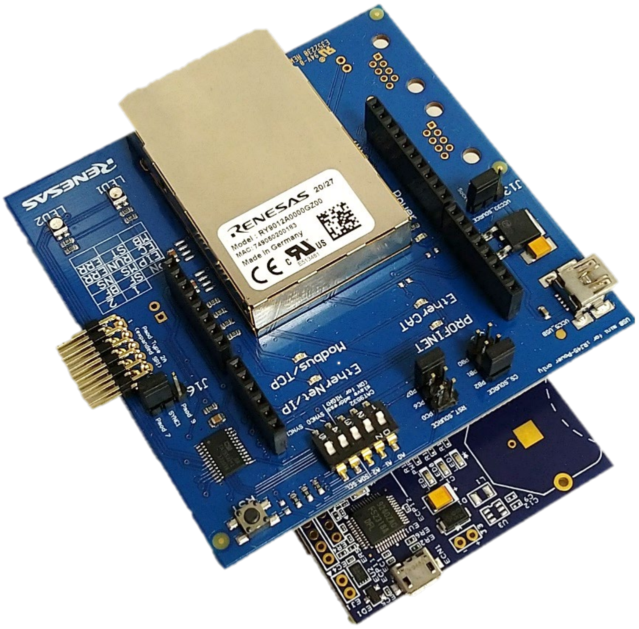


Figure 1-1 R-IN32M3 Module + RL78/G14 Fast Prototyping Board

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	e² studio 2024-04 Windows| Renesas	
RL family GNU Toolchain	GCC for Renesas RL78	V4.9.2.202201	-	Included with e2studio
Management Tool, simple software PLC	ICE	V1.5.1	-	port industrial automation GmbH Including with Sample package
Software PLC of EtherCAT	TwinCAT	V3.1	https://www.beckhoff.com/	Beckhoff Automation GmbH

1.2.2 Hardware environment

The operation of this sample software is verified with a hardware environment connected to the RL78/G14 MCU Group Evaluation Kit (RL78/G14 Fast Prototyping Board) with an adapter board equipped with an R-IN32M3 module (YCONNECT-IT-I-RJ4501).

If you use RL78/G14 Fast Prototyping Board, you do not need to prepare the emulator separately for the execution of this sample software because the emulator circuit equivalent to the E2 emulator Lite is built into the board.

Also, this sample software includes multiple applications. Multi-protocol application and Remote I/O application can be executed by connecting the Digilent Pmod^{TM} 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
RL78/G14 Fast Prototyping Board	RTK5RLG140C00000BJ	Renesas Electronics Corporation	RL78/G14 Fast Prototyping 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.

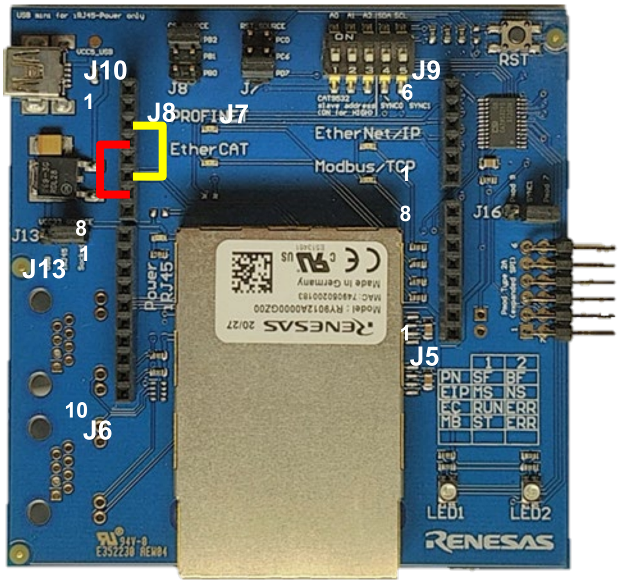


Figure 2-1 Adaptor board with R-IN32M3 module

Plug the male Arduino connector on the back of the adapter board with R-IN32M3 module into the socket of the RL78/G14 Fast Prototyping Board.

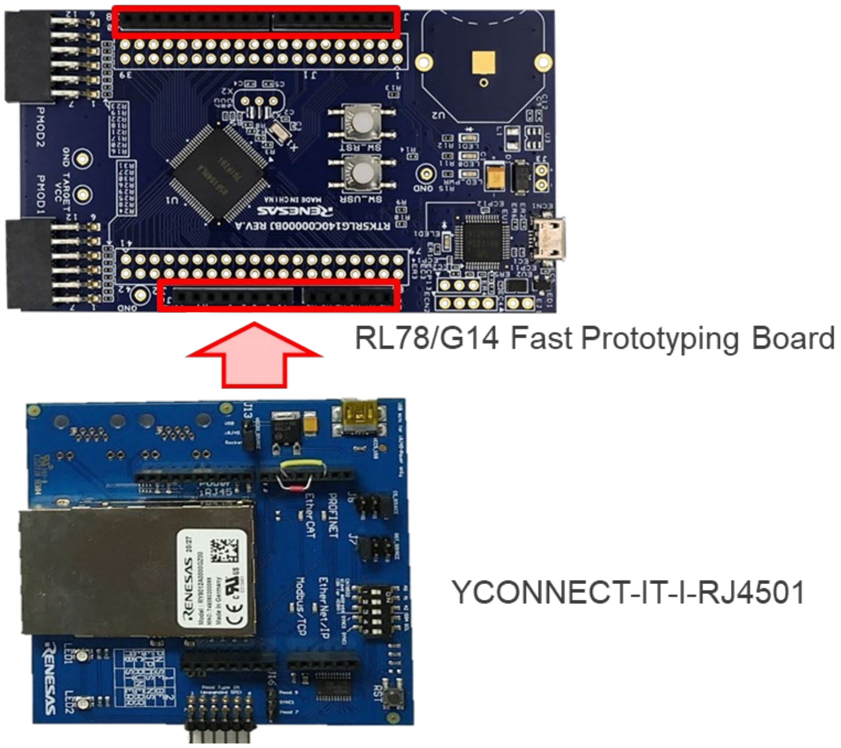


Figure 2-2 Arduino^{TM} Connection

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 upper stage (1-6pin) of the Pmod2 connector on the RL78/G14 Fast Prototyping Board.

It is also used as the input switch of sample application software for Remote I/O, and the EtherCAT Explicit Device ID.

Table 2-1 Connection of Pmod2 and Pmod SWT for Multi-protocol selector

PMOD2 Upper	RL78/G14 Fast Prototyping Board	Pmod SWT
1	P16	Selector-ID1
2	P13	Selector-ID2
3	P14	Selector-ID3
4	P15	Selector-ID4
5	GND	GND
6	+3.3V	VCC

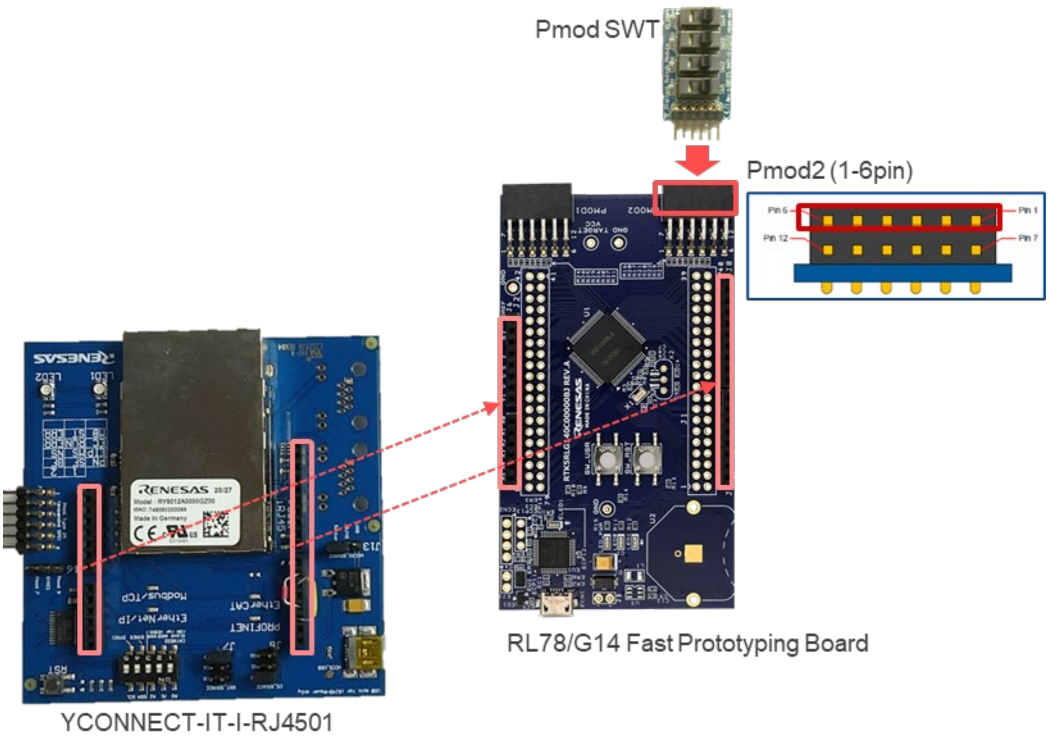


Figure 2-3 multi-protocol selector Connection

2.3 Remote I/O application

Sample application software for Remote I/O is confirmed in a configuration in which the switch input (Pmod SWT) is connected to the upper of the Pmod2 connector (1-6pin) and the LED output (Pmod LED) is connected to the lower (7-12pin) of the Pmod1 connector. (Figure 2-4)

It is also used as the selector input of sample application for multi-protocol, and the EtherCAT Explicit Device ID.

Table 2-2 Connection of Pmod1 and Pmod SWT

PMOD2 Upper	RL78/G14 Fast Prototyping Board	Pmod SWT
1	P16	SW1
2	P13	SW2
3	P14	SW3
4	P15	SW4
5	GND	GND
6	+3.3V	VCC

Table 2-3 Connection of Pmod2 and Pmod LED

PMOD1 Lower	RL78/G14 Fast Prototyping Board	Pmod LED
7	P140	LED1
8	P130	LED2
9	P147	LED3
10	P146	LED4
11	GND	GND
12	+3.3V	VCC

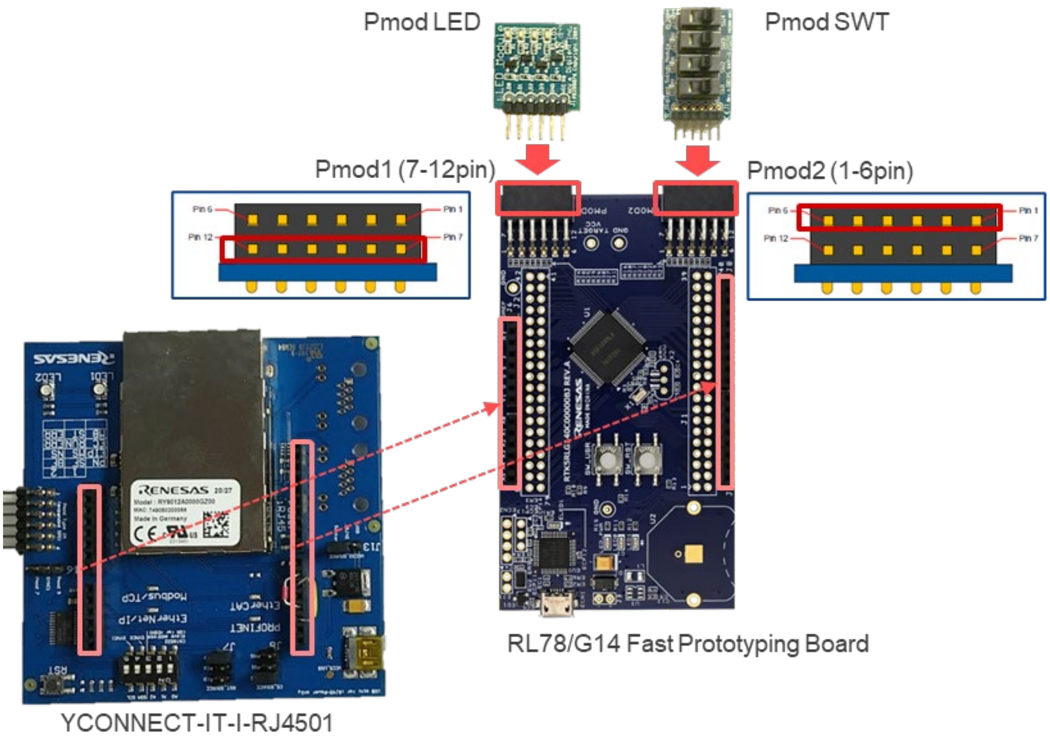


Figure 2-4 Remote I/O Connection

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 RL78/G14 Fast Prototyping Board.

It is also used as the selector input of sample application for multi-protocol, and the switch input of sample application software for Remote I/O.

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

PMOD2 Upper	RL78/G14 Fast Prototyping Board	Pmod SWT
1	P16	ECAT-ID1
2	P13	ECAT-ID2
3	P14	ECAT-ID3
4	P15	ECAT-ID4
5	GND	GND
6	+3.3V	VCC

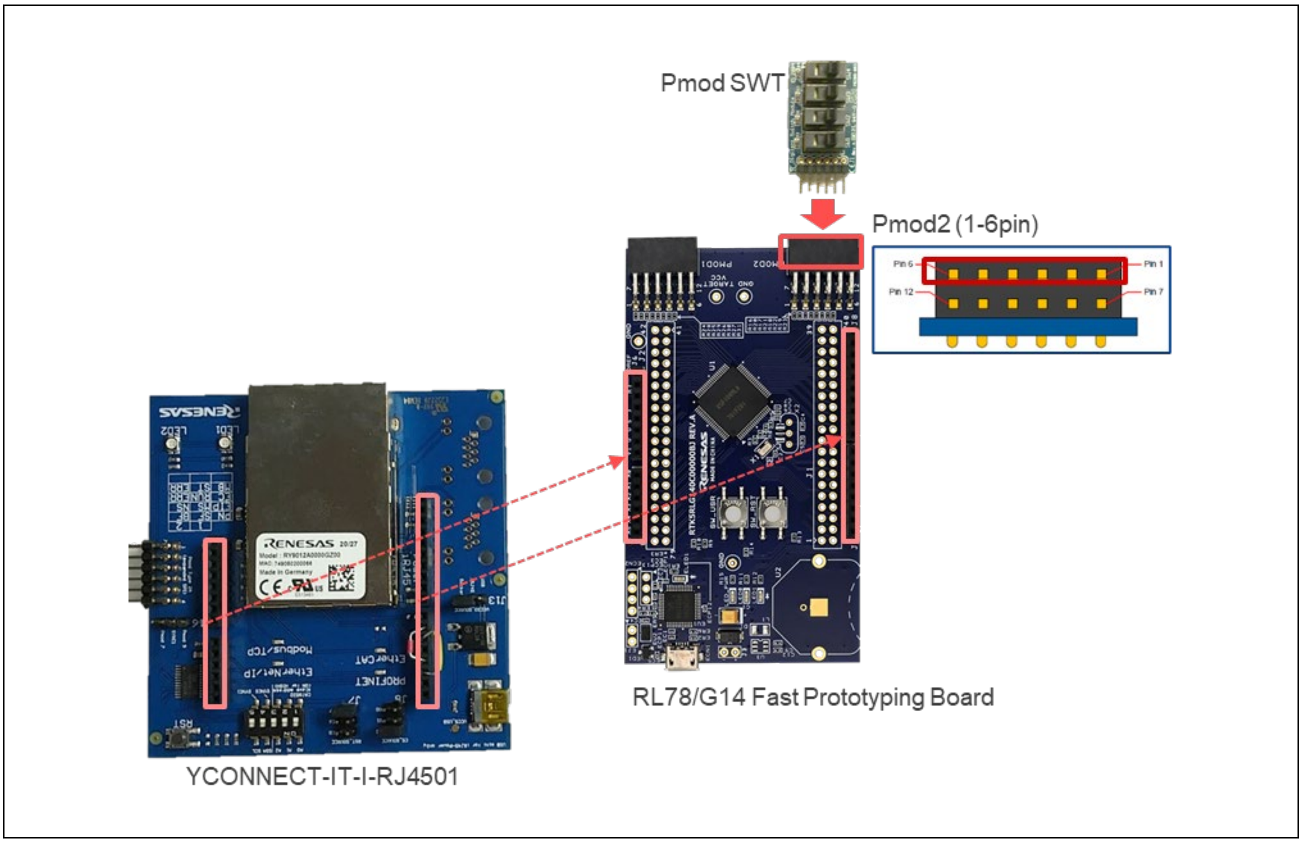


Figure 2-5 EtherCAT Explicit Device ID Connection

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.

```
RL78_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
 │
 ├──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	More detail
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

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.

3.3 Set up of development environment

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

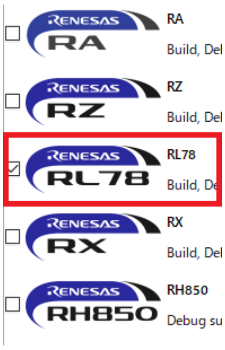
3.3.1 Install

(1) IDE e2studio and GCC for Renesas RL78

Download e2studio in the following web site and install it on your PC. GCC for Renesas RL78, the GNU Toolchain for the RL family, will be installed along with e2studio.

There are four points to note regarding the installation.

1. Do not forget to check for "RL78" in [Device Family] screen during the installation. (Multiple selections can be made together with others)



□ RENESAS RA — RA Build, Del

□ RENESAS RZ — RZ Build, Del

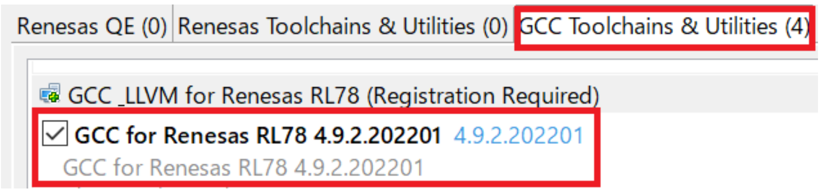
☑ RENESAS RL78 — RL78 Build, De

□ RENESAS RX — RX Build, Del

□ RENESAS RH850 — RH850 Debug su

Figure 3-1 Select device family

2. Select “GCC Toolchains & Utilities” tag in [Additional Software] screen during the installation. After that, check "GCC for Renesas RL78 4.9.2.202201" and install GCC.



Renesas QE (0) | Renesas Toolchains & Utilities (0) | GCC Toolchains & Utilities (4)

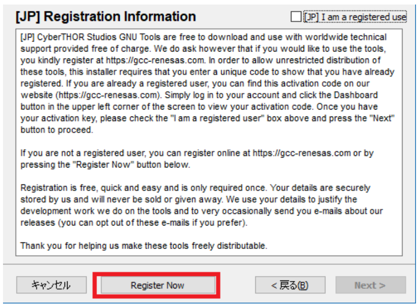
GCC _LLVM for Renesas RL78 (Registration Required)

☑ GCC for Renesas RL78 4.9.2.202201 4.9.2.202201
GCC for Renesas RL78 4.9.2.202201

Figure 3-2 Select additional software

3. During the installation of the GCC compiler, a CyberTHOR Studios Limited user registration may be required.

If you do not have an account, please register from "Register Now". Alternatively, you can also register from the [Open Source Tools for Renesas (llvm-gcc-renesas.com)](https://llvm-gcc-renesas.com).



[JP] Registration Information

□ [JP] I am a registered use

[JP] CyberTHOR Studios GNU Tools are free to download and use with worldwide technical support provided free of charge. We do ask however that if you would like to use the tools, you kindly register at https://gcc-renesas.com. In order to allow unrestricted distribution of these tools, this installer requires that you enter a unique code to show that you have already registered. If you are already a registered user, you can find this activation code on our website (https://gcc-renesas.com). Simply log in to your account and click the Dashboard button in the upper left corner of the screen to view your activation code. Once you have your activation key, please check the "I am a registered user" box above and press the "Next" button to proceed.

If you are not a registered user, you can register online at https://gcc-renesas.com or by pressing the "Register Now" button below.

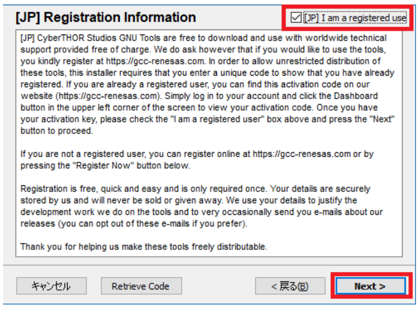
Registration is free, quick and easy and is only required once. Your details are securely stored by us and will never be sold or given away. We use your details to justify the development work we do on the tools and to very occasionally send you e-mails about our releases (you can opt out of these e-mails if you prefer).

Thank you for helping us make these tools freely distributable.

Figure 3-3 Install GCC compiler (no account)

After registering or if you have an account, check “registered use” and select [Next>]. In the next pop-up, enter the registered e-mail and Authentication Code to proceed with the GCC installation.

4. Make sure that [Change PATH environment variable automatically] is checked and proceed with the installation.



[JP] Registration Information

☑ [JP] I am a registered use

[JP] CyberTHOR Studios GNU Tools are free to download and use with worldwide technical support provided free of charge. We do ask however that if you would like to use the tools, you kindly register at https://gcc-renesas.com. In order to allow unrestricted distribution of these tools, this installer requires that you enter a unique code to show that you have already registered. If you are already a registered user, you can find this activation code on our website (https://gcc-renesas.com). Simply log in to your account and click the Dashboard button in the upper left corner of the screen to view your activation code. Once you have your activation key, please check the "I am a registered user" box above and press the "Next" button to proceed.

If you are not a registered user, you can register online at https://gcc-renesas.com or by pressing the "Register Now" button below.

Registration is free, quick and easy and is only required once. Your details are securely stored by us and will never be sold or given away. We use your details to justify the development work we do on the tools and to very occasionally send you e-mails about our releases (you can opt out of these e-mails if you prefer).

Thank you for helping us make these tools freely distributable.

Figure 3-4 Installation of GCC compiler (account possession)

3.3.2 Connection

(1) No additional connection

After stacking the Adapter board with R-IN32M3 Module on RL78/G14 Fast Prototyping 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 RL78/G14 Fast Prototyping Board.

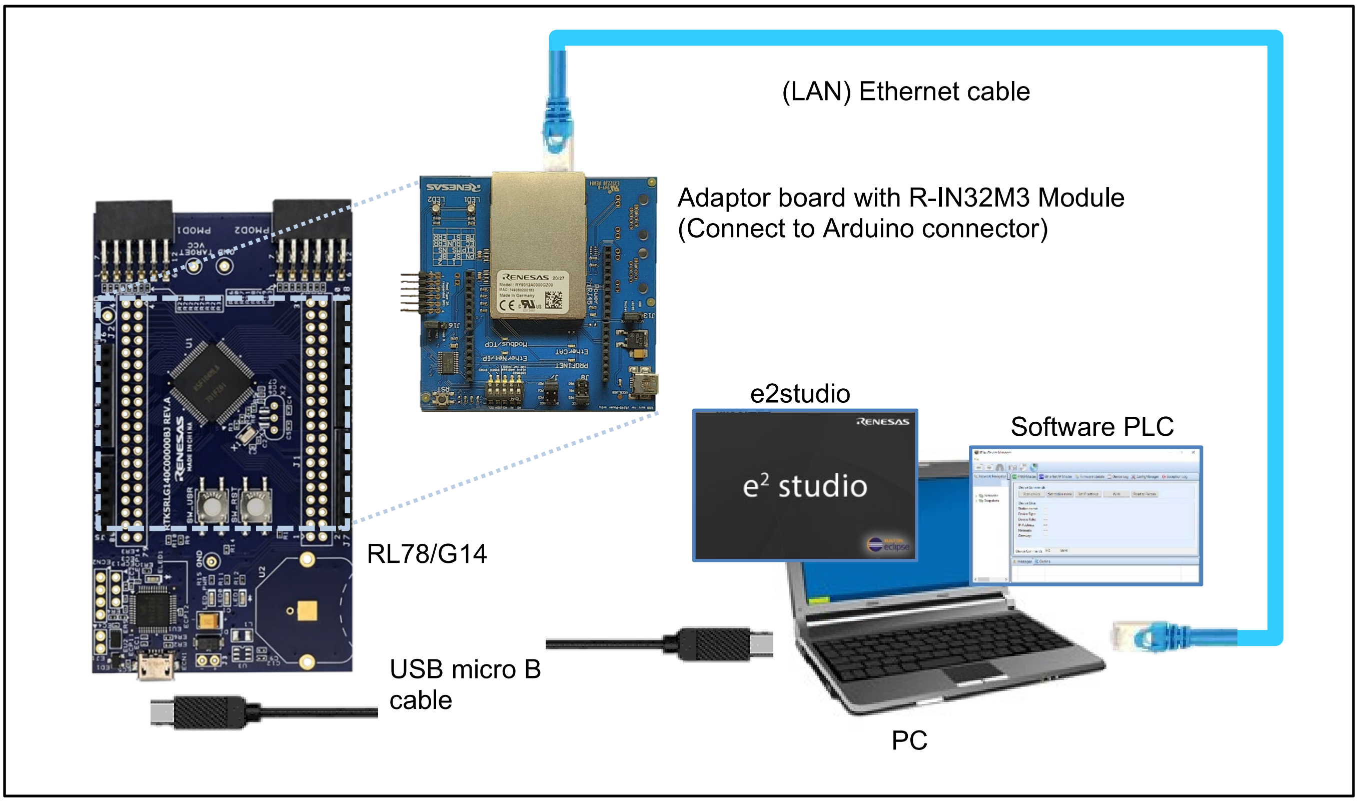


Figure 3-5 Connection configuration

(2) Additional Pmod SWT and Pmod LED connection

Connect Pmod SWT (refer to Table 1-2) to Pmod2 terminal Upper (1-6 pin) and Pmod LED (refer to Table 1-2) to Pmod1 terminal Lower (7-12 pin) on RL78/G14 Fast Prototyping Board (For details, please refer to Chapter 2.3). Power is supplied to those boards by connecting a USB micro B cable to RL78/G14 Fast Prototyping Board.

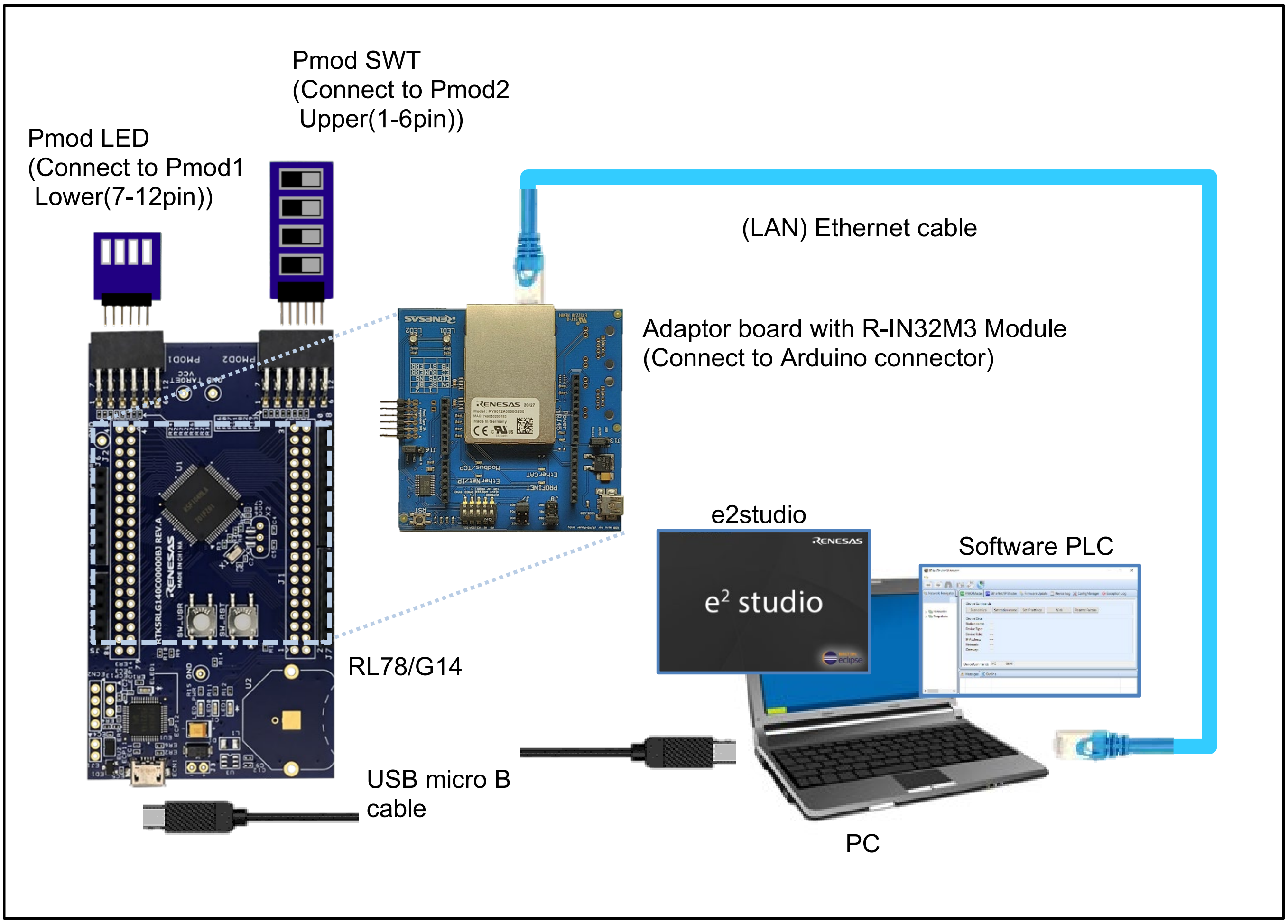


Figure 3-6 Connection configuration (connection with Pmod SWT and Pmod LED)

3.3.3 Import project

(1) Unzip package

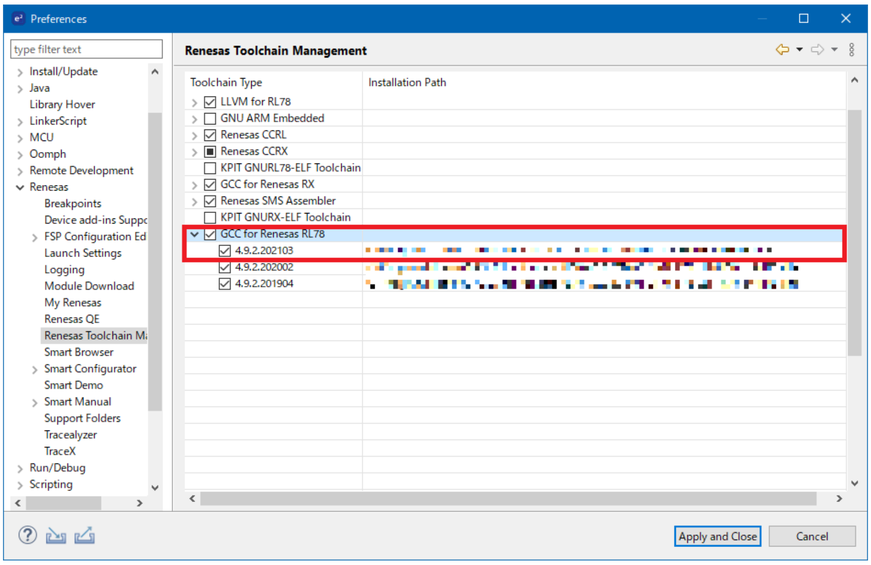
First, unzip the archived package of this sample software (RL78_uCCM_V***.zip) and store it in arbitrary folder. Because e2studio cannot recognize project properly if file path is too long in the folder hierarchy, place it in shorter path. Also, do not use multi-byte character, such as Japanese, in the folder path.

(2) Execute e2studio

Execute "e2studio.exe" to start e2studio in the following folder (default case) installed:

\Renesas\e2_studio\eclipse\e2studio.exe

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 "GCC for Renesas RL78".



Renesas Toolchain Management

Toolchain Type	Installation Path
☑ LLVM for RL78	
☐ GNU ARM Embedded	
☑ Renesas CCRL	
■ Renesas CCRX	
☐ KPIT GNURL78-ELF Toolchain	
☑ GCC for Renesas RX	
☑ Renesas SMS Assembler	
☐ KPIT GNURX-ELF Toolchain	
☑ GCC for Renesas RL78	
☑ 4.9.2.202103	[redacted]
☑ 4.9.2.202002	[redacted]
☑ 4.9.2.201904	[redacted]

Figure 3-7 Renesas Toolchain Management

(3) Import project

Import the sample project into e2studio from the following steps:

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

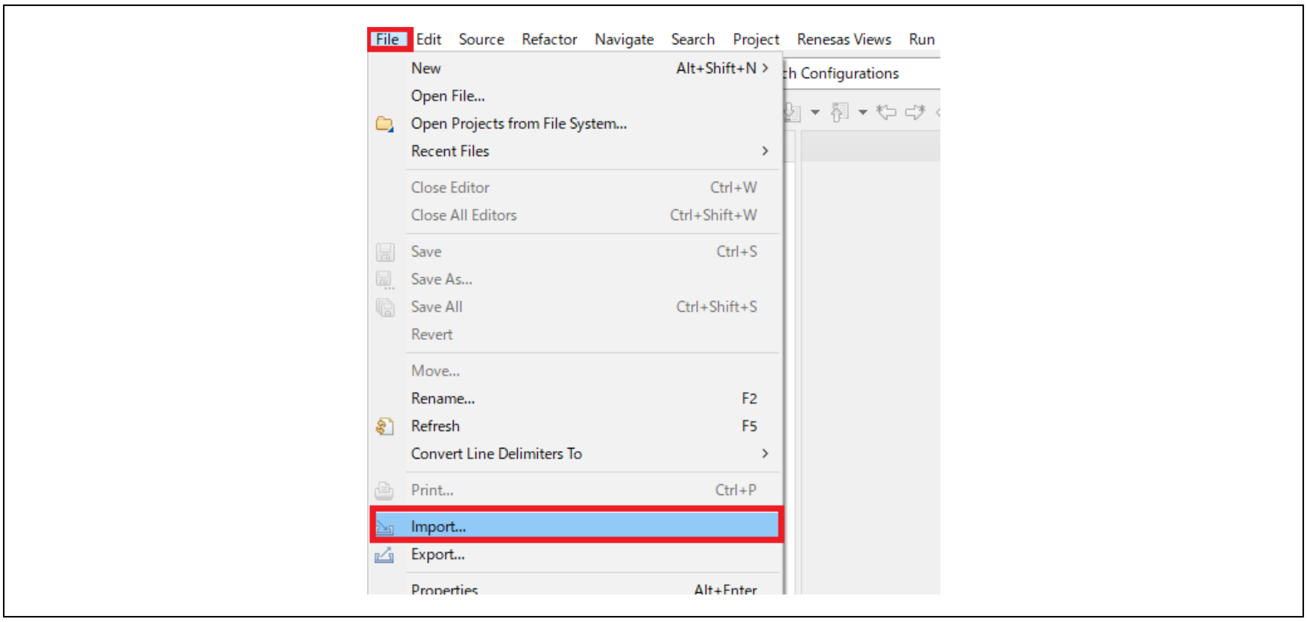
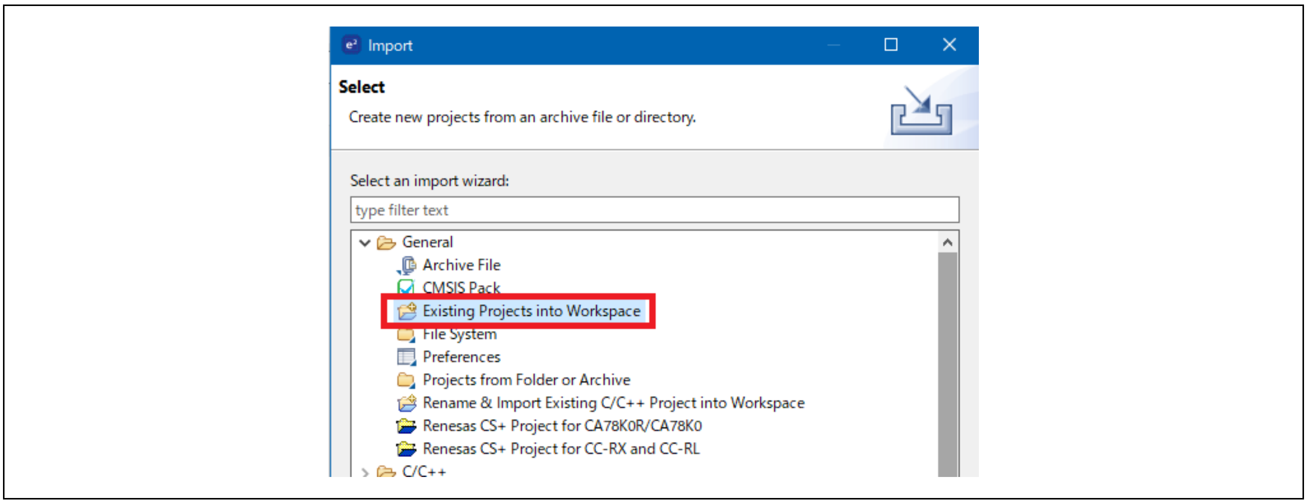


Figure 3-8 Import

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


Import

Select

Create new projects from an archive file or directory.

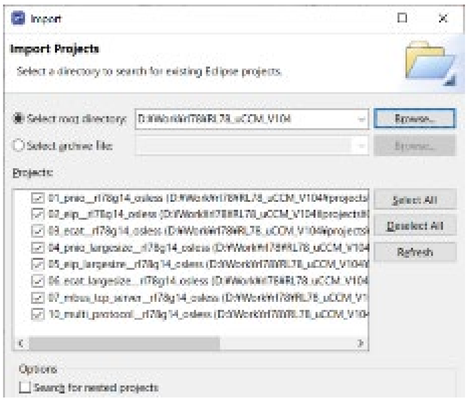
Select an import wizard:

type filter text

General
Archive File
CMSIS Pack
Existing Projects into Workspace
File System
Preferences
Projects from Folder or Archive
Rename & Import Existing C/C++ Project into Workspace
Renesas CS+ Project for CA78K0R/CA78K0
Renesas CS+ Project for CC-RX and CC-RL
C/C++

Figure 3-9 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 "RL78_uCCM_V***" stored in arbitrary folder at 3.3.3(1) and select [OK].



● Select root directory: D:¥Work¥rl78¥RL78_uCCM_V104

○ Select archive file:

☑ 01_pnio__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104¥projects

☑ 02_eip__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104¥projects¥

☑ 03_ecat__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104¥projects¥

☑ 04_pnio_largesize__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104

☑ 05_eip_largesize__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104¥

☑ 06_ecat_largesize__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104

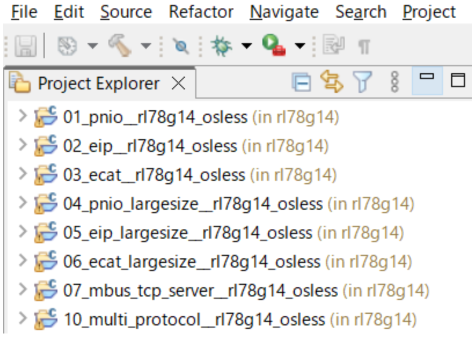
☑ 07_mbus_tcp_server__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V1

☑ 10_multi_protocol__rl78g14_osless (D:¥Work¥rl78¥RL78_uCCM_V104

☐ Search for nested projects

Figure 3-10 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.



01_pnio__rl78g14_osless (in rl78g14)

02_eip__rl78g14_osless (in rl78g14)

03_ecat__rl78g14_osless (in rl78g14)

04_pnio_largesize__rl78g14_osless (in rl78g14)

05_eip_largesize__rl78g14_osless (in rl78g14)

06_ecat_largesize__rl78g14_osless (in rl78g14)

07_mbus_tcp_server__rl78g14_osless (in rl78g14)

10_multi_protocol__rl78g14_osless (in rl78g14)

Figure 3-11 Imported projects

3.3.4 Build project

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

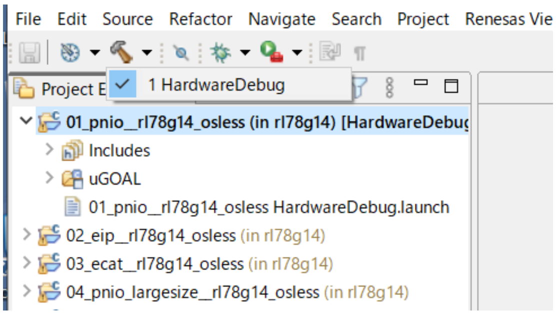
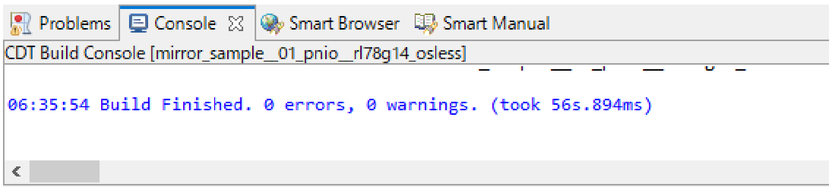

File Edit Source Refactor Navigate Search Project Renesas Vie
Project E
1 HardwareDebug
01_pnio__rl78g14_osless (in rl78g14) [HardwareDebug
Includes
uGOAL
01_pnio__rl78g14_osless HardwareDebug.launch
02_eip__rl78g14_osless (in rl78g14)
03_ecat__rl78g14_osless (in rl78g14)
04_pnio_largesize__rl78g14_osless (in rl78g14)

Figure 3-12 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.



CDT Build Console [mirror_sample__01_pnio__rl78g14_osless]

```
06:35:54 Build Finished. 0 errors, 0 warnings. (took 56s.894ms)
```

Figure 3-13 Build finished

3.3.5 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…].

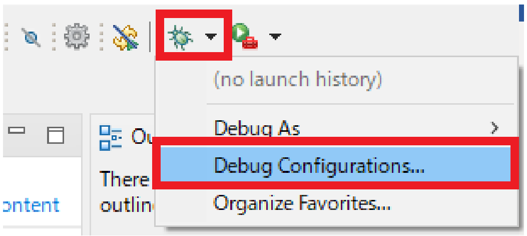
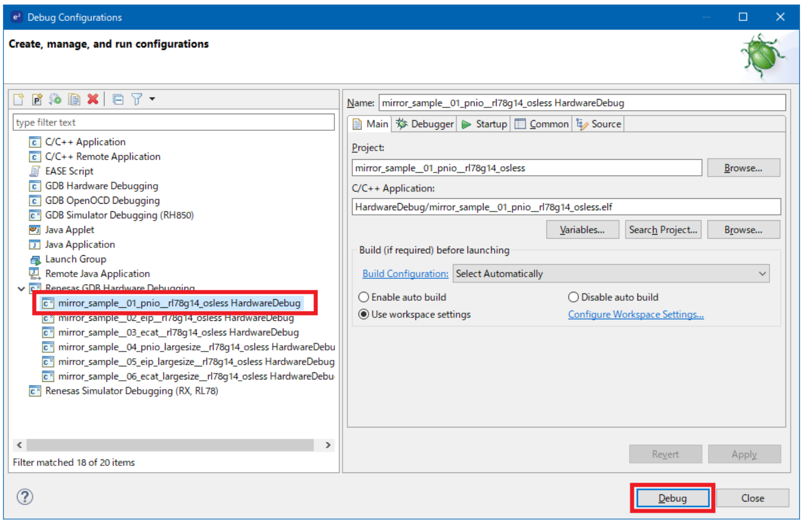


Figure 3-14 Debug Configurations

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



- C/C++ Application
- C/C++ Remote Application
- EASE Script
- GDB Hardware Debugging
- GDB OpenOCD Debugging
- GDB Simulator Debugging (RH850)
- Java Applet
- Java Application
- Launch Group
- Remote Java Application
- Renesas GDB Hardware Debugging
  - mirror_sample__01_pnio__rl78g14_osless HardwareDebug
  - mirror_sample__02_eip__rl78g14_osless HardwareDebug
  - mirror_sample__03_ecat__rl78g14_osless HardwareDebug
  - mirror_sample__04_pnio_largesize__rl78g14_osless HardwareDebu
  - mirror_sample__05_eip_largesize__rl78g14_osless HardwareDebug
  - mirror_sample__06_ecat_largesize__rl78g14_osless HardwareDebu
- Renesas Simulator Debugging (RX, RL78)

Filter matched 18 of 20 items

Name: mirror_sample__01_pnio__rl78g14_osless HardwareDebug

Project: mirror_sample__01_pnio__rl78g14_osless

C/C++ Application: HardwareDebug/mirror_sample__01_pnio__rl78g14_osless.elf

Build (if required) before launching

Build Configuration: Select Automatically

○ Enable auto build ○ Disable auto build

● Use workspace settings Configure Workspace Settings...

Figure 3-15 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

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. <u>Evaluation Environment Setup</u>

-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 3.3.4 to 3.3.5. When the sample application is run, the protocol display LED (PROFINET) turn on.

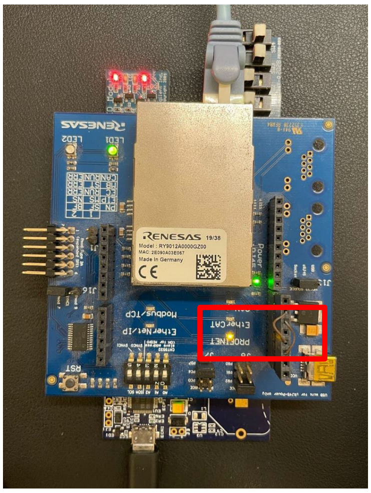


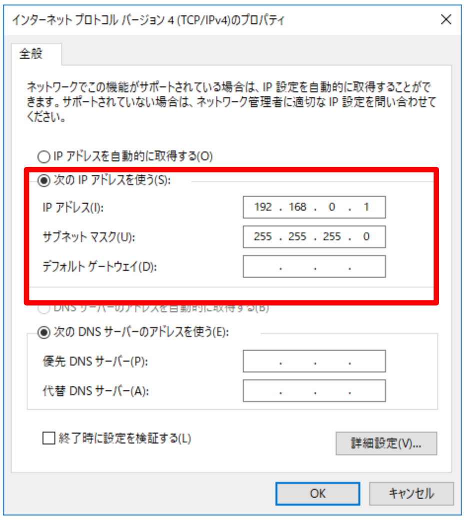
Figure 3-16 Protocol LED: PROFINET

-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).

Table 3-3 IP Address

IP address	192.168.0.1
Netmask	255.255.255.0



インターネット プロトコル バージョン 4 (TCP/IPv4)のプロパティ

全般

ネットワークでこの機能がサポートされている場合は、IP 設定を自動的に取得することができます。サポートされていない場合は、ネットワーク管理者に適切な IP 設定を問い合わせてください。

○ IP アドレスを自動的に取得する(O)

● 次の IP アドレスを使う(S):

IP アドレス(I): 192 . 168 . 0 . 1

サブネット マスク(U): 255 . 255 . 255 . 0

デフォルト ゲートウェイ(D): . . .

○ DNS サーバーのアドレスを自動的に取得する(B)

● 次の DNS サーバーのアドレスを使う(E):

優先 DNS サーバー(P): . . .

代替 DNS サーバー(A): . . .

□ 終了時に設定を検証する(L)

Figure 3-17 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.

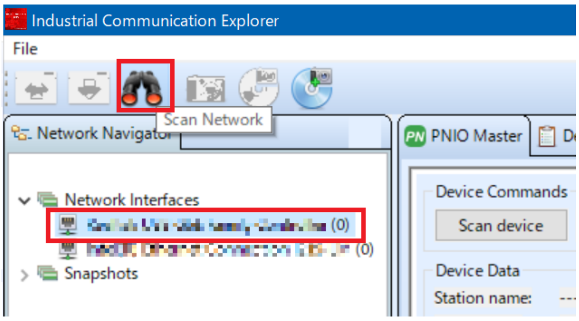
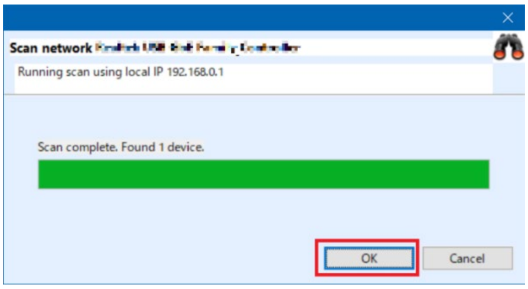


Figure 3-18 Scan network

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


Scan network [redacted]

Running scan using local IP 192.168.0.1

Scan complete. Found 1 device.

Figure 3-19 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].

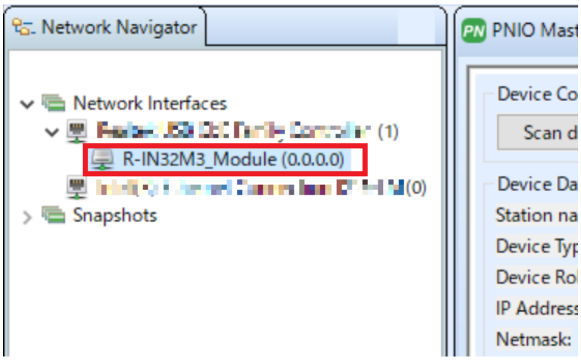


Figure 3-20 Select R-IN32M3 Module

-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.

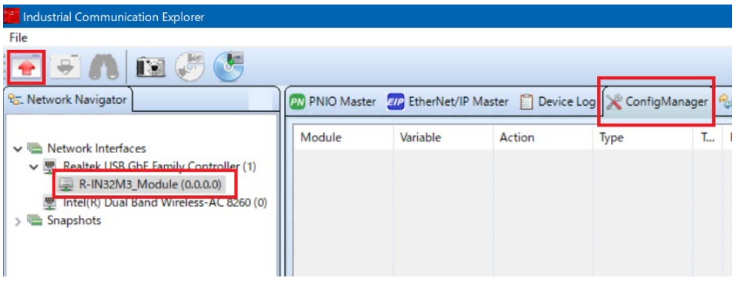
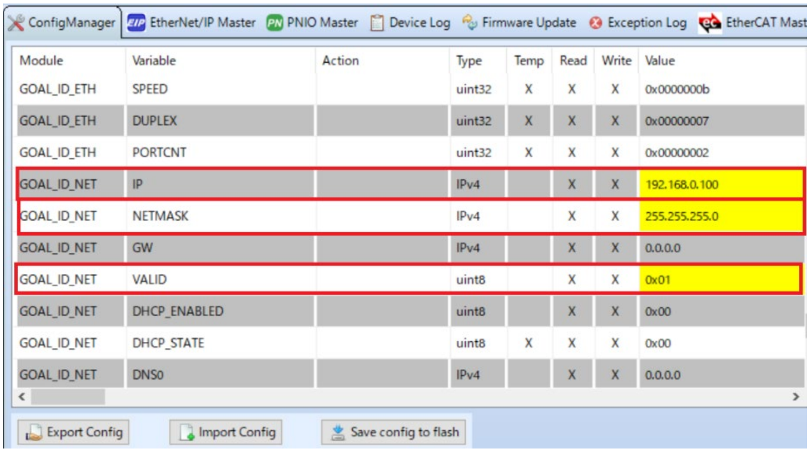

Industrial Communication Explorer
File
Network Navigator
PNIO Master
EtherNet/IP Master
Device Log
ConfigManager
Module
Variable
Action
Type
T...
Network Interfaces
Realtek USB GbE Family Controller (1)
R-IN32M3_Module (0.0.0.0)
Intel(R) Dual Band Wireless-AC 8260 (0)
Snapshots

Figure 3-21 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.

Table 3-4 GOAL_ID_NET

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



Module	Variable	Action	Type	Temp	Read	Write	Value
GOAL_ID_ETH	SPEED		uint32	X	X	X	0x0000000b
GOAL_ID_ETH	DUPLEX		uint32	X	X	X	0x00000007
GOAL_ID_ETH	PORTCNT		uint32	X	X	X	0x00000002
GOAL_ID_NET	IP		IPv4		X	X	192.168.0.100
GOAL_ID_NET	NETMASK		IPv4		X	X	255.255.255.0
GOAL_ID_NET	GW		IPv4		X	X	0.0.0.0
GOAL_ID_NET	VALID		uint8		X	X	0x01
GOAL_ID_NET	DHCP_ENABLED		uint8		X	X	0x00
GOAL_ID_NET	DHCP_STATE		uint8	X	X	X	0x00
GOAL_ID_NET	DNS0		IPv4		X	X	0.0.0.0

Figure 3-22 Set IP address on R-IN32M3 module

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

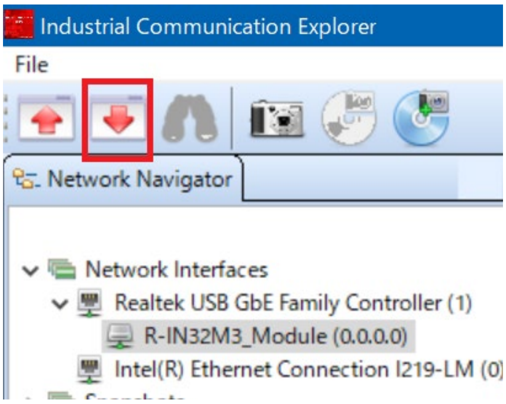
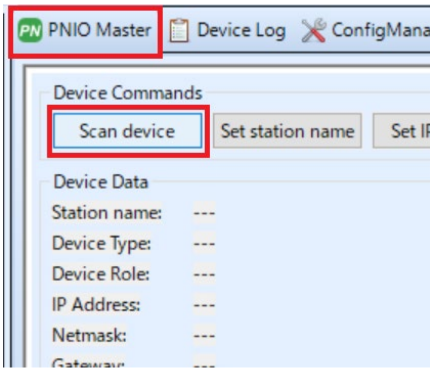

Industrial Communication Explorer
File
Network Navigator
Network Interfaces
Realtek USB GbE Family Controller (1)
R-IN32M3_Module (0.0.0.0)
Intel(R) Ethernet Connection I219-LM (0)

Figure 3-23 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.

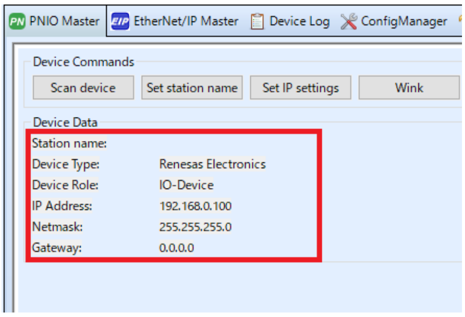
-8. Select [PNIO Master] panel, and then select [Scan device].



Device Data
Station name: ---
Device Type: ---
Device Role: ---
IP Address: ---
Netmask: ---
Gateway: ---

Figure 3-24 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.



Device Data
Station name:
Device Type: Renesas Electronics
Device Role: IO-Device
IP Address: 192.168.0.100
Netmask: 255.255.255.0
Gateway: 0.0.0.0

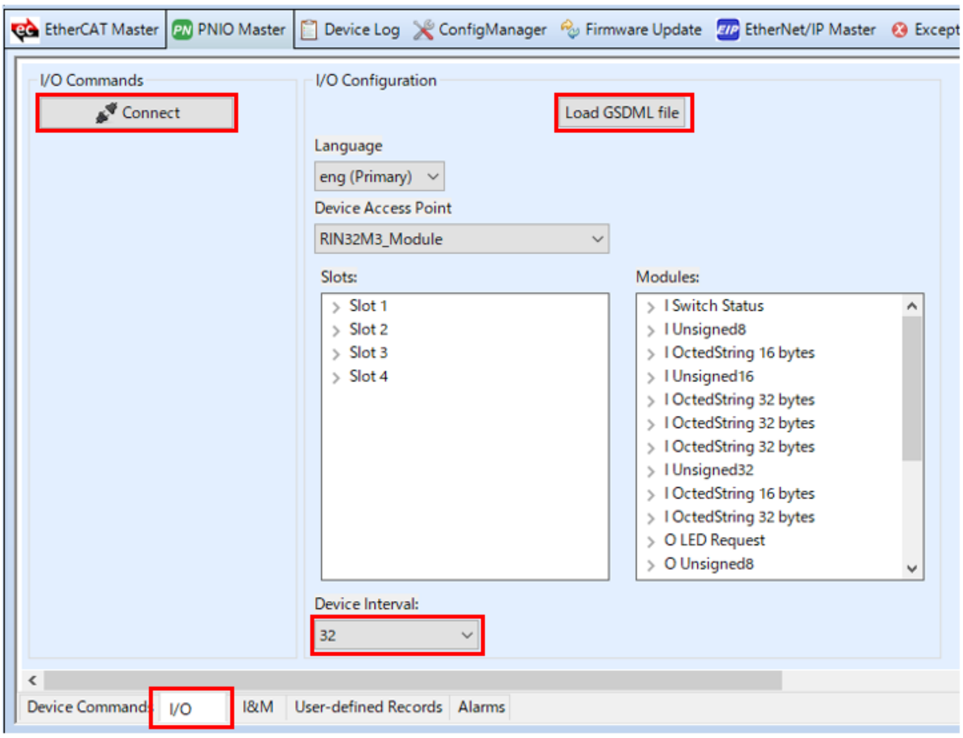
Figure 3-25 Device Data

-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 3-5 GSDML File

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.



Language
eng (Primary)

Device Access Point
RIN32M3_Module

Slots:
- Slot 1
- Slot 2
- Slot 3
- Slot 4

Modules:
- I Switch Status
- I Unsigned8
- I OctedString 16 bytes
- I Unsigned16
- I OctedString 32 bytes
- I OctedString 32 bytes
- I OctedString 32 bytes
- I Unsigned32
- I OctedString 16 bytes
- I OctedString 32 bytes
- O LED Request
- O Unsigned8

Device Interval:
32

Figure 3-26 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

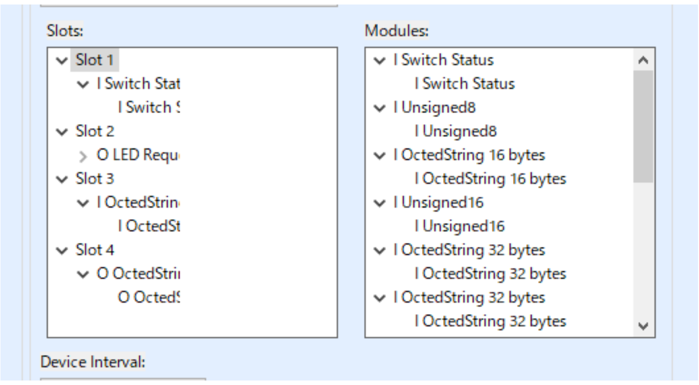
  Target Project : 01_pnio, 04_pnio_largesize, (10_multi_protocol)
- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 01_pnio, 04_pnio_largesize, (10_multi_protocol)
- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 04_pnio_largesize

Table 3-6 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



```
Slots:
Slot 1
    I Switch Stat
        I Switch
Slot 2
    O LED Requ
Slot 3
    I OctedStrin
        I OctedSt
Slot 4
    O OctedStri
        O Octed

Modules:
I Switch Status
    I Switch Status
I Unsigned8
    I Unsigned8
I OctedString 16 bytes
    I OctedString 16 bytes
I Unsigned16
    I Unsigned16
I OctedString 32 bytes
    I OctedString 32 bytes
I OctedString 32 bytes
    I OctedString 32 bytes

Device Interval:
```

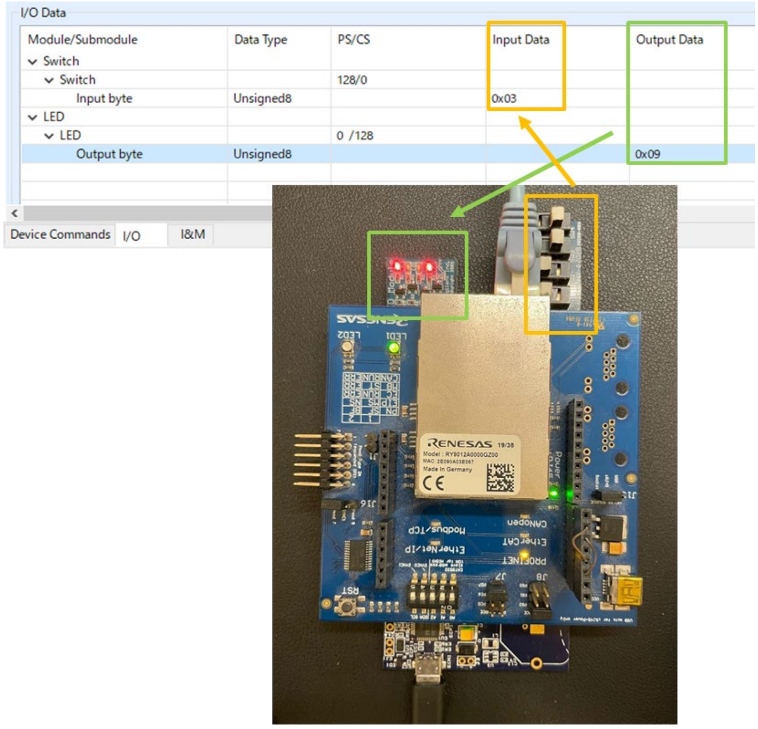
Figure 3-27 Application define (ex. 01_pnio)

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.

Table 3-7 I/O app.

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.



I/O Data

Module/Submodule	Data Type	PS/CS	Input Data	Output Data
⌄ Switch				
⌄ Switch		128/0		
Input byte	Unsigned8		0x03	
⌄ LED				
⌄ LED		0 /128		
Output byte	Unsigned8			0x09

Figure 3-28 Remote I/O control [PROFINET]

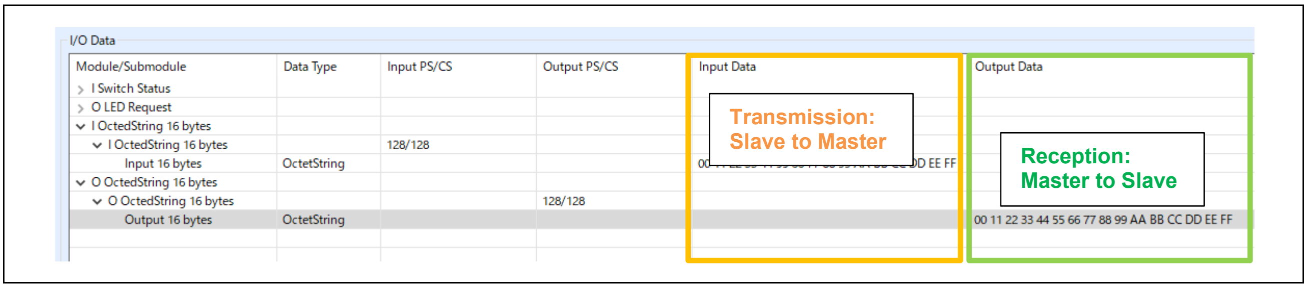
Mirror control

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.

Table 3-8 Mirror app.

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



Module/Submodule	Data Type	Input PS/CS	Output PS/CS	Input Data	Output Data
> I Switch Status					
> O LED Request					
∨ I OctedString 16 bytes					
∨ I OctedString 16 bytes		128/128			
Input 16 bytes	OctetString			00 [illegible] DD EE FF	
∨ O OctedString 16 bytes					
∨ O OctedString 16 bytes			128/128		
Output 16 bytes	OctetString				00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

Figure 3-29 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 3-9 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

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 3.3.4 to 3.3.5. When the sample application is run, the protocol display LED (EtherNet/IP) turn on.

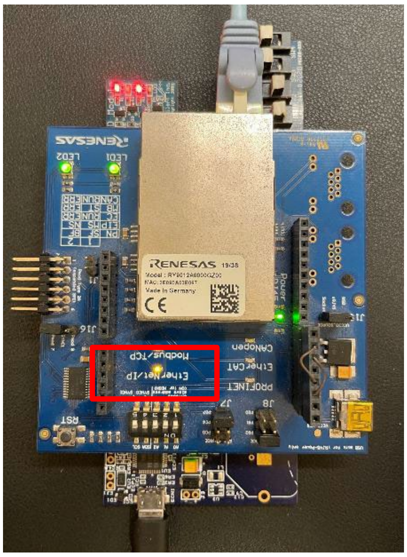


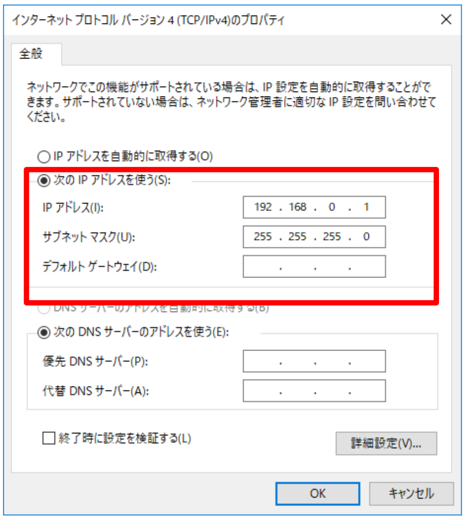
Figure 3-30 Protocol LED: EtherNet/IP

-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).

Table 3-10 IP Address

IP address	192.168.0.1
Netmask	255.255.255.0



インターネット プロトコル バージョン 4 (TCP/IPv4)のプロパティ

全般

ネットワークでこの機能がサポートされている場合は、IP 設定を自動的に取得することができます。サポートされていない場合は、ネットワーク管理者に適切な IP 設定を問い合わせてください。

○ IP アドレスを自動的に取得する(O)

● 次の IP アドレスを使う(S):

IP アドレス(I): 192 . 168 . 0 . 1

サブネット マスク(U): 255 . 255 . 255 . 0

デフォルト ゲートウェイ(D): . . .

○ DNS サーバーのアドレスを自動的に取得する(B)

● 次の DNS サーバーのアドレスを使う(E):

優先 DNS サーバー(P): . . .

代替 DNS サーバー(A): . . .

□ 終了時に設定を検証する(L)

Figure 3-31 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.

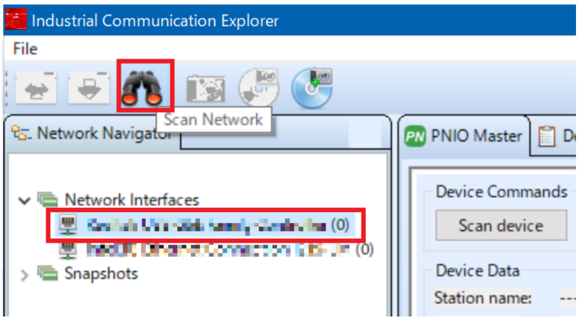
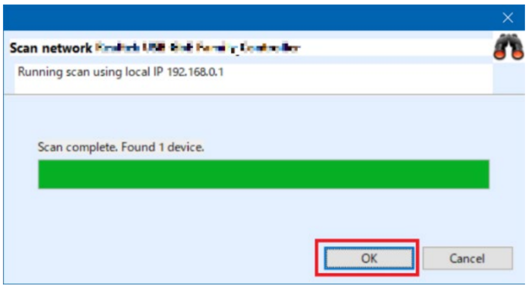


Figure 3-32 Scan network

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


Scan network [redacted]

Running scan using local IP 192.168.0.1

Scan complete. Found 1 device.

Figure 3-33 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].

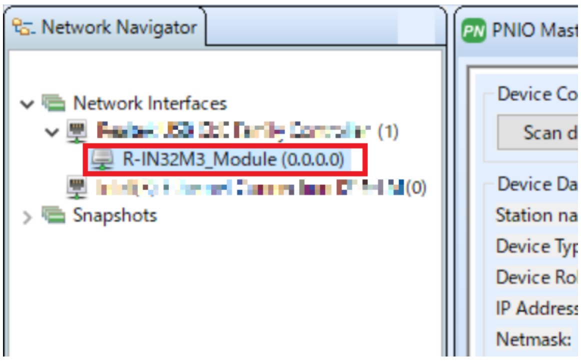


Figure 3-34 Select R-IN32M3 Module

-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.

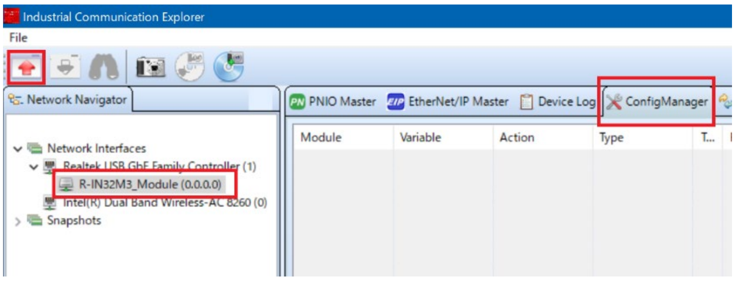
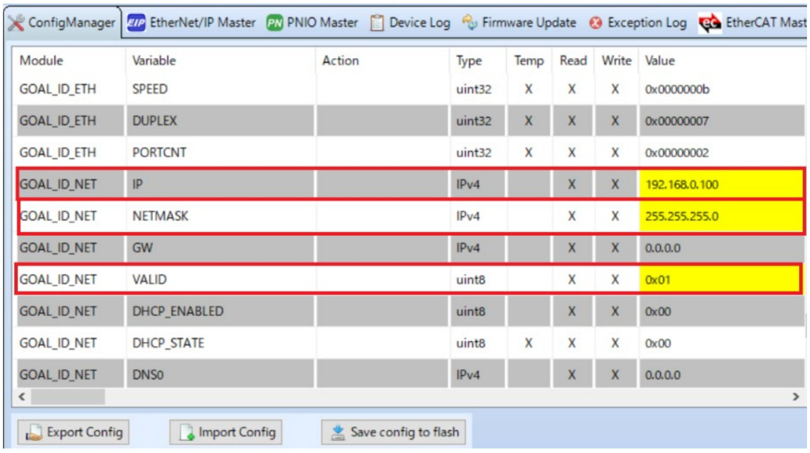

Industrial Communication Explorer
File
Network Navigator
Network Interfaces
Realtek USB GbE Family Controller (1)
R-IN32M3_Module (0.0.0.0)
Intel(R) Dual Band Wireless-AC 8260 (0)
Snapshots
PNIO Master
EtherNet/IP Master
Device Log
ConfigManager
Module
Variable
Action
Type
T...

Figure 3-35 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.

Table 3-11 GOAL_ID_NET

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



Module	Variable	Action	Type	Temp	Read	Write	Value
GOAL_ID_ETH	SPEED		uint32	X	X	X	0x0000000b
GOAL_ID_ETH	DUPLEX		uint32	X	X	X	0x00000007
GOAL_ID_ETH	PORTCNT		uint32	X	X	X	0x00000002
GOAL_ID_NET	IP		IPv4		X	X	192.168.0.100
GOAL_ID_NET	NETMASK		IPv4		X	X	255.255.255.0
GOAL_ID_NET	GW		IPv4		X	X	0.0.0.0
GOAL_ID_NET	VALID		uint8		X	X	0x01
GOAL_ID_NET	DHCP_ENABLED		uint8		X	X	0x00
GOAL_ID_NET	DHCP_STATE		uint8	X	X	X	0x00
GOAL_ID_NET	DNS0		IPv4		X	X	0.0.0.0

Figure 3-36 Set IP address on R-IN32M3 module

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

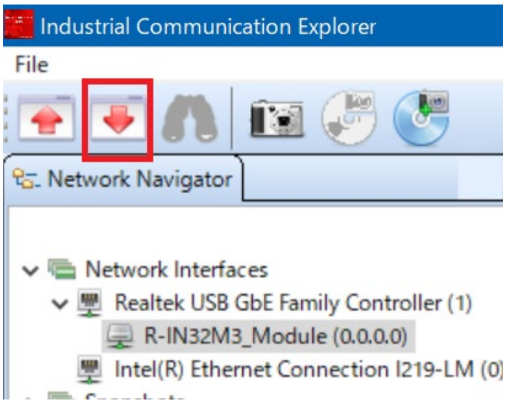
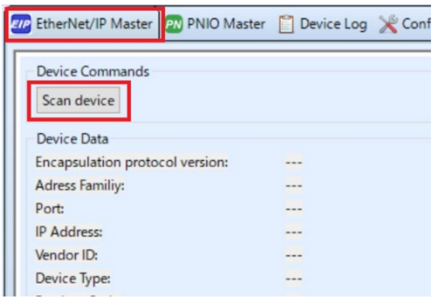

Industrial Communication Explorer
File
Network Navigator
Network Interfaces
Realtek USB GbE Family Controller (1)
R-IN32M3_Module (0.0.0.0)
Intel(R) Ethernet Connection I219-LM (0)

Figure 3-37 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.

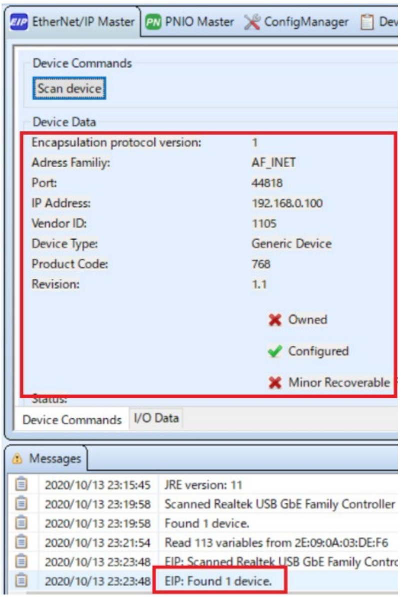


Device Data

Encapsulation protocol version:	---
Adress Familiy:	---
Port:	---
IP Address:	---
Vendor ID:	---
Device Type:	---

Figure 3-38 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.



Device Data

Encapsulation protocol version:	1
Adress Familiy:	AF_INET
Port:	44818
IP Address:	192.168.0.100
Vendor ID:	1105
Device Type:	Generic Device
Product Code:	768
Revision:	1.1

✗ Owned
✓ Configured
✗ Minor Recoverable
Status:

```
2020/10/13 23:15:45  JRE version: 11
2020/10/13 23:19:58  Scanned Realtek USB GbE Family Controller
2020/10/13 23:19:58  Found 1 device.
2020/10/13 23:21:54  Read 113 variables from 2E:09:0A:03:DE:F6
2020/10/13 23:23:48  EIP: Scanned Realtek USB GbE Family Contro
2020/10/13 23:23:48  EIP: Found 1 device.
```

Figure 3-39 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)
- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 02_eip, 05_eip_largesize, (10_multi_protocol)
- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 05_eip_largesize

Application defied:

Table 3-12 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

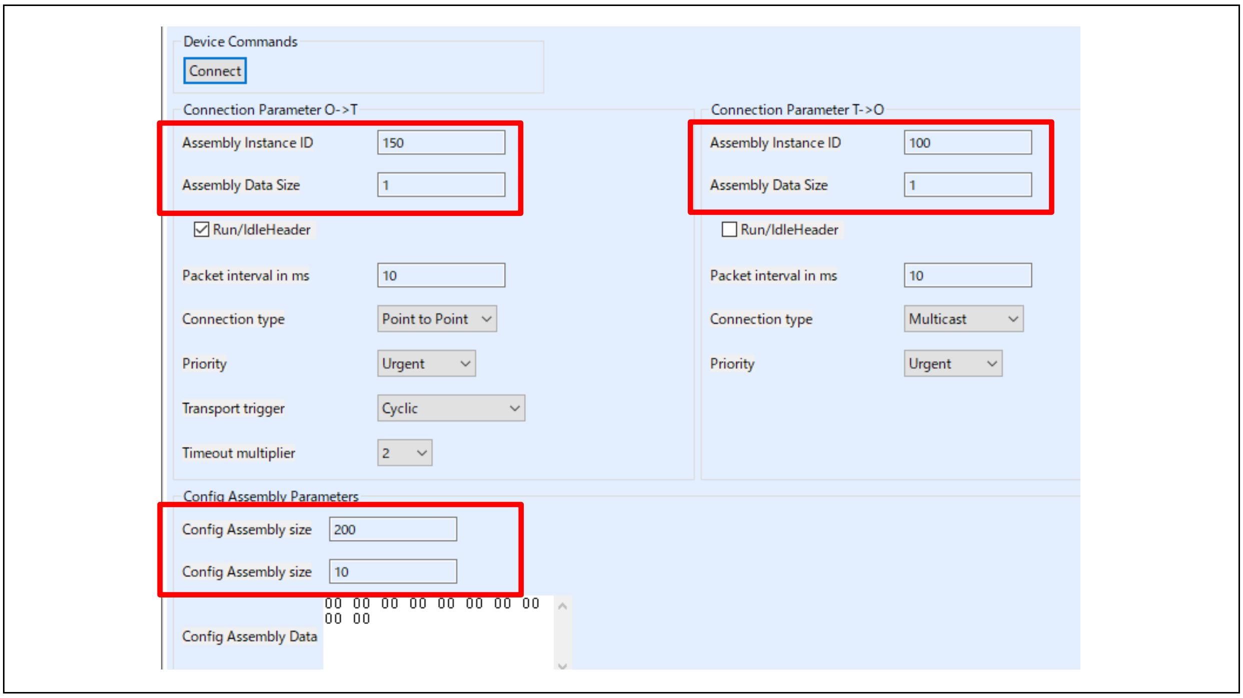
Table 3-13 Configuration

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

Remote-IO (LED/Switch)

Refer to Table 3-12 and Table 3-13 to set the connection parameters.

Packet interval in ms is left at the default value.



Device Commands

Connect

Connection Parameter O->T

Assembly Instance ID: 150

Assembly Data Size: 1

☑ Run/IdleHeader

Packet interval in ms: 10

Connection type: Point to Point

Priority: Urgent

Transport trigger: Cyclic

Timeout multiplier: 2

Connection Parameter T->O

Assembly Instance ID: 100

Assembly Data Size: 1

☐ Run/IdleHeader

Packet interval in ms: 10

Connection type: Multicast

Priority: Urgent

Config Assembly Parameters

Config Assembly size: 200

Config Assembly size: 10

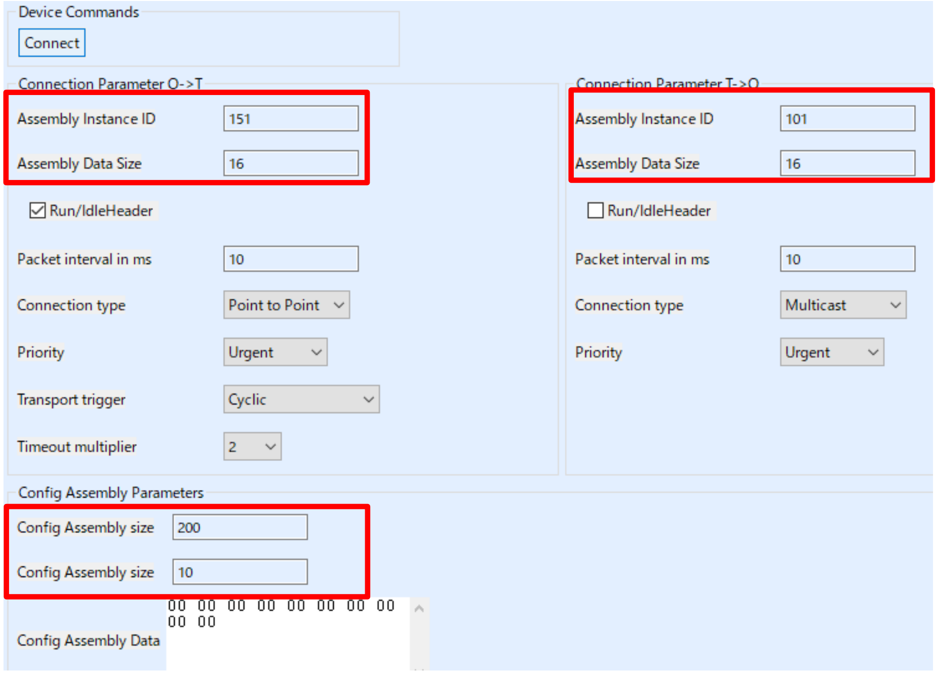
Config Assembly Data:

```
00 00 00 00 00 00 00 00
00 00
```

Figure 3-40 Remote-IO application parameter

Mirror control

Refer to Table 3-12 and Table 3-13 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.



Device Commands

Connection Parameter O->T

Assembly Instance ID: 151
Assembly Data Size: 16
☑ Run/IdleHeader
Packet interval in ms: 10
Connection type: Point to Point
Priority: Urgent
Transport trigger: Cyclic
Timeout multiplier: 2

Connection Parameter T->O

Assembly Instance ID: 101
Assembly Data Size: 16
□ Run/IdleHeader
Packet interval in ms: 10
Connection type: Multicast
Priority: Urgent

Config Assembly Parameters

Config Assembly size: 200
Config Assembly size: 10
Config Assembly Data:

```
00 00 00 00 00 00 00 00
00 00
```

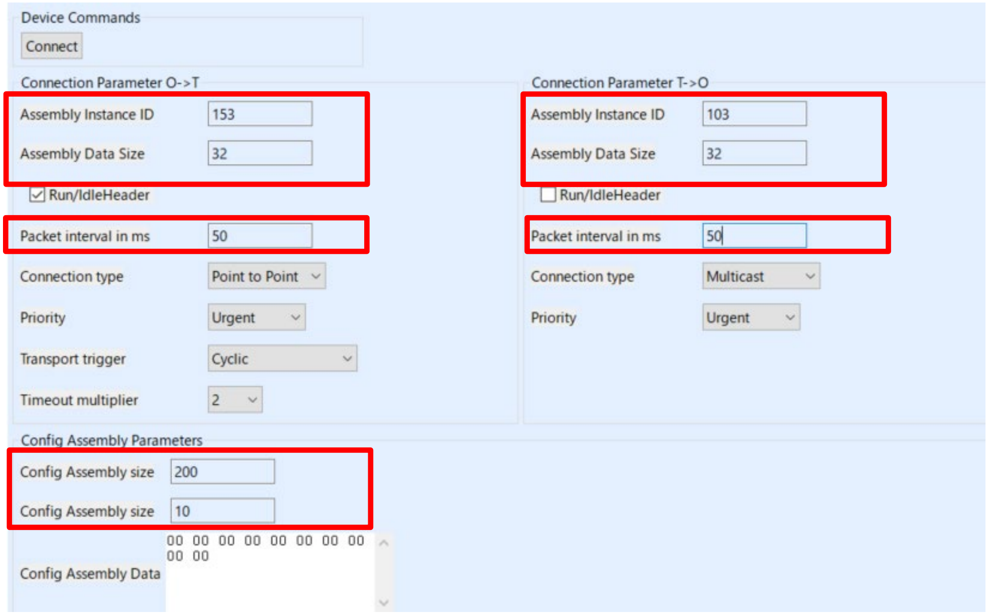
Figure 3-41 Mirror application parameter

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 3-12 and Table 3-13 to set the connection parameters. Figure 3-42 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.



Connection Parameter O->T

Assembly Instance ID: 153

Assembly Data Size: 32

☑ Run/IdleHeader

Packet interval in ms: 50

Connection type: Point to Point

Priority: Urgent

Transport trigger: Cyclic

Timeout multiplier: 2

Connection Parameter T->O

Assembly Instance ID: 103

Assembly Data Size: 32

□ Run/IdleHeader

Packet interval in ms: 50

Connection type: Multicast

Priority: Urgent

Config Assembly Parameters

Config Assembly size: 200

Config Assembly size: 10

Config Assembly Data: 00 00 00 00 00 00 00 00 00 00

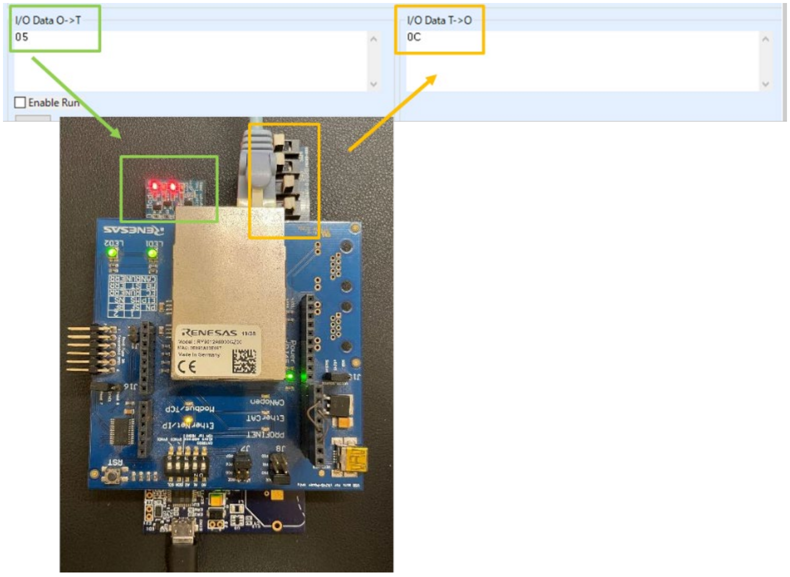
Figure 3-42 Mirror application (RPC) parameter

-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.



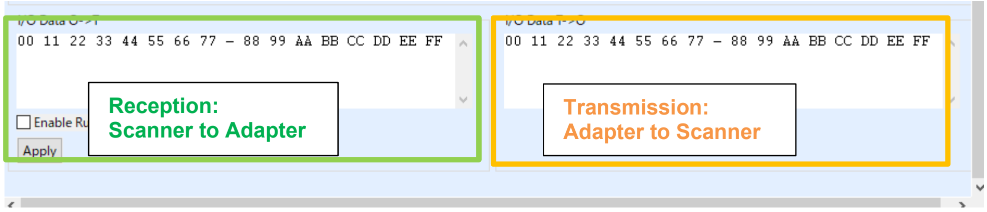
I/O Data O->T
05

I/O Data T->O
0C

Figure 3-43 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.



I/O Data O->T
00 11 22 33 44 55 66 77 - 88 99 AA BB CC DD EE FF

I/O Data T->O
00 11 22 33 44 55 66 77 - 88 99 AA BB CC DD EE FF

Reception:
Scanner to Adapter

Transmission:
Adapter to Scanner

Figure 3-44 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 3-14 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

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 3.3.4 to 3.3.5. When the sample application is run, the protocol display LED (EtherCAT) turn on.

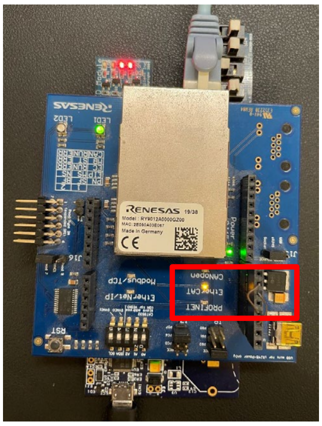


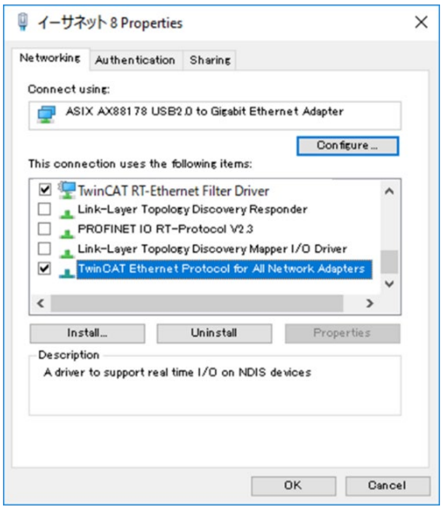
Figure 3-45 Protocol LED: EtherCAT

-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


イーサネット 8 Properties

Networking | Authentication | Sharing

Connect using:

ASIX AX88178 USB2.0 to Gigabit Ethernet Adapter

Configure...

This connection uses the following items:

☑ TwinCAT RT-Ethernet Filter Driver
☐ Link-Layer Topology Discovery Responder
☐ PROFINET IO RT-Protocol V2.3
☐ Link-Layer Topology Discovery Mapper I/O Driver
☑ TwinCAT Ethernet Protocol for All Network Adapters

Install... | Uninstall | Properties

Description

A driver to support real time I/O on NDIS devices

OK | Cancel

Figure 3-46 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 3-15 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].

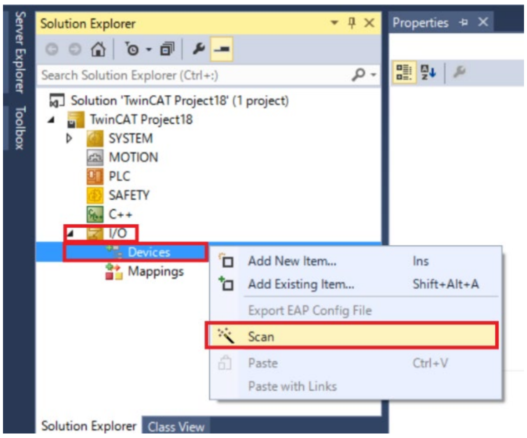


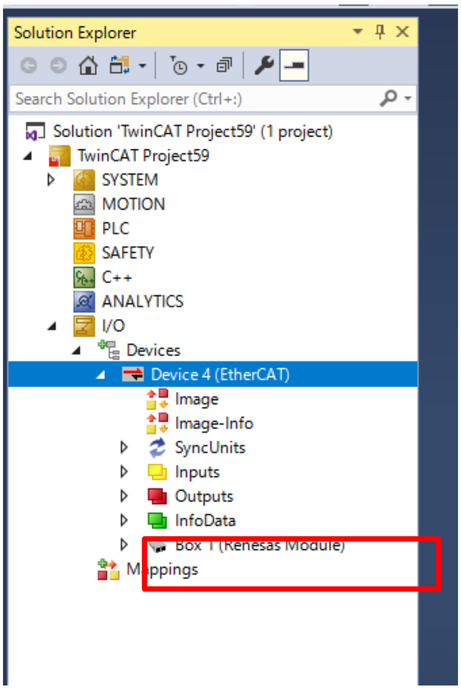
Figure 3-47 Scan network

-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].



Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'TwinCAT Project59' (1 project)
  - TwinCAT Project59
    - SYSTEM
    - MOTION
    - PLC
    - SAFETY
    - C++
    - ANALYTICS
    - I/O
      - Devices
        - Device 4 (EtherCAT)
          - Image
          - Image-Info
          - SyncUnits
          - Inputs
          - Outputs
          - InfoData
          - Box 1 (Renesas Module)
      - Mappings

Figure 3-48 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)
- **Mirror:** Sends data received from the master and mirrored back

  Target Project : 03_ecat, 06_ecat_largesize , (10_multi_protocol)
- **Mirror (RPC) :** Sends data received from the master and mirrored back

  Target Project : 06_ecat_largesize

Table 3-16 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

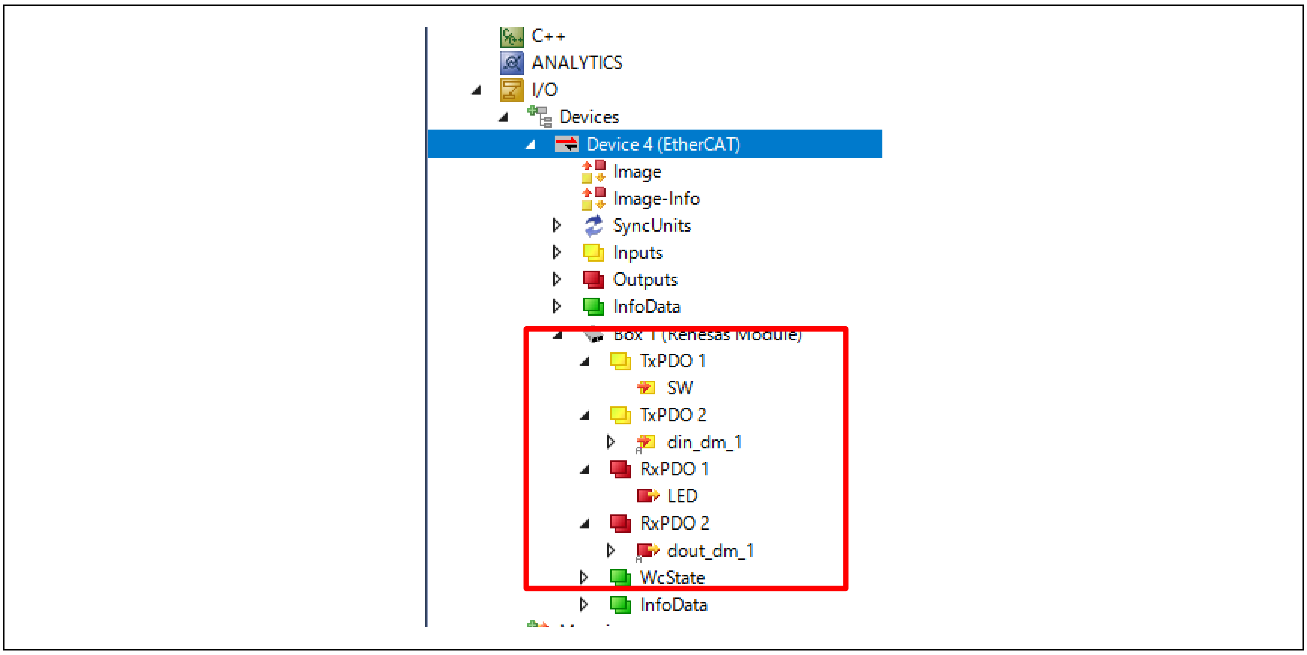


Figure 3-49 Application define (ex. 03_ecat)

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.

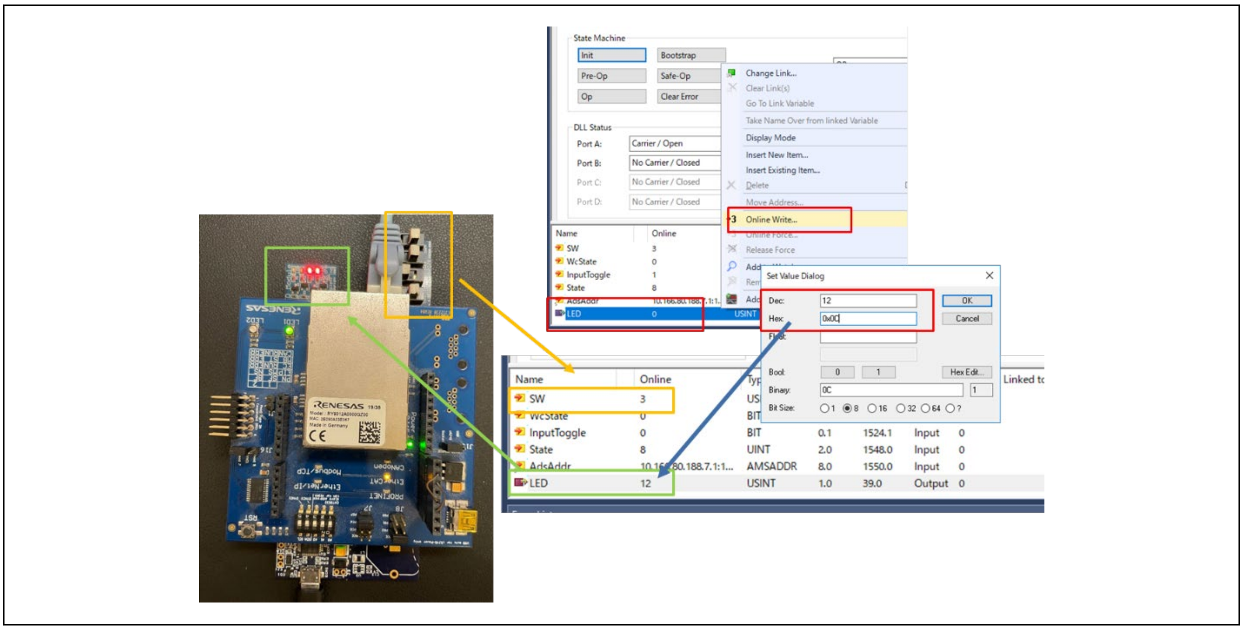


Figure 3-50 Remote I/O control [EtherCAT]

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.

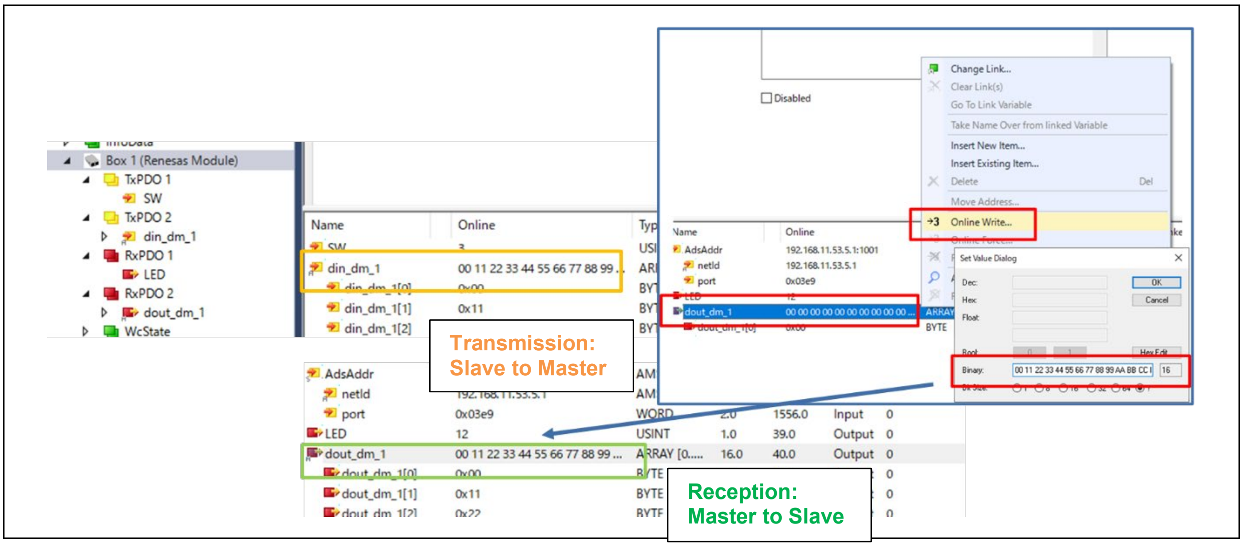


Figure 3-51 Mirror control [EtherCAT]

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 3-17 Modbus TCP 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 3-18 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 2.3.

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.4 to 3.3.5.

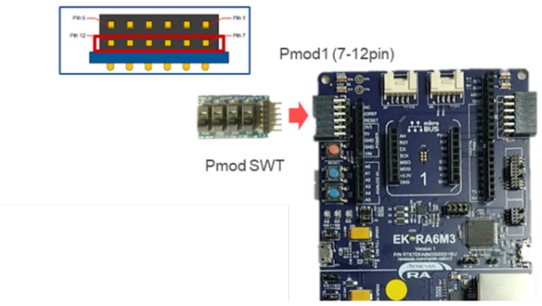


Figure 3-52 General purpose switch (SW9)

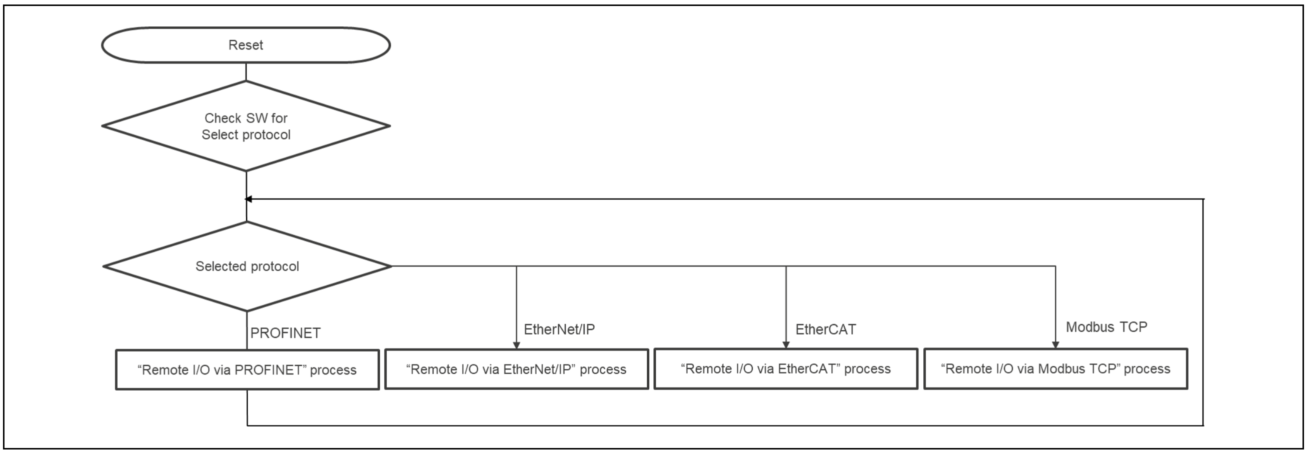

Reset
Check SW for Select protocol
Selected protocol
PROFINET
EtherNet/IP
EtherCAT
Modbus TCP
"Remote I/O via PROFINET" process
"Remote I/O via EtherNet/IP" process
"Remote I/O via EtherCAT" process
"Remote I/O via Modbus TCP" process

Figure 3-53 Multi-protocol selector flow

Table 3-19 Pmod SWT Setting

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

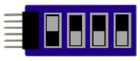


Figure 3-54 PROFINET

EtherNet/IP：Pmod SWT [off-on-off-off]

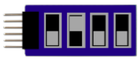


Figure 3-55 EtherNet/IP

EtherCAT：Pmod SWT [off-off-on-off]

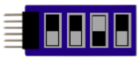


Figure 3-56 EtherCAT

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

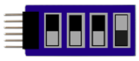


Figure 3-57 Modbus TCP Server

-2. Set Network Adapter

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

Table 3-20 Network Adapter

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.

Table 3-21 Master Connection

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

3.5 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.

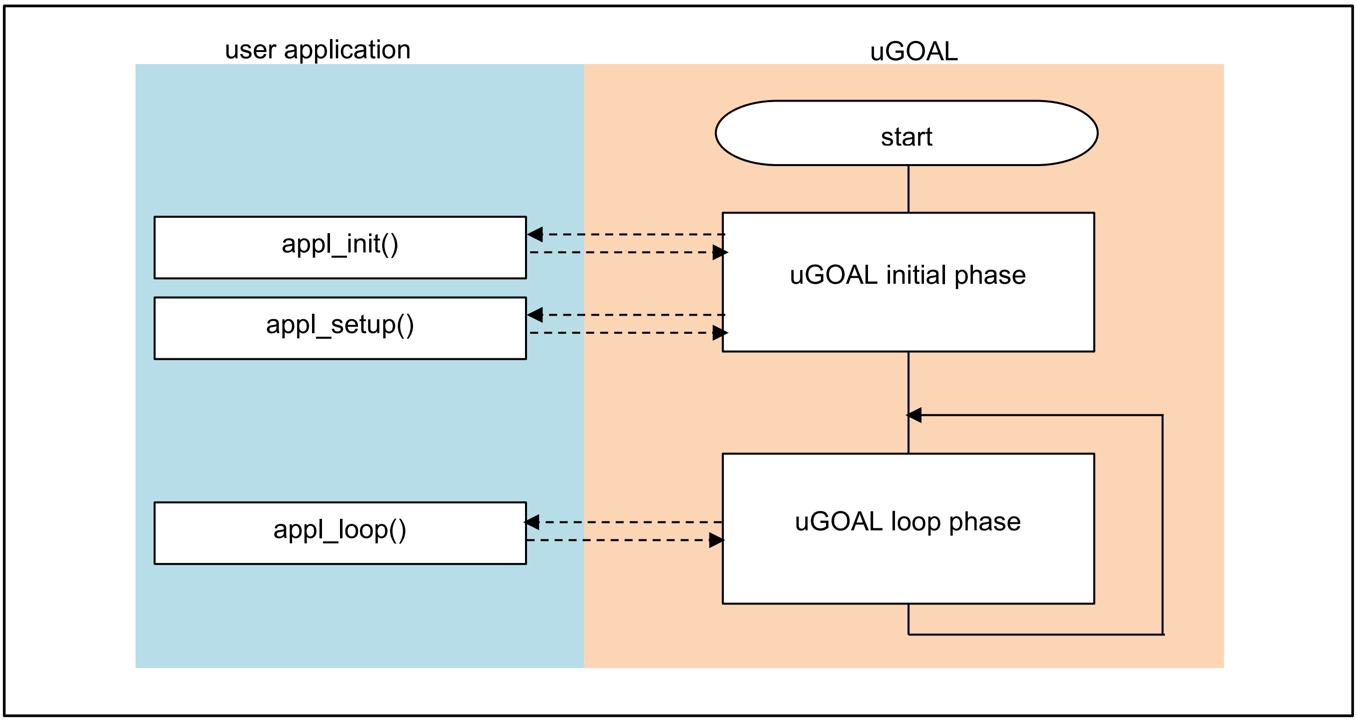


Figure 3-58 Overall flow of the program

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 3-22 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 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);
    if (GOAL_RES_ERR(res)) {                                         ①
        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)) {                                         ③
        goal_logErr( “failed to create a new PROFINET instance” );
        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”);                                                 ④
        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;
    }

. . .                                                                                           ⑤

    /* 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 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 */                                               ①
        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)) {                                                                              ①
        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 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”);                         ①
        return res;
    }
・・・

    res = goal_eipNew(&pHdlEip, 0, main_eipCallback);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(″failed to create eip instance %″FMT_x32, res);         ②
        return res;
    }

    res = main_eipApplInit(pHdlEip);
    if (GOAL_RES_ERR(res)) {
        goal_logErr(“failed to initialize assembly and attribute configuration”);   ③
        return res;
    }

・・・

}
```

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 */                                                ①
        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();                                                                   ①
    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”);            ①
        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”);
    }
                                                                           ②
・・・

    /* 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 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” );                    ③
        return res;
    }

    res = appl_ecatCreateObjects(pHdlEcat);
    if (GOAL_RES_ERR(res)) {
        goal_logErr( “failed to initialize object dictionary” );                     ④
        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)) {                                                          ⑤
        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);   ⑥
        if (GOAL_RES_OK(res)) {
            goal_logInfo(″event generation enabled″);
        }
    }
#endif

 . . .

    return res;
}
```

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;                                          ①

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

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

・・・
}
```

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

In this sample software, the function to output log message for debug is prohibited due to the memory allocation limitation of RL/78G14.

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 4-1 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)

```
GOAL_STATUS_T goal_maNetIpSet(
    GOAL_MA_NET_T *pNetHdl,                    /**< pointer to store NET handler */
    uint32_t addrIp,                           /**< IP address */
    uint32_t addrMask,                         /**< subnet mask */
    uint32_t addrGw,                            /**< gateway */
    GOAL_BOOL_T flgTemp                        /**< temporary IP config flag */
);
```

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 4-2 provides a list of how to set up an IP address.

Table 4-2 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.

4.4 Board Stand-alone Operation

The emulator is placed in the forced reset state by short-circuiting EJ1 header (Emulator reset header) on RL78/G14 Fast Prototyping Board. This allows stand-alone operation of the RL78/G14 independently of control by the IDE (ex: e2studio) while the IDE is applying a forcible reset(only through-holes are provided; an actual header component is not mounted).

For details, refer to “RL78/G14 Fast Prototyping Board User's Manual” (R20UT4573EJ****).

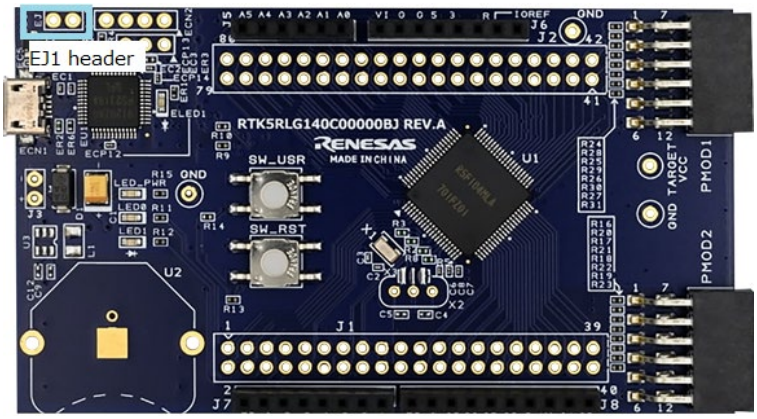


Figure 4-1 EJ1 header

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct/15/2021	-	First Edition
1.01	Jan/11/2022	24	Add Remote I/O sample application
		51	Add Modbus TCP sample application
1.02	Aug/5/2022	-	Minor correction
1.03	May/31/2023	24	Review of description with sample program update
1.04	Dec/15/2023	40	Added explanation about Mirror (RPC) in section 3.4.2
		30	Replaced Figure 3-26 and modified the explanation
		6	Updated Table 1-1
		4	Updated list of Related documents
		-	Minor correction
1.05	May/31/2024	6	Updated Table 1-1 エラー! 参照元が見つかりません。
		30	Updated Table 3-5

Trademark

- ARM and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.
- Ethernet is a registered trademark of Fuji Xerox Co., Ltd.
- EtherCAT® and TwinCAT® are registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- EtherNet/IP is a registered trademark of ODVA Inc.
- PROFINET is a registered trademark of PROFIBUS Nutzerorganisation e.V. (PNO)
- Modbus is a registered trademark of Schneider Electric SA.
- Additionally, all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

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/.