Application Note

R-IN32M3 Module (RY9012A0)

RX66T Sample Application (uGOAL Edition)

R30AN0399EJ0105
Rev.1.05
May.31.2024

Introduction

This document describes sample software for host microcomputers (RX66T) mounted on the R-IN32M3 Module Evaluation Board (SEM1320), which is manufactured by Shimafuji Electric Co., Ltd.

Target Device

RX66T (R5F566TKADFP)

R-IN32M3 Module (RY9012A0)

Contents

List of Abbreviations and Acronyms

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

Terms	Description
This board	R-IN32M3 Module Evaluation Board: SEMB1320 (manufactured by Shimafuji Electric Co., Ltd.), which is the target board for the sample programs explained in this document
This sample software (SW)	The sample program for the host microcomputer (RX66T) that is explained in this document
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	User's Implementation Guide (uGOAL Edition)	R30AN0402EJ****
Application Note	R-IN32M3 Module Management Tool Instruction Guide	R30AN0390EJ****
Application Note	R-IN32M3 Module Modbus TCP Start-Up Manual	R30AN0406EJ****
Application Note	Sensorless Vector Control for Permanent Magnet Synchronous Motor (Implementation)	R01AN4244EJ****
User's Manual	Renesas Solution Starter Kit 24V Motor Control Evaluation System for RX23T (Motor RSSK)	R20UT3697EJ****
Application Note	RX Smart Configurator User's Guide: e² studio	R20AN0451ES****
Application Note	Software PLC Connection Guide TwinCAT	R30AN0380EJ****

1. Overview

1.1 Overview

This document describes sample software for host microcomputers (RX66T) mounted on theR-IN32M3 Module Evaluation Board (SEMB1320), which is manufactured by Shimafuji Electric Co., Ltd.

The SEMB1320 is equipped with the RX66T host MCU that is SPI-connected to the R-IN32M3 module.

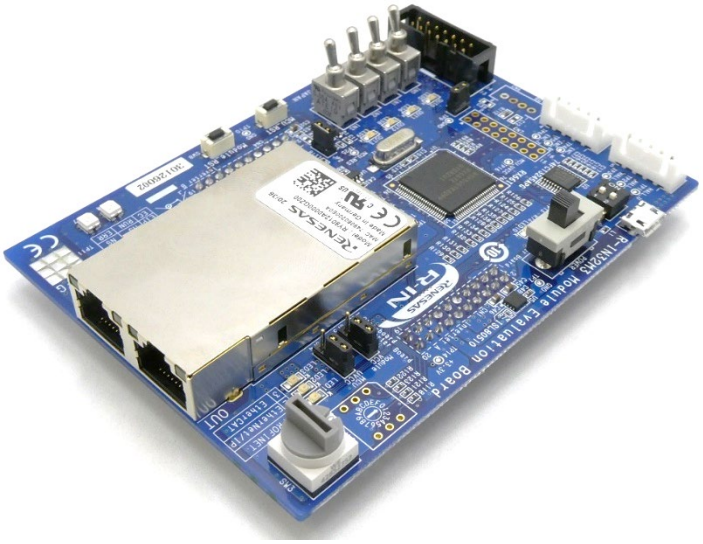

Figure 1-1 R-IN32M3 Module Evaluation Board

This board can be connected to an optional inverter board, which is included in "24V Motor Control Evaluation System for RX23T", to evaluate motor control by industrial Ethernet protocol communication.

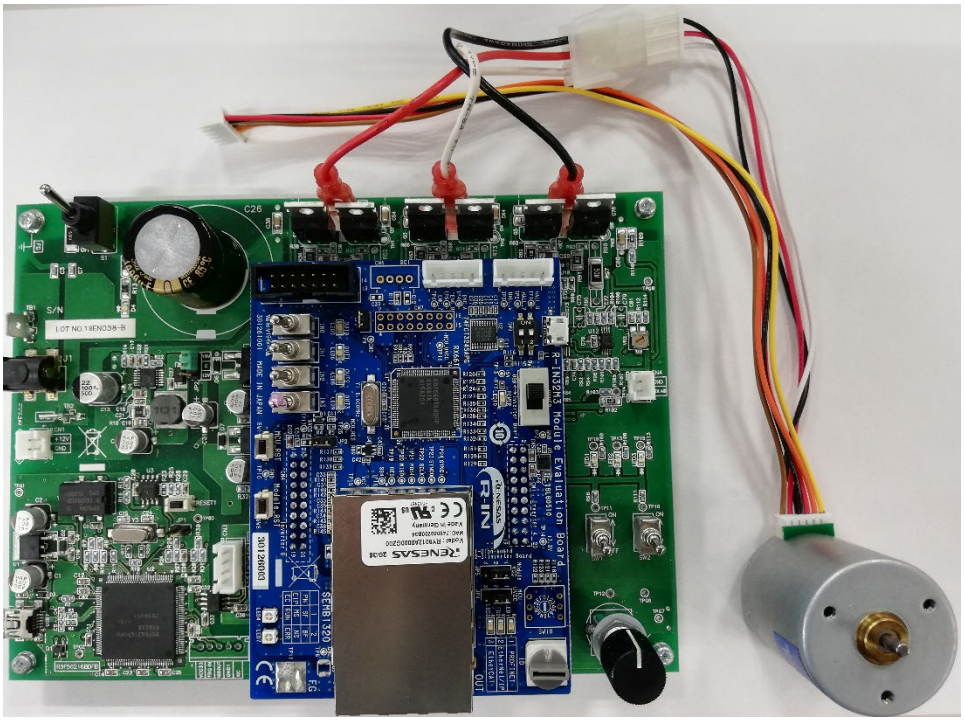

Figure 1-2 Photos of this board connected to the Inverter Board

1.2 Operating environment

The operating environment of this sample software is shown in Table 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	
RX family GNU Toolchain	GCC for Renesas RX	V8.3.0.202311	-	Included in installer of e2studio, otherwise download/install it individually, see chapter 4.5
RX family C/C++ Compiler package	CC-RX	V3.06.00	C/C++ Compiler Package for RX Family | Renesas	Included in installer of e2studio
FIT module	RX Driver Package	V1.42	RX Family RX Driver Package | Renesas	Included in installer of e2studio, otherwise download/install it individually, see chapter 4.6
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

One of the following emulators is required separately to run this sample software.

Table 1-2 Supported emulators

Abbreviation	Name	Description
E1	E1 emulator	On-chip debugging emulator and flash programmer Type name：R0E000010KCE00
E2 Lite	E2 emulator Lite	On-chip debugging emulator and flash programmer Type name：RTE0T0002LKCE00000R

2. Hardware configuration

2.1 List of specifications

Table 2-1 shows a list of specifications for this board.

Table 2-1 Board specifications

Item		Functions and specification
Input Power	Input voltage	5V (Supply from USB Micro B or inverter board.) or 3.3V (from inverter board)
MCU	Type name	R5F566TKADFP
	Flash memory	1MB
	RAM	128KB
	Data flash memory	32KB
MCU Input Clock	Oscillator	8MHz
Connectors	USB Micro B	only input power (data line is not connected)
	JTAG	E1 / E2 Lite emulator 2.54mm pitch 14pins
	Inverter Board	CNA：2.54mm pitch 20pins CNB：2.54mm pitch 20pins
	Hall sensor	B5B-XH-A
	Encoder	B5B-XH-A
	SCI	B4B-XH-A (not mounted)
Switches	Power input switch	Slide switch SPDT
	Generic	DIP switch 2bit toggle switch 4bit rotary switch Hexadecimal (not mounted)
	EtherCAT ID	rotary switch Hexadecimal
	CPU Reset	Push switch 1bit
	R-IN32M3 Module Reset	Push switch 1bit
LED	5V power in	Red 1bit
	Generic	Green 4bit
	Protocol display	Green 3bit
	Protocol status	Bi-Color(2bit) x 2
Pin Headers for External Extensions		2.54mm pitch 16 pins (not mounted)
consumption	while executing motor program	60mA typ.
	while executing ethernet communication program	260mA typ.
Operating Temperature		0 ~ 45℃
Board Dimension		80mm × 110mm t = 1.6mm

2.2 Appearance of the board

Figure 2-1 shows the appearance of this board, and Table 2-2 shows the main parts and external interfaces.

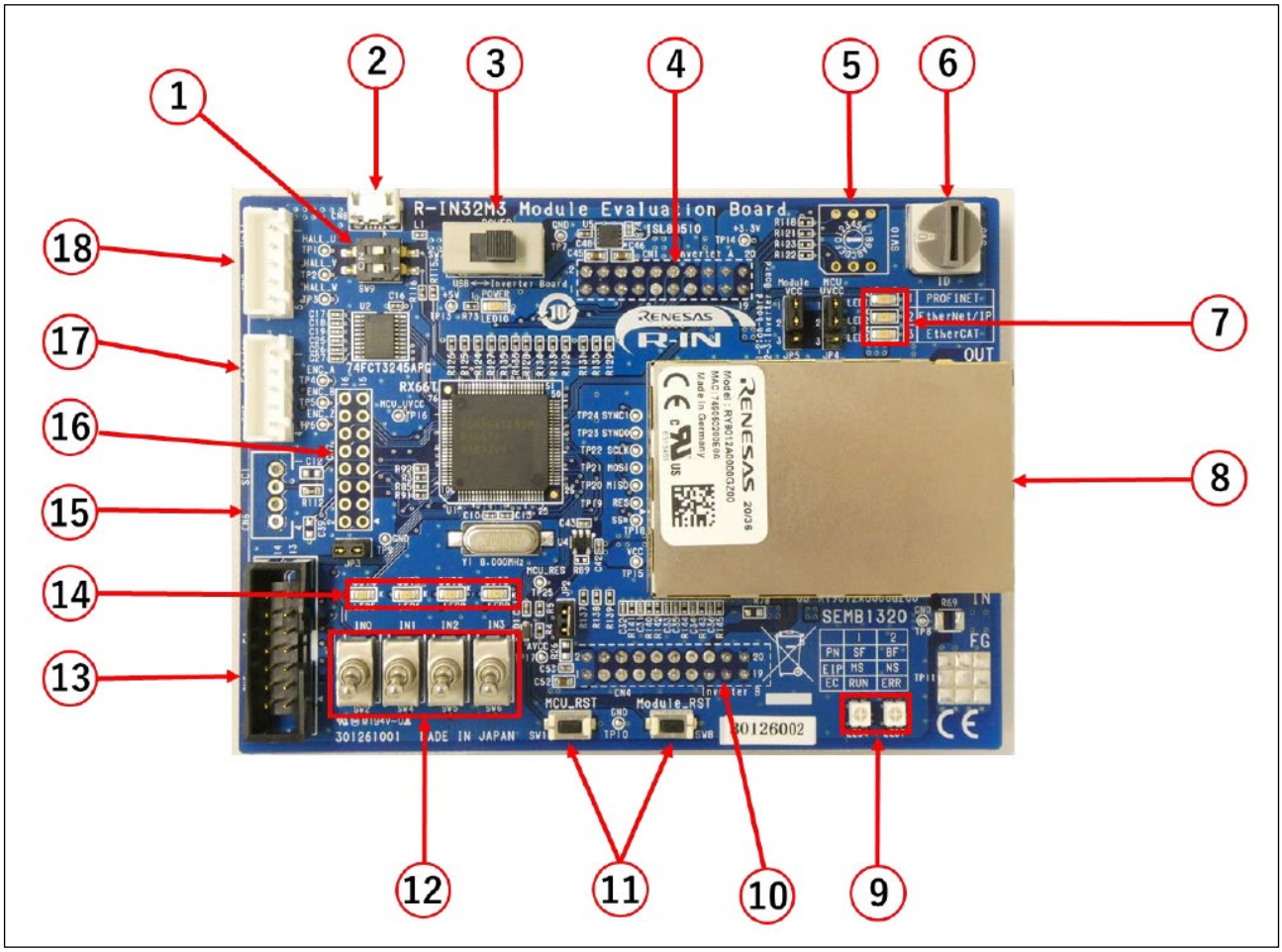

Figure 2-1 Board appearance

Table 2-2 Main parts and external interfaces

No.	Component Description	No.	Component Description
1	General purpose DIP switch	10	Inverter Board connector B
2	USB Micro B	11	Reset switch
3	Power input selector switch	12	General purpose Input switch
4	Inverter Board connector A	13	JTAG connector
5	General-purpose rotary switch (not mounted)	14	General purpose Output LED
6	EtherCAT ID switch	15	SCI connector (not mounted)
7	Protocol display LED	16	Pin header for external expansion (not mounted)
8	R-IN32M3 Module	17	Encoder connector
9	Protocol status LED	18	Hall sensor connector

2.3 Block diagram

The block diagram of this board is shown below.

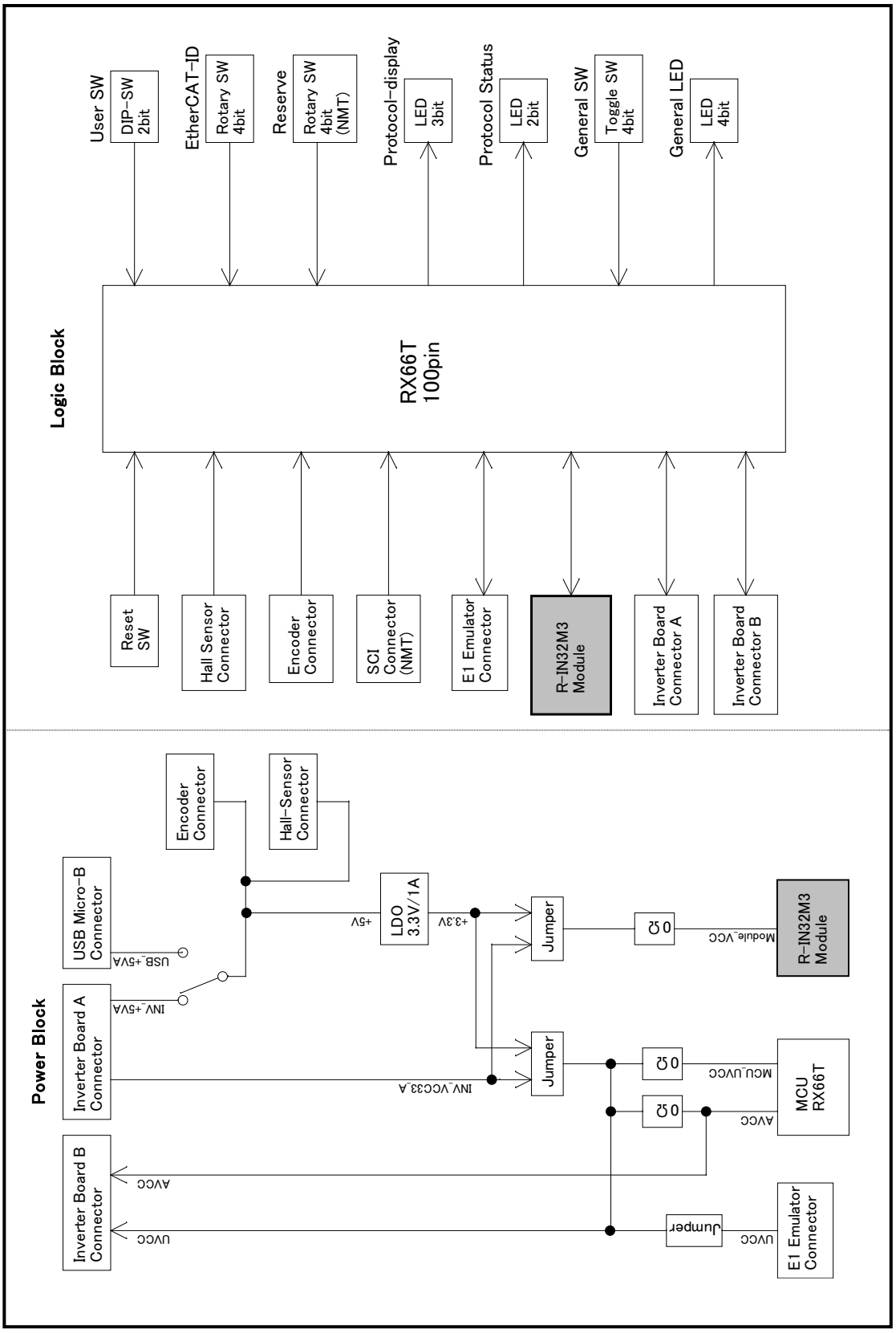

Figure 2-2 Block diagram

2.4 Features

2.4.1 Power supply

The board is powered by a USB Micro B connector or an Integer Board connector.

Normally, if you want to use this board alone, set SW7 at the USB Micro B connector side.

3.3V power supply method to RX66T MCU and the Module can be selected with jumper pins. For more information, 2.5.5(3) JP4、2.5.5(4) JP5

Figure 2-3 shows power supply configuration diagram of this board.

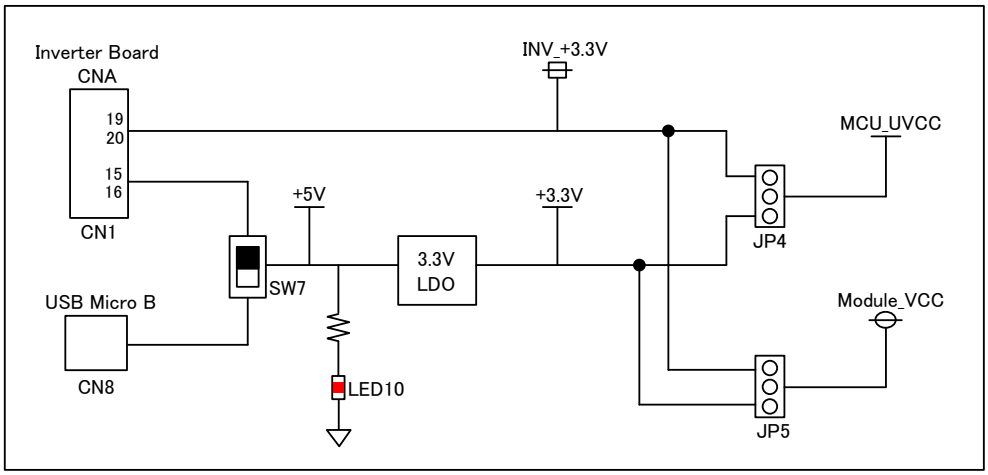

Figure 2-3 Power diagram

2.4.2 GND

The board's GND and AVSS are connected by R39 near CN4.

In the shipping state, JP2 is shorted, and the PGAVSS0 terminal of the MCU is connected to the AVSS.

The FG terminal of the R-IN32M3 Module is connected to the TP11 and can be connected to the GND by mounting R69.

The connection diagram of the GND of this board is shown below.

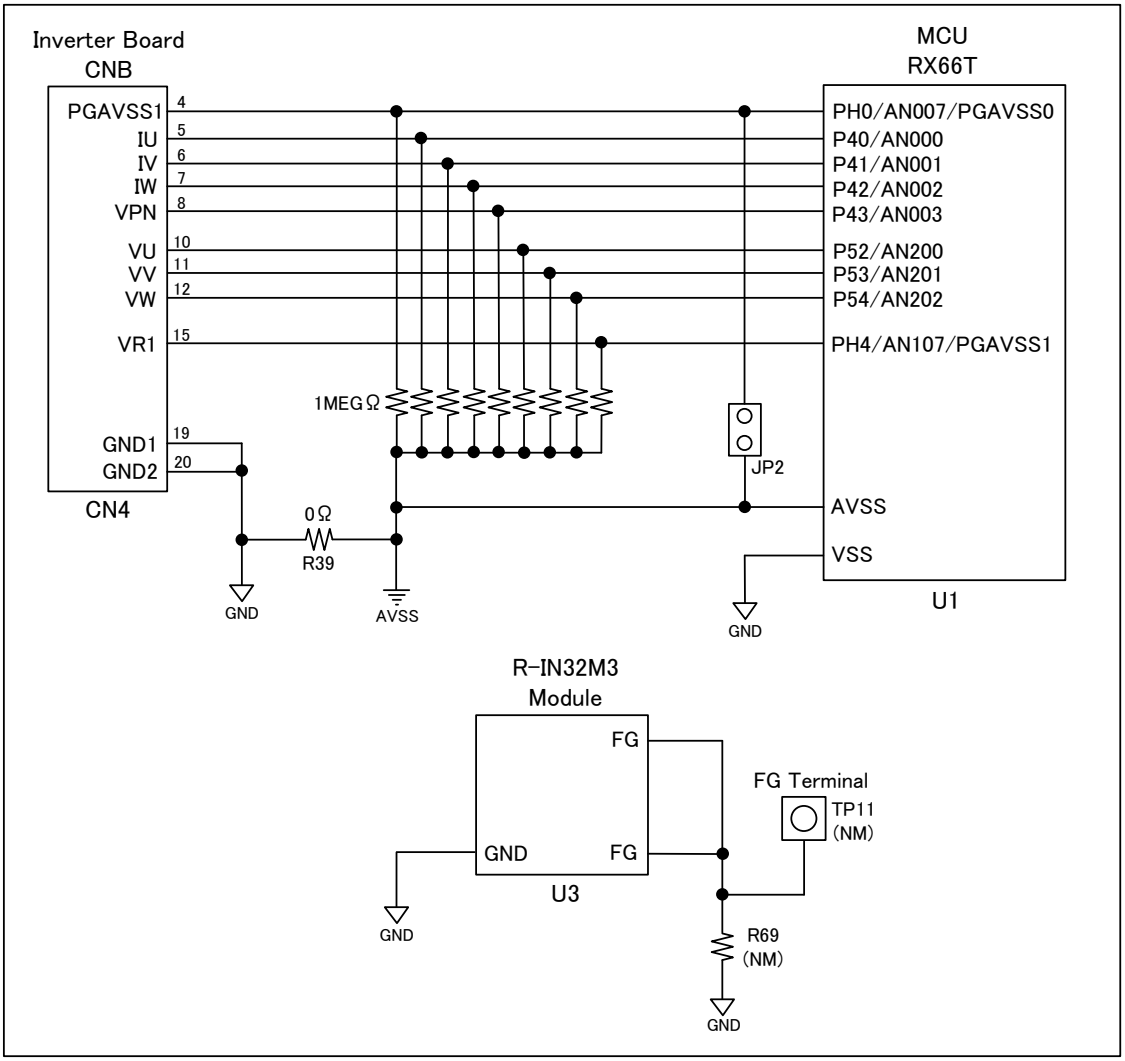

Figure 2-4 GND

2.4.3 RESET and JTAG

This board has three methods of reset, "Power ON Reset", "Reset by JTAG Emulator", and "Reset by External Switch". The reset and JTAM diagram of this board shows in Figure 2-5

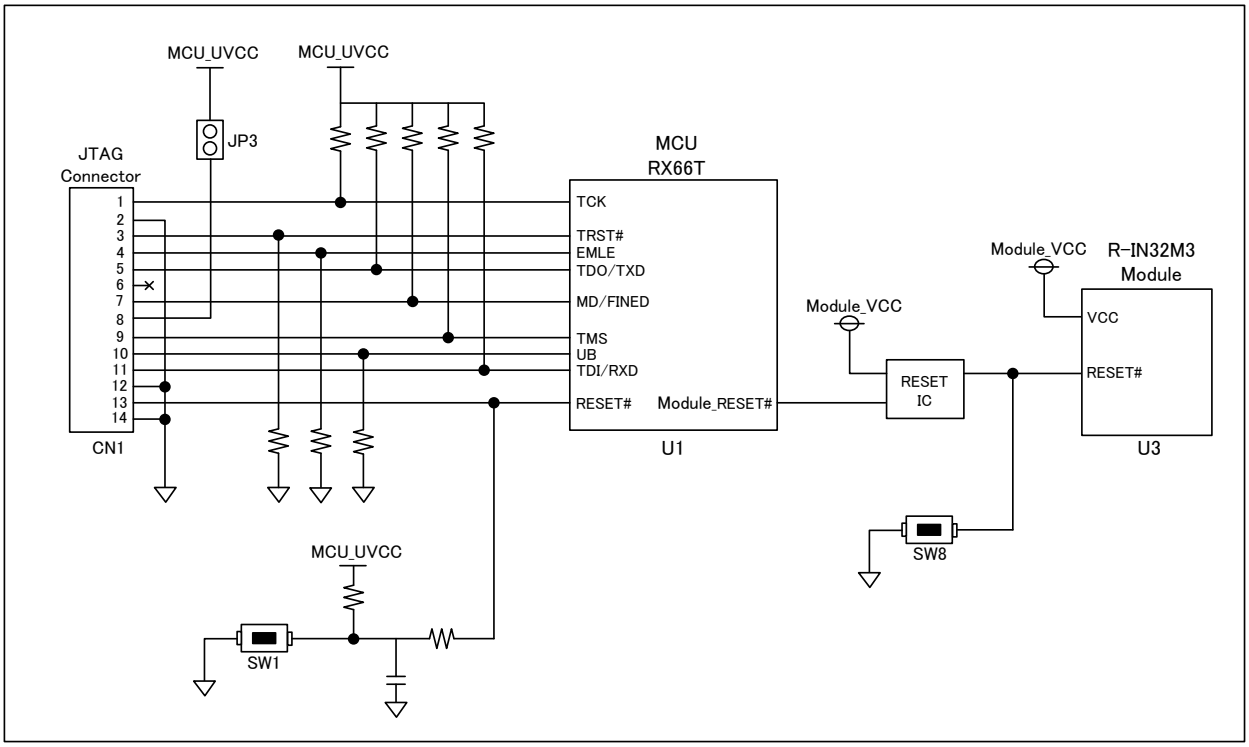

Figure 2-5 RESET diagram

2.4.4 R-IN32M3 Module

For more information about the R-IN32M3 Module mounted on this board, see the R-IN32M3 Module (RY9012A0) User's Manual Hardware (R19UH0122EJ****).

The communication between the R-IN32M3 Module and the MCU is done via 4-wire SPI.

The SPI connection is shown in Figure 2-6. Each signal line in the SPI is not processed on this board because a Pull-Up or Pull-Down resistor is granted in the R-IN32M3 Module.

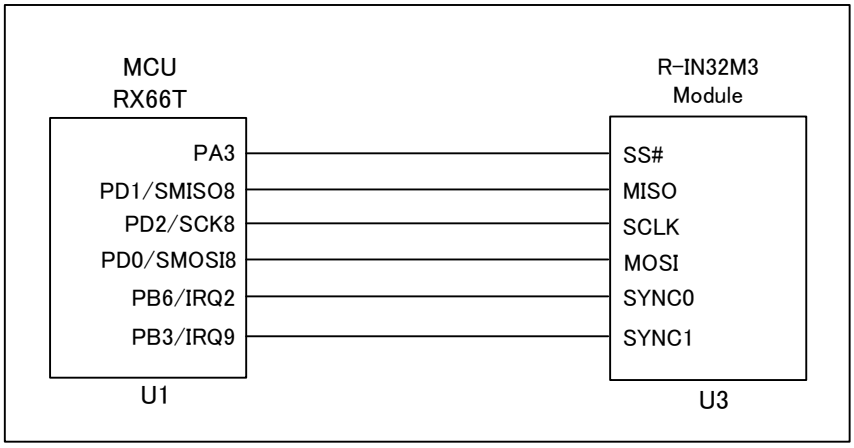

Figure 2-6 SPI

2.4.5 Emulator connection

The RX66T program is rewritten using E1 or E2 Lite, an on-chip debugging emulator from Renesas Electronics. The program is written by connecting E1 or E2 Lite to the Emulator connector on this board and the USB on PC.

Do not supply power from E1, E2 Lite in the integrated development environment.

2.5 External interfaces

2.5.1 Ethernet connector

The R-IN32M3 Module on this board has two RJ45 network connectors.

The Ethernet switch function of the R-IN32M3 allows external connections in several network topologies, such as dizzy chain connections. The internal PHY layer of R-IN32M3-EC can handle a variety of industrial communication protocols and supports 10BASE-T and 100BASE-TX/FX.

2.5.2 LED

This board is equipped with a 5V power display LED, a protocol display LED, a protocol status LED representing the status of each protocol, and a general-purpose output LED.

(1) 5V power display (LED10)

The LED10 (Red) is lit by a +5V power supply from the USB Micro B connector or the Inverter Board connector. See Figure 2-3 for the configuration.

(2) Protocol display (LED1~3)

Depending on the industrial ethernet protocol selected, the project in the sample software is executed on RX66T. Depending on the protocol running, one of LED1-3 (Green) will light up.

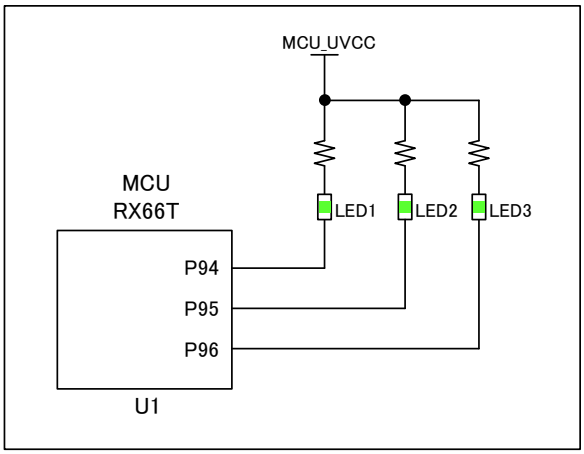

Figure 2-7 protocol display LED

LED display depending on the protocol is shown in Table 2-3.

Table 2-3 protocol display LED

protocol	LED1 (P94)	LED2 (P95)	LED3 (P96)
PROFINET	ON	OFF	OFF
EtherNet/IP	OFF	ON	OFF
EtherCAT	OFF	OFF	ON

(3) Protocol status (LED4,7)

LED4 and LED7 are Bi-Color LEDs (Green / Red) that display the LED status specified in each protocol standard.

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

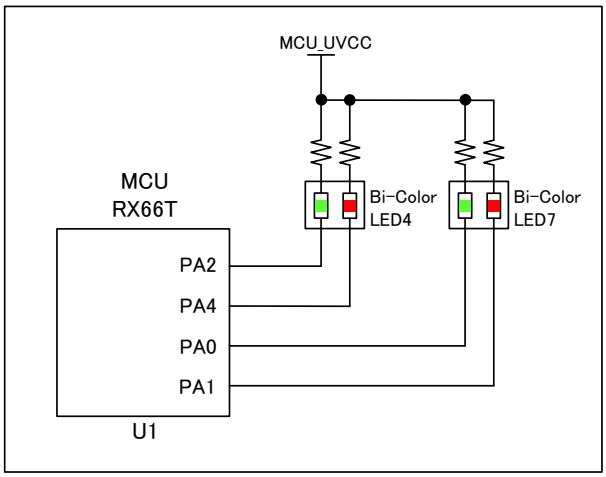

Figure 2-8 protocol status LED

Table 2-4 Protocol Status LED

Mode		LED7		LED4	
		GREEN	RED	GREEN	RED
		PA0	PA1	PA2	PA4
1	PROFINET	Connection	BF	DCP indicator (Blink)	SF
2	EtherNet/IP	NS	NS	MS	MS
3	EtherCAT	OFF	ERR	RUN	OFF

SF: system failure, BF: bus failure, DCP: discovery and configuration protocol
MS: module status indicator, NS: network status indicator,

(4) General-purpose output LED (LED5,6,8,9)

4bit green LEDs (LED5, LED6, LED8, and LED9) are available for general purpose I/O applications.

In the sample application for Remote I/O, these are used as the LED outputs.

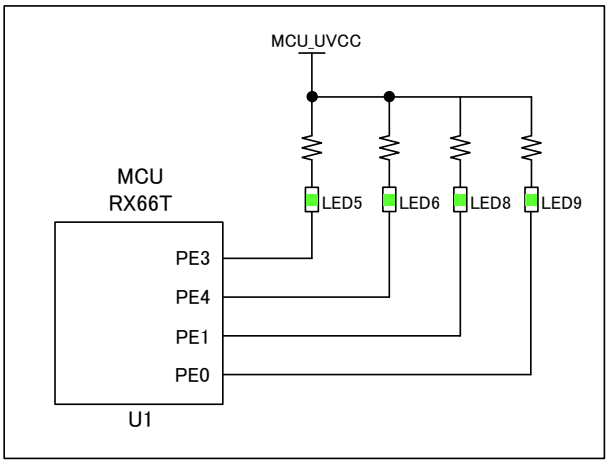

Figure 2-9 General purpose LED

2.5.3 Switch

This board has several switches for EtherCAT Explicit Device ID, input for general-purpose I/O applications, general-purpose DIP, input power, and resets.

(1) EtherCAT Explicit Device ID Switch (SW3)

When EtherCAT is selected as the protocol setting, the Explicit Device ID is set by SW3.

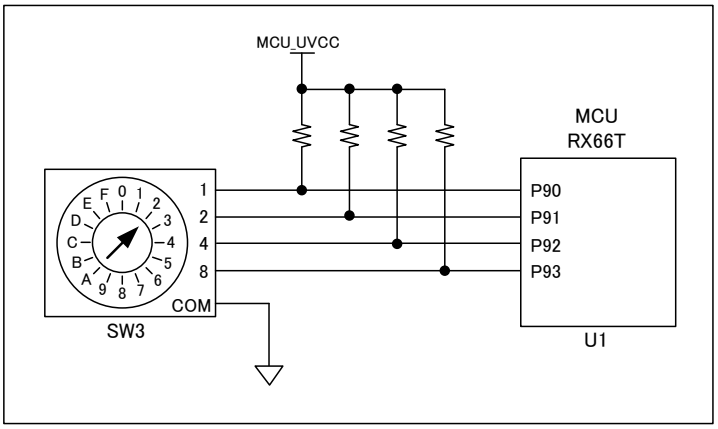

Figure 2-10 EtherCAT Explicit Device ID Switch

Table 2-5 EtherCAT ID setting

EtherCAT Device ID	SW3	P93	P92	P91	P90
0	0	0	0	0	0
1	1	0	0	0	1
2	2	0	0	1	0
3	3	0	0	1	1
4	4	0	1	0	0
-	-				
15	F	1	1	1	1

(2) General-purpose input SW (SW2,4,5,6)

SW2,4,5,6 (toggle switch) is available as an input switch for each 4bit for general purpose I/O applications. In the sample application for Remote I/O, these are used as the switch inputs.

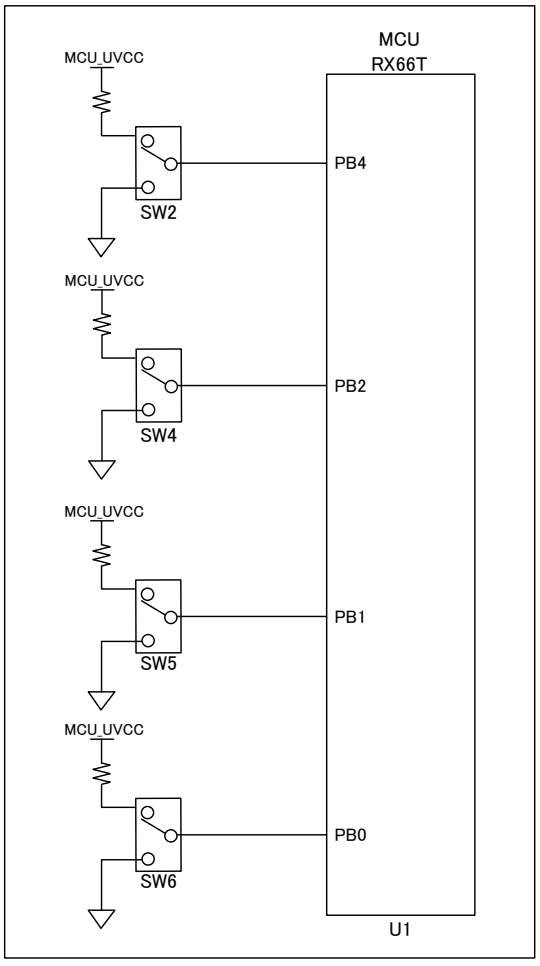

Figure 2-11 General purpose input SW

(3) General-input SW (SW9)

Figure 2-12 shows DIP switch, SW9, input for general purpose.

In the multi-protocol sample application, this is used as a selector input for setting the protocol (PROFINET, EtherNet / IP, EtherCAT and Modbus TCP).

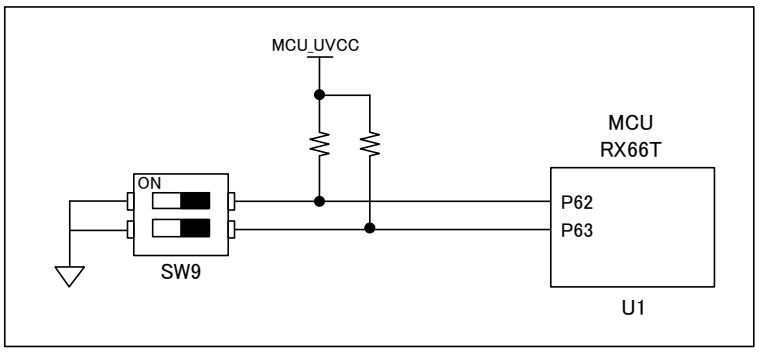

Figure 2-12 Generic SW

(4) Power SW (SW7)

SW7 is a switch that selects the 5V power supply to this board. Select from the USB Micro B connector or the Inverter Board connector to power it.

See Figure 2-3for the configuration.

Table 2-6 SW7

SW7 Select	Description
1-2 ("USB" silkscreen side)	USB connector (CN8) supplies 5V power to this board
2-3 ("Inverter Board" silkscreen side)	5V power is supplied from the Integer Board connector A (CN1).

(5) Reset SW (SW1, SW8)

SW1 is push switch to reset the RX66T microcomputer and SW8 is one to reset the R-IN32M3 module.

See Figure 2-5 for reset configuration.

2.5.4 Connector

Table 2-7 shows the connectors on this board (except for the RJ-45 connector with the R-IN32M3 Module).

Table 2-7 Connector List

CN	Type name	description
CN1	SFH11-PBPC-D10-ST-BK	Inverter Board connector A
CN2	B5B-XH-A	Hall sensor connector
CN3	B5B-XH-A	Encoder connector
CN4	SFH11-PBPC-D10-ST-BK	Inverter Board connector B
CN5	XG4C-1431	JTAG connector
CN6	B4B-XH-A (not mounted)	SCI connector (not mounted)
CN7	A1-16PA-2.54DSA (not mounted)	Pin Headers for External Extensions(not mounted)
CN8	10118194-0001LF	USB Micro B

(1) Inverter Board Connector (CN1, CN4)

This board is equipped with an Inventor Board connector for use as a CPU card for "24V Motor Control Evaluation System". The following is a pin assignment for the Integer Board connector:

Table 2-8 Inverter Board Connector A (CN1)

pin	signal	MCU Port	Note	pin	signal	MCU Port	Note
1	LED1#	P22		2	LED2#	P21	
3	LED3#	P20		4	VRL	P24	
5	FO#	P70		6	NC	-	NC
7	WN	P76/ MTIOC4D		8	VN	P75/ MTIOC4C	
9	UN	P74/ MTIOC3D		10	WP	P73/ MTIOC4B	
11	VP	P72/ MTIOC4A		12	UP	P71/ MTIOC3B	
13	SW1#	P23		14	SW2#	P27	
15	+5VA1	-		16	+5VA2	-	
17	GND	-		18	GND	-	
19	VCC33_A1	-		20	VCC33_A2	-	

Table 2-9 Inverter Board Connector B (CN4)

pin	signal	MCU Port	Note	pin	signal	MCU Port	Note
1	AVCC1	-		2	AVCC2	-	
3	NC	-	NC	4	PGAVSS1	PH0/ PGAVSS1	
5	IU	P40/AN000		6	IV	P41/AN001	
7	IW	P42/AN002		8	VPN	P43/AN003	
9	TEMP(VOT)	-	NC	10	VU	P52 /AN200	
11	VV	P53/AN201		12	VW	P54/ AN202	
13	VAC	-	NC	14	IPFC	-	NC
15	VR1	PH4/AN107		16	VN	-	NC
17	VCCIO1	-		18	VCCIO2	-	
19	GND1	-		20	GND2	-	

(2) Hall sensor connector (CN2)

The Hall sensor connector (CN2) on this board is only compatible with 5V Hall sensors.

Because 5V is connected to the VCC pin, connecting the Hall sensor of 3.3V might be broken.

The signal line converts the 5V input signal to 3.3V by the level conversion IC on the board and connects it to the MCU.

(This sample software is not supported.)

The pin assignment of the Hall sensor connector is shown below.

Table 2-10 Hall sensor connector

pin	signal	MCU Port	Note
1	VCC	-	5V output
2	GND	-	
3	HALL_U	P61/ IRQ5	Converted to 3.3V on this board
4	HALL_V	P60/IRQ4	Converted to 3.3V on this board
5	HALL_W	P55/IRQ3	Converted to 3.3V on this board

(3) Encoder connector (CN3)

The Encoder connector on this board is only compatible with Encoder with 5V operation.

Because 5V is connected to the VCC pin, connecting encoder of 3.3V might be broken.

The signal line converts the 5V input signal to 3.3V by the level conversion IC on the board and connects it to the MCU.

(This sample software is not supported.)

The pin assignments for the Encoder connector are as follows:

Table 2-11 Encoder connector (CN3)

pin	signal	MCU Port	Note
1	VCC	-	5V output
2	GND	-	
3	ENC_A	P33/MTCLKA	Converted to 3.3V on this board
4	ENC_B	P32/MTCLKB	Converted to 3.3V on this board
5	ENC_Z	PA5/MTIOC1A	Converted to 3.3V on this board

(4) JTAG Connector (CN5)

The pin assignments for JTAG connector is shown below.

Table 2-12 JTAG Connector (CN5)

pin	signal	MCU Port	Note	pin	signal	MCU Port	Note
1	TCK/FINEC	PD4/TCK		2	GND	-	
3	TRST#	PD7/TRST#		4	EMLE	EMLE	
5	TDO /TXD1	PD3/TXD1		6	NC	-	NC
7	MD/FINED	MD/FINED		8	VCC	-	
9	TMS	PD6_TMS		10	UB	P00_UB	
11	TDI/RXD1	PD5_RXD1		12	GND	-	
13	RESET#	RESET#		14	GND	-	

(5) SCI Connector (CN6 : Not Mounted)

The pin assignments for SCI connector, CN6 is shown below.

CN6 is not mounded on this board.

Table 2-13 SCI Connector (CN6)

pin	signal	MCU Port
1	VCC	-
2	TXD	P81/TXD6
3	RXD	P80/RXD6
4	GND	-

(6) Connectors for external expansion (CN7 : Not Mounted)

The external expansion connector (CN7) has an unused pin connected to the MCU.

The pin assignments for external expansion connectors are shown below. CN7 connectors (pin headers) are not implemented.

Table 2-14 Connectors for external extension (CN7)

pin	signal	MCU Port	Note	pin	signal	MCU Port	Note
1	MCU_UVCC	-		2	MCU_UVCC	-	
3	MCU_PE5	PE5		4	MCU_P01	P01	
5	MCU_PE2	PE2		6	MCU_P10	P10	
7	MCU_P11	P11		8	MCU_P82	P82	
9	MCU_P44	P44		10	MCU_P45	P45	
11	MCU_P46	P46		12	MCU_P47	P47	
13	MCU_PB7	PB7		14	GND	-	
15	GND	-		16	GND	-	

(7) USB micro B (CN8)

CN8 is USB Micro B connector for 5V power supply to this board. See Figure 2-3 for the power supply diagram. By switching SW7 to the one described as "USB" with silkscreen, 5V power is supplied from CN8 to the board.

2.5.5 Jumper

Below is a list of jumper pins on this board. Normally, please use it in the shipping state.

Table 2-15 List of Jumper pins

pin	signal	Feature
JP2	XJ8C-0211	Select connection of PGAVSS0 and AVSS
JP3	XJ8C-0211	Select connection of JTAG_VCC and MCU_UVCC
JP4	XJ8D-0311	Select input of MCU_UVCC
JP5	XJ8D-0311	Select input of Module_VCC

The settings for each jumper pin are shown below.

(1) JP2

JP2 is a jumper pin for connecting the PGAVSS0 pin and AVSS of the MCU. (See Figure 2-4)

The configuration table of JP2 is shown below.

When using the inverter board, please use JP2 of this board while keeping it short (shipped state).

Table 2-16 JP2

JP	Description	shipped state
1-2	Connect PGAVSS0 terminal and AVSS # Please use this normally.	○
not connect	Do not connect PGAVSS0 terminal and AVSS	

(2) JP3

JP3 is a jumper pin for connecting the VCC pin of JTAG connector (CCN5) to the MCU_UVCC. (See Figure 2-5). The configuration table of JP3 is shown below.

Table 2-17 JP3

JP	Description	shipped state
1-2	Connect VCC pin and MCU_UVCC of the JTAG connector. # Please use this normally.	○
not connect	Do not connect VCC pin and MCU_UVCC of the JTAG connector.	

(3) JP4

JP4 is jumper to select MCU_UVCC input. (See Figure 2-3)

The configuration table of JP4 is shown below.

Table 2-18 JP4

JP	Description	shipped state
1-2	Use 3.3V LDO output on this board as MCU_UVCC input # Please use this normally.。	○
2-3	Use VCC33_A of Inverter Board Connector A as MCU_UVCC.	
not connect	There is no power to the MCU. # Select this setting in case of power supply from JTAG is required.	

(4) JP5

JP5 is jumper to select Module_VCC input. (See Figure 2-3)

The configuration table of JP5 is shown below.

Table 2-19 JP5

JP	Description	shipped state
1-2	3.3V LDO output on this board is selected as Module_VCC # Please use this normally.	○
2-3	VCC33_A from Inverter Board connector A is selected as Module_VCC.	
not connect	No power input to R-IN32M3 Module	

2.6 Difference between RX66T CPU Card

This board is developed referring to the RX66T CPU card (RTK0EMX870C00000BJ) and is configured to add an industrial Ethernet communication module, but there are several differences in MCU and the peripheral circuits. For this reason, in order for the sample software for the original RX66T CPU card to work with this board, it is necessary to make changes that take into account the following hardware differences.

Table 2-20 shows the major differences between original RX55T CPU card and this board about MPU peripheral circuits and R-IN32M3 Module-related circuit.

Table 2-20 Differences b/w RX66T CPU card and this board

#	Item		This board (SEMB1320)	RX66T CPU Card RTK0EMX870C00000BJ	Connection	Note
1	MPU	Type Name	R5F566TKADFP	R5F566TEADFP		
2		ROM	1024KByte	512KByte		
3		RAM	128KByte	64KByte		
4		Package	LFQFP / 100 pin	The same on the left		
5		Operating power supply voltage	3.3V (MPU spec: 2.7~5.5V)	5V (MPU spec: 2.7~5.5V)		
6	Encoder Input	ENCA	58pin (P33 / MTCLKA)	The same on the left	CN3	
7		ENCB	59pin (P32 / MTCLKB)	The same on the left	CN3	
8		ENCZ	36pin (PA5 / MTIOC1A)	The same on the left	CN3	
9	Hall sensor input	HU	76pin (P61 / IRQ5)	17pin (PE0 / IRQ7)	CN2	
10		HV	77pin (P60 / IRQ4)	16pin (PE1 / IRQ15)	CN2	
11		HW	78pin (P55 / IRQ3)	1pin (PE5 / IEQ0)	CN2	
12	Volume in	VR_1	86pin (PH4 / AN107)	68pin (P21 / AN217)	CN4	Need Volt Conv (*1)
13	DC Link Volt detect	VPN	87pin (P43 / AN003)	75pin (P62 / AN208)	CN4	
14	waveform monitor tool	TXD6	97pin (P81 / TXD6)	35pin (PB0 / TXD6)	CN6	Unusable
15		RXD6	98pin (P80 / RXD6)	34pin (PB1 / RXD6)	CN6	Unusable
16	Inverter board interface	SW2#	64pin (P27)	97pin (P81)	CN1	error release
17		SW1#	66pin (P23)	98pin (P80)	CN1	enable motor control
18		LED1#	67pin (P22)	9pin (PE3)	CN1	Normal
19		LED2#	68pin (P21)	26pin (PB7)	CN1	Error
20		LED3#	69pin (P20)	32pin (PB3)	CN1	

(*) By changing the input supply voltage from 5V to 3.3V, the voltage loading process of sample software must be corrected. Here, this sample software is already corrected.

3. Sample software configuration

R-IN32M3 Module sample software can be downloaded 1.2 Operating environment.

The sample software is compatible with Little-endian. For Big-endian support, refer to 4.4 Big-endian support and create a project.

3.1 Folder structure

The folder structure of this sample software is shown below.

RX66T_uCCM_V***	
├──appl	User application
│ ├──01_pnio	PROFINET sample application
│ ├──02_eip	EtherNet/IP sample application
│ ├──03_ecat	EtherCAT sample application
│ ├──04_pnio_largesize	PROFINET Large data size sample application
│ ├──05_eip_largesize	EtherNet/IP Large data size sample application
│ ├──06_ecat_largesize	EtherCAT Large data size sample application
│ ├──07_modbus_tcp_slave	Modbus TCP sample application
│ ├──10_multi_protocol	multi-protocol [01_pnio, 02_eip, 03_ecat, 07_modbus] sample application
│ ├──11_pnio_http	01_pnio sample Enhanced [web saver and host MCU update function]
│ ├──12_eip_http	02_eip sample Enhanced [web saver and host MCU update function]
│ ├──13_ecat_http	03_ecat sample Enhanced [web saver and host MCU update function]
│ └──17_fwup_bootloader	Boot Loader for host MCU update function
│	
├──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 Sample project Overview

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

Table 3-1 Protocol and feature

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

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

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

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

- ✓ 04_pnio_largesize, 05_eip_largesize, 06_ecat_largesize project has a sample project for large data transfer using RPC communication. See “User's Implementation Guide (uGOAL Edition) [R30AN0402EJ****]” for details on RPC communication.
- ✓ This document only describes the data communication function. For details about the firmware update function, please refer to the "R-IN32M3 Module (RY9012A0) Firmware Update Guide (R30AN0401EJ****)".

3.3 Setup 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 RX

Download e2studio in the following web site and install it on your PC.

Download Link: 1.2 Operating environment

There are four points to note regarding the installation.

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

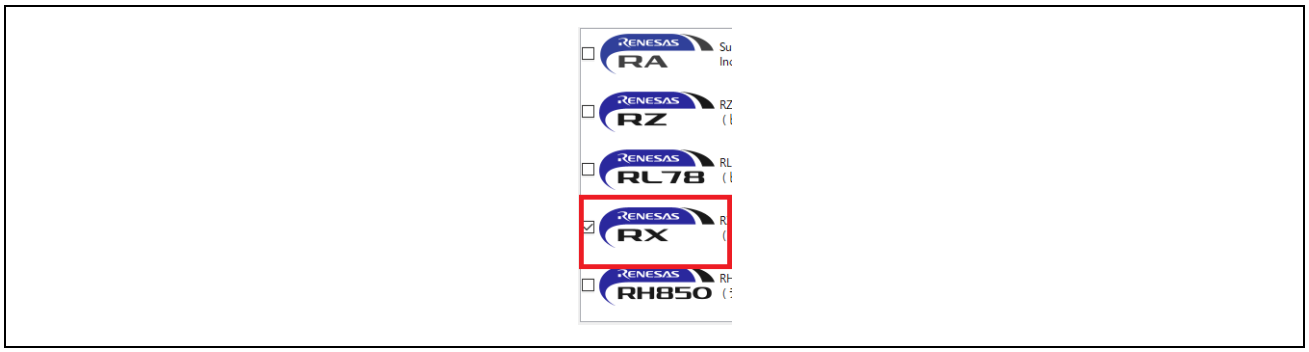

Figure 3-1 Select device family

2. Select target compiler from “GCC Toolchains & Utilities” category in [Additional Software] screen during the installation.

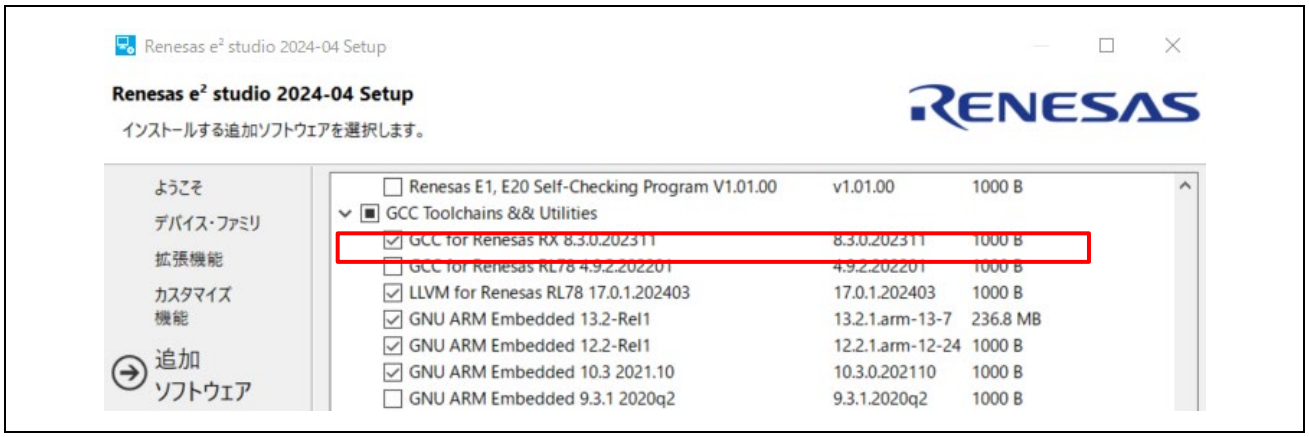

Figure 3-2 Select additional software

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).

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.

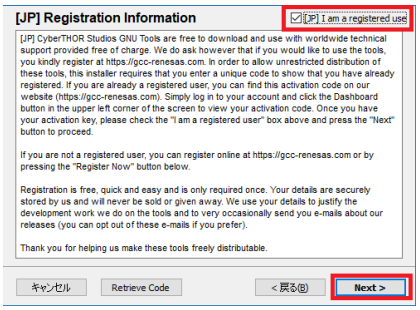

Figure 3-4 Installation of GCC compiler (account possession)

(2) CC-RX

If you use CC-RX, the C/C++ compiler package for the RX family.

When installing e2studio, select and install the version listed in Table 1-1. If the relevant version is not included, please refer to Chapter 4.7 for individual installation.

e2strudio install:

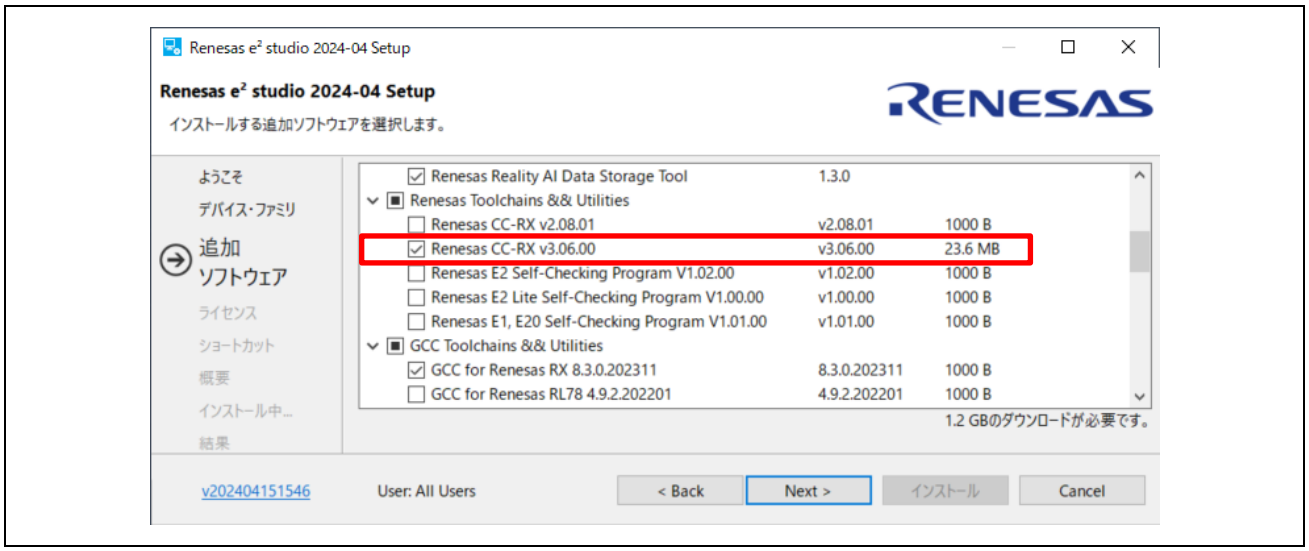

Figure 3-5 Installation CC-RX

Because it is a free trial version, the license has an expiration date and is a 60-day trial version. After 60 days, the ROM size will be limited to 128Kbyte. For more information, please refer to “CC-RX Compiler User's Manual”.

3.3.2 Development system structure

Connect this board to the E1 emulator and your PC as follows:

After setting the power input switch SW7 of this board to the "USB" side (yellow arrow in Figure 3-6), power is supplied to this board by connecting a USB micro B cable to this board.

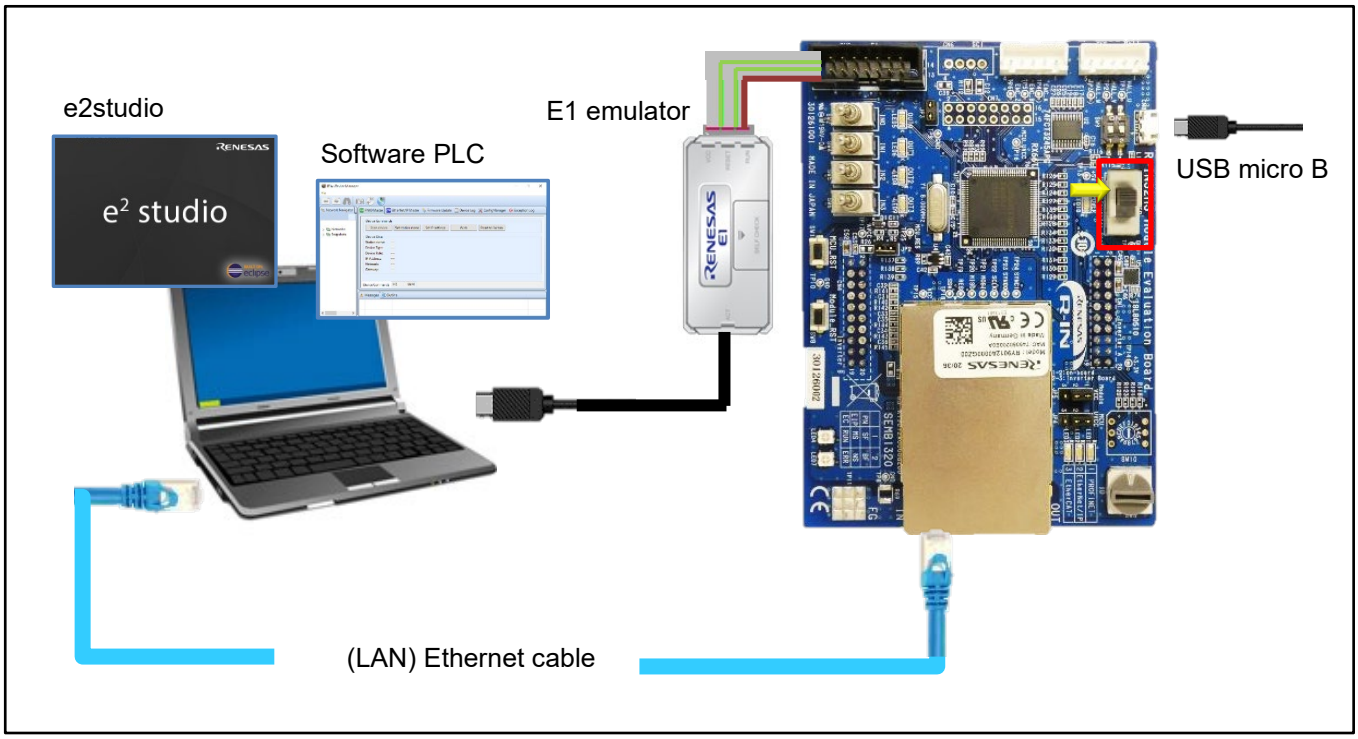

Figure 3-6 Connection of one board operation

3.3.3 Import project

(1) Unzip package

First, unzip the archived package of this sample software (RX66T_uCCM_V***.zip) and store it in arbitrary folder. Because e2studio cannot recognize project properly if file path is too long in the folder hierarchy, place it in shorter path. Also, do not use multi-byte character, such as Japanese, in the folder path.

(2) Execute e2studio

Execute "e2studio.exe" to start e2studio.

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

This note is described based on the use of GCC 8.3.0.202311. If you don't find it on the dialog here, you may need to download and install it individually. Go to GNU Tools https://llvm-gcc-renesas.com/, you can pick the intended GCC package from the list on [PRODUCTS] tab > [RX] > [Download Toolchain]. For details, see chapter 4.5.

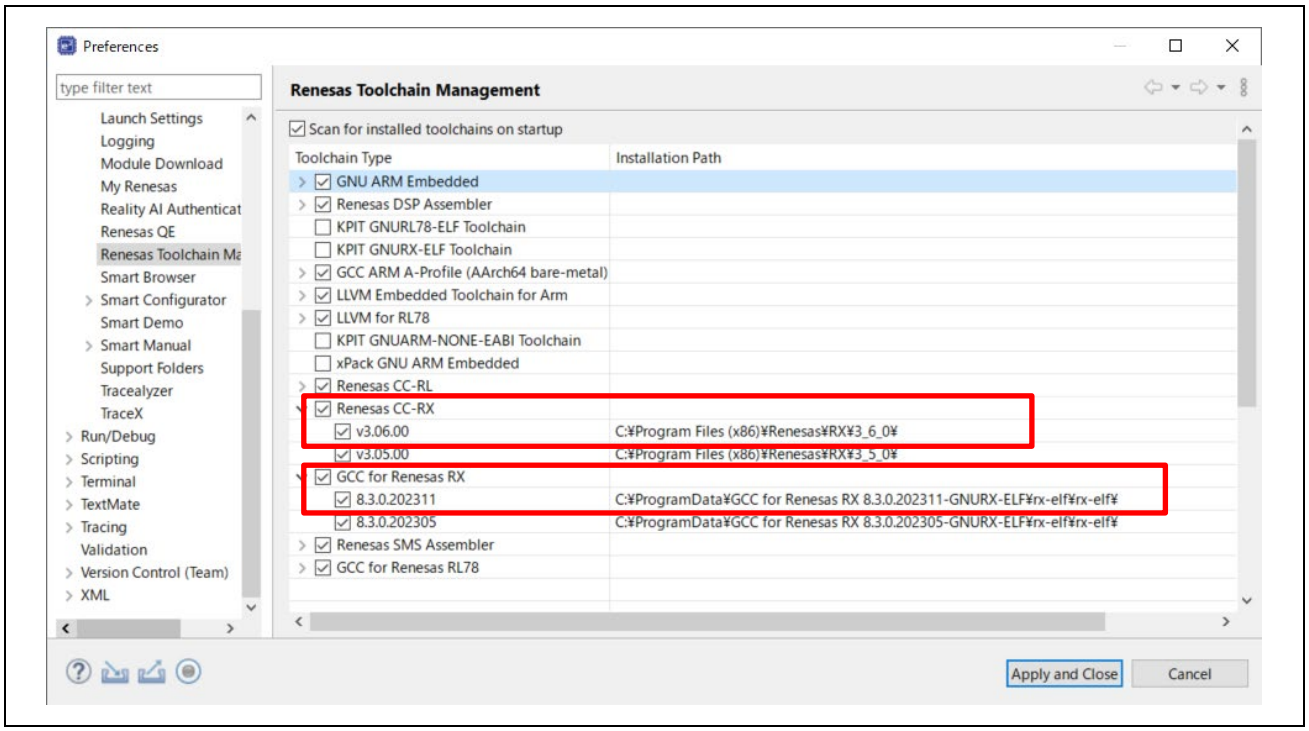

Figure 3-7 Renesas Toolchain Management

(3) Import project

Import the sample project into e2studio from the following steps:

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

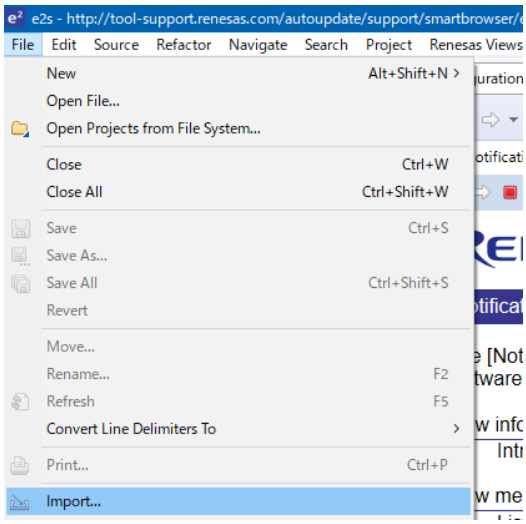

Figure 3-8 Import

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

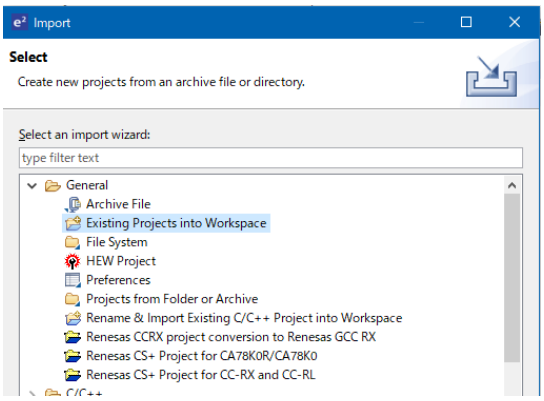

Figure 3-9 Select

In the [Import Projects] dialog, select [Select root directory] check box, and then select [Browse]. Select the package of this sample software "RX66T_uCCM_V***" stored in arbitrary folder at 3.3.3(1) and select [OK].

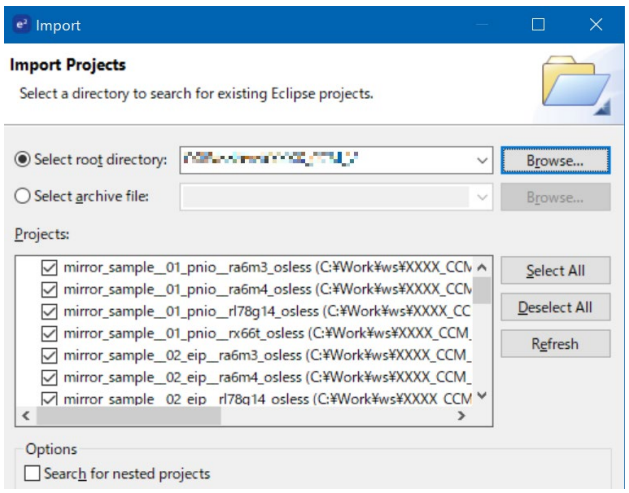

Figure 3-10 Import Projects

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

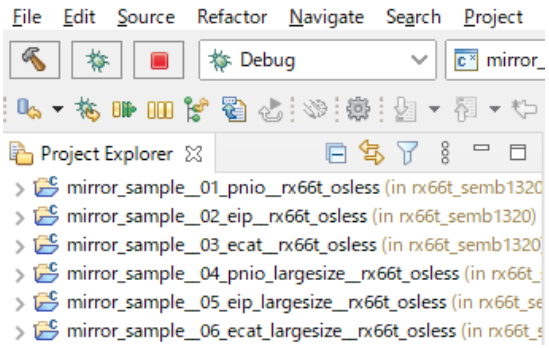

Figure 3-11 Imported projects

3.3.4 FIT Module

(1) Download FIT Module

Sample projects in this sample software apply the FIT module. Download the FIT module from Smart Configurator in e2studio. If every project uses the same version of FIT module, the download should be carried out only once on the PC. (It is not necessary to carry out it for each project after that.)

If you have already downloaded it, please go to the next section. For more information, see "RX Smart Configurator User's Guide: e² studio" (R20AN0451ES****).

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

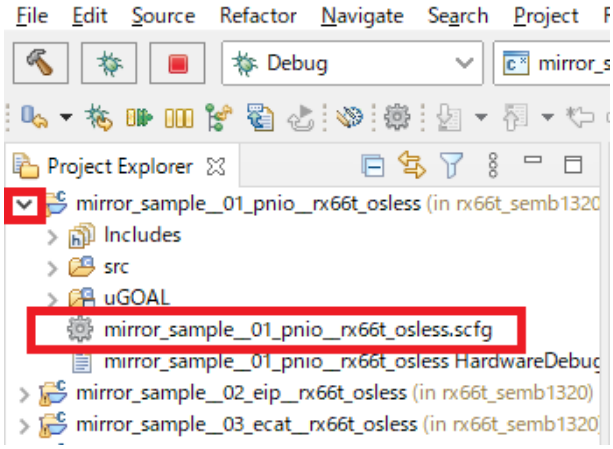

Figure 3-12 Project Explore

If you see a dialog below, select [Open Perspective].

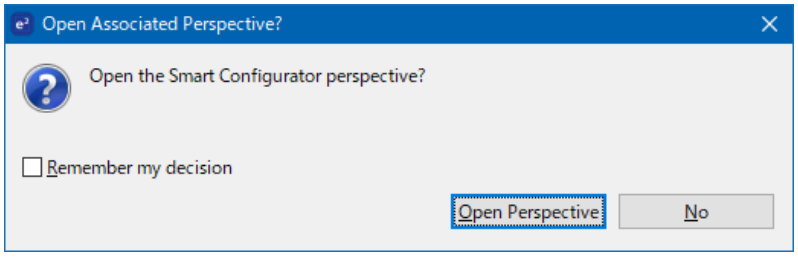

Figure 3-13 Open Perspective

Go to the [Components] tab of the Smart Configurator perspective screen and select the [Add component] button.

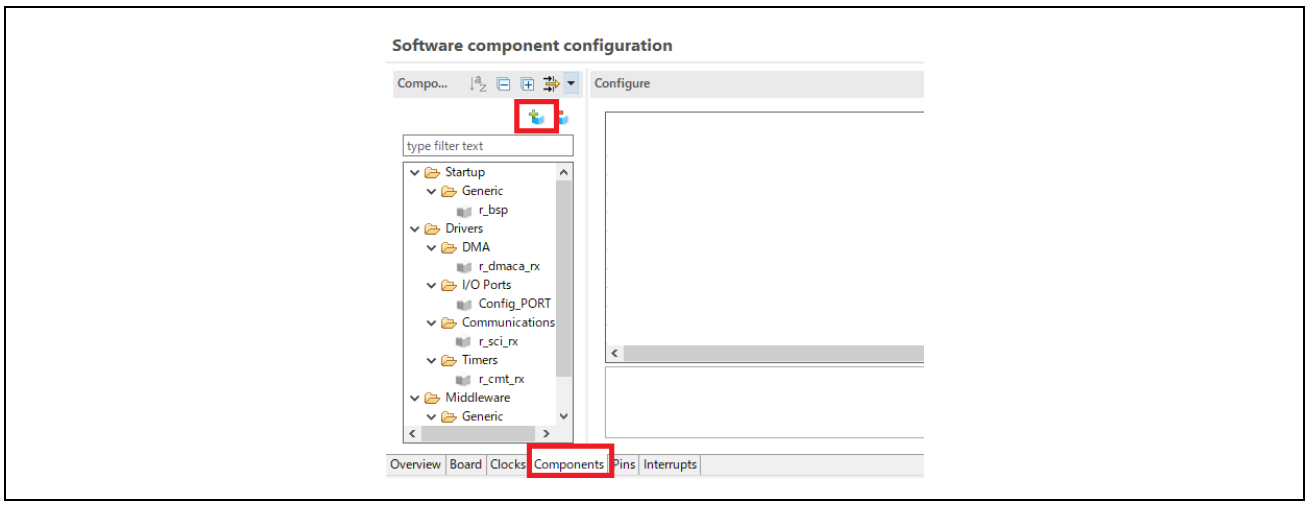

Figure 3-14 Add component

In the [Software Component Selection] dialog, select [Download more software components]. If the region selection dialog appears, select your region.

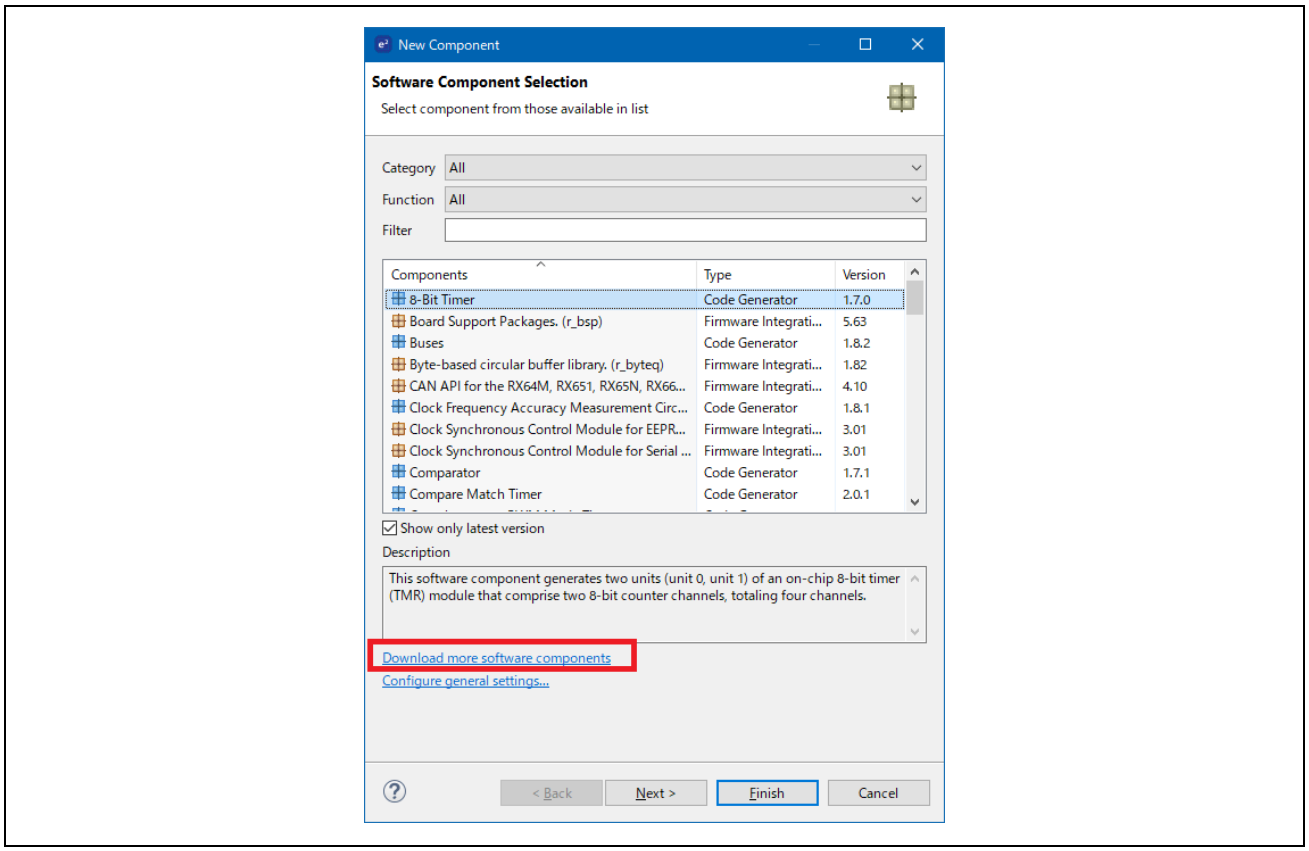

Figure 3-15 Download selection of Software Component

After checking "RX Family Driver Package", select Download. If the license dialog appears, select the [Accept] button. RX Driver Package Ver.1.42 should be selected. If you don't have it on the dialog here, you may need to download/install it individually. For details, see chapter 4.6.

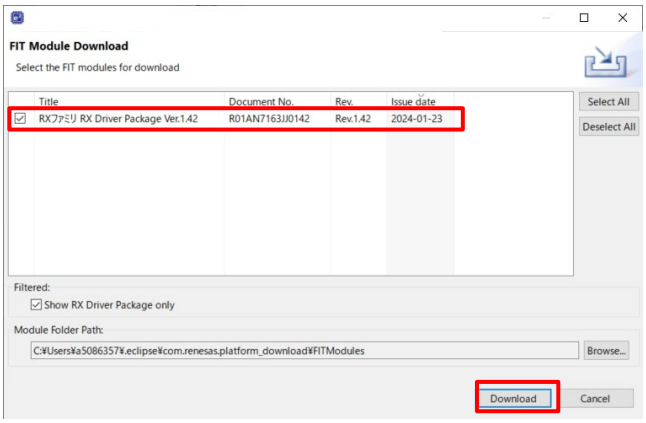

Figure 3-16 FIT Module Download

After the download is complete, close the [Software Component Selection] dialog. If successfully downloaded, each element of the FIT module will be activated.

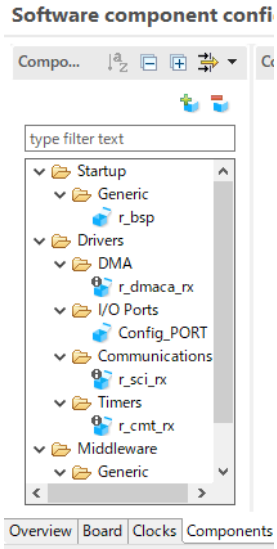

Figure 3-17 Activated FIT module

If each component is not activated, or if each component is not latest version, go to the [Components] tab of the Smart Configurator perspective screen and select the [Add component] button (Figure 3-14). So that, in the [Software Component Selection] dialog, select [Configure general settings…].

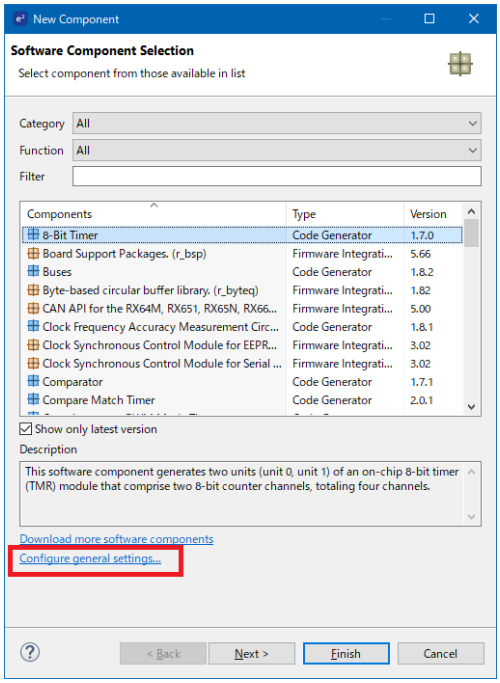

Figure 3-18 General settings of Software Component

In component settings, check [Allow blocked FIT modules to be displayed] and select [Apply].

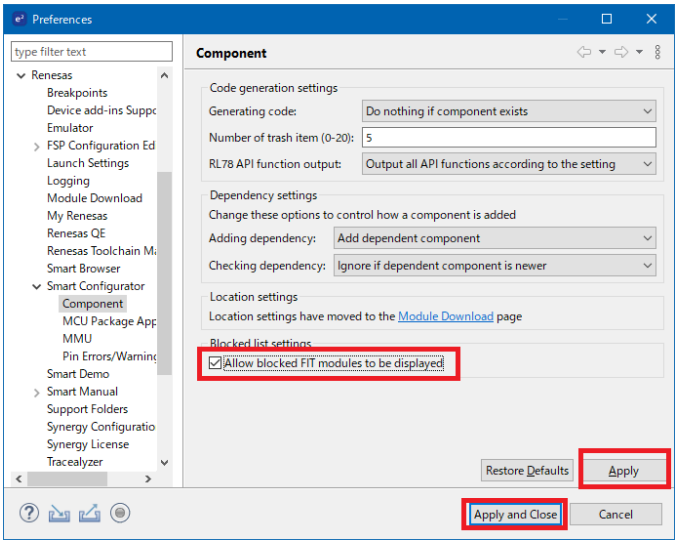

Figure 3-19 Check of component display setting

Right-click each component and select [Change version…].

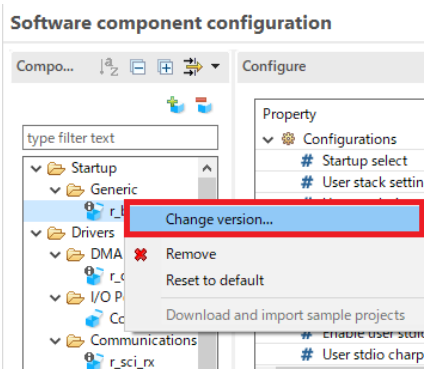

Figure 3-20 Change version of component

Select [Next] -> [Finish].

If there is latest version in available versions, select [Next] -> [Finish] after selecting its version.

(2) Code generation for FIT Module

Generate the source code of the FIT module to use. Code generation needs to be done on a project-by-project level. Go to [Overview] tab of the Smart Configurator perspective screen and select [Generate Code] button to generate the required source code.

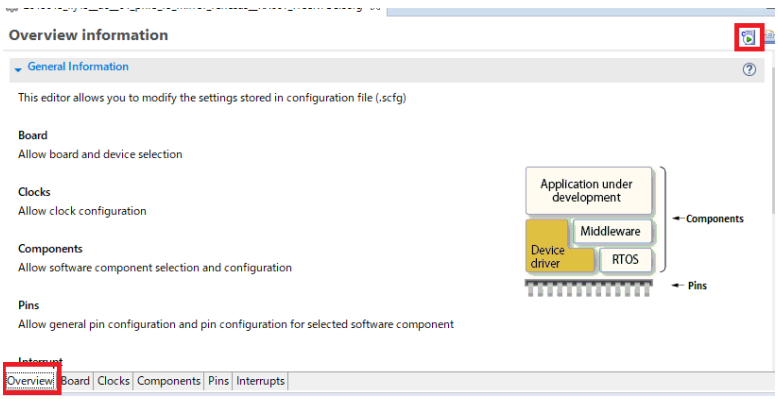

Figure 3-21 Generate code

Now, it is ready to build the project.

3.3.5 Build project

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

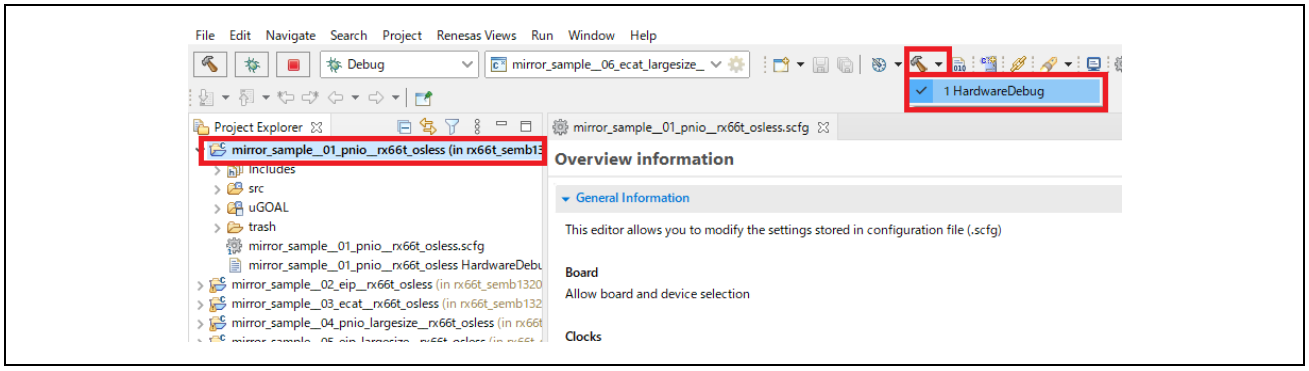

Figure 3-22 Build project

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

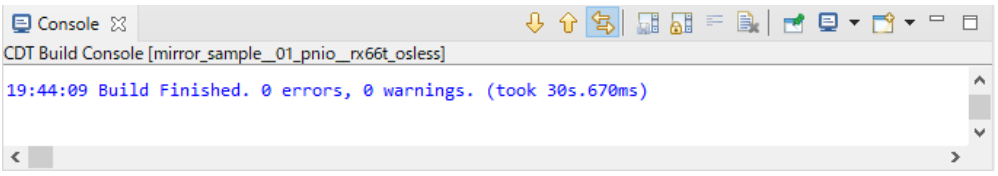

Figure 3-23 Build finished

3.3.6 Debug

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

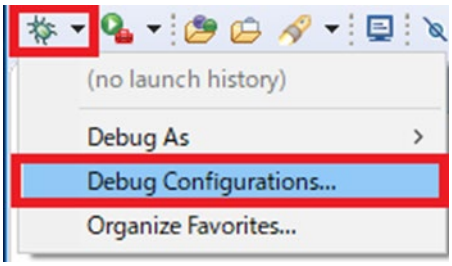

Figure 3-24 Debug Configurations

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.

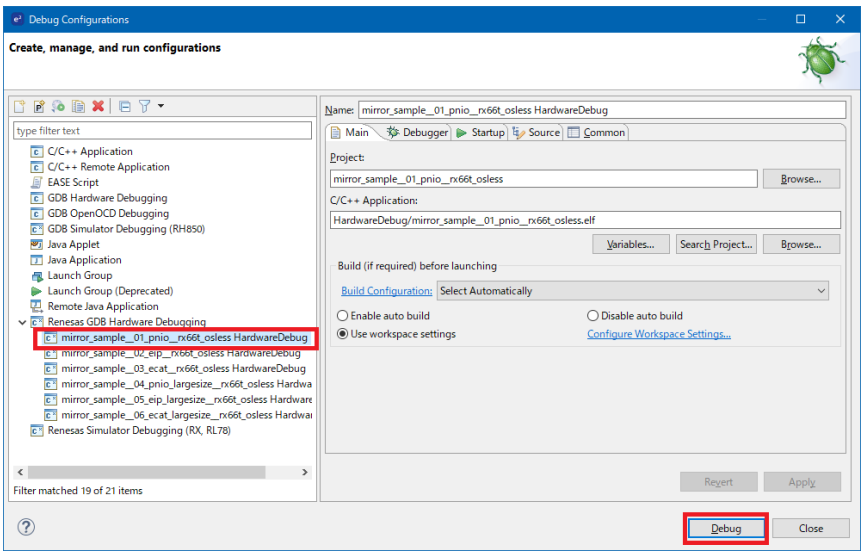

Figure 3-25 Debug start

When using E2 Lite emulator, open [Debugger] tab and select the emulator to use at [Debug hardware]. Here, attention to set [Power Target from The Emulator (MAX 200mA)] to "No".

After changing [Debugger] configuration, select [Apply] and [Debug].

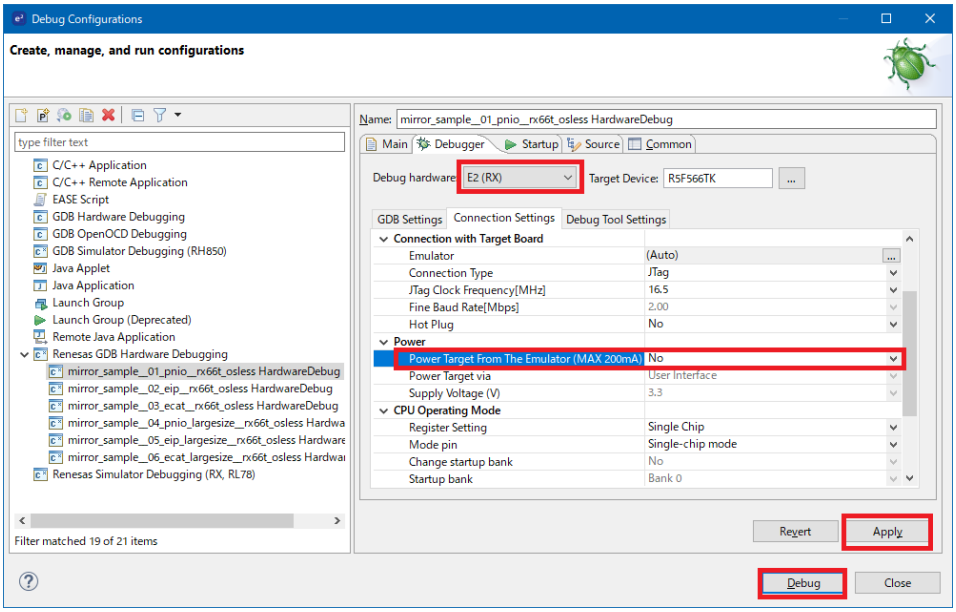

Figure 3-26 Selecting Debug hardware

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

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

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

3.4 Protocol communication and Application control

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

3.4.1 PROFINET

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

Table 3-2 PROFINET Sample software

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

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

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.4 to 3.3.6. When the sample application is run, the protocol display LED (LED1: PROFINET) turn on.

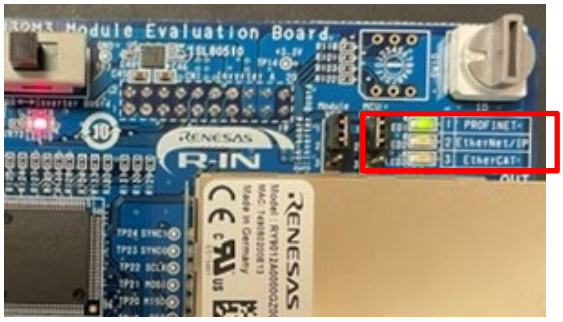

Figure 3-27 Protocol LED: PROFINET

-2. Set IP address

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

IP address	192.168.0.1
Netmask	255.255.255.0

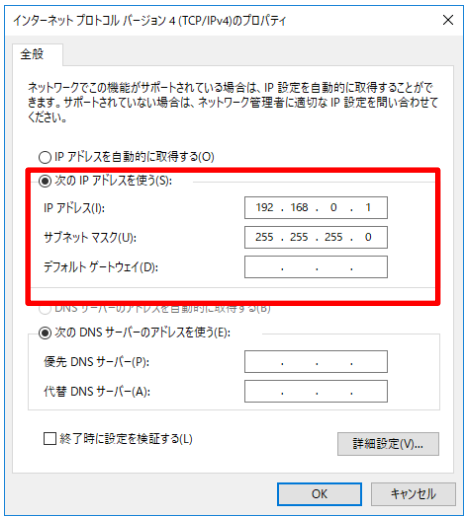

Figure 3-28 Set Static IP address

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.

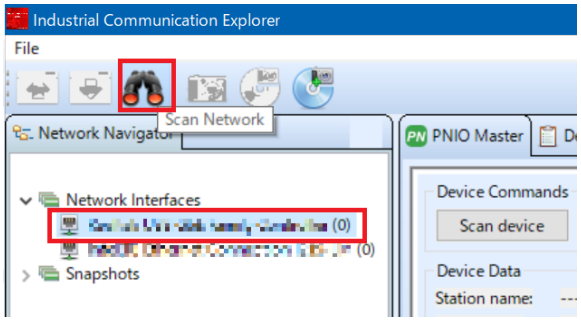

Figure 3-29 Scan network

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

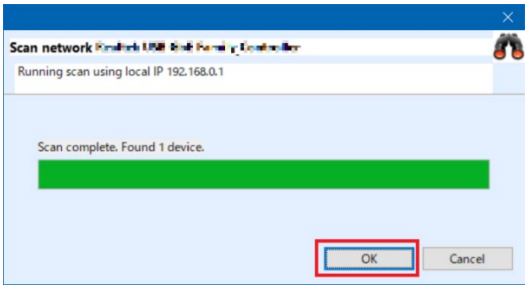

Figure 3-30 Scan completed

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

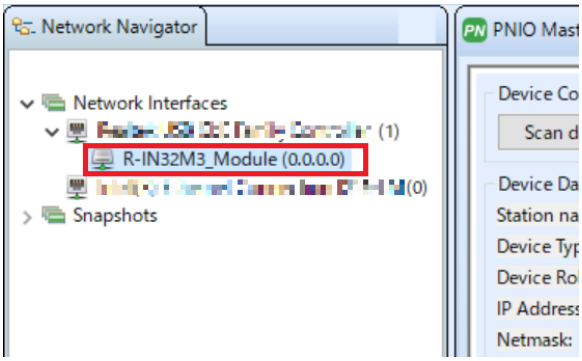

Figure 3-31 Select R-IN32M3 Module

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

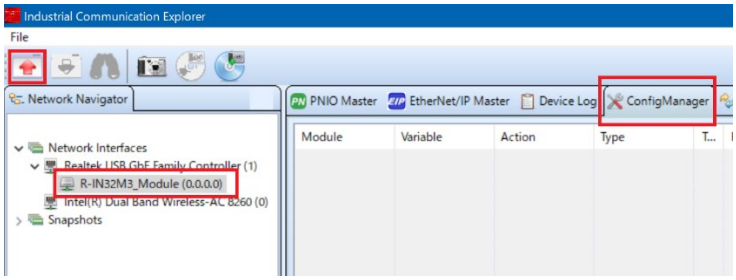

Figure 3-32 ConfigManager

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

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

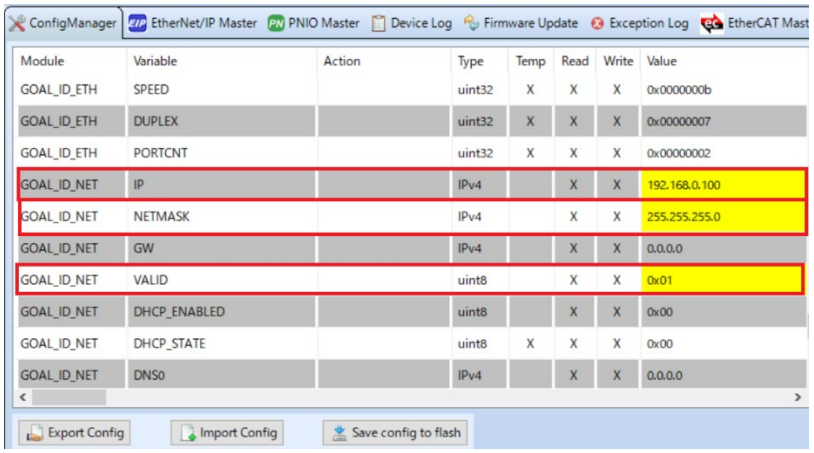

Figure 3-33 Set IP address on R-IN32M3 module

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

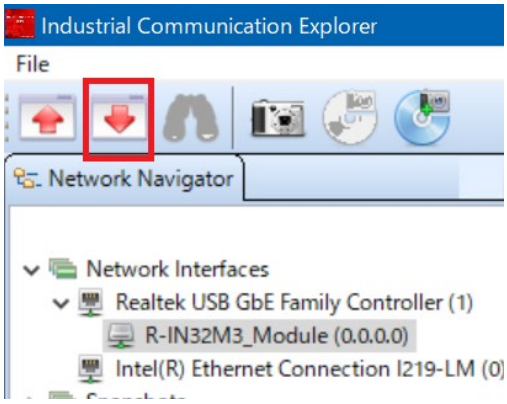

Figure 3-34 Download Config variables

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

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

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

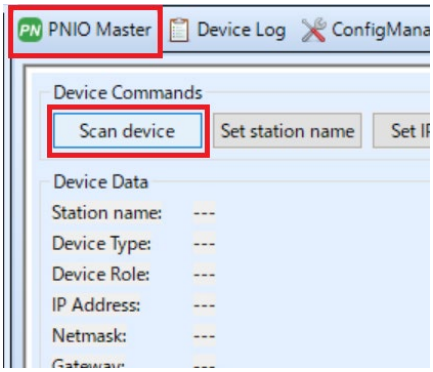

Figure 3-35 PNIO Master

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

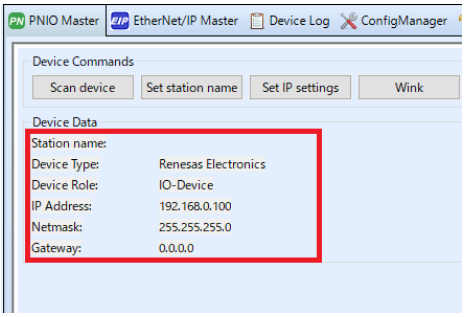

Figure 3-36 Device Data

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

Table 3-3 GSDML Files

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

Verify that [Slots:] and [Modules] display contents as set in GSDML, select [32] from pull-down of [Device Interval] then push [Connect] button. If the connection is successful, this button switches to [Disconnect] button. In addition, the protocol status LED (LED4) lights up.

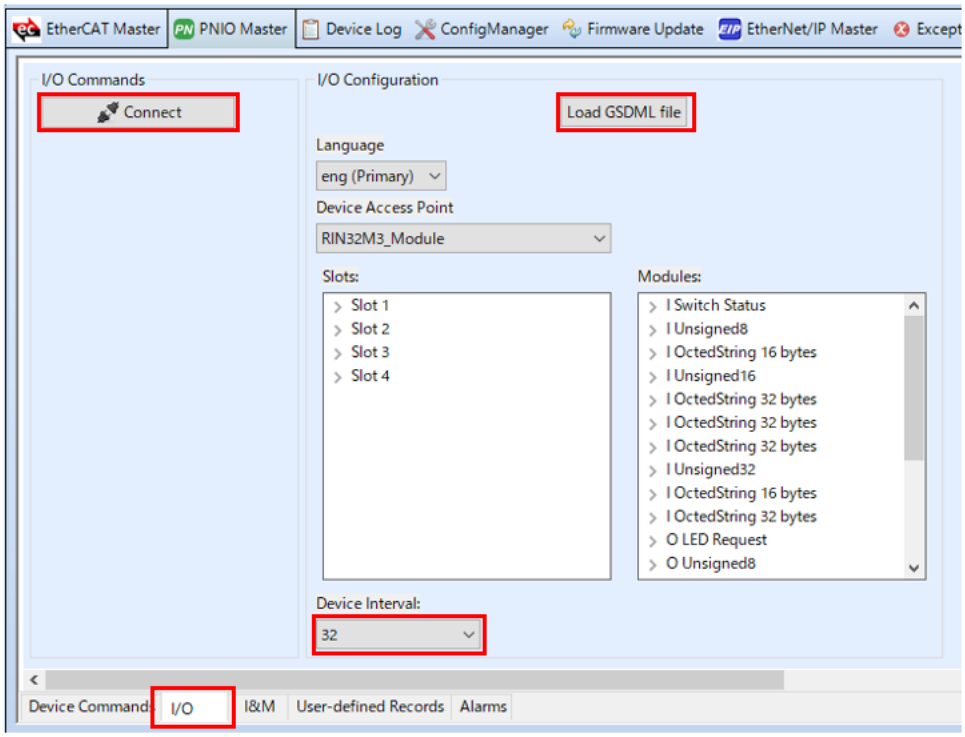

Figure 3-37 GSDML

-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 projects: 01_pnio, 04_pnio_largesize, (10_multi_protocol), 11_pnio_http

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

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

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

  Target projects: 04_pnio_largesize

Table 3-4 Application defied:

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

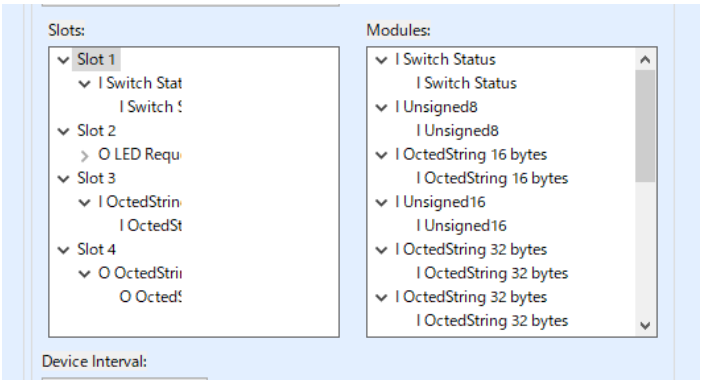

Figure 3-38 Application define (ex. 01_pnio)

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 app.		Remote I/O control
Switch (Slot 1)	SW2, 4, 5, 6	Input Data value changes by operating general-purpose switches SW2 : bit0 SW4 : bit1 SW5 : bit2 SW6 : bit3
LED (Slot 2)	LED5, 6, 8, 9	General-purpose output LED changes by registering a value to Output Data. bit0 : LD5 bit1 : LD6 bit2 : LD8 bit3 : LD9

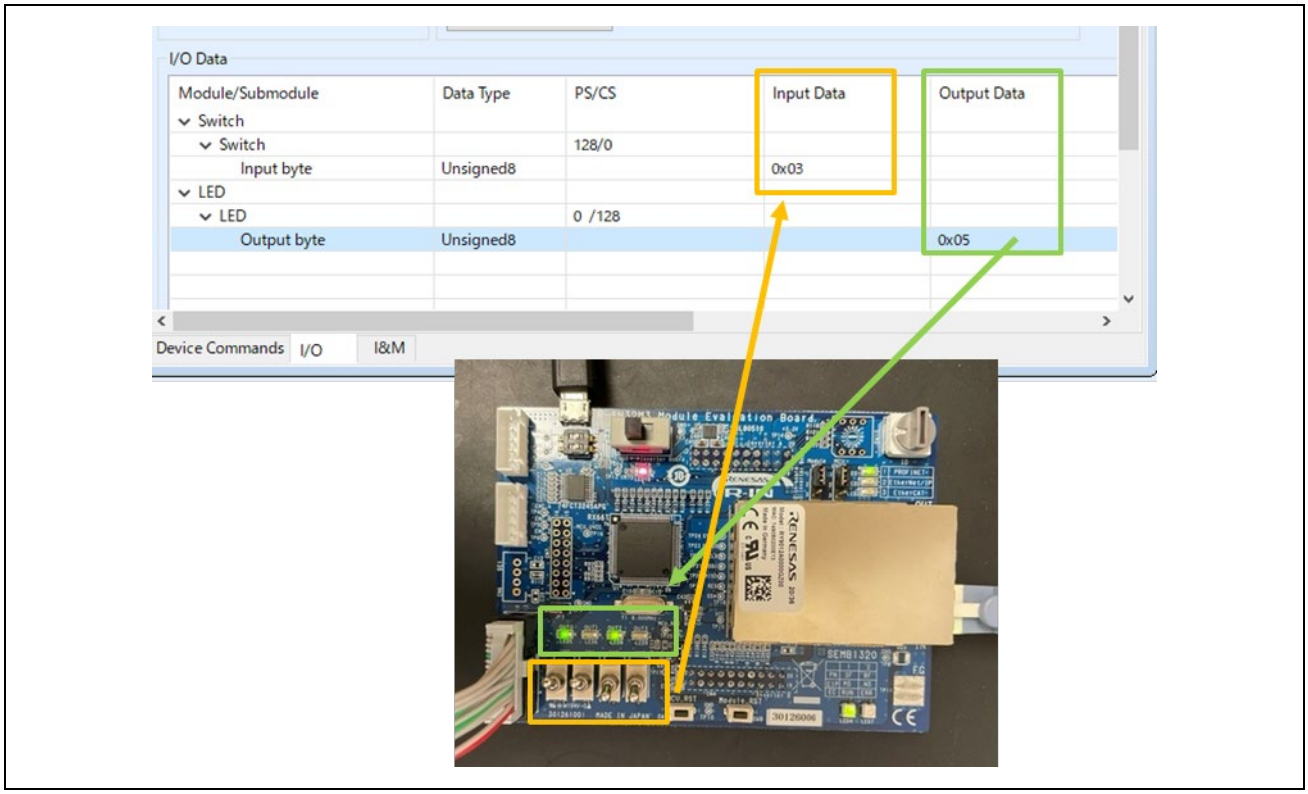

Figure 3-39 Remote I/O control [PROFINET]

Mirror control

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

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

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

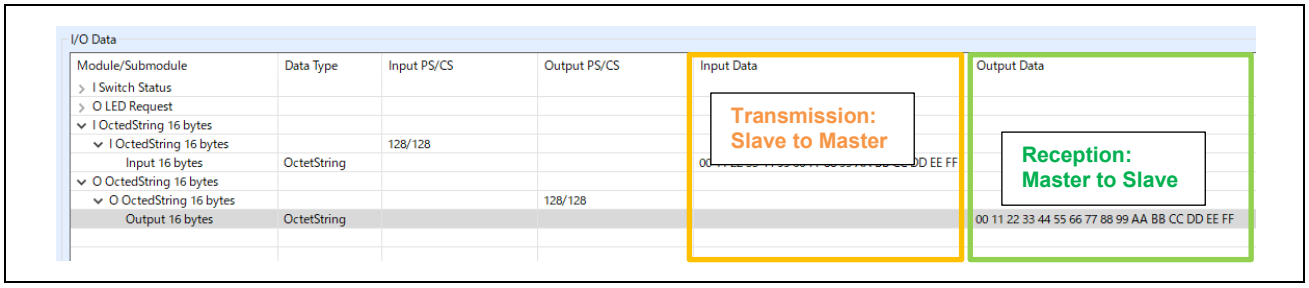

Figure 3-40 Mirror control [PROFINET]

-12. [Disconnect] terminates communication.

3.4.2 EtherNet/IP

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

Table 3-5 EtherNet/IP Sample software

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

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

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.4 to 3.3.6. When the sample application is run, the protocol display LED (LED2: EtherNet/IP) turn on.

Figure 3-41 Protocol LED: EtherNet/IP

-2. Set IP address

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

IP address	192.168.0.1
Netmask	255.255.255.0

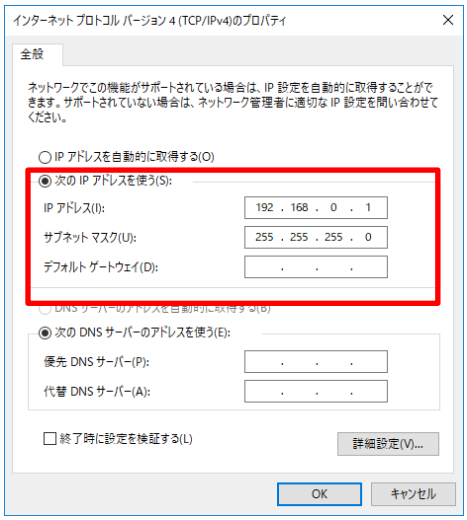

Figure 3-42 Set Static IP address

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.

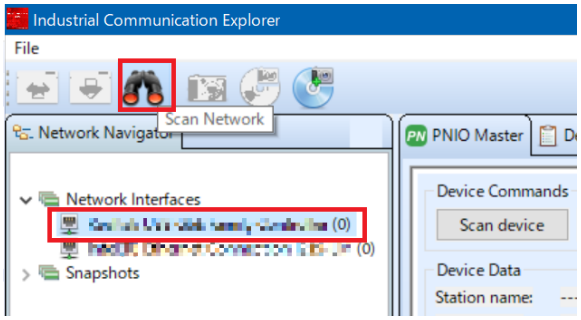

Figure 3-43 Scan network

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

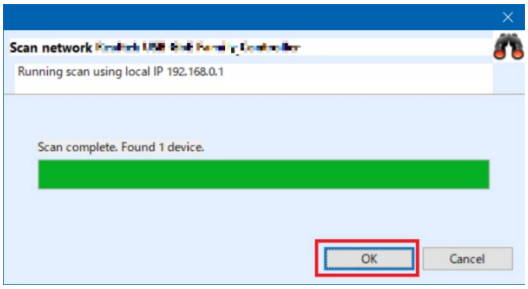

Figure 3-44 Scan completed

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

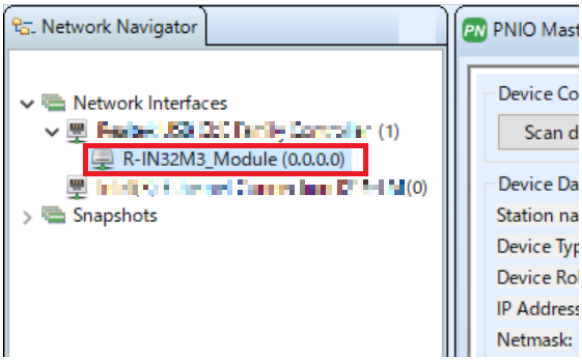

Figure 3-45 Select R-IN32M3 Module

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

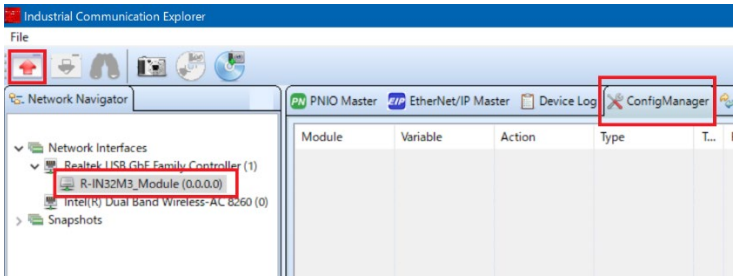

Figure 3-46 ConfigManager

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

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

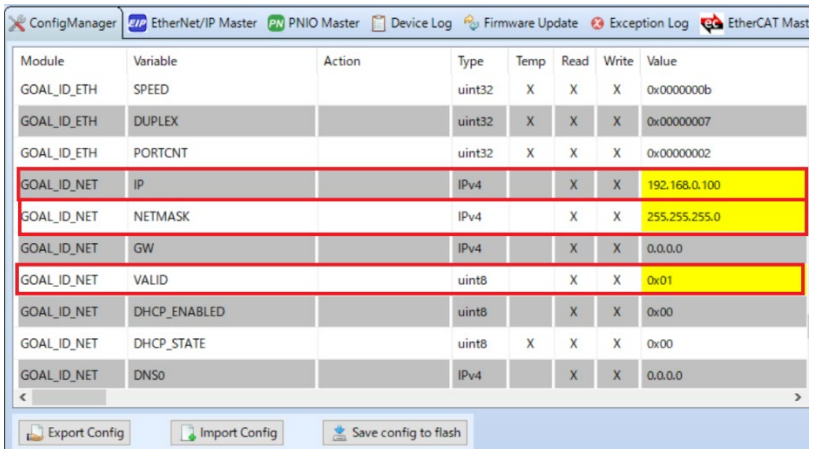

Figure 3-47 Set IP address on R-IN32M3 module

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

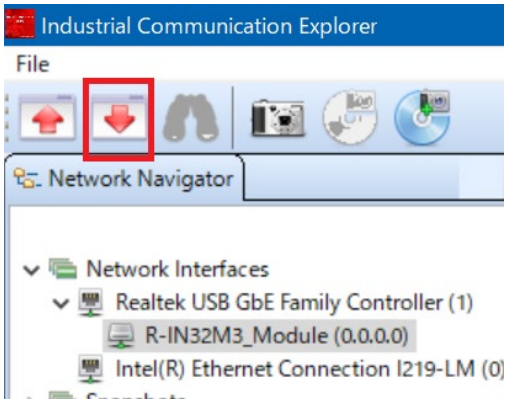

Figure 3-48 Download Config variables

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

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

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

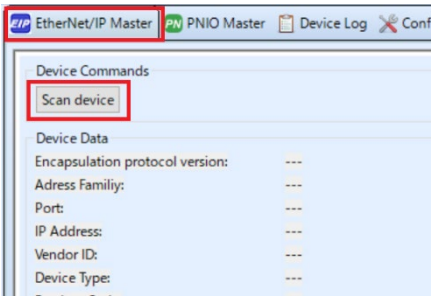

Figure 3-49 Scan device

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

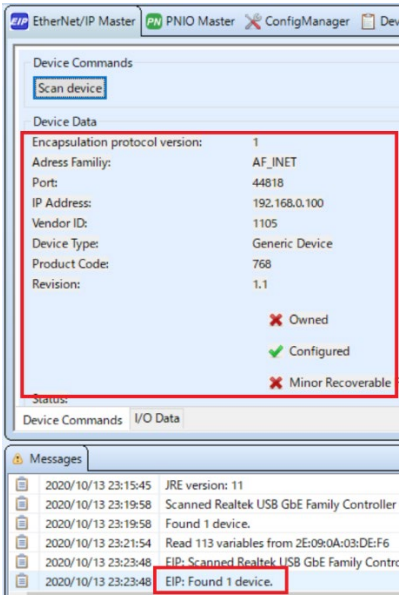

Figure 3-50 Device Data

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

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

  Target project: 02_eip, 05_eip_largesize, (10_multi_protocol), 12_eip_http
- **Mirror:** Sends data received from the master and mirrored back

  Target project: 02_eip, 05_eip_largesize, (10_multi_protocol), 12_eip_http
- **Mirror:** Sends data received from the master and mirrored back

  Target project: 05_eip_largesize

Application defied:

Table 3-6 Data application

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

Table 3-7 Configuration

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

Remote-IO (LED/Switch)

Refer to Table 3-6 and Table 3-7 to set the connection parameters. Packet interval in ms is left at the default value.

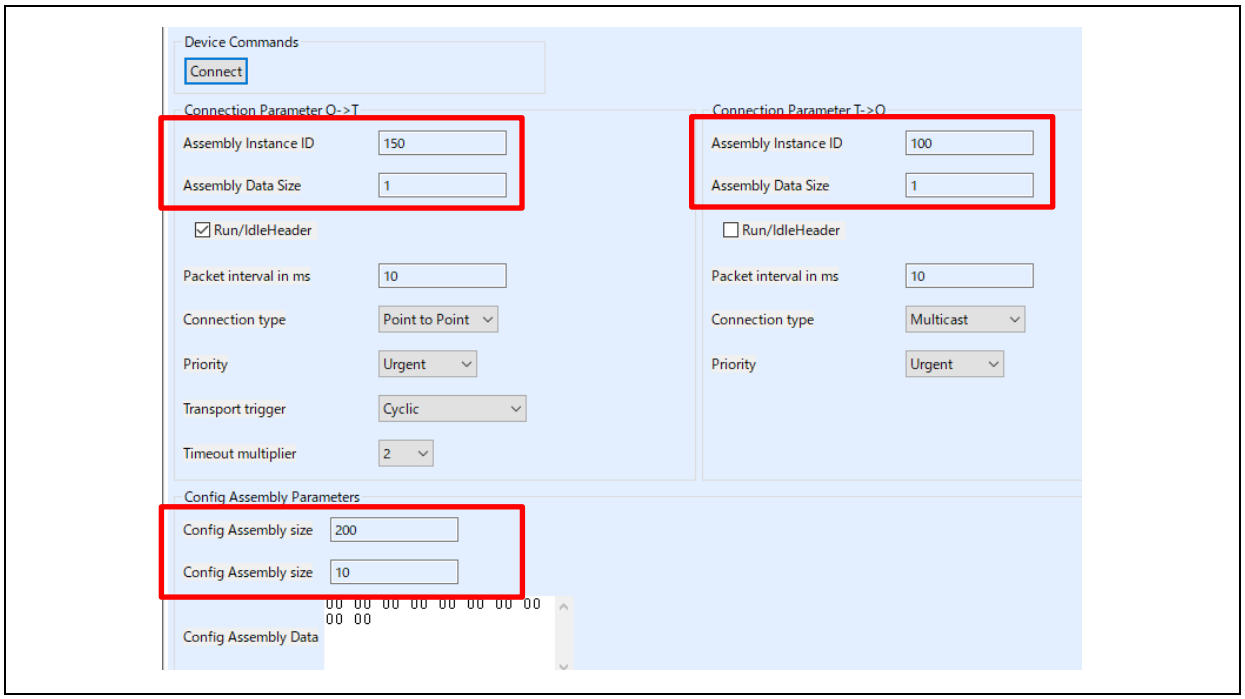

Figure 3-51 Remote-IO application parameter

Mirror control

Refer to -Table 3-6 and Table 3-7 to set the connection parameters.
Here is an example of mirror control for the 02_eip sample.

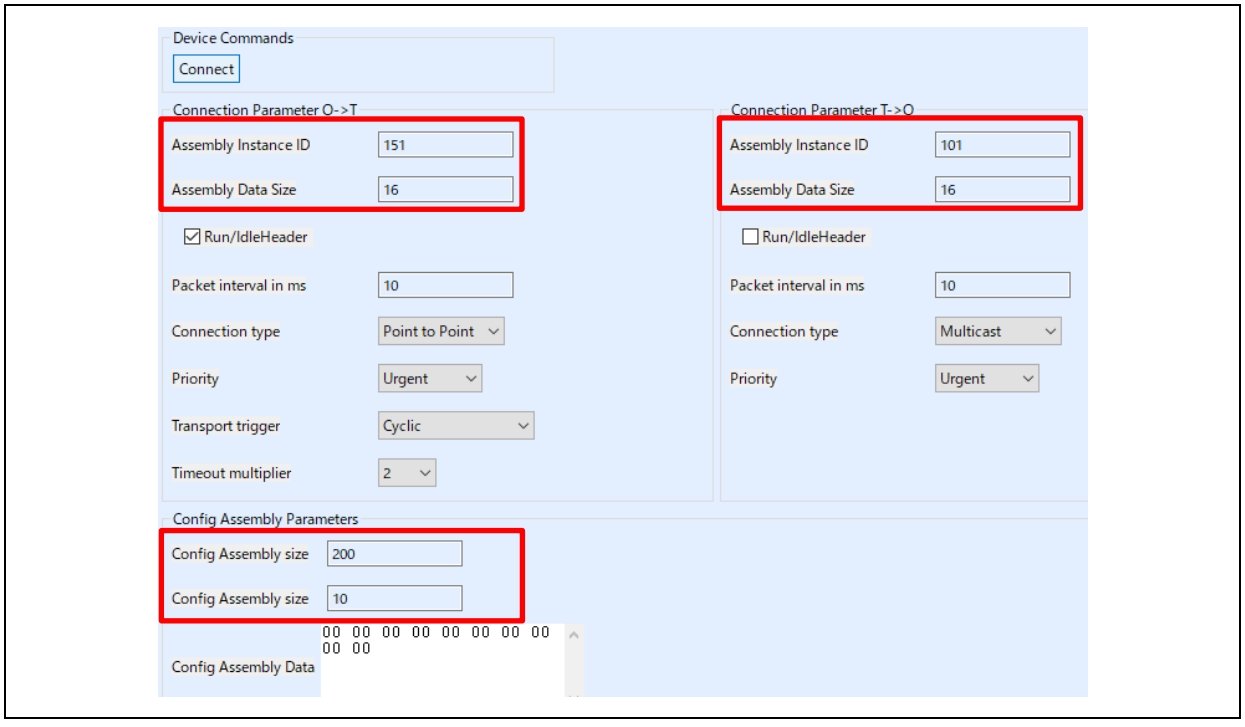

Figure 3-52 Mirror application parameter

Mirror control (RPC)

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

Refer to Table 3-6 and Table 3-7 to set the connection parameters. Figure 3-53 shows an example of a communication configuration for Mirror Data Reception_1 (152) and Mirror Data Transmission_1 (102).

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

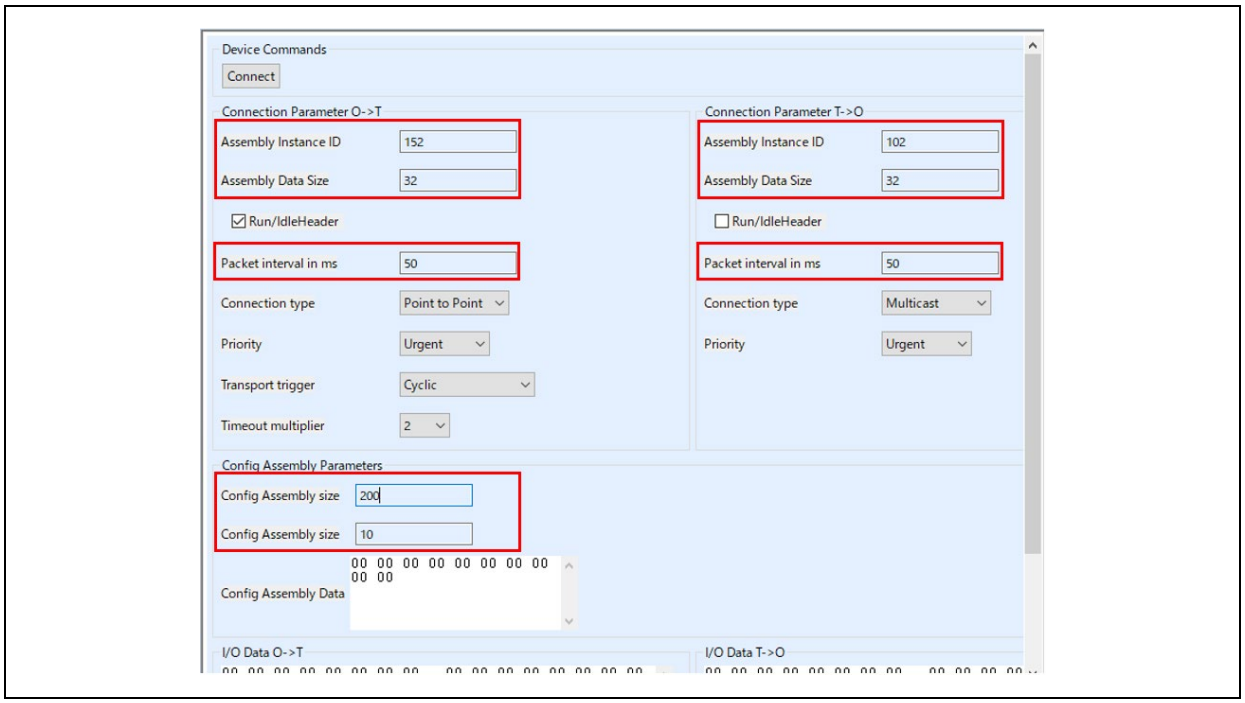

Figure 3-53 Mirror application(RPC) parameter

-11. Select the [Connect] button, which switches to the [Disconnect] button if the connection is successfully established. Also, the protocol status LED (LED4) on this board will light up.

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

Remote-IO (LED/Switch)

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

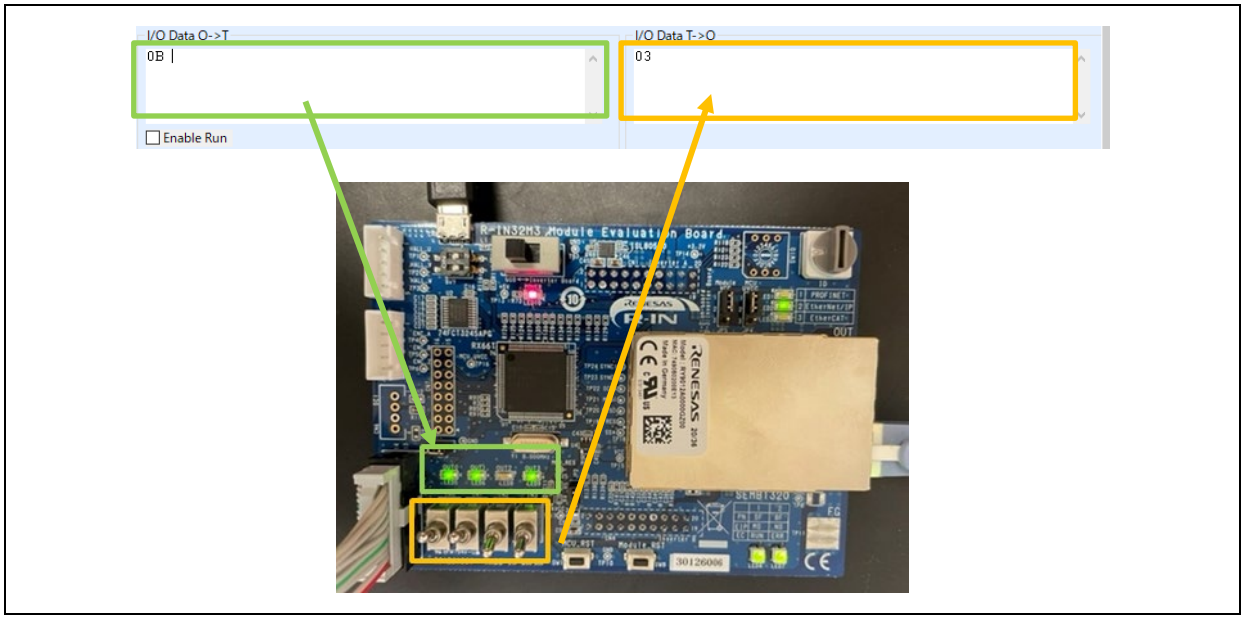

Figure 3-54 Remote-IO (LED/Switch) control [EtherNet/IP]

Mirror control

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

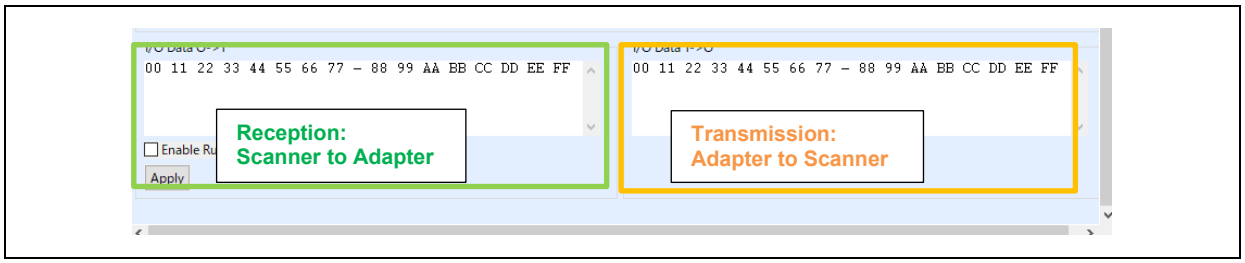

Figure 3-55 Mirror control [EtherNet/IP]

-13. [Disconnect] terminates communication.

3.4.3 EtherCAT

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

Table 3-8 EtherCAT Sample software

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

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

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

Refer to Chapter 3.3. to prepare the development environment.
Build the project and run the sample application, referring to Chapters 3.3.4 to 3.3.6. When the sample application is run, the protocol display LED (LED1: EtherCAT) turn on.

Figure 3-56 Protocol LED: EtherCAT

-2. Set Network Adapter

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

Drivers:

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

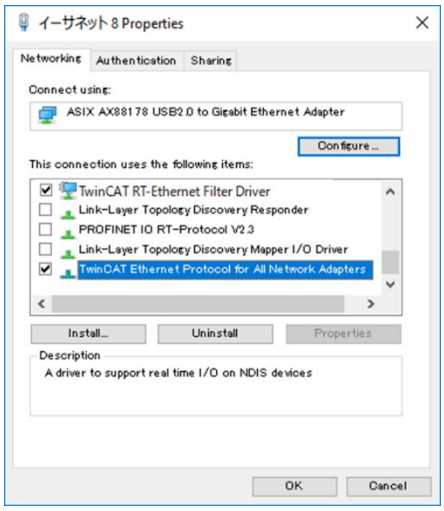

Figure 3-57 Network Adapter: EtherCAT

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

-3. ESI file

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

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

Table 3-9 ESI Files

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

[Folder for ESI storage]

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

2. Master connection

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

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

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

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

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

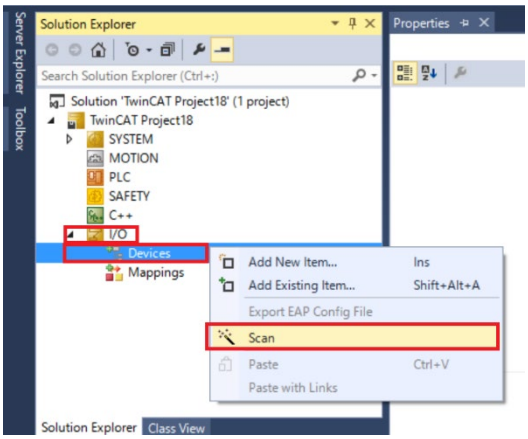

Figure 3-58 Scan network

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

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

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

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

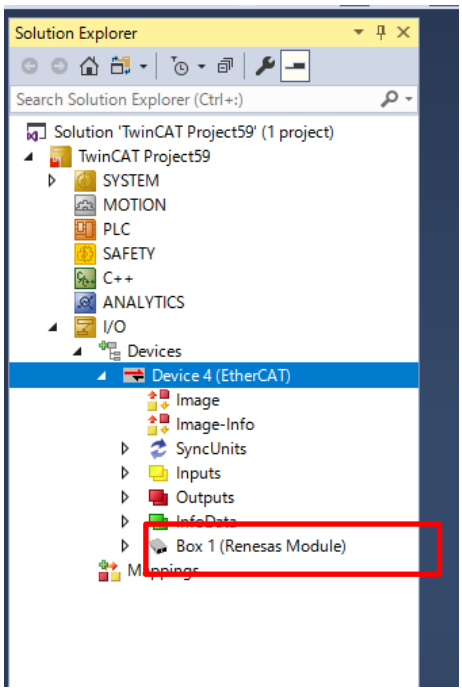

Figure 3-59 TwinCAT connection

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

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

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

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

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

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

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

  Target project: 06_ecat_largesize

Table 3-10 Application defied:

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

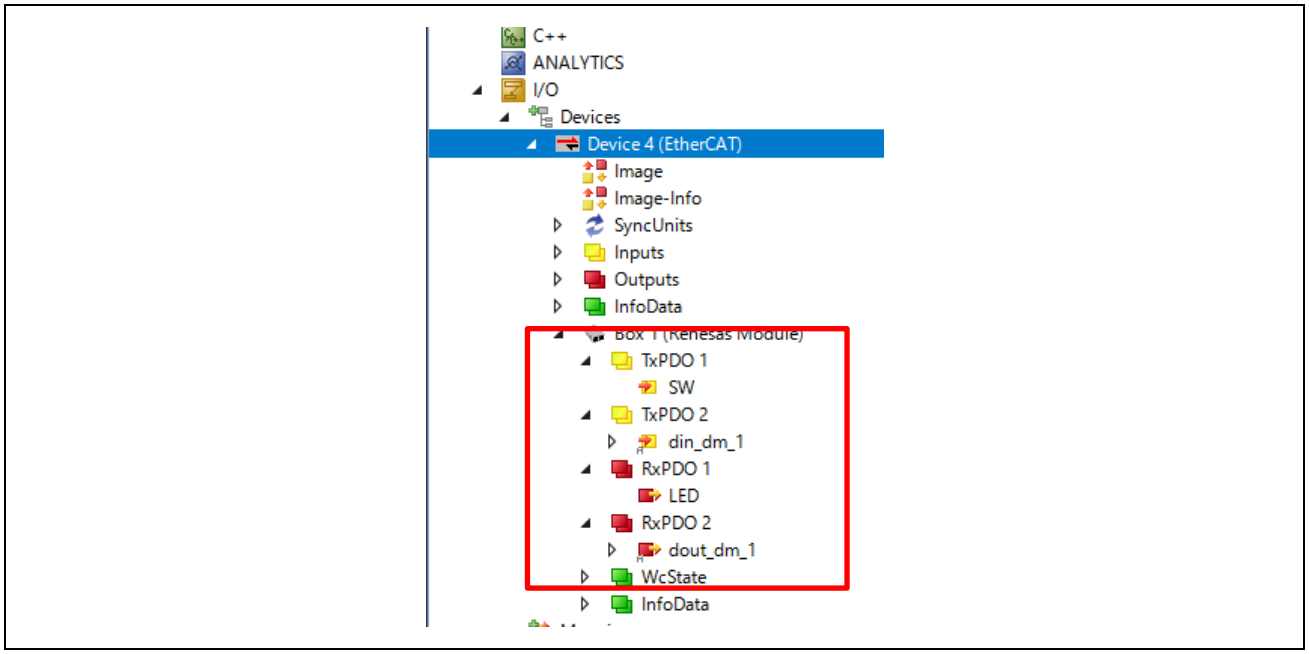

Figure 3-60 Application define (ex. 03_ecat)

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.

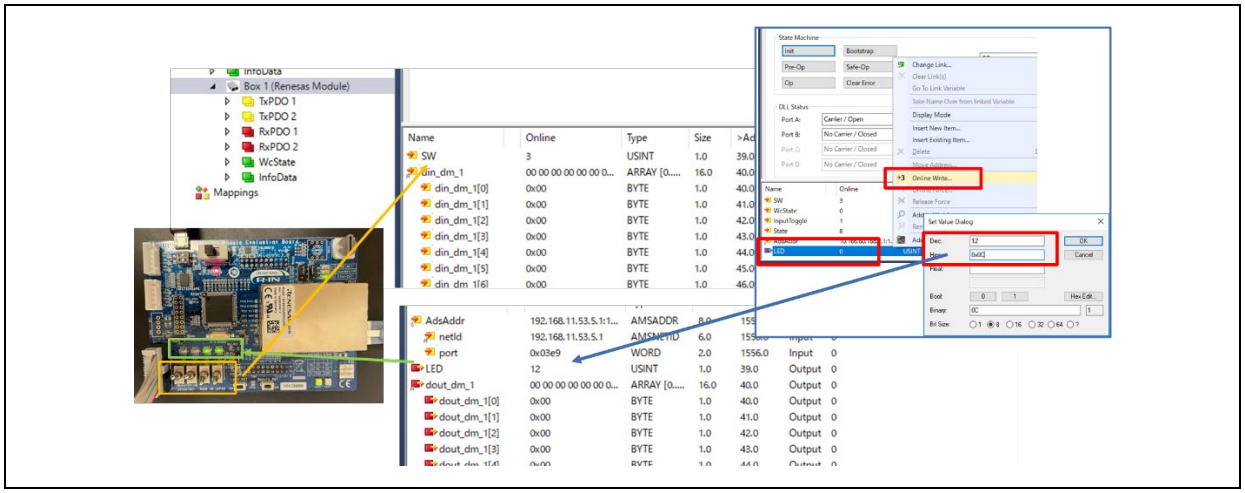

Figure 3-61 Remote I/O control [EtherCAT]

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.

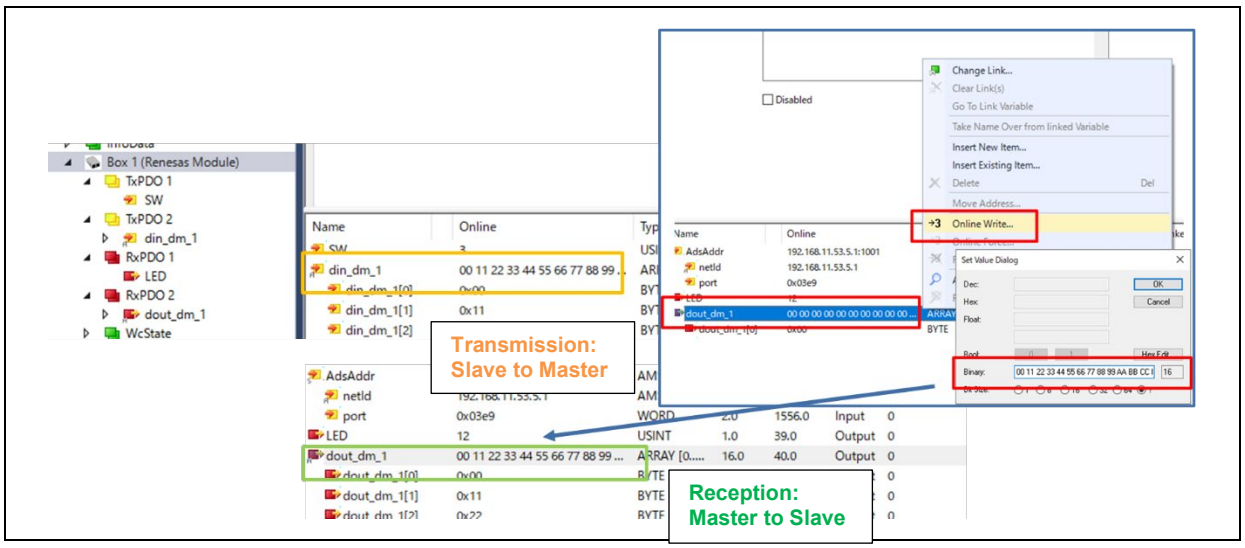

Figure 3-62 Mirror control [EtherCAT]

3.4.4 Modbus TCP

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

The target sample is below.

Table 3-11 Modbus Sample software

Sample software	Overview
07_mbus_tcp_server	Modbus TCP sample application

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

3.4.5 multi-protocol

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

Table 3-12 multi-protocol Sample software

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

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

1. Evaluation Environment Setup

-1. Evaluation Board Preparation

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 3.3.6. The protocol is executed according to the value of the general purpose switch (SW9).

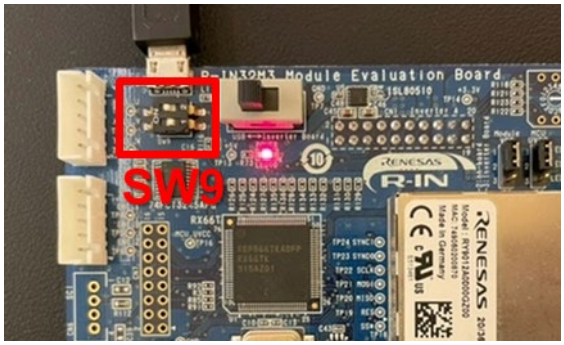

Figure 3-63 General purpose switch (SW9)

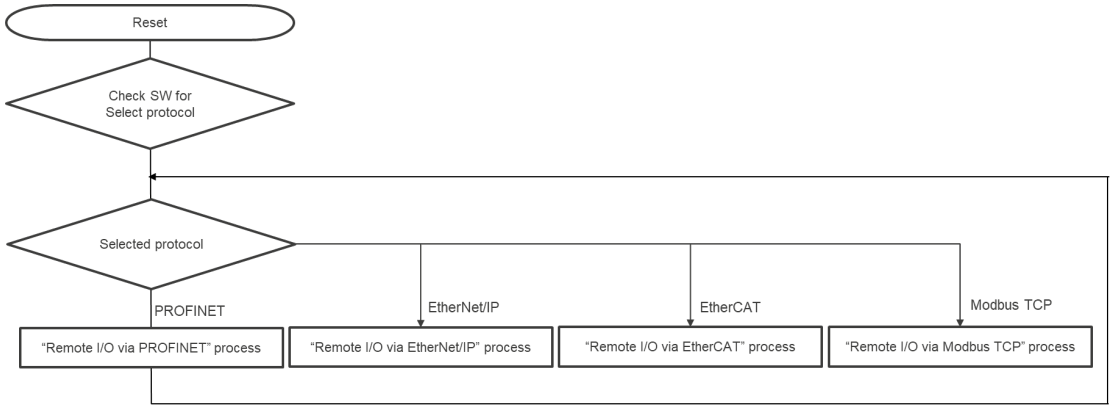

Figure 3-64 Multi-protocol selector flow

SW9	protocol	Protocol LED
0	Modbus TCP Server	All light-off
1	PROFINET	LED1 light-on
2	EtherNet/IP	LED2 light-on
3	EtherCAT	LED3 light-on

Modbus TCP Server：SW9 set to [0]

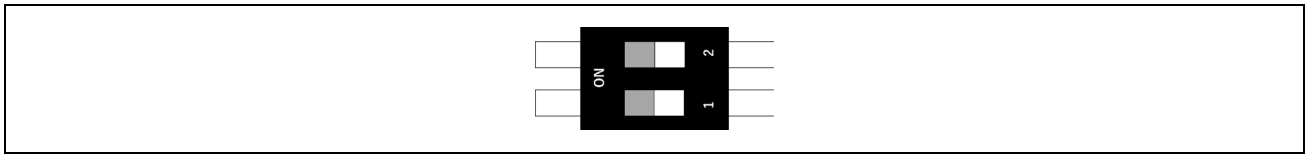

Figure 3-65 Modbus TCP Server

PROFINET：SW9 set to [1]

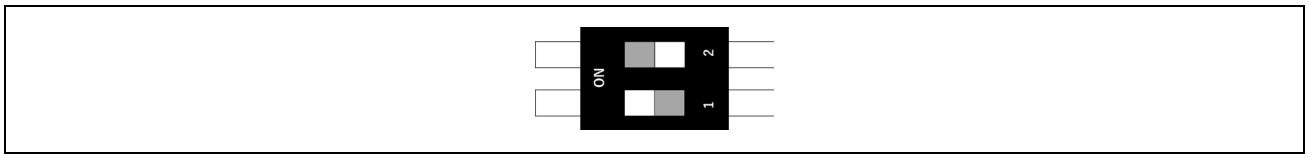

Figure 3-66 PROFINET

EtherNet/IP：SW9 set to [2]

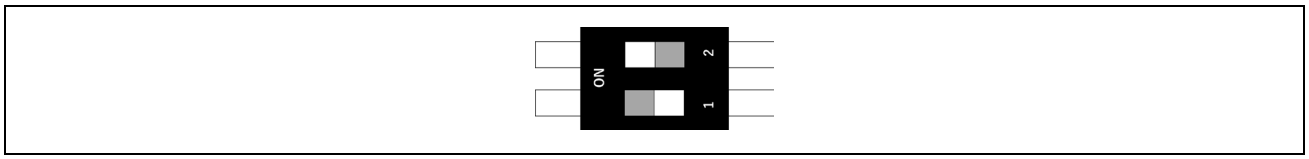

Figure 3-67 EtherNet/IP

EtherCAT：SW9 set to [3]

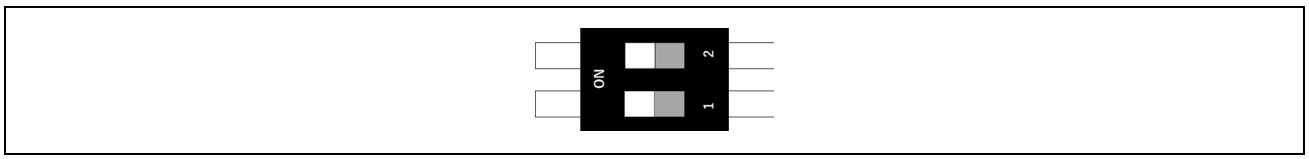

Figure 3-68 EtherCAT

-2. Set Network Adapter

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

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

2. Master connection

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

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

3.4.6 Web saver

The web browser access procedure using the sample program with the web server function.
The web content provided as a sample shows 2.5.2(3) Protocol status (LED4,7) in *index.html*. These html data are provided in *goal_http_fs.h*.

The target sample is below.

Table 3-13 Web Server Sample software

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

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

Evaluation Environment Setup

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

Protocol	Refer
PROFINET	3.4.1 PROFINET
EtherNet/IP	3.4.2 EtherNet/IP
EtherCAT	3.4.3 EtherCAT

The web browser access procedure

PROFINET, EtherNet/IP

The following conditions are used as an example.

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

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

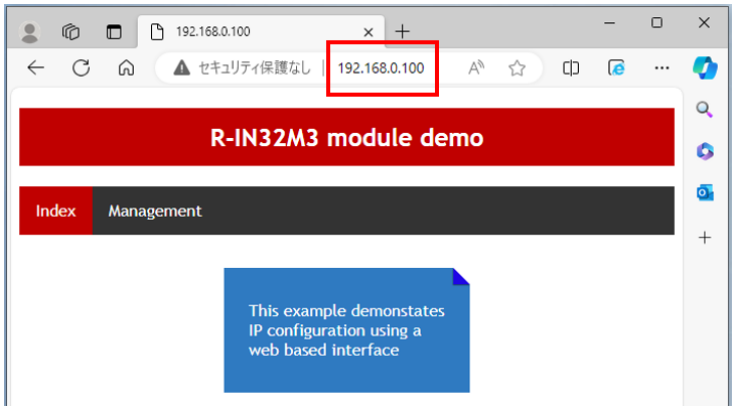

Figure 3-69 web browser access

EtherCAT

The following conditions are used as an example.

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

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

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

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

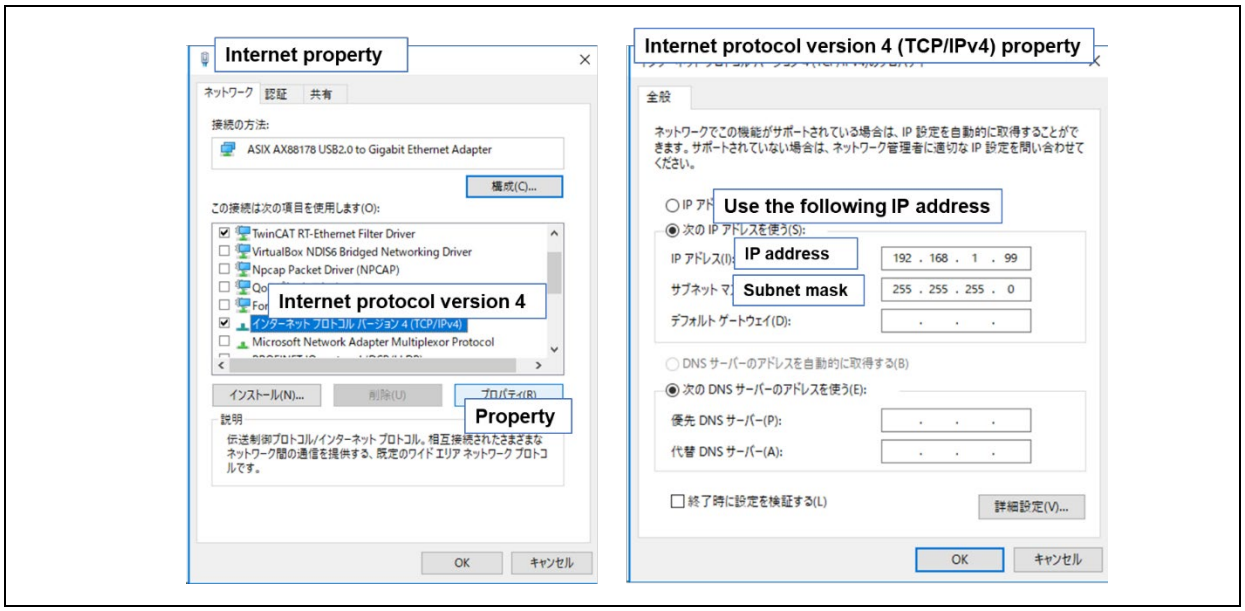

Figure 3-70 EtherCAT web access Network adapter

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

Select Slave > EtherCAT tab > Advanced Settings...

Mailbox > select “EoE”

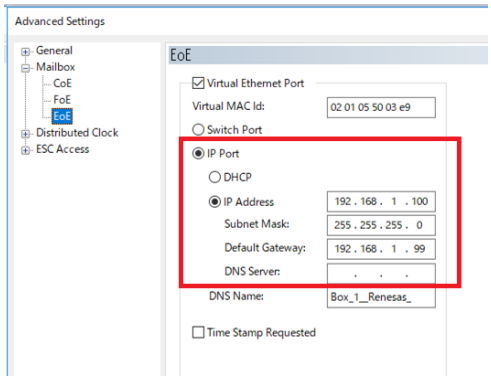

Figure 3-71 TwinCAT Network setting

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

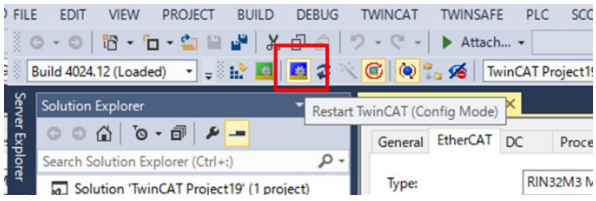

Figure 3-72 re-connect TwinCAT

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

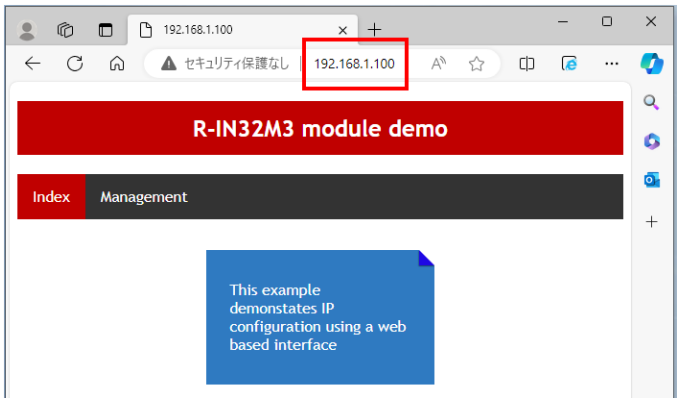

Figure 3-73 web browser access

3.5 Application Implement Guide

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

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

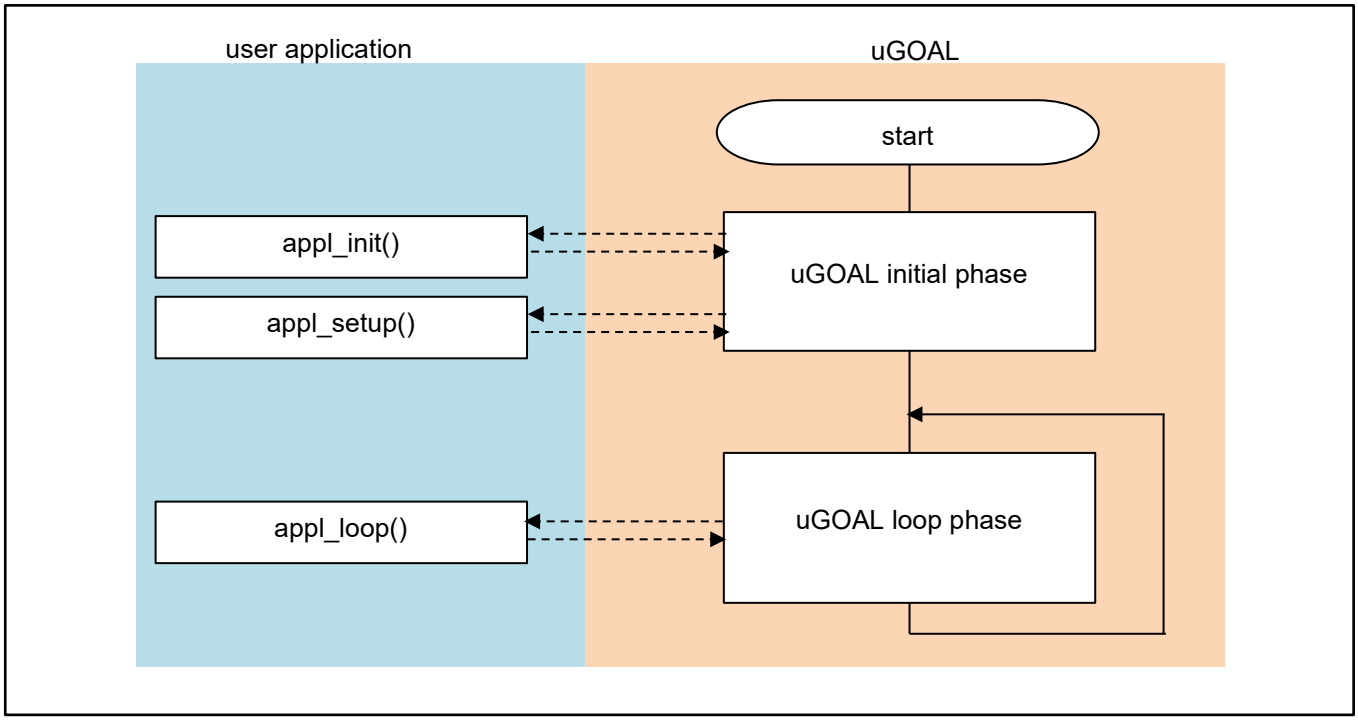

Figure 3-74 Overall flow of the program

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

Table 3-14 User applications and unique processing

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

3.5.1 PROFINET

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

(1) appl_init

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

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

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

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

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

. . .

    return res;
}
```

① ①

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

(2) appl_setup

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

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

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

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

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

・・                                                                                      ②

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

・・・

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

・・・
```

① Create an instance of SNMP.

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

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

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

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

. . .

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

. . .

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

. . .

    /* PROFINET configuration 80uccessful */
    goal_logInfo(“PROFINET ready”);

. . .

    return res;
}
```

④ ⑤

④ Create an instance of a sub-slot.

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

(3) appl_loop

Process the data after initialization of uGOAL.

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

. . .

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

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

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

. . .

}
```

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

3.5.2 EtherNet/IP

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

(1) appl_init

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

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

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

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

・・・

    return res;
}
```

① ①

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

(2) appl_setup

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

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

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

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

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

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

・・・

}
```

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

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

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

(3) appl_loop

Process the data after initialization of uGOAL.

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

 . . .

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

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

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

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

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

3.5.3 EtherCAT

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

(1) appl_init

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

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

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

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

    return res;
}
```

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

(2) appl_setup

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

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

```
GOAL_STATUS_T appl_setup(
    void
)
{

. . .

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

. . .

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

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

. . .

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

① ②

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

② Initialization of SII. (Disabled by default)

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

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

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

 . . .

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

 . . .

    return res;
}
```

③ ④ ⑤ ⑥

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

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

⑤ Set up firmware update via FoE.

⑥ Initialize the module for setting the EtherCAT Explicit Device ID. The EtherCAT Explicit Device ID switch (SW3) is required to set the ID. For details, please refer to Chapter 2.5.3(1).

(3) appl_loop

Process the data after initialization of uGOAL.

```
void appl_loop(
    void
)
{

 . . .

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

        read_analog16_input1 = write_analog16_output1;
        read_analog16_input2 = write_analog16_output2;

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

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

 . . .

}
```

① ①

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

4. Appendix

4.1 uGOAL API

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

4.2 Logging

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

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

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

Table 4-1 Log message output way and compile macro

Output way	Compile macro	Default value
1	CONFIG_UGOAL_LOGGING	0

The output method is described below.

4.2.1 Using TeraTerm

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

The step is shown below.

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

(1) Connect this board and PC using USB-UART converter cable.

- Connect USB-UART converter cable TX line to Pin3 in CN6 on this board.
- Connect USB-UART converter cable RX line to Pin2 in CN6 on this board.
- Connect USB-UART converter cable GND line to Pin4 in CN6 on this board.

The CN6 SCI connector (B4B-XH-A) is not mounted (refer to Chapter 2.5.4(5)), so it required to be mounted separately.

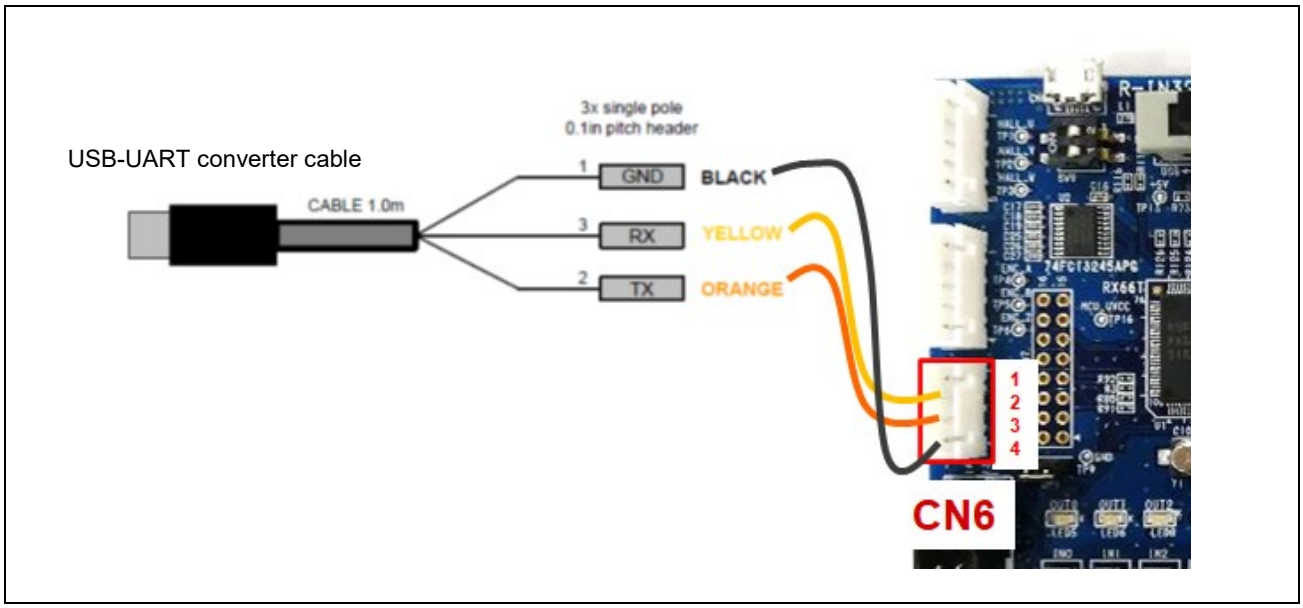

Figure 4-1 Connection of this board and USB-UART converter cable

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

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

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

Ex)

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

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

(4) Build the project and run the sample application, referring to Chapters 3.3.5 to 3.3.6.

(5) Connect to this sample application and start cyclic communication by operating Management tool referring to Chapter 3.4.

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

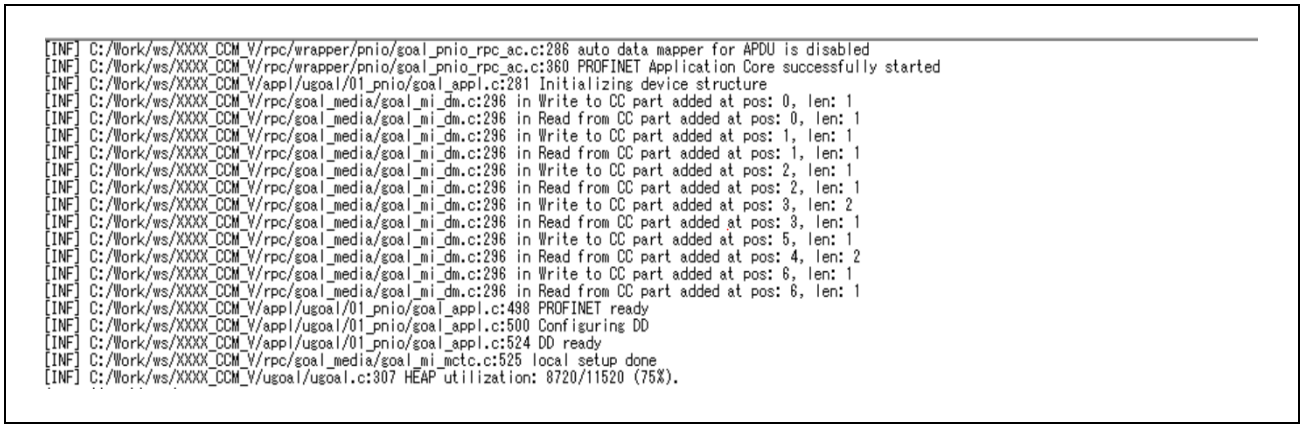

Figure 4-2 Log message on TeraTerm

4.3 IP Address Setting

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

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

Table 4-2 IP Configuration (GOAL_ID_NET)

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

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

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

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

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

Table 4-3 IP address setting list

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

4.4 Big-endian

The sample software is compatible with little endian. For Big-endian support, please refer to this step to create a project.

1. Create Project

1-1. Create a new project
Select “New” and create a project.

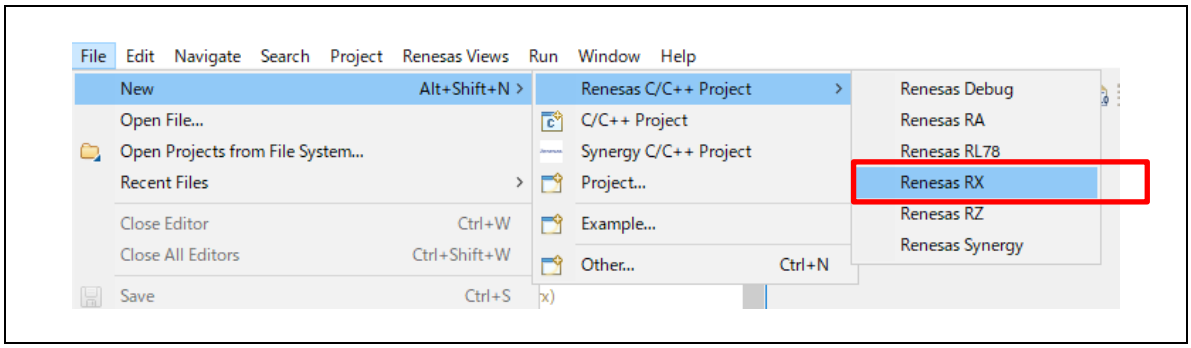

Figure 4-3 Create a project

1-2. Compiler
Select the compiler environment [GCC or CC-RX].

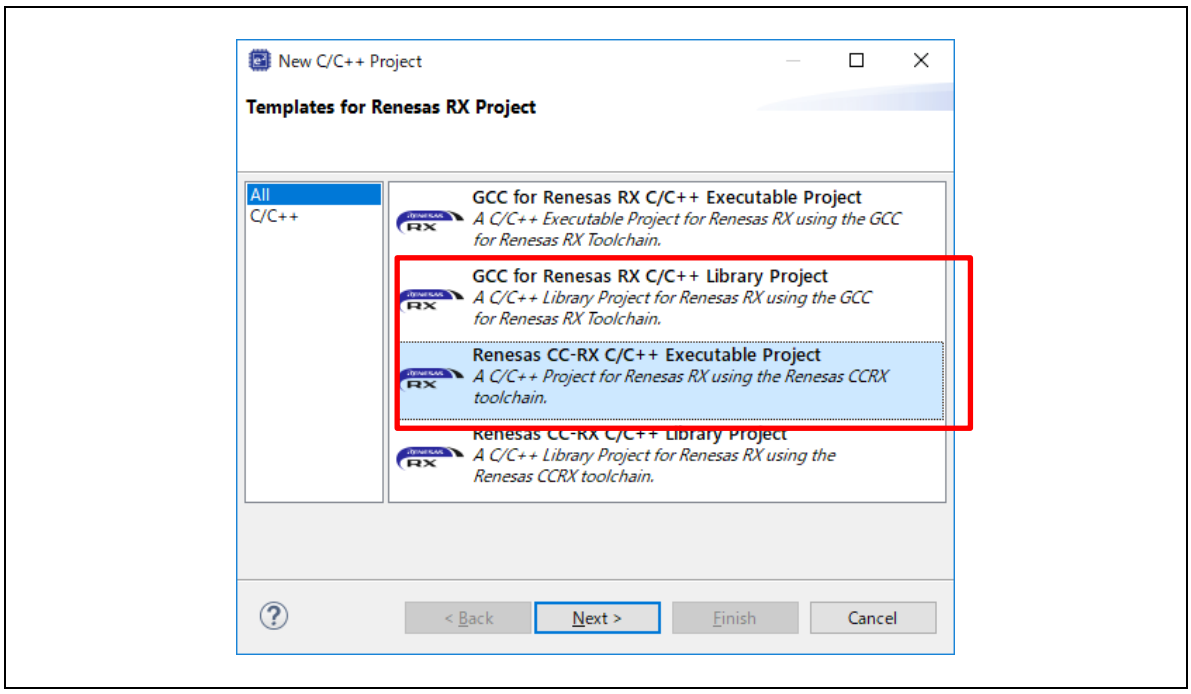

Figure 4-4 Compiler

1-3. Project name
Set Project name and program location.

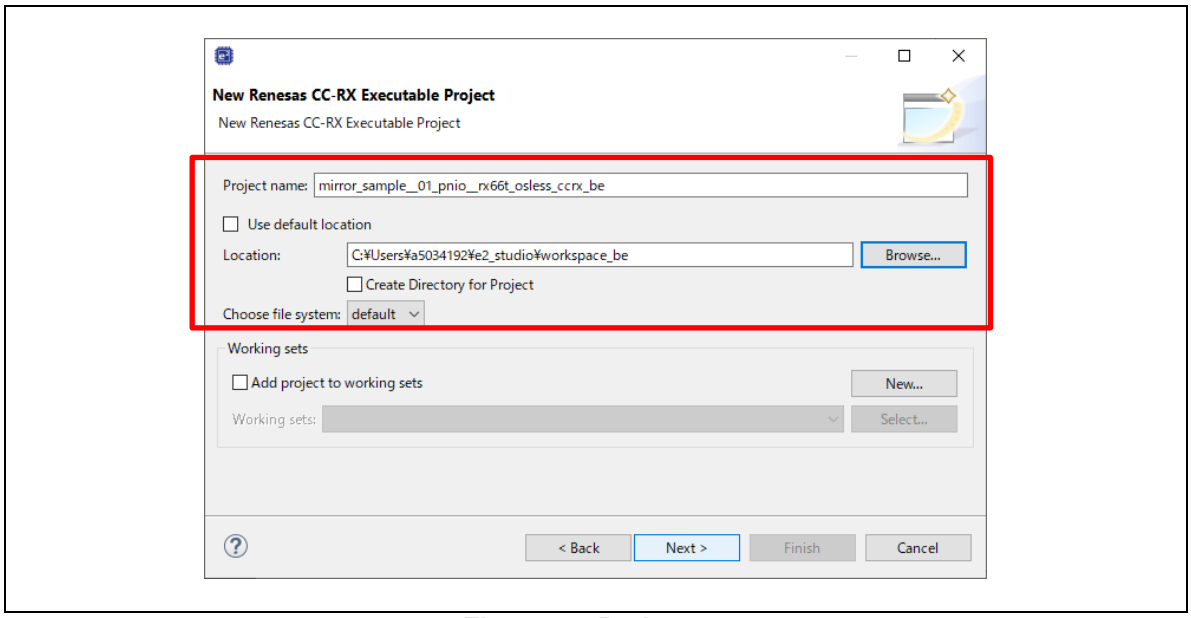

Figure 4-5 Project name

1-4. Project settings
Set project.

Target Device : R5F566TKAxFP * SEMB1320

Endian : Big

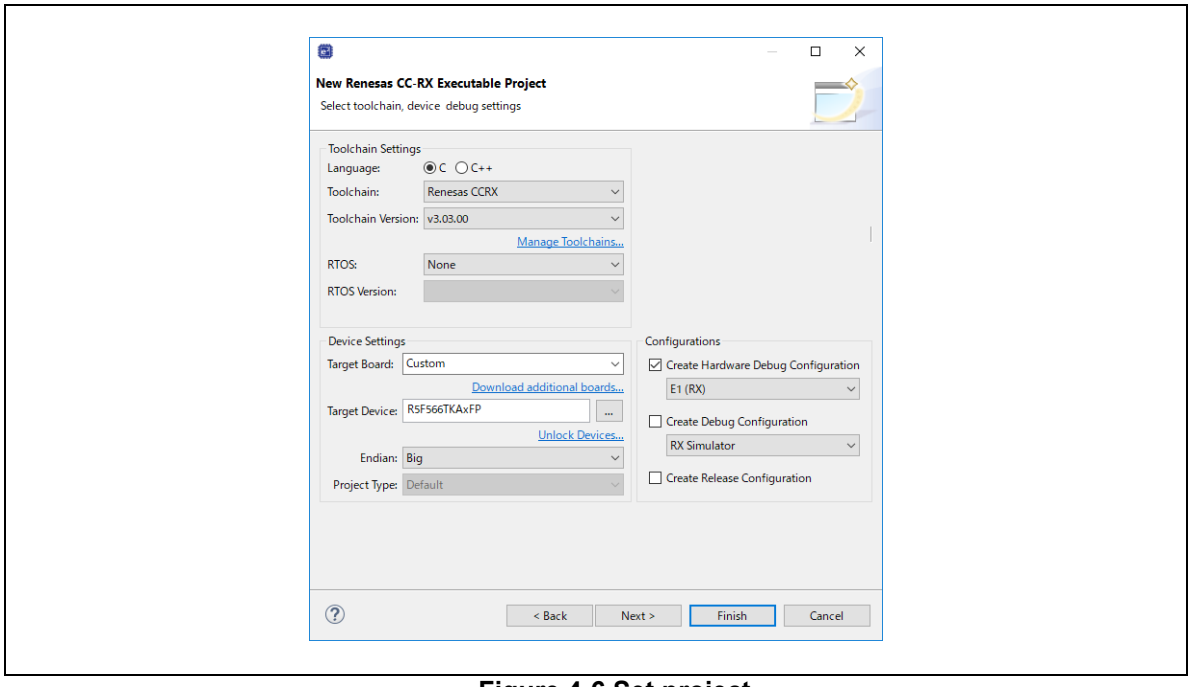

Figure 4-6 Set project

1-5. Smart Configurator * only CC-RX compiler
Enable “Use Smart Configurator” and create a project.

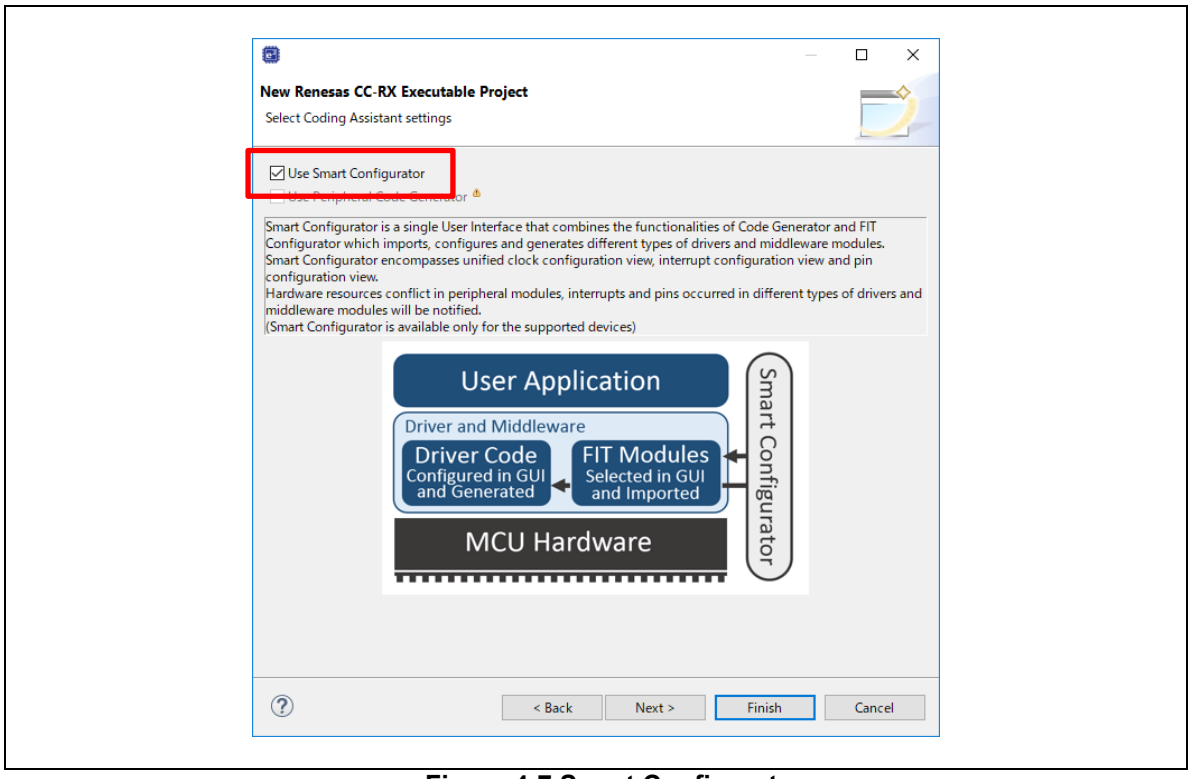

Figure 4-7 Smart Configurator

2. Project porting

2-1. Porting from a sample project
I. Close the configurator that is automatically displayed when the project is created.
II. Delete the configurator file.

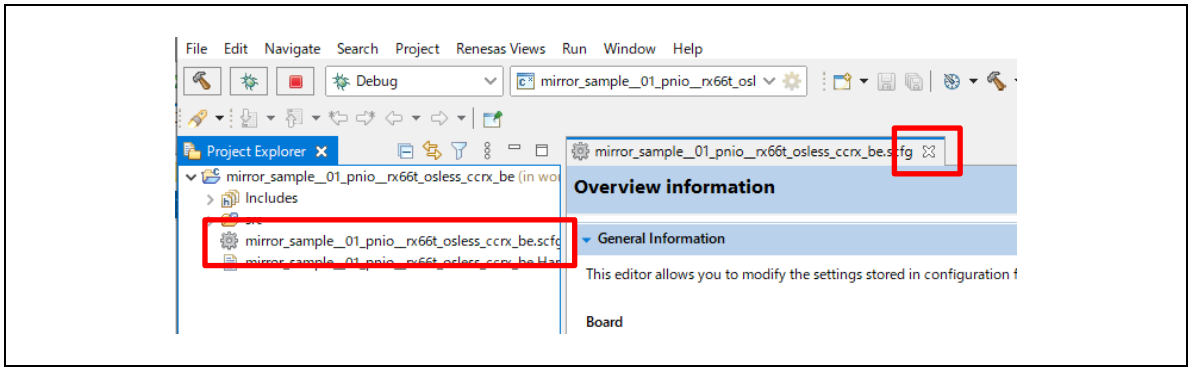

Figure 4-8 Close and Delete Configurator file

III. Import the R-IN32M3 Module sample project that you want to be ported Big-endian.
For import procedure, refer to 3.3.3 Import project.

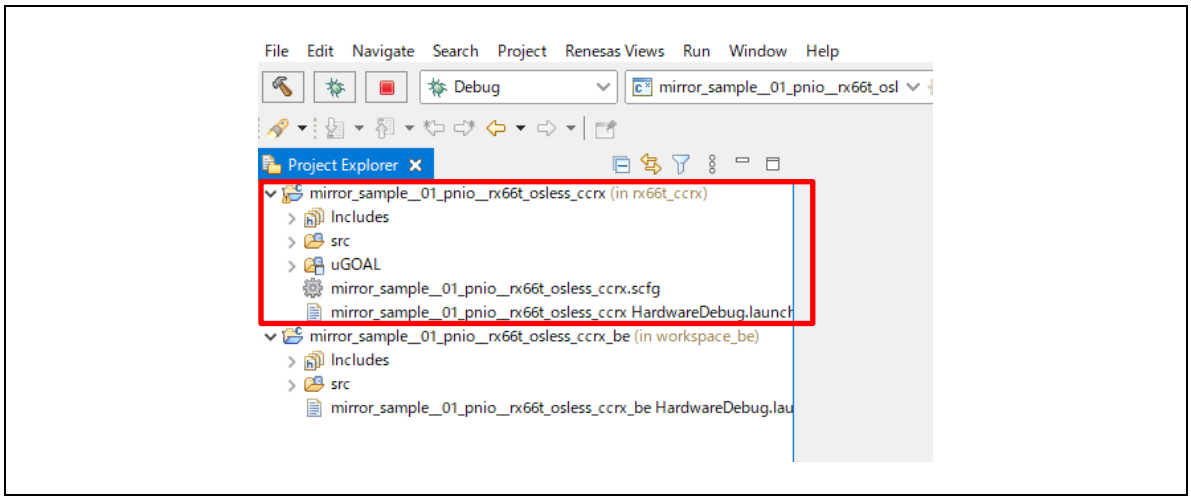

Figure 4-9 Import a sample project

IV. Copy the “uGOAL” folder and “Configurator” file from the R-IN32M3 Module sample project to be ported to the project on Big-endian side.

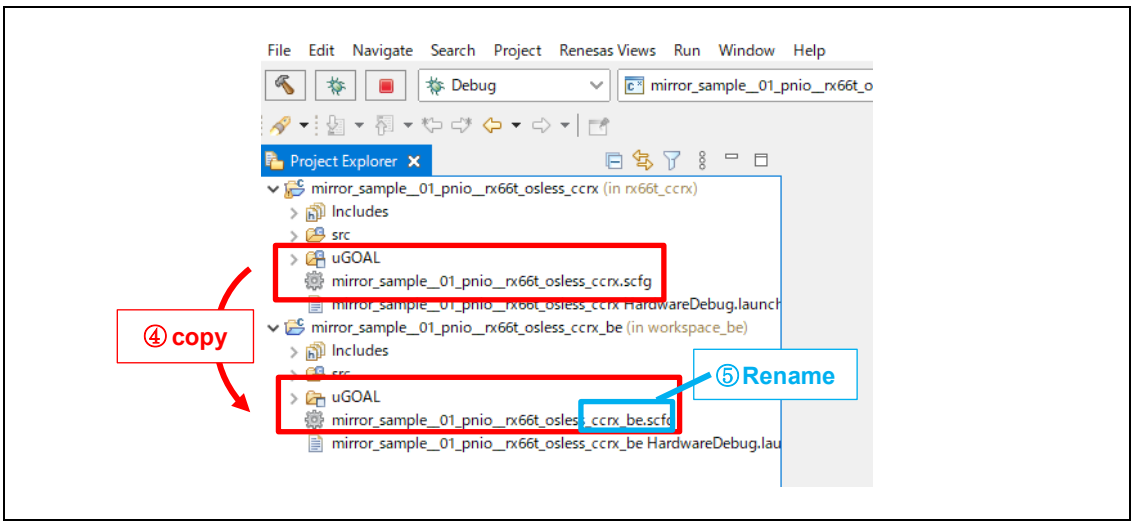

Figure 4-10 Copy and Rename

2-2. Project settings

I. Open the properties of the “uGOAL” folder and disable “Execute resource from build”. Apply and close the property.

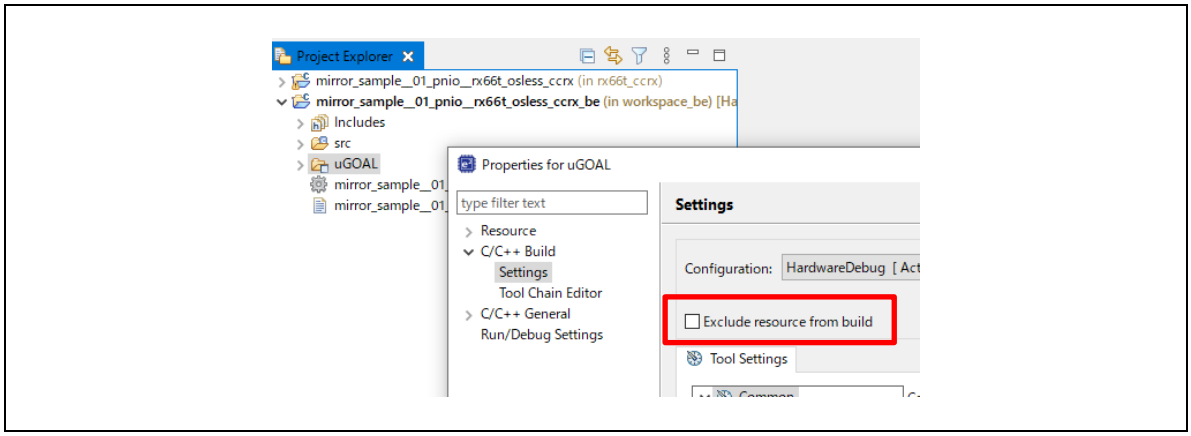

Figure 4-11 Build enables

II. Open the properties of the file (mirror_sample__01_pnio__rx66t_osless_ccrx.c) that contains the main function that was automatically generated when the project was created, and enable “Execute resource from build”. Apply and close the property.

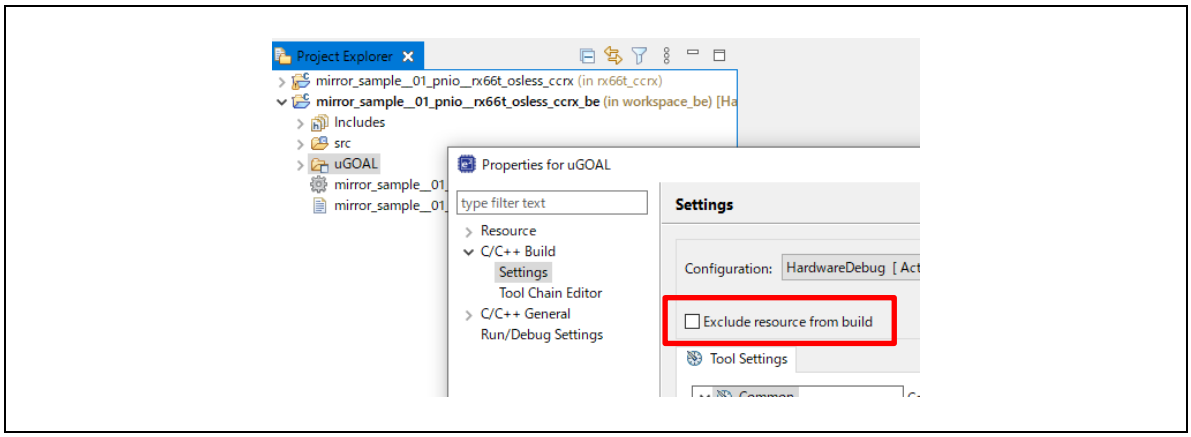

Figure 4-12 Build Disable

2-3. Build settings

Open the project properties and set the build.

I. Copy the Build Variables settings from the sample project settings you are porting from and apply them.

Variables name	GoalDirPath
Type	String
Value	${ProjDirPath}/../../../../..

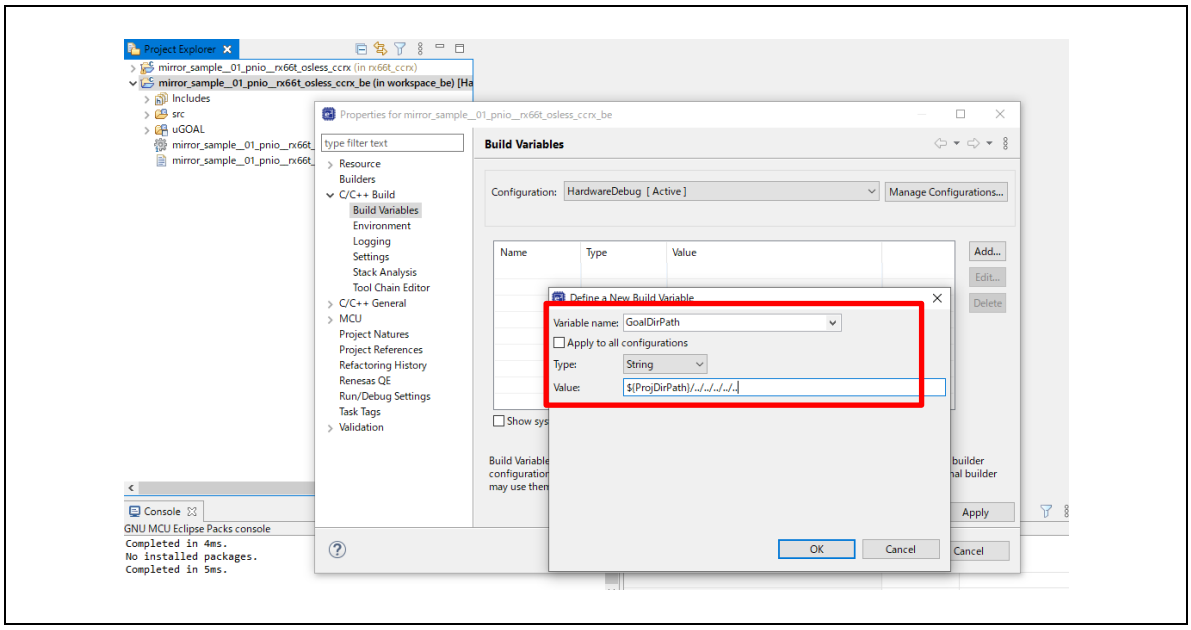

Figure 4-13 Set Variable

II. Include file path
[CCRX: Settings > Compiler > Source], [GCC: Settings > Compiler > Includes]

Copy the include file path from the sample project settings you are porting from.
The copy target is the path with "$ {GoalDirPath}" at the beginning.

The following include file directory is an example of a big endian port from “mirror_sample__01_pnio__rx66t_osless_ccrx”.

"${GoalDirPath}/."	"${GoalDirPath}/rpc/wrapper"
"${GoalDirPath}/appl"	"${GoalDirPath}/rpc/wrapper/ccm"
"${GoalDirPath}/appl/mirror_sample/01_pnio"	"${GoalDirPath}/rpc/wrapper/dd"
"${GoalDirPath}/ext"	"${GoalDirPath}/rpc/wrapper/dd/protos"
"${GoalDirPath}/ext/uthash"	"${GoalDirPath}/rpc/wrapper/dd/protos/dd"
"${GoalDirPath}/plat/rx66t_semb1320"	"${GoalDirPath}/rpc/wrapper/dd/protos/dd/rpc"
"${GoalDirPath}/plat/rx66t_semb1320/Drivers/r_gpio_rx"	"${GoalDirPath}/rpc/wrapper/pnio"
"${GoalDirPath}/rpc"	"${GoalDirPath}/sapi"
"${GoalDirPath}/rpc/goal_mctc"	"${GoalDirPath}/ugoal"
"${GoalDirPath}/rpc/goal_media"	

III. Macro
[CCRX: Settings > Compiler > Source], [GCC: Settings > Compiler > Includes]

Copy the macro from the sample project settings you are porting from.

The following include file directory is an example of a big endian port from “mirror_sample__01_pnio__rx66t_osless_ccrx”.

Copy	_RX66T_DEMO_
Copy	APPL_STANDALONE
Copy	CONFIG_UGOAL_HEAP_BUFFER_ALIGNMENT=4
Add	CONFIG_UGOAL_BIG_ENDIAN=1

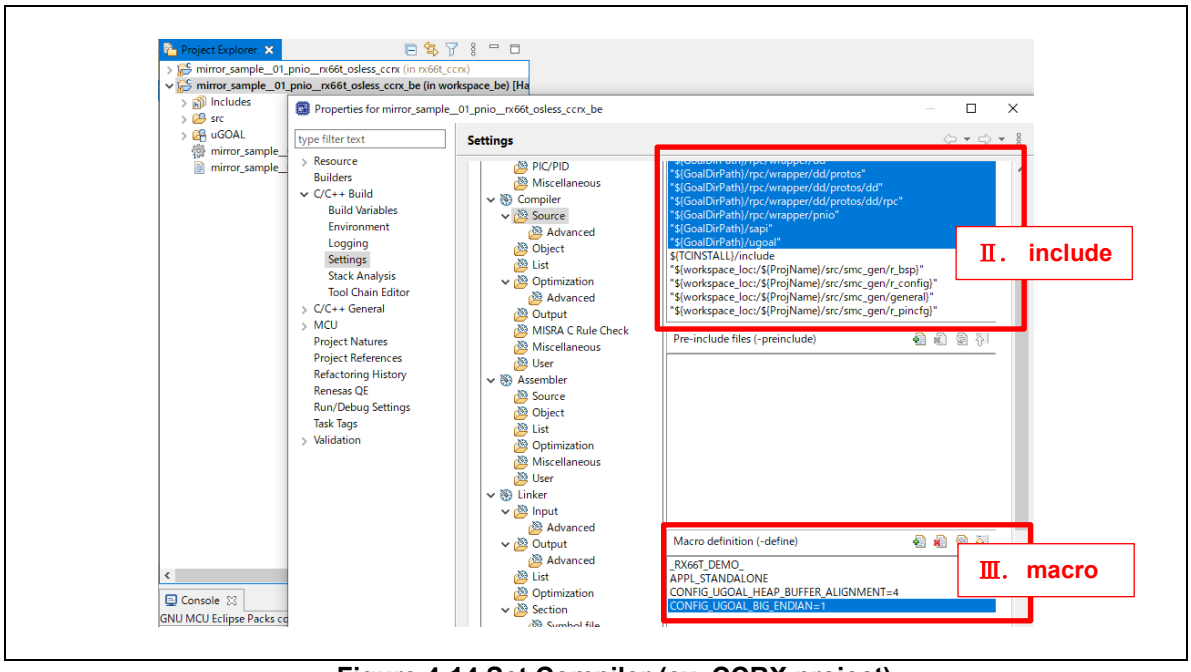

Figure 4-14 Set Compiler (ex. CCRX project)

IV. Set object
[CCRX: Settings > Compiler > Source]

Enable the objects.

Allocates uninitialed variables to 4-byte boundary alignment sections (-nostuff=B)
Allocates initialed variables to 4-byte boundary alignment sections (-nostuff=D)
Allocates const qualified variables to 4-byte boundary alignment sections (-nostuff=C)

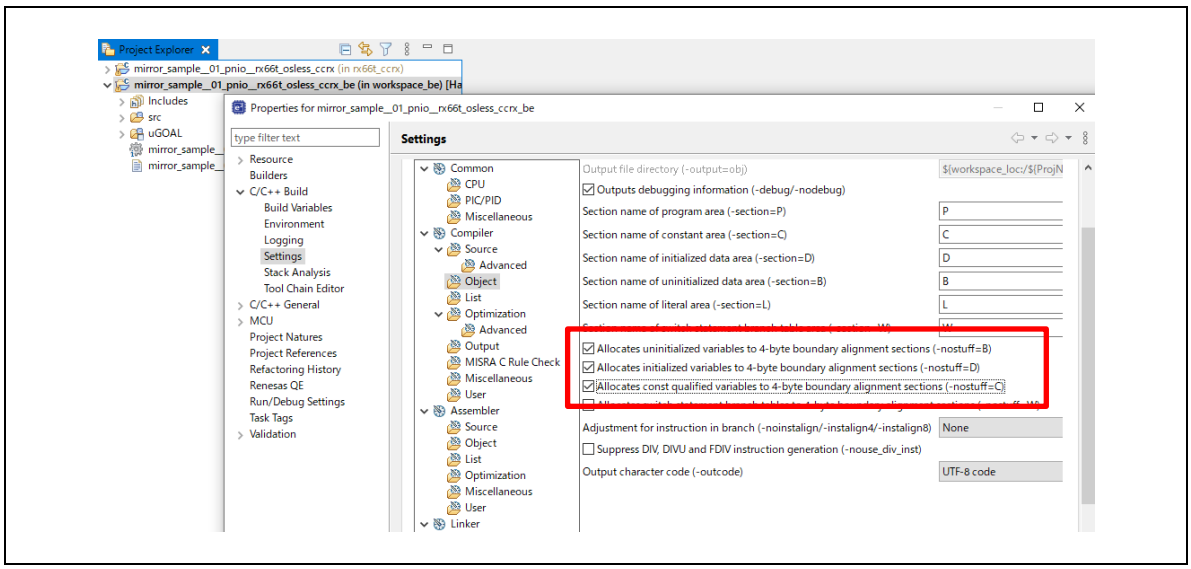

Figure 4-15 Set Object (ex. CCRX project)

"Apply and Close" to close the property window.

This Completeness the creation of Big-endian project.
After that, proceed with the development environment construction by following the procedure from 3.3.4 FIT Module.

4.5 Individual installation for GCC toolchain

This note is described based on the use of GCC 8.3.0.202202. If you don't have the toolchain on your environment, you may need to download/install it individually.
Go to GNU Tools https://llvm-gcc-renesas.com/, you can reach the intended GCC package from the list on [PRODUCTS] tab > [RX] > [Download Toolchain].

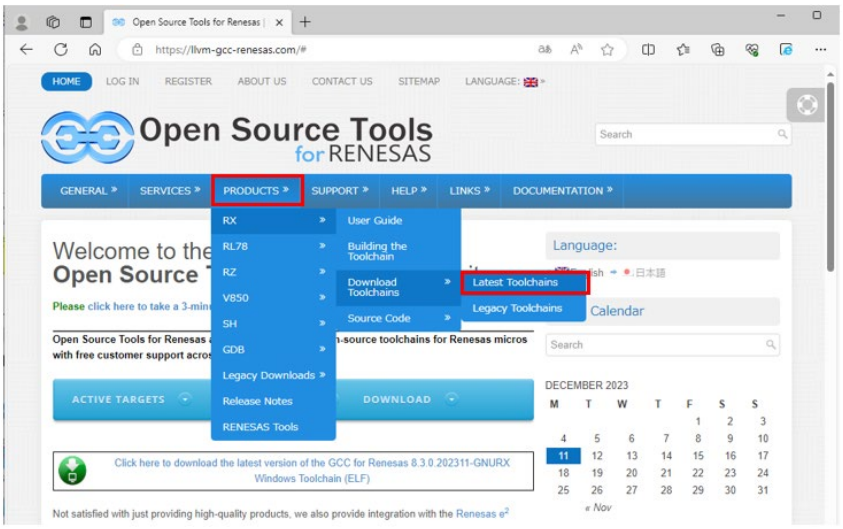

Figure 4-16 Open Source Tools web page

Push [Download] button of GCC for Renesas 8.3.0. 202311-GNURX Toolchain.

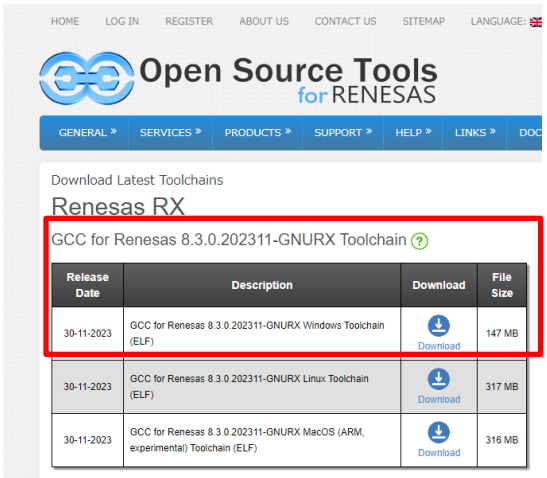

Figure 4-17 Download page

Once downloaded, run the installer to adopt whole component. After installation, please make sure whether the intended GCC toolchain is activated on your e2studio environment (reference: 3.3.3).

4.6 Individual installation for FIT module

This note is described based on the use of RX Driver Package V1.42 as a- FIT module on e2studio. If you cannot select the FIT module, you may need to download/install it individually.
Go to RX Driver Package page https://www.renesas.com/jp/ja/software-tool/rx-driver-package on Renesas HP to download it. If the relevant version is not listed, go to [Previous version] at the bottom of the page to download the relevant version.

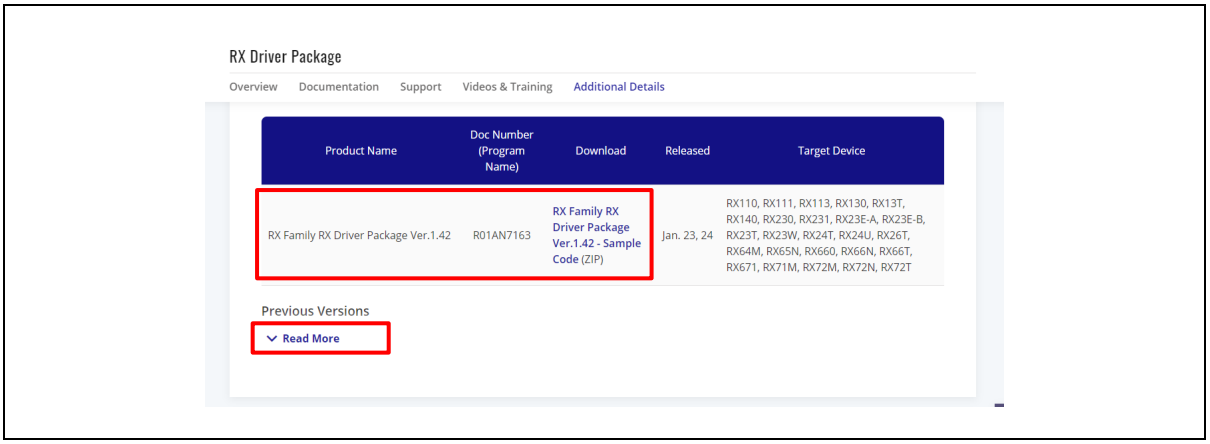

Figure 4-18 Download page

Once downloaded, open a folder which is already created as following file path, then store the zip file [r01an7163xx0142-rx-fit.zip] into the folder.

C:\Users**UserName**\.eclipse\com.renesas.platform_download\FITModules\Downloaded

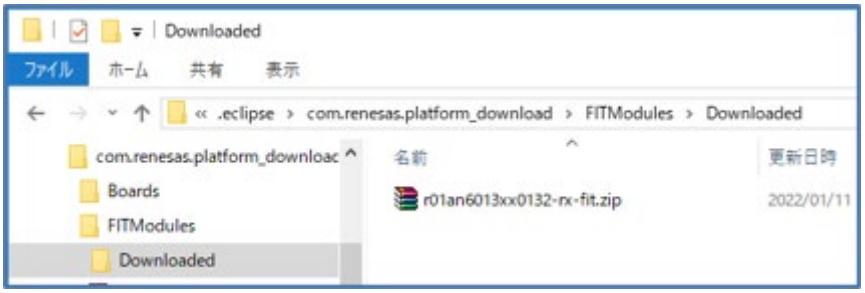

Figure 4-19 Zip file stored into [Downloaded]

Copy whole ingredients of [FITModules] folder of the package, then paste them directly into a folder below;

C:\Users**UserName**\.eclipse\com.renesas.platform_download\FITModules

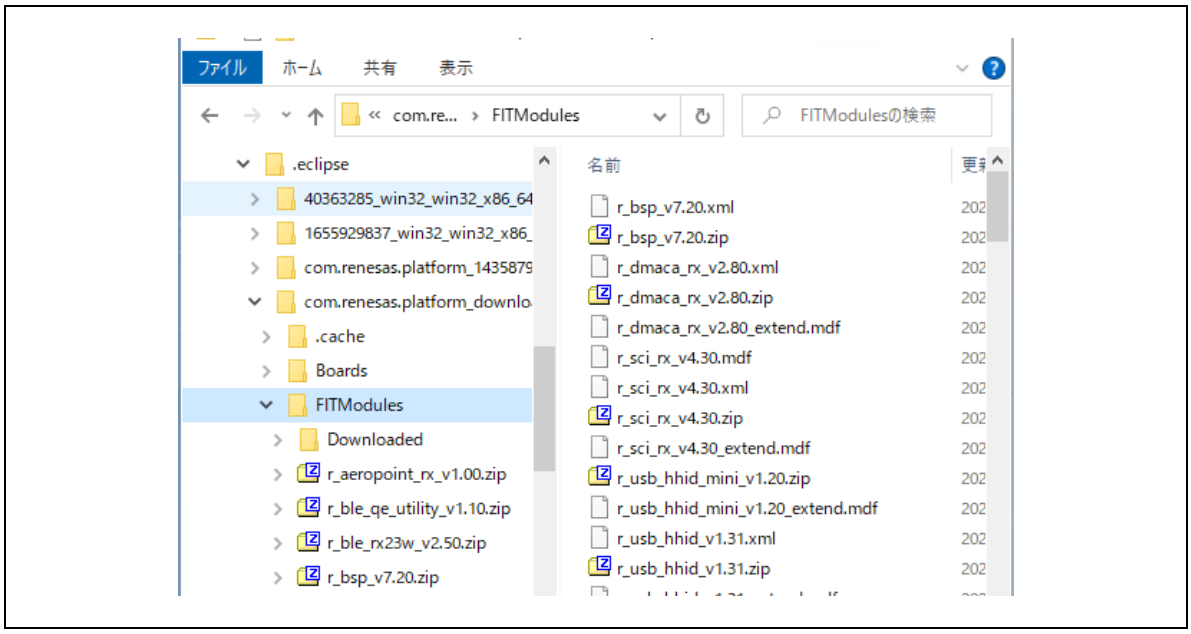

Figure 4-20 Ingredients stored into [FITModules]

After installation, please make sure whether the intended FIT module can be activated on your e2studio environment (reference: 3.3.4).

4.7 Individual Installation for CC-RX

If the version of the RX compiler (CC-RX) you need is not listed in e2studio, follow the steps below to obtain CC-RX individually. Here are the steps to download CC-RX V3.06.00.

Download CC-RX V3.06.00 from C/C++ Compiler Package for RX Family [CC-RX] | Renesas

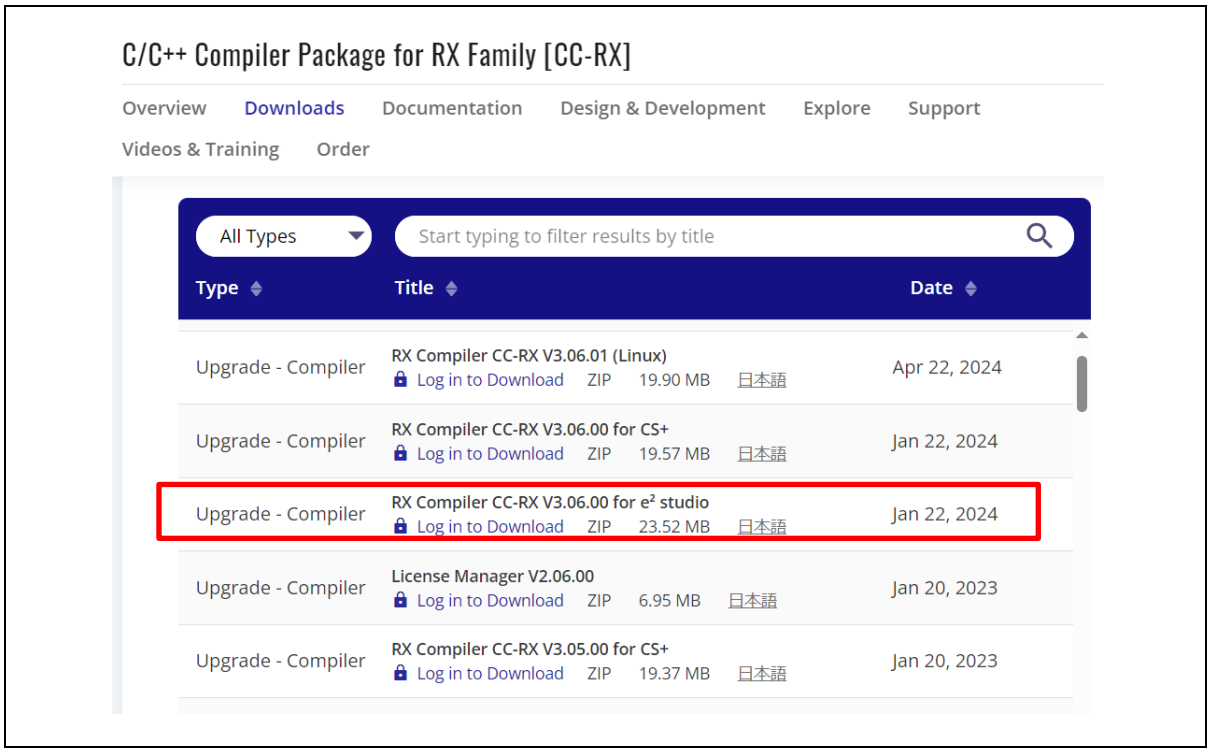

Figure 4-21 CC-RX Page

Run the downloaded installer to install the complete set of components. After installation, confirm that the relevant compiler has been added in e2studio's toolchain management dialog (Refer to Chapter 3.3.3)

Revision History

Rev.	Date	Description	
		Chapter	Summary
1.00	Oct/15, 2021	-	First Edition
1.01	Jan/11, 2022	3.4	Add Remote I/O sample application
		3.4.4	Add Modbus TCP sample application
		3.3.1	Add RX family C/C++ Compiler (CC-RX)
1.02	Aug/5, 2022	3.4.1.8 3.4.1.9	Add web saver function sample
		4.4	Add section of Big-endian support
1.03	May/31, 2023	3.4	Review of description with sample program update
1.04	Dec/15, 2023	Related Doc.	Add application note for TwinCAT
		1.2	Add FIT module version
		3.3.3	Add instruction text for legacy GCC package download
		3.3.4	Add instruction text for legacy FIT module download
		3.4.1	Update figure of GSDML appearance, add instruction text
		3.4.2	Add instruction about Mirror (RPC)
		3.4.3	Add instruction about transmission/reception application
		3.4.6	Update figure of webserver appearance
		4.5	Add instruction for manual installation of GCC
		4.6	Add instruction for manual installation of FIT
1.05	May/31, 2024	1.2	Update each version in Table 1-1
		3.3.1	Replace Figure 3-2 and Figure 3-5
		3.3.3	Replace Figure 3-7 and update compiler version
		3.3.4	Replace Figure 3-16 and update FIT version
		3.4.1	Update GSD file name
		3.4.6	Delete Modbus TCP from Evaluation Environment Setup
		4.5	Replace Figure 4-17 and update compiler version
		4.6	Replace Figure 4-18 and update FIT version
		4.7	Add chapter 4.7

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/.