

Renesas RA Family

Modbus TCP Sample Program

Introduction

This application note evaluates Modbus® communication with the RA microcomputer evaluation board. Modbus protocol is a communication protocol developed by Modicon Inc. (Schneider Electric SA.) for programmable logic controllers (PLCs), and its specifications are open to the public. For details, refer to the protocol specifications (PI-MBUS-300 Rev.J).

Target Device

RA6M3, RA6M4, RA6M5

RA8D1, RA8M1, RA8T1

RA8T2, RA8D2, RA8P1, RA8M2

Supported Evaluation Boards

EK-RA6M3, EK-RA6M4, EK-RA6M5

EK-RA8D1, EK-RA8M1, MCK-RA8T1

MCK-RA8T2, EK-RA8D2, EK-RA8P1, EK-RA8M2, EK-RA8T2

Contents

1. Overview	4
1.1 Abbreviations/Definitions	4
1.2 References	5
2. Features	7
3. Sample Program Package Configuration	8
3.1 Modbus Sample Project	8
3.2 Modbus Sample Application	8
3.3 Modbus Demo Application	8
4. Operating Environment Requirements	9
5. Evaluation Board Connection Setup	10
6. Setting Up the Modbus Sample Project	13
6.1 Import Modbus Sample Project	13
6.1.1 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Importing Procedures	13
6.1.2 MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2 Importing Procedures	19
6.2 Creation of New Modbus Project	21
6.2.1 Common Procedures for all Evaluation Boards	21
6.2.2 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Creating Procedures	35
6.2.3 MCK-RA8T2 Creating Procedures	39
6.2.4 EK-RA8D2 Creating Procedures	48
6.2.5 EK-RA8P1 Creating Procedures	56
6.2.6 EK-RA8M2 Creating Procedures	65
6.2.7 EK-RA8T2 Creating Procedures	71
6.3 Changing of Configuration	78
7. Execution of Modbus Sample Project	80
8. Modbus Communication Using Modbus Demo Application	83
8.1 IP Address Setting	83
8.2 Setting up the Modbus Demo Application	84
8.3 Modbus Demo Application Specification	85
9. Appendix	86
9.1 Appendix A: Modbus Protocol Stack Configuration	86
9.2 Appendix B: IP List Related Parameters	88
9.3 Appendix C: DHCP Mode	89
9.4 Appendix D: User-defined Function	90
9.4.1 Register Function Code	90
9.4.2 User-defined Functions	90

9.5 Appendix E: Multiple Client Communication 92

9.6 Appendix F. Application Programming Interface..... 93

Revision History95

1. Overview

This document describes a sample program that uses the Modbus TCP protocol stack on an RA evaluation board. It provides an overview of the program's functions and demonstrates how to develop and implement applications using the protocol stack.

This sample program package supports the Ethernet-based Modbus TCP protocol.

This application note provides:

- Sample program package configuration
- Operating environment requirements
- Evaluation board connection setup
- Procedure for creating, modifying, and building the Modbus sample project using the Flexible Software Package (FSP) and e² studio Integrated Development Environment (IDE).
- Instructions for connecting with a client and operating a simple demo.

1.1 Abbreviations/Definitions

Table 1.1 Abbreviations/Definitions

Index	Abbreviations /Definitions	Description
1	IP	Internet Protocol
2	TCP	Transmission Control Protocol
3	USB	Universal Serial Bus
4	PC	Personal Computer
5	SW	Switch
6	EWARM	Embedded Workbench® for Arm
7	LED	Light Emitting Diode
8	Wireshark	Free packet capture tool to check packets flowing on LAN

1.2 References

Technical information about Modbus is available through the Modbus Organization website. Information on the RA evaluation board is available from Renesas.

Table 1.2 Technical Inputs (1/2)

Index	Technical Inputs
1	Modbus Application Protocol Specification Vxxx
2	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 Quick Start Guide / r20qs0011euxxxx
3	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 v1 User's Manual / r20ut4623euxxxx
4	Evaluation Kit for RA6M4 Microcontroller Group EK-RA6M4 Quick Start Guide / r20qs0016egxxxx
5	Evaluation Kit for RA6M4 Microcontroller Group EK-RA6M4 v1 User's Manual / r20ut4836egxxxx
6	Evaluation Kit for RA6M5 Microcontroller Group EK-RA6M5 Quick Start Guide / r20qs0021egxxxx
7	Evaluation Kit for RA6M5 Microcontroller Group EK-RA6M5 v1 User's Manual / r20ut4829egxxxx
8	Evaluation Kit for RA8D1 Microcontroller Group EK-RA8D1 Quick Start Guide / r20qs0065egxxxx
9	Evaluation Kit for RA8D1 Microcontroller Group EK-RA8D1 v1 User's Manual / r20ut5205egxxxx
10	Evaluation Kit for RA8M1 Microcontroller Group EK-RA8M1 Quick Start Guide / r20qs0035egxxxx
11	Evaluation Kit for RA8M1 Microcontroller Group EK-RA8M1 v1 User's Manual / r20ut5149egxxxx
12	MCK-RA8T1 Quick Start Guide / r12qs0067ejxxxx
13	MCK-RA8T1 User's Manual / r12uz0133ejxxxx
14	MCK-RA8T2 Quick Start Guide / r12qs0088ejxxxx
15	MCK-RA8T2 User's Manual / r12uz0172ejxxxx
16	Evaluation Kit for RA8D2 Microcontroller Group EK-RA8D2 v1 Quick Start Guide / r20qs0077egxxxx
17	Evaluation Kit for RA8D2 Microcontroller Group EK-RA8D2 v1 User's Manual / r20ut5523egxxxx
18	Evaluation Kit for RA8P1 Microcontroller Group EK-RA8P1 v1 Quick Start Guide / r20qs0051egxxxx
19	Evaluation Kit for RA8P1 Microcontroller Group EK-RA8P1 v1 User's Manual / r20ut5309egxxxx
20	Evaluation Kit for RA8M2 Microcontroller Group EK-RA8M2 v1 Quick Start Guide / r20qs0069egxxxx
21	Evaluation Kit for RA8M2 Microcontroller Group EK-RA8M2 v1 User's Manual / r20ut5451egxxxx

Table 1.3 Technical Inputs (2/2)

Index	Technical Inputs
22	Evaluation Kit for RA8T2 Microcontroller Group EK-RA8T2 v1 Quick Start Guide / r20qs0097egxxxx
23	Evaluation Kit for RA8T2 Microcontroller Group EK-RA8T2 v1 User's Manual / r20ut5714egxxxx

2. Features

The Modbus protocol stack for the RA evaluation board allows for quick and easy development of Modbus TCP applications.

Modbus function codes supported by the initialization API are also specified. The following nine function codes can be implemented in this stack:

- 1(0x01) – Read coils
- 2(0x02) – Read discrete input
- 3(0x03) – Read holding registers
- 4(0x04) – Read input registers
- 5(0x05) – Write single coil
- 6(0x06) – Write single register
- 15(0x0F) – Write multiple coils
- 16(0x10) – Write multiple registers
- 23(0x17) – Read/Write multiple registers

For more information about Modbus, refer to the following site:

<http://www.modbus.org>

Note: The version number may differ depending on the update. Refer to the latest manual.

3. Sample Program Package Configuration

This sample program package consists of three components:

- Modbus sample project using the Modbus protocol stack.
- Modbus sample application using Modbus protocol stack.
- Modbus sample demo application

3.1 Modbus Sample Project

- Modbus_TCP / project / EK-RA6M5
 - This folder contains a folder called "RA_Modbus" which includes a Modbus sample project for EK-RA6M5 boards that uses the Modbus protocol stack.
 - The sample project is created for GCC compilers. To use another compiler, refer to section "[6.2 Creation of new Modbus Project](#)" and create another project.
 - The sample project is created for EK-RA6M5. To run it on EK-RA6M3, EK-RA6M4, EK-RA8D1, EK-RA8M1, MCK-RA8T1, refer to step (4) of "[6.1 Import Modbus Sample Project](#)" and modify the project with respect to different boards.
- Modbus_TCP / project / MCK-RA8T2
- Modbus_TCP / project / EK-RA8D2
- Modbus_TCP / project / EK-RA8P1
- Modbus_TCP / project / EK-RA8M2
- Modbus_TCP / project / EK-RA8T2
 - These folders contain a folder called "RA_Modbus" which includes a Modbus sample project for MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2 boards that uses the Modbus protocol stack.
 - The sample projects are created for GCC compilers. To use another compiler, refer to section "[6.2 Creation of new Modbus Project](#)" and create another project.

3.2 Modbus Sample Application

- Modbus_TCP / src / modbus_func.c, modbus_user.c, new_thread0_entry.c
 - Users can register their own implementations of Modbus function codes in the Modbus protocol stack.
 - The code in this directory provides examples of the Modbus protocol stack initialization process and the Modbus function codes processing using the Modbus protocol stack API.

3.3 Modbus Demo Application

- Modbus_tool / ModbusDemoApplication.exe
 - This executable file is a Modbus demo application used for Modbus communication. It can be used to demonstrate the operation of the Modbus sample application.

4. Operating Environment Requirements

The sample program package described in this document runs in the following environment.

Table 4.1 **Operating environment**

Item	Description
Board	RA evaluation board
Integrated development environment	IAR Systems - IAR Embedded Workbench® for Arm Version 9.70.4 or later Renesas Electronics - e² Studio 2026-07 or later - Renesas RA Smart Configurator 2026-07 or later
Toolchain	IAR Embedded Workbench for Arm - IAR C/C++ Compiler for Arm 9.70.4 or later e² Studio - GCC Arm Embedded (13.2.1.arm-13-7) or later - LLVM for Arm (22.1.0) or later - Arm Compiler 6.24 or later
MCU software package	FSP (Flexible Software Package) v6.6.0 or later
Emulator	J-LINK® OB
Communications protocol	Modbus TCP
Client tool	ModbusDemoApplication.exe: Modbus Demo Application

5. Evaluation Board Connection Setup

Connect the PC to a supported evaluation board. Power is supplied by connecting a USB micro-B cable to the board. For Modbus TCP communication, use an RJ45 connector and connect to the PC with a LAN cable.

The connection setting diagrams for the EK RA8M1 evaluation board, MCK-RA8T1 board, MCK-RA8T2, EK-RA8T2 board, and Switch SW4 setting table are shown below.

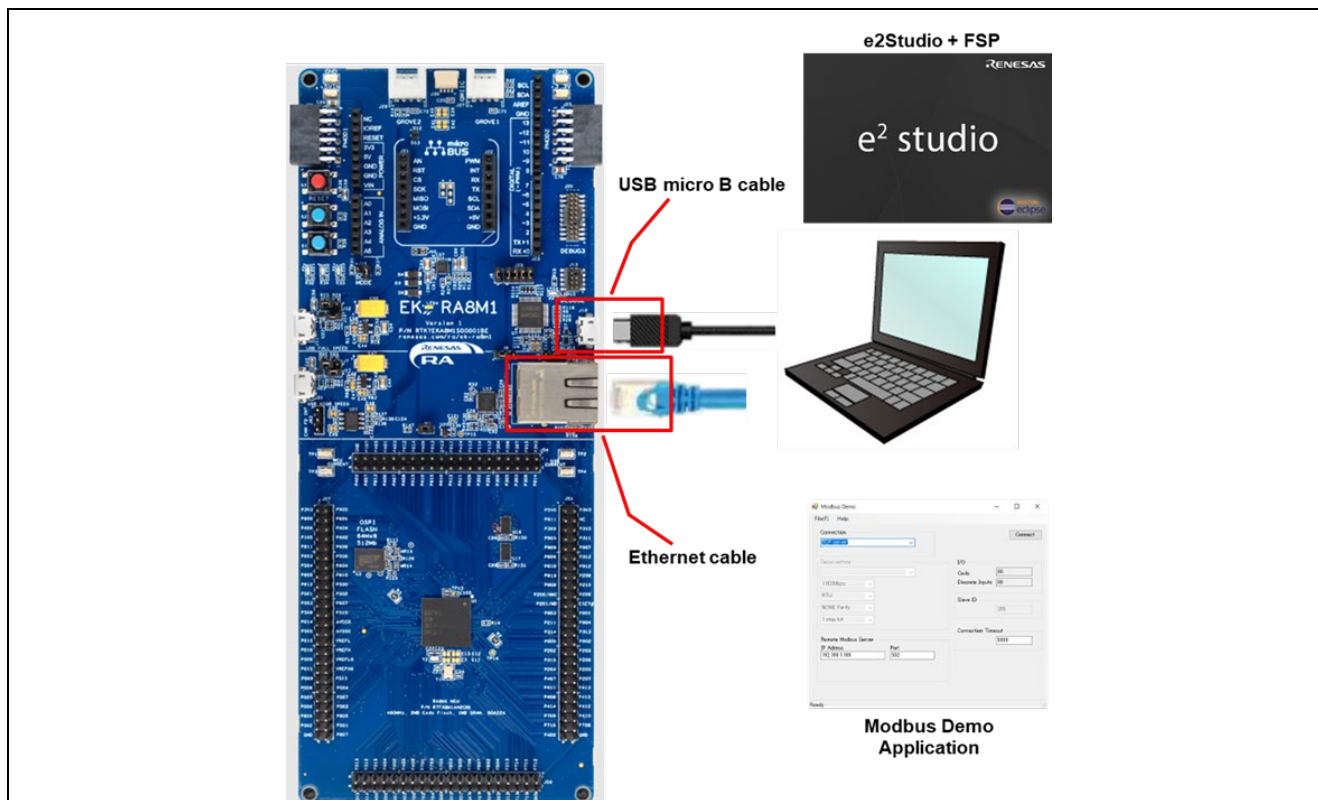


Figure 5.1: EK evaluation board connection setup for RA (Example: EK-RA8M1)

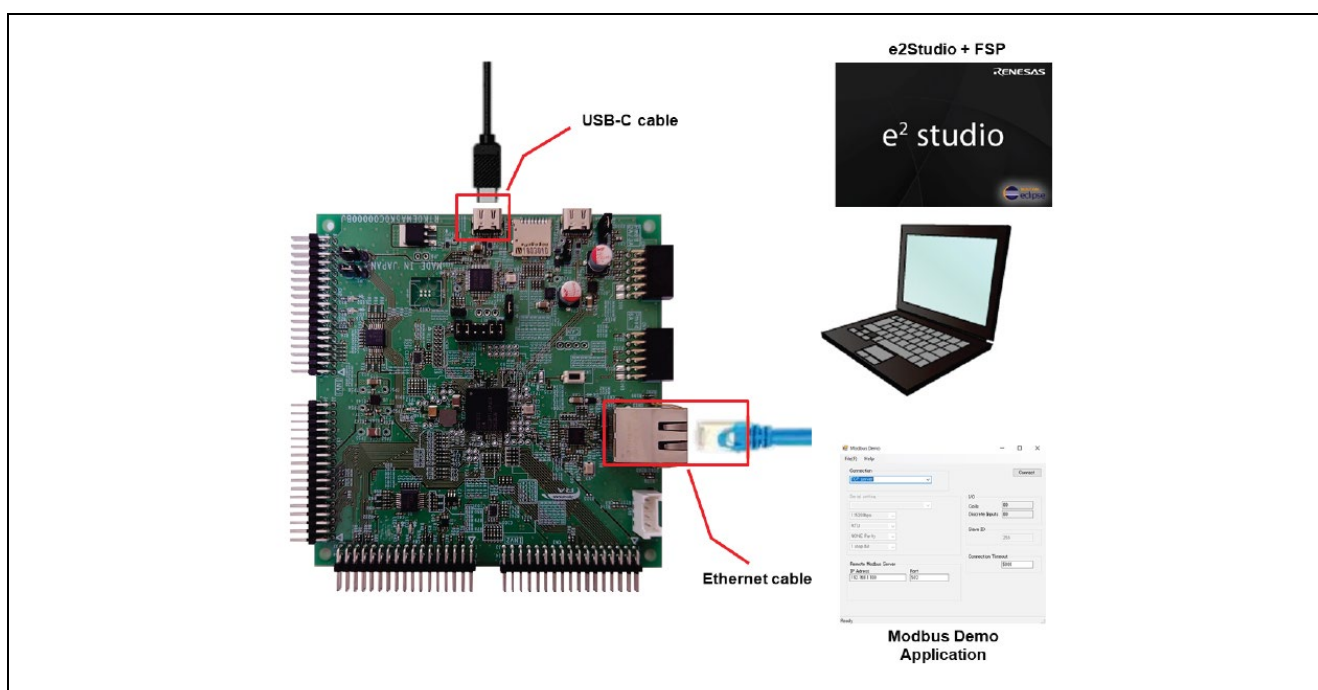


Figure 5.2: MCK-RA8T1 board connection setup

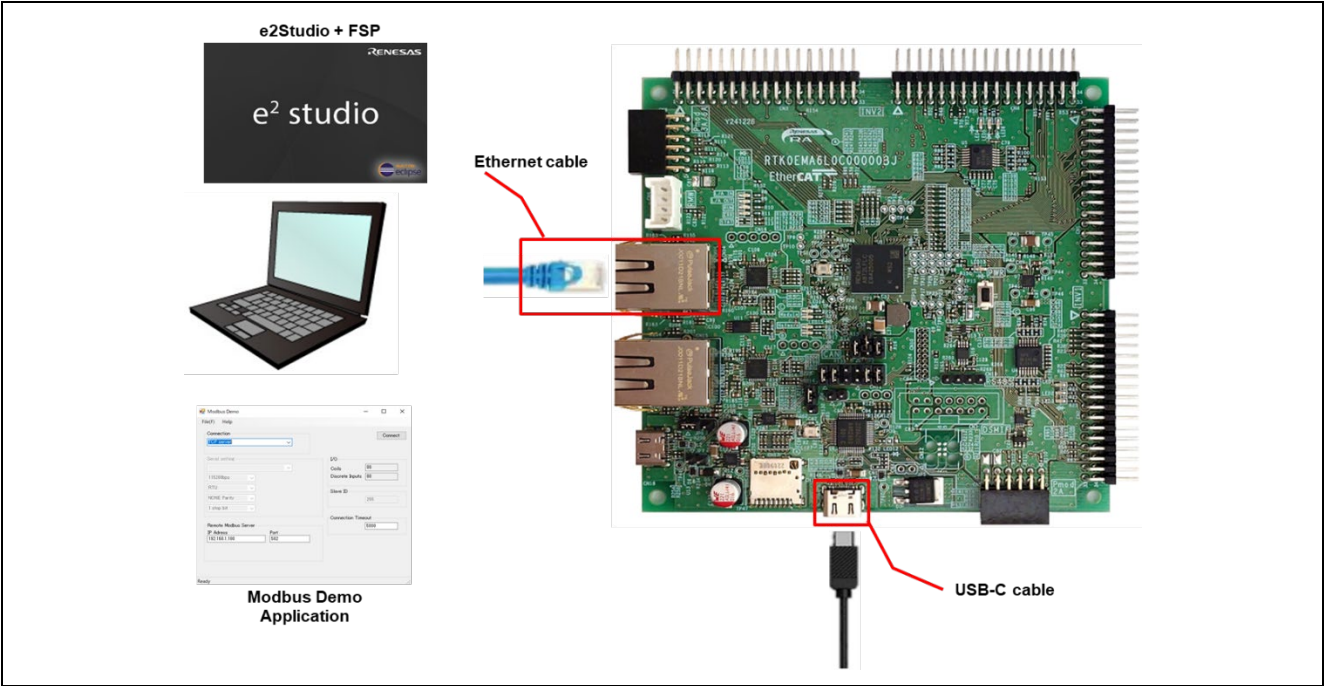


Figure 5.3: MCK-RA8T2 board connection setup

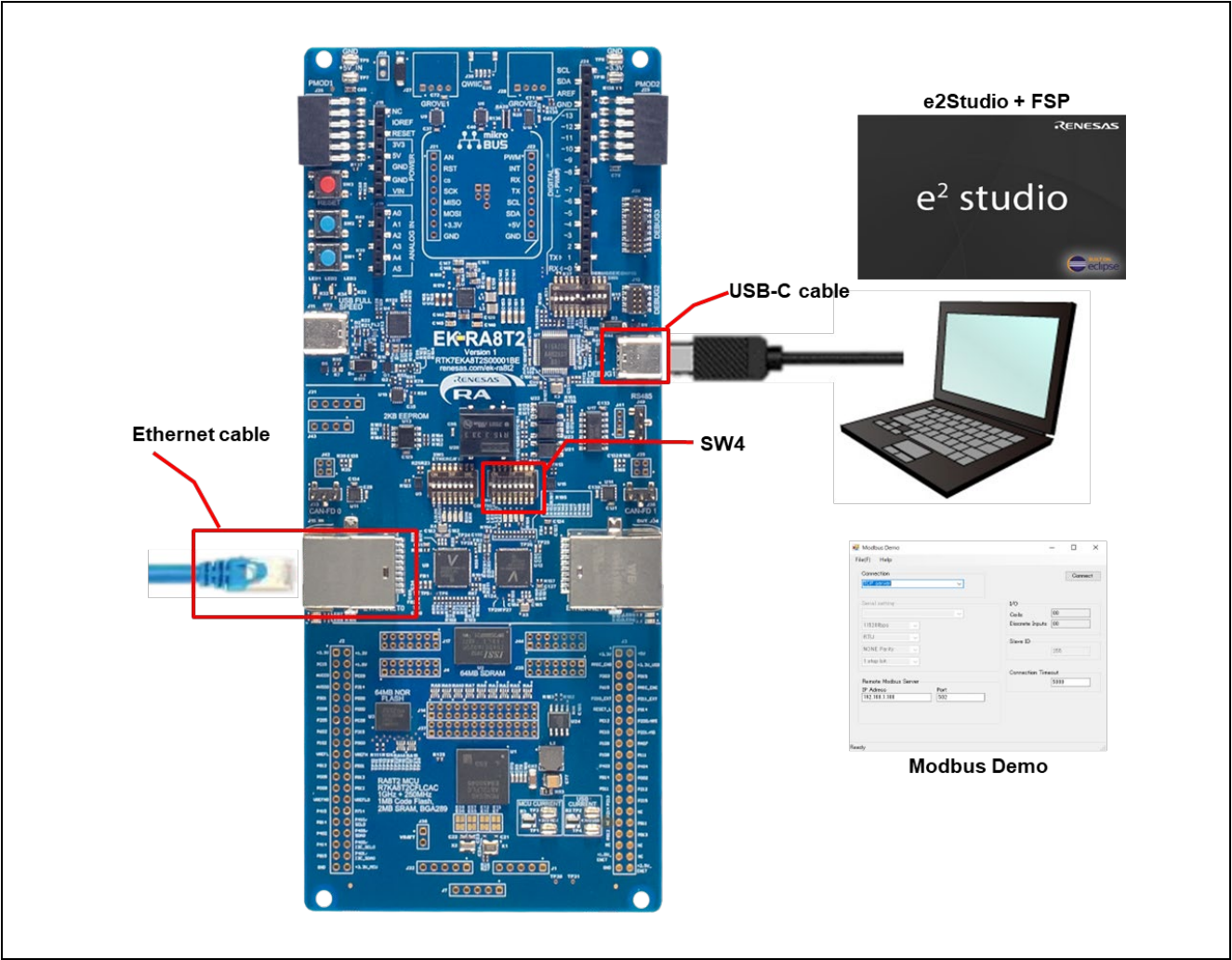


Figure 5.4: EK-RA8T2 board connection setup

Table 5.1 Switch SW4 setting

SW4	Setting	Description
SW4-6	OFF	Ethernet PHY COMA_MODE pin set low

6. Setting Up the Modbus Sample Project

This section describes the procedure for importing/creating a Modbus sample project. Before proceeding, read section "4. Operating Environment Requirements" first and complete the installation of the tools.

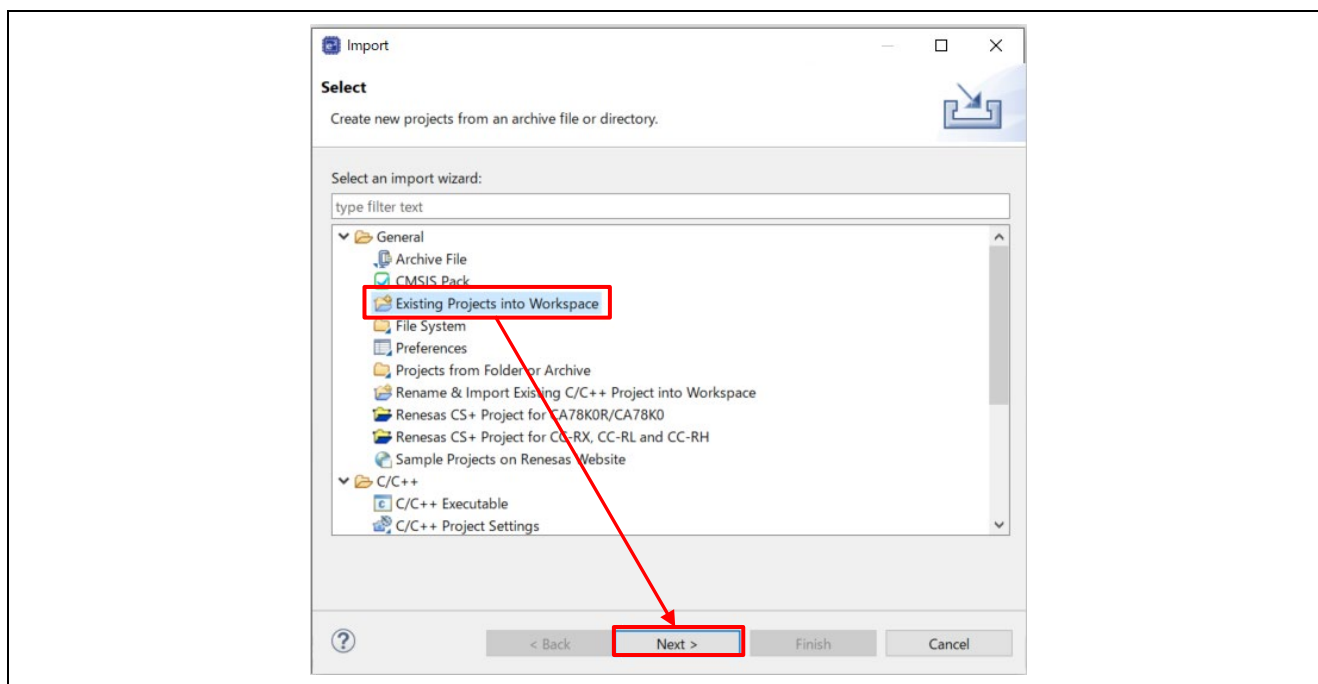
6.1 Import Modbus Sample Project

This section describes the procedure for importing the Modbus sample project and changing the evaluation board.

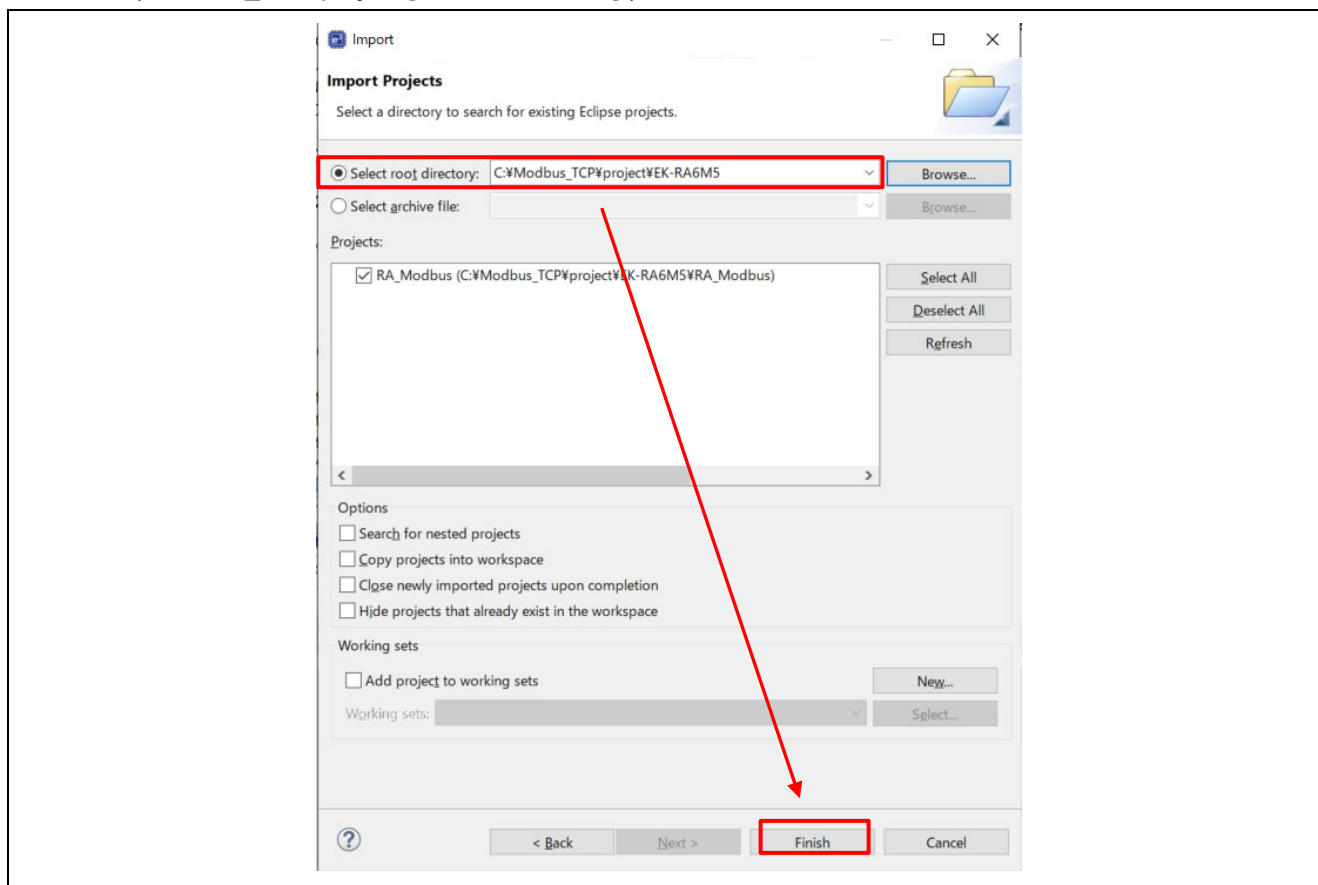
6.1.1 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Importing Procedures

This section describes the procedures for importing EK-RA6Mx (EK-RA6M3, EK-RA6M4, EK-RA6M5) / EK-RA8x1 (EK-RA8D1, EK-RA8M1) / MCK-RA8T1.

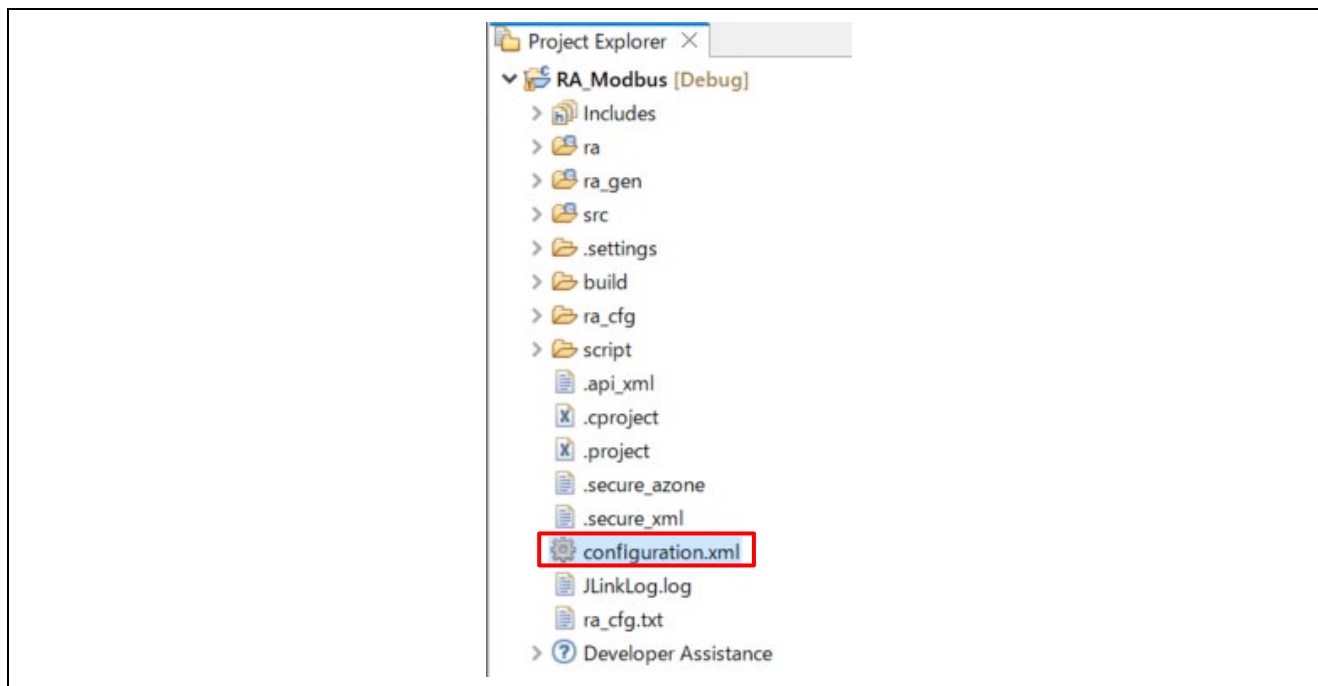
1. Import the sample project. Start e² studio, select [File] → [Import], and when the Import window appears, select [General] → [Existing Projects into Workspace].



Click "Select root directory" and select the Modbus sample project folder ("Modbus_TCP/project/[evaluation board]").

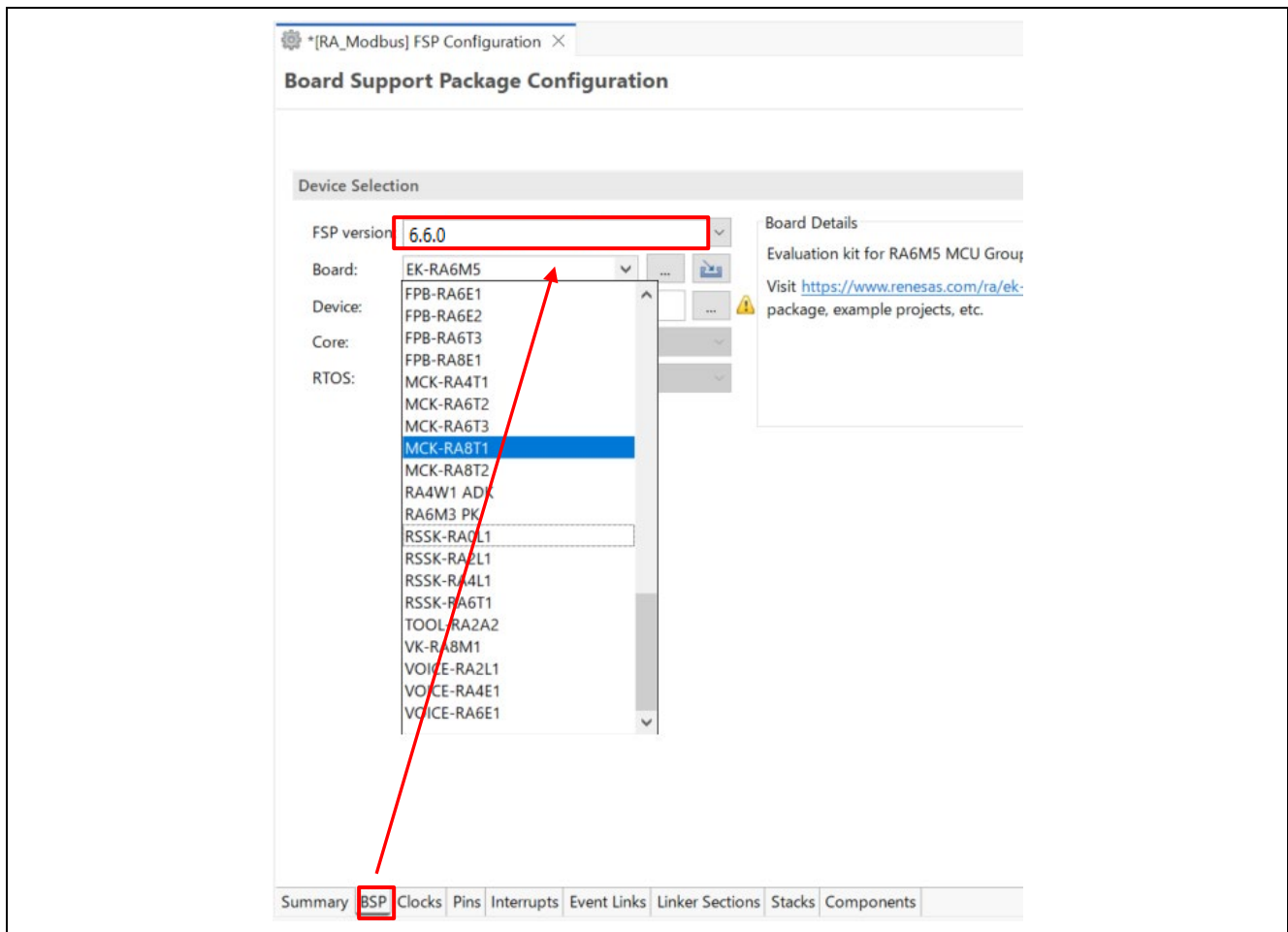


2. Open "configuration.xml" of the "RA_Modbus" project.

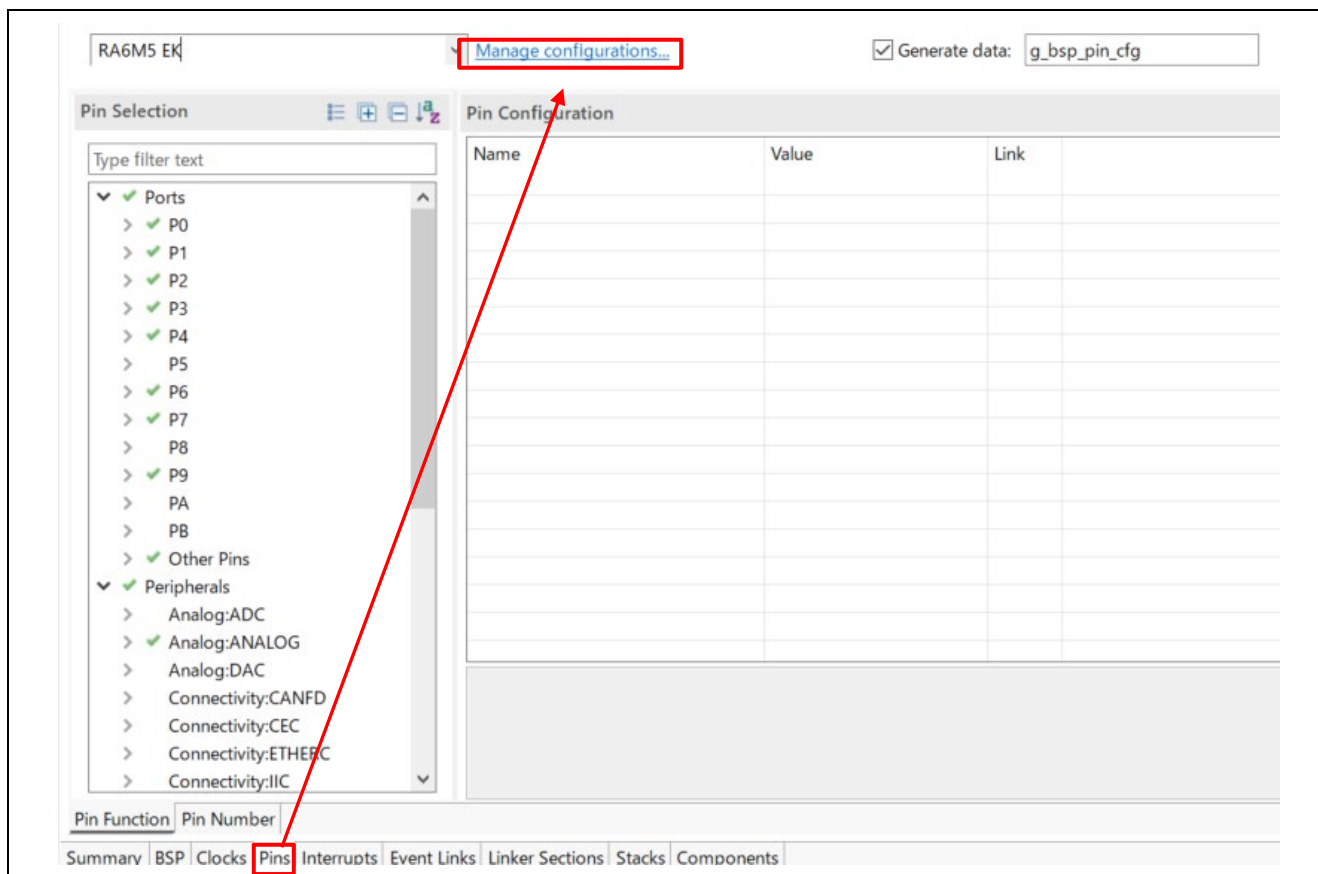


3. The "RA_Modbus" project under the "Modbus_TCP/project/EK-RA6M5" folder is created for EK-RA6M5. To run it on another supported evaluation board, modify the project by following steps (1) to (4) below.

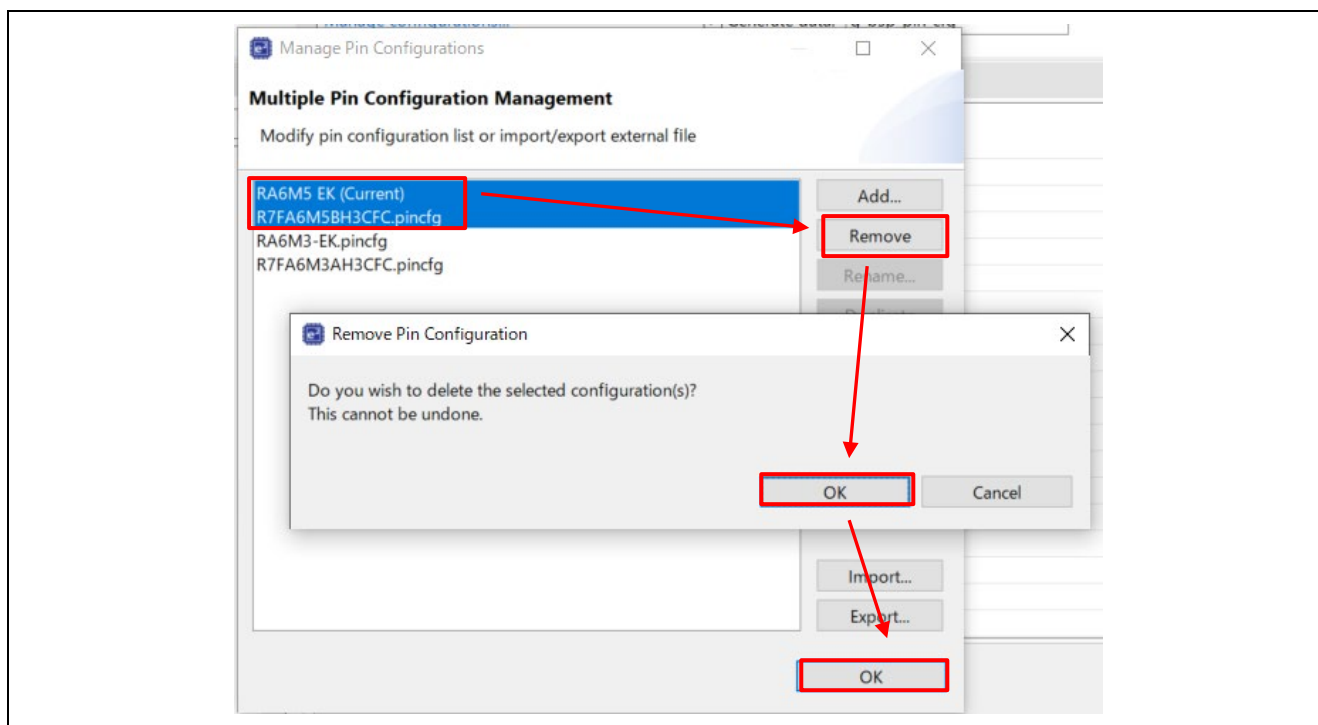
- (1) Click the "BSP" tab, then select "EK-RA6M3", "EK-RA6M4", "EK-RA8D1", "EK-RA8M1", or "MCK-RA8T1" from "Board".



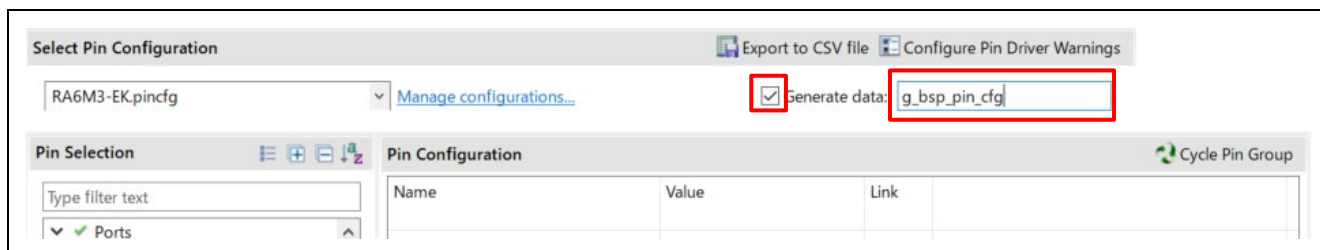
- (2) Click "Manage configurations" on the "Pins" tab.



- (3) Select "RA6M5 EK (Current)" and "R7FA6M5BH3CFC.pincfg" and click "Remove". Click "OK" in the "Remove Pin Configuration" pop-up, then click "OK" in "Manage Pin Configurations".

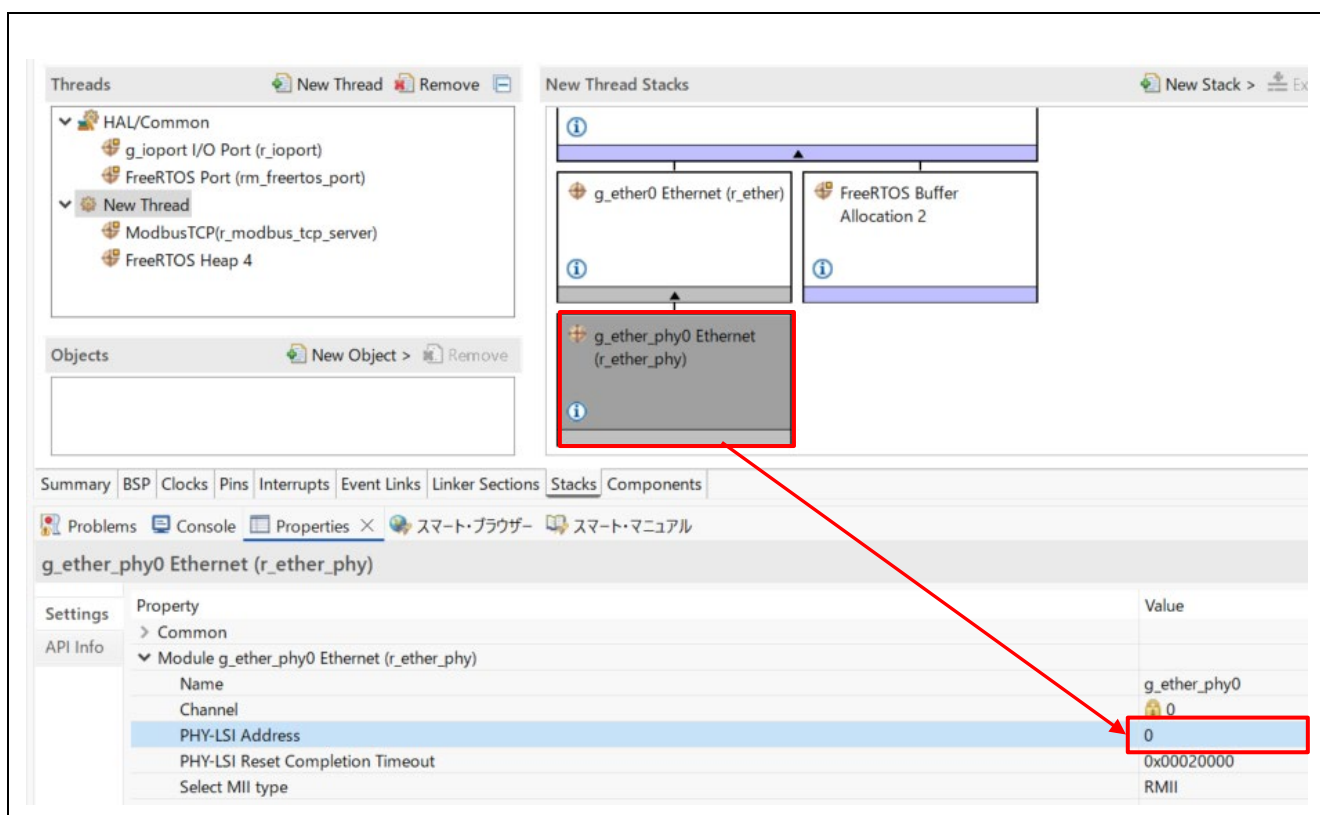


- (4) Click the "Generate data" checkbox and enter "g_bsp_pin_cfg" in the text box.



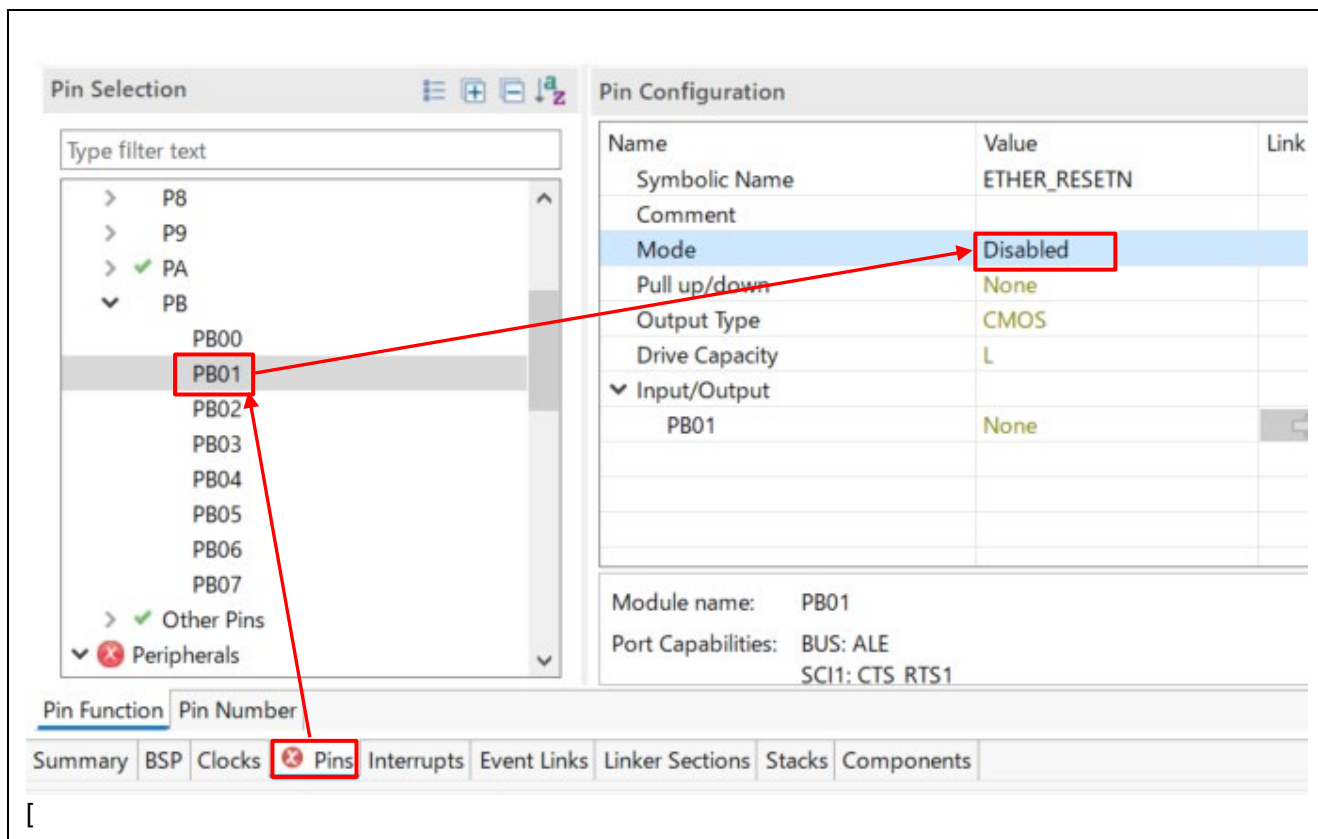
4. For "EK-RA8D1", "EK-RA8M1" and "MCK-RA8T1", change "PHY-LSI Address" in "Stacks" → "g_ether_phy0 Ethernet (r_ether_phy)" → "Module g_ether_phy0 Ethernet (r_ether_phy)" to the following value.

PHY-LSI Address: **5**



5. For "EK-RA8D1" and "MCK-RA8T1", click the "Pins" tab, then select Reset pin [Reset Pin for each evaluation board] (see below), go to "Pin Configuration" and change the "Mode" to "Disabled".

EK-RA8D1: **P706**, MCK-RA8T1: **PB01**

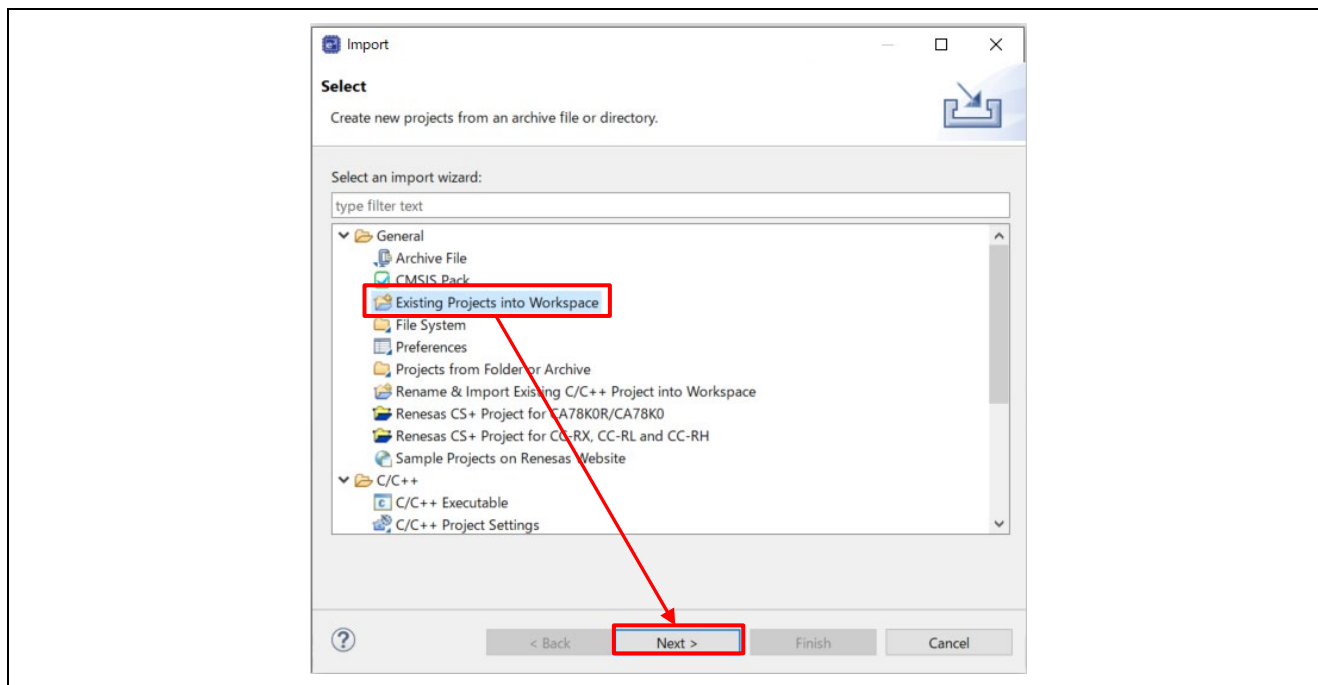


6. Execute the Modbus Sample Project.
Refer to "[7. Execution of Modbus Sample Project](#)" and follow the steps.

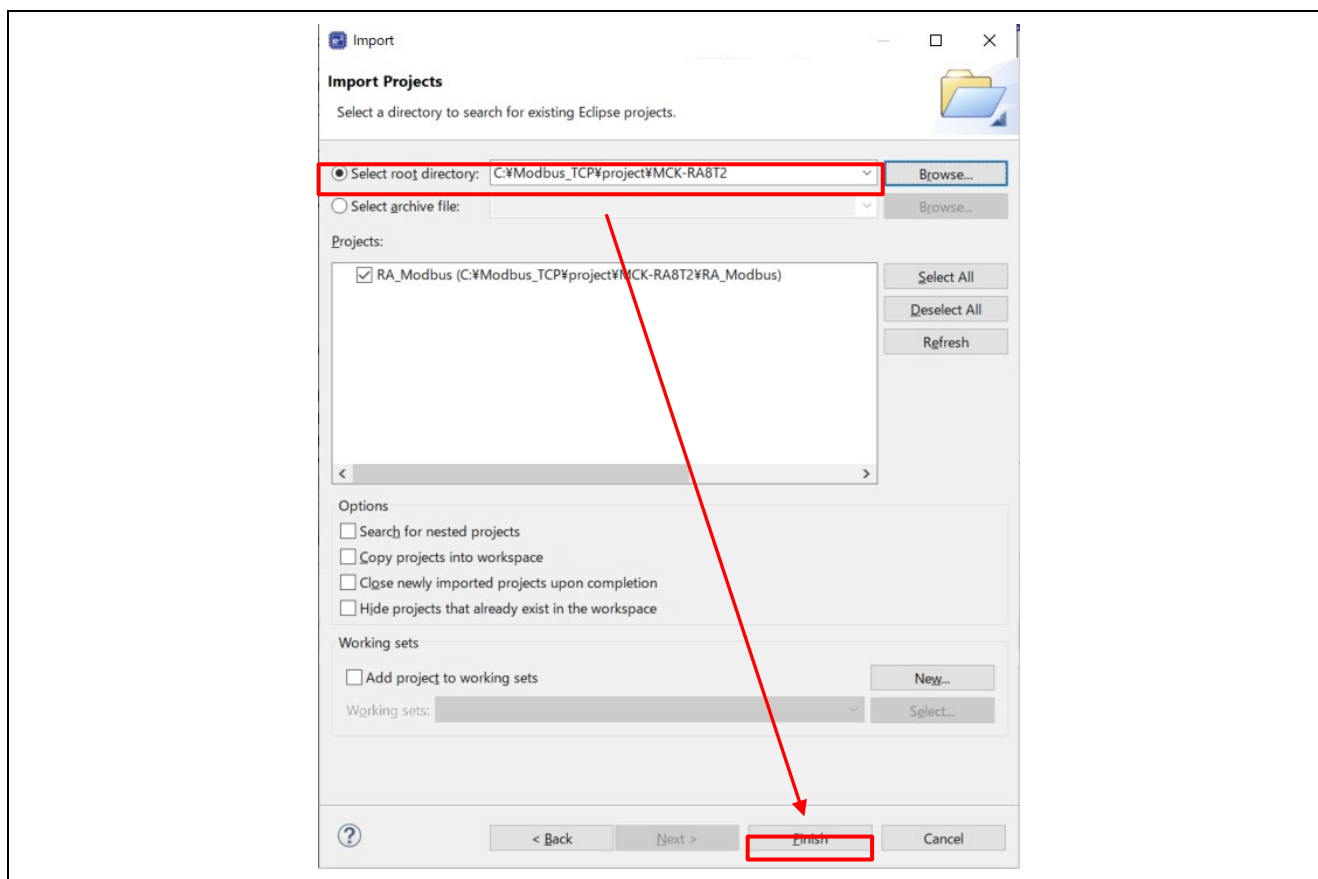
6.1.2 MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2 Importing Procedures

This section describes the procedures for importing MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2.

1. Import the sample project. Start e² studio, select [File] → [Import], and when the Import window appears, select [General] → [Existing Projects into Workspace].



Click "Select root directory" and select the Modbus sample project folder ("Modbus_TCP/project/[evaluation board]").



2. Execute the Modbus Sample Project.
Refer to "[7. Execution of Modbus Sample Project](#)" and follow the steps.

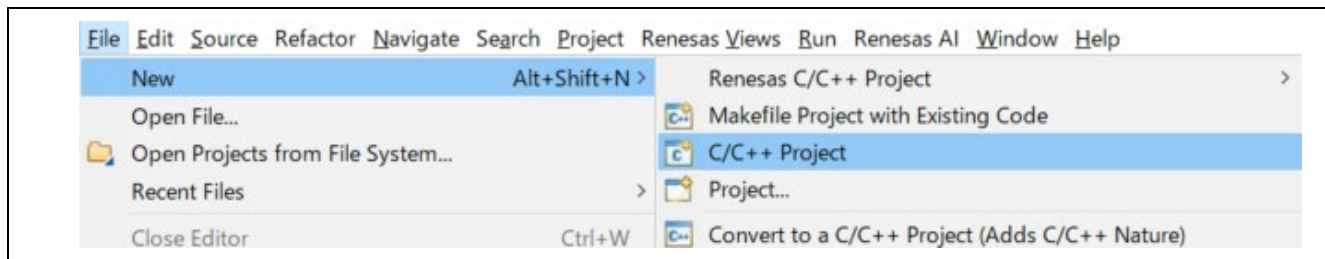
6.2 Creation of New Modbus Project

This section describes the procedure for creating a new Modbus sample project.

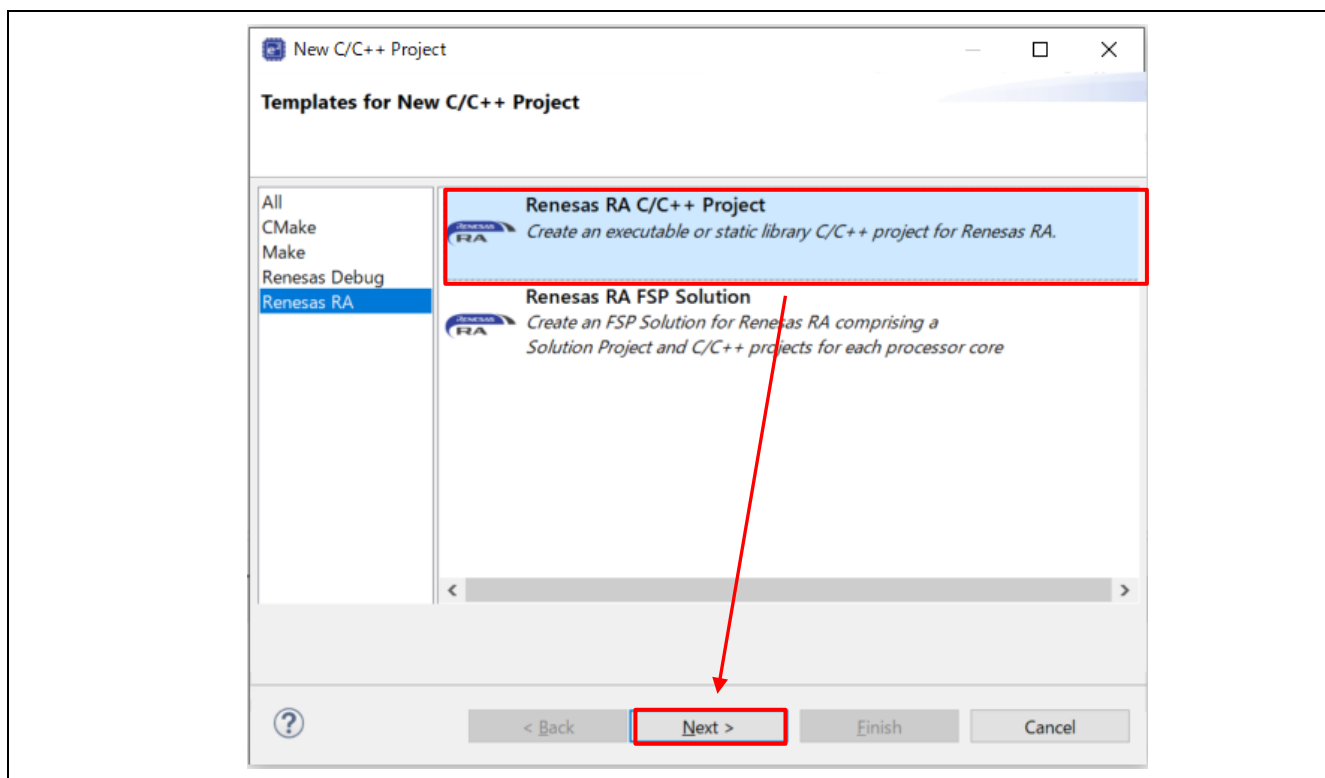
6.2.1 Common Procedures for all Evaluation Boards

This section describes the procedures common to all evaluation boards.

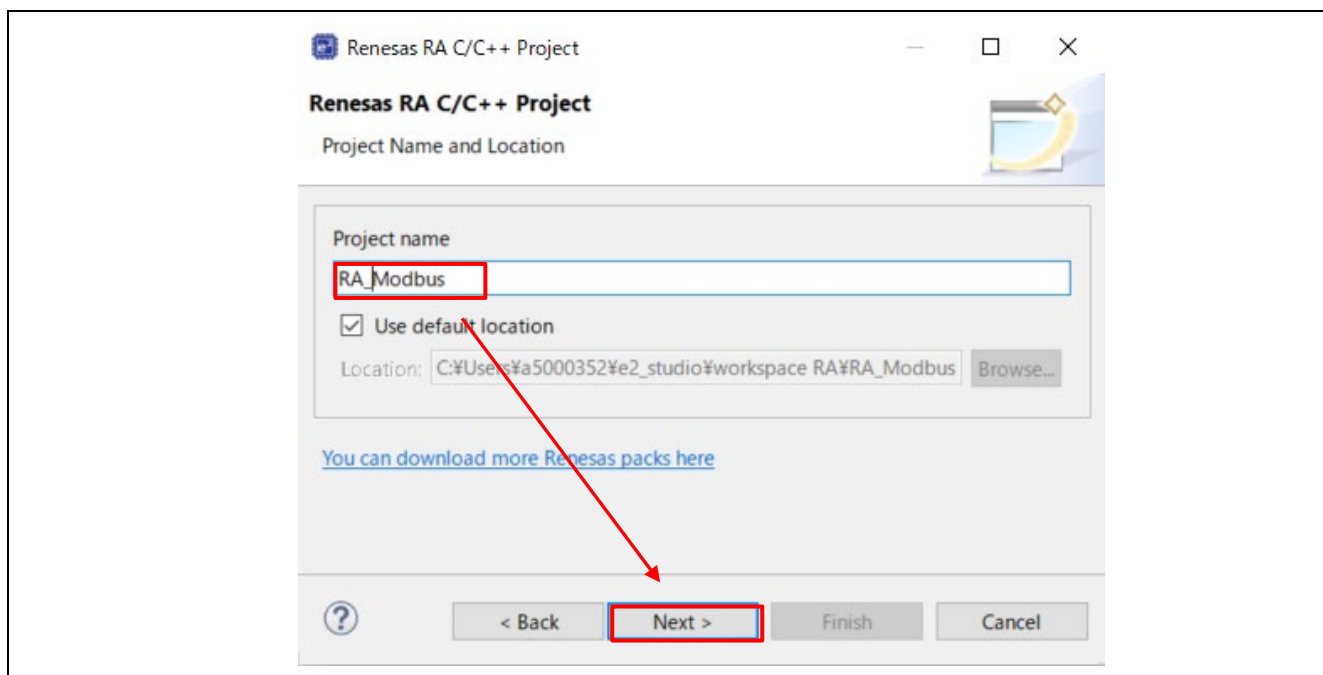
1. Set up a new project. Start e² studio and select [File] → [New] → [C/C++ Project].



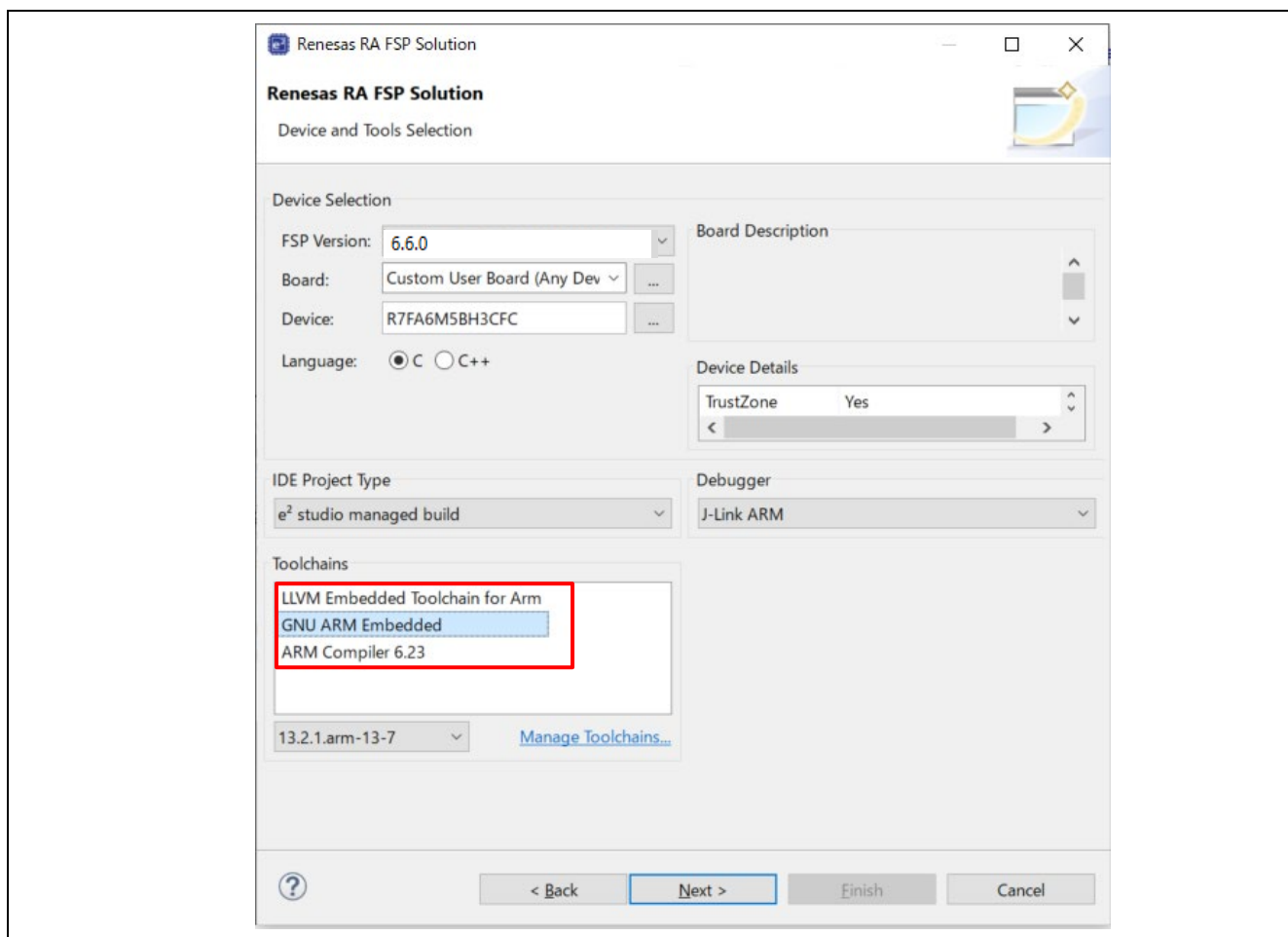
2. Select "Renesas RA C/C++ Project".



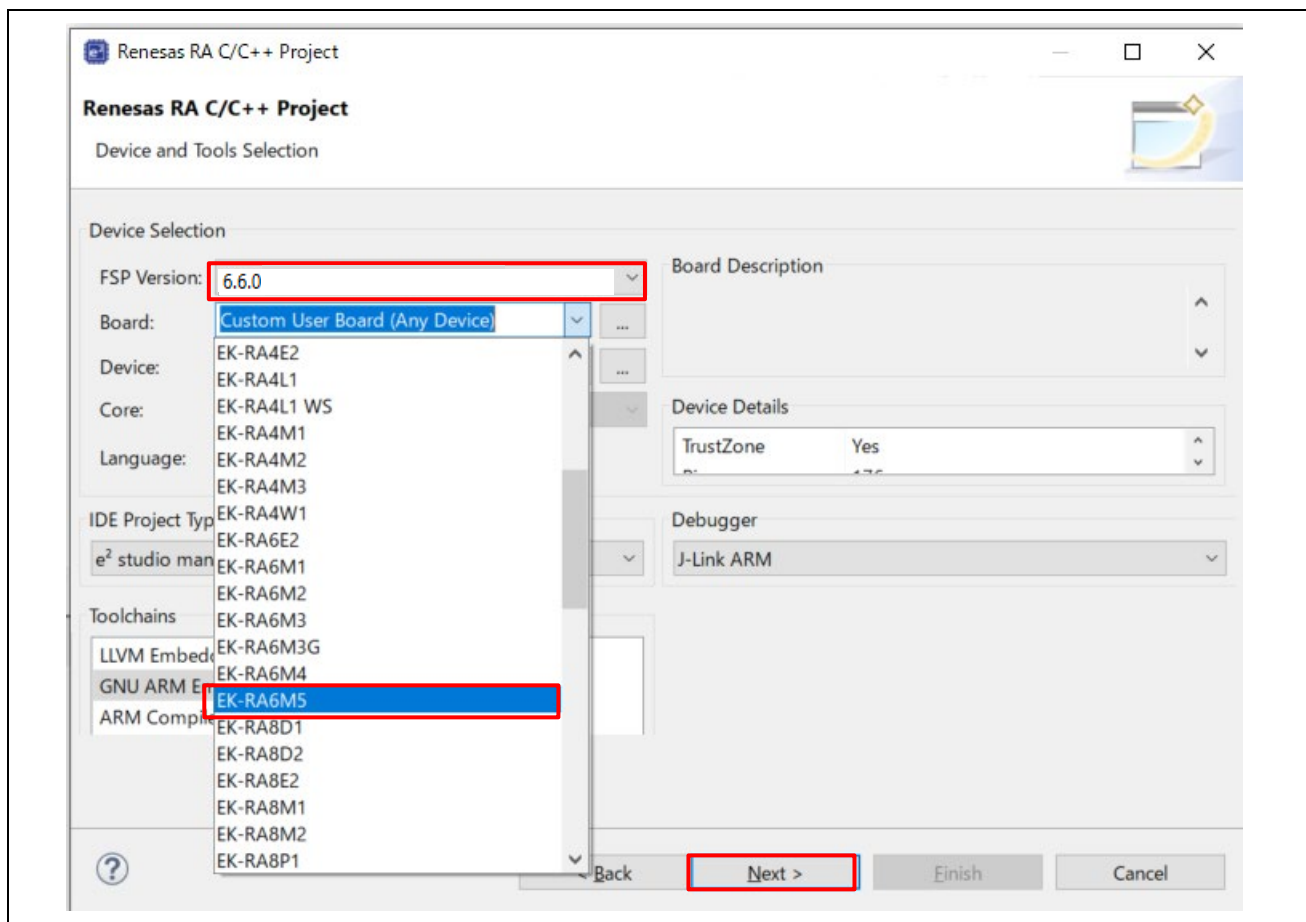
3. Enter any project name.



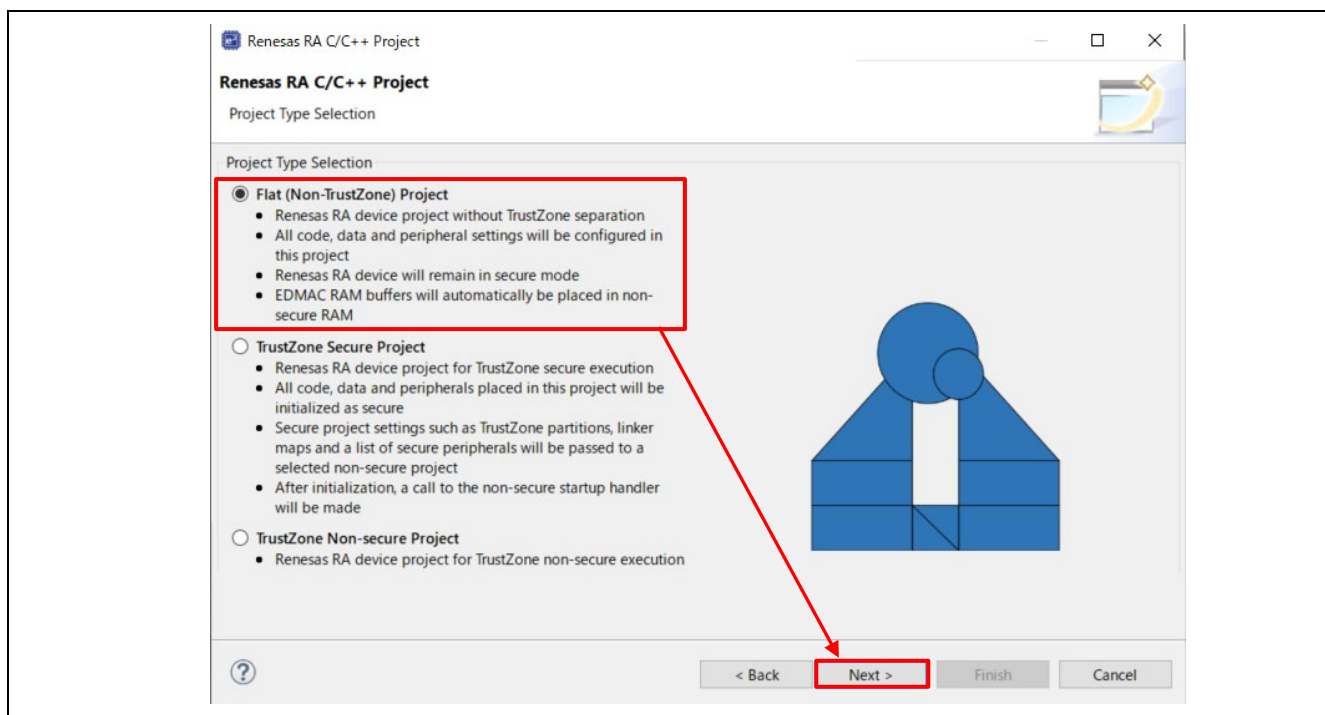
4. Select a toolchain from "Toolchains".



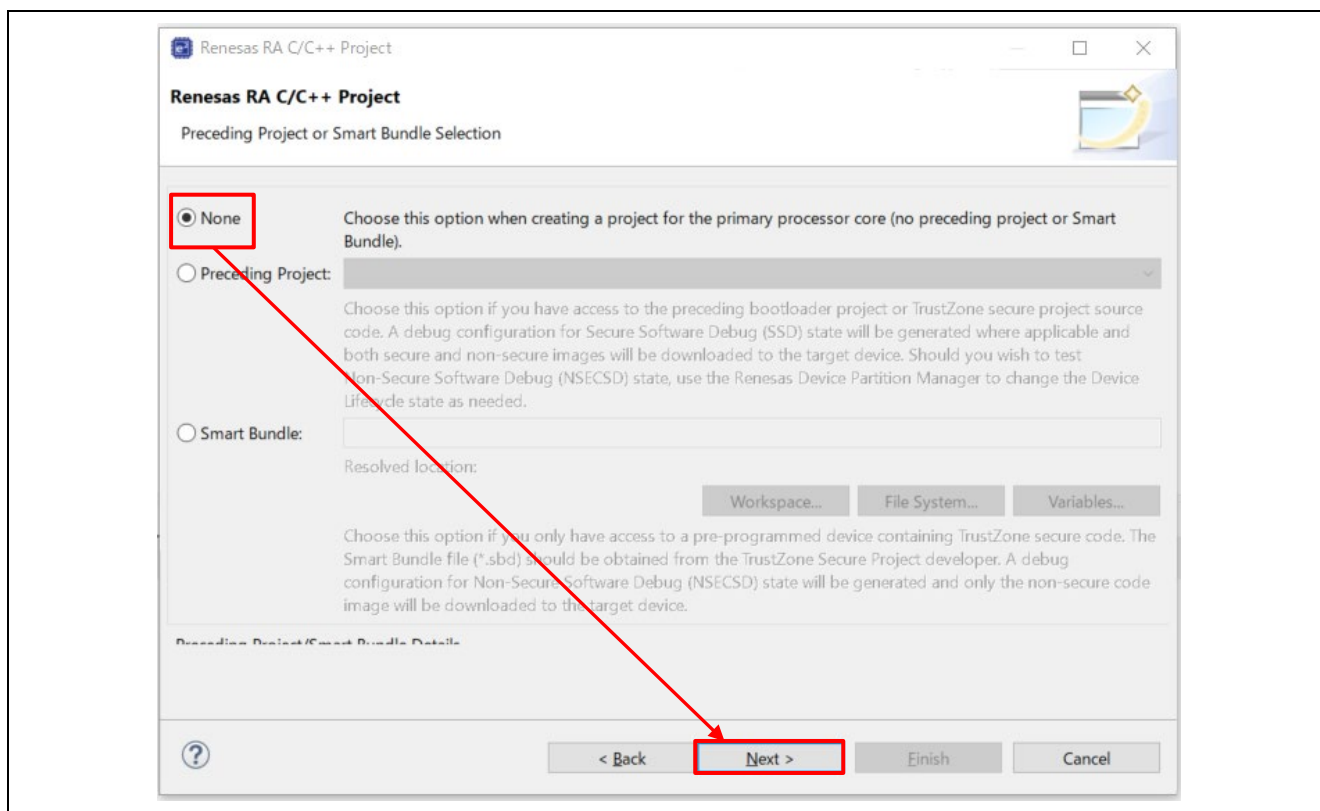
5. Select the FSP Version and Board.



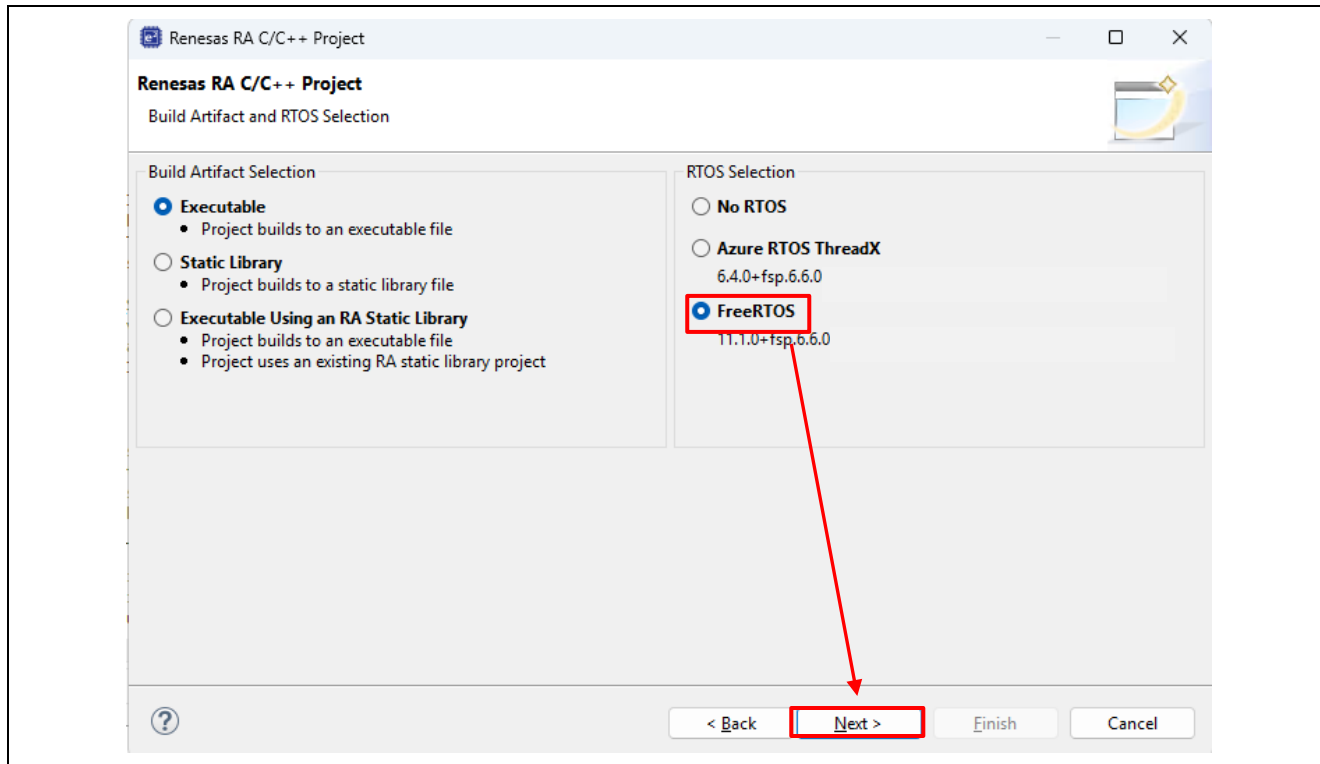
6. Select the project type for "Flat (Non-TrustZone) Project".



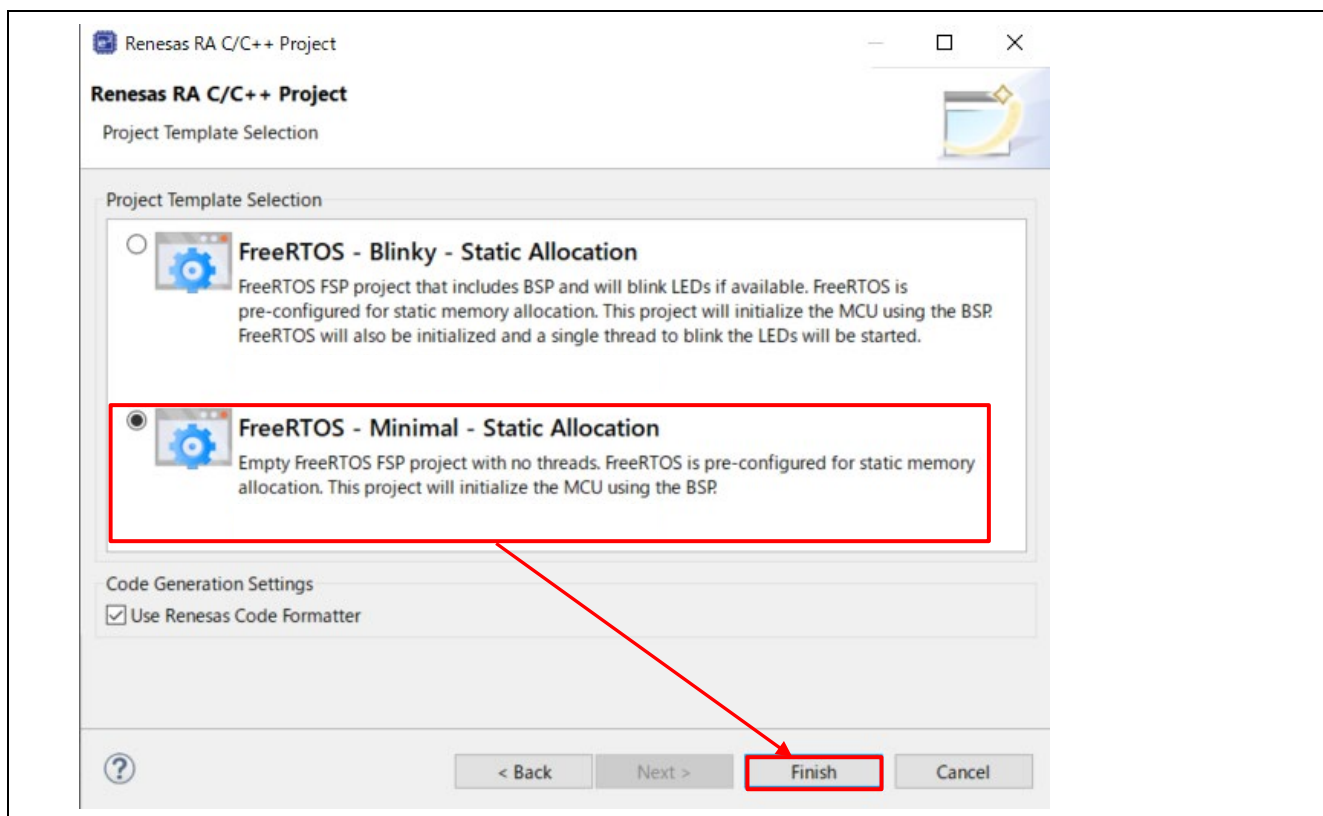
7. Select "None".



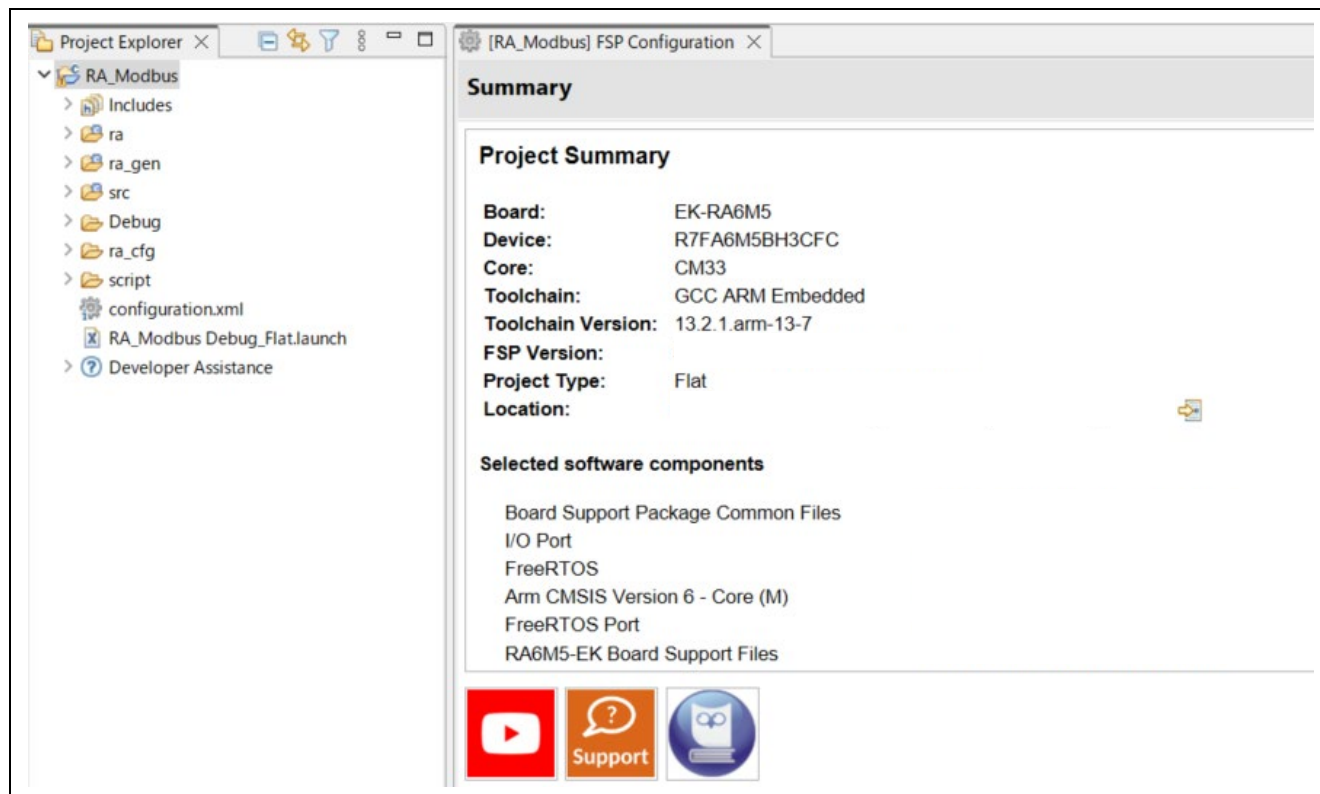
8. Select "FreeRTOS (xxx)".



9. Select "FreeRTOS - Minimal - Static Allocation" under Project Template Selection.

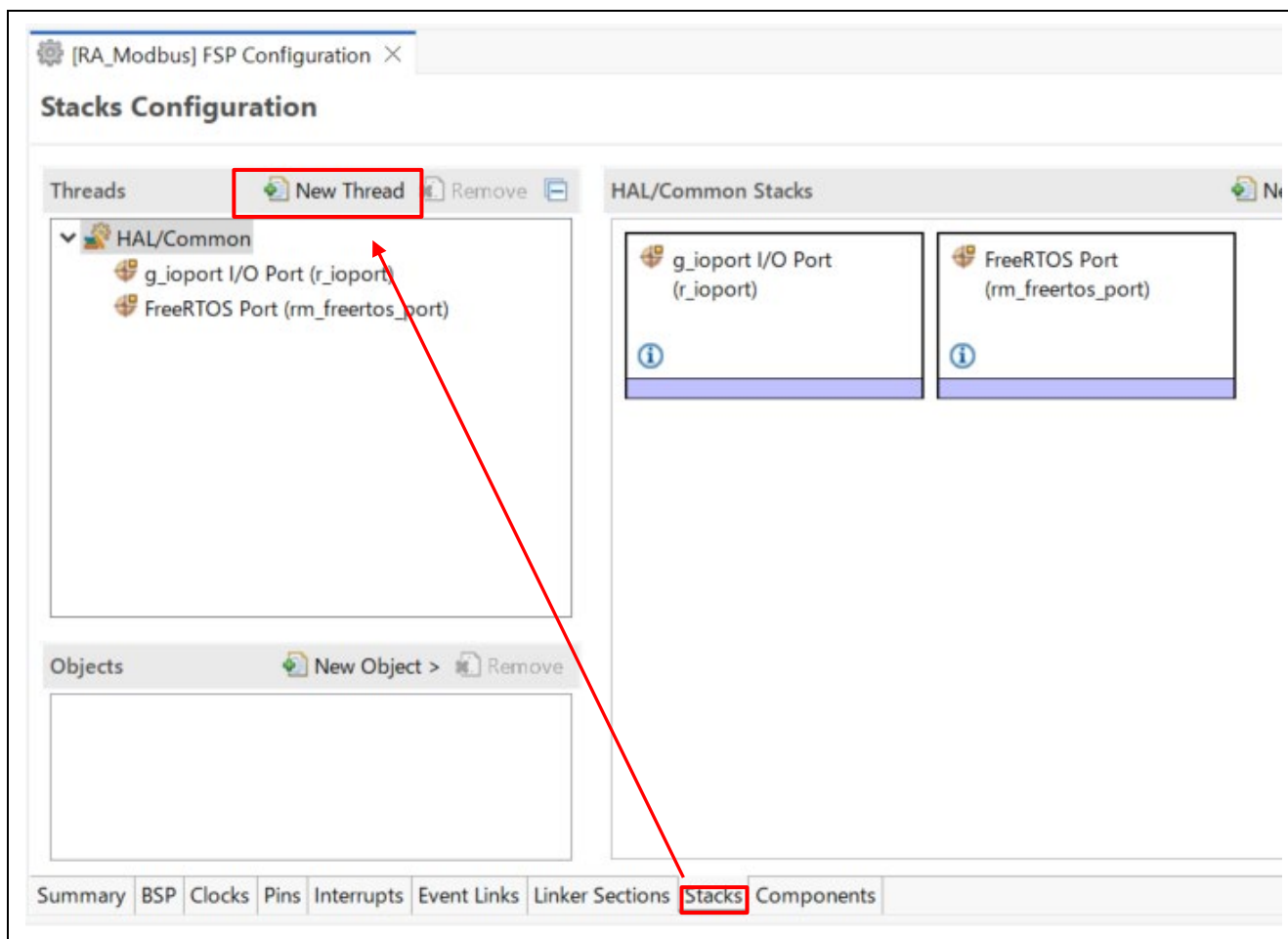


Once the project is created, it appears in the "Project Explorer," and the "FSP Configuration" tab is displayed.

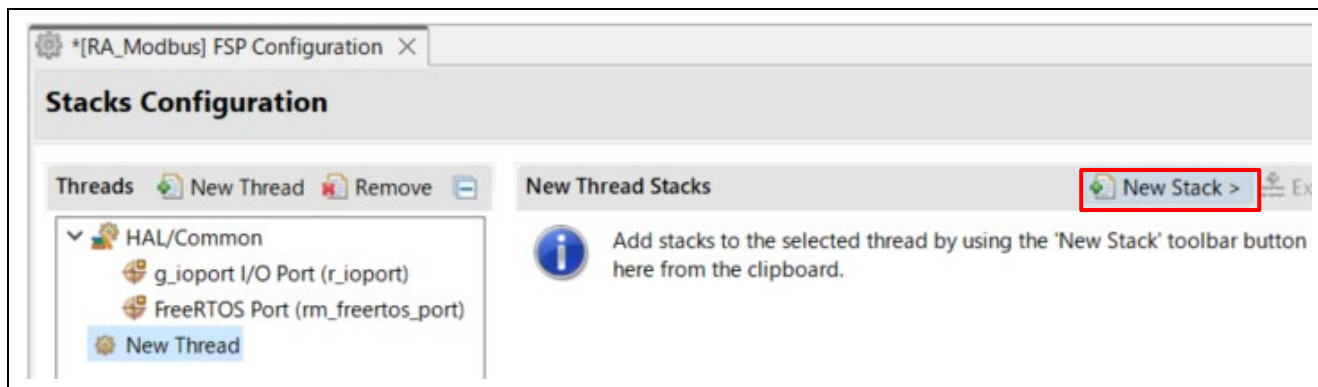


10. Create New Thread

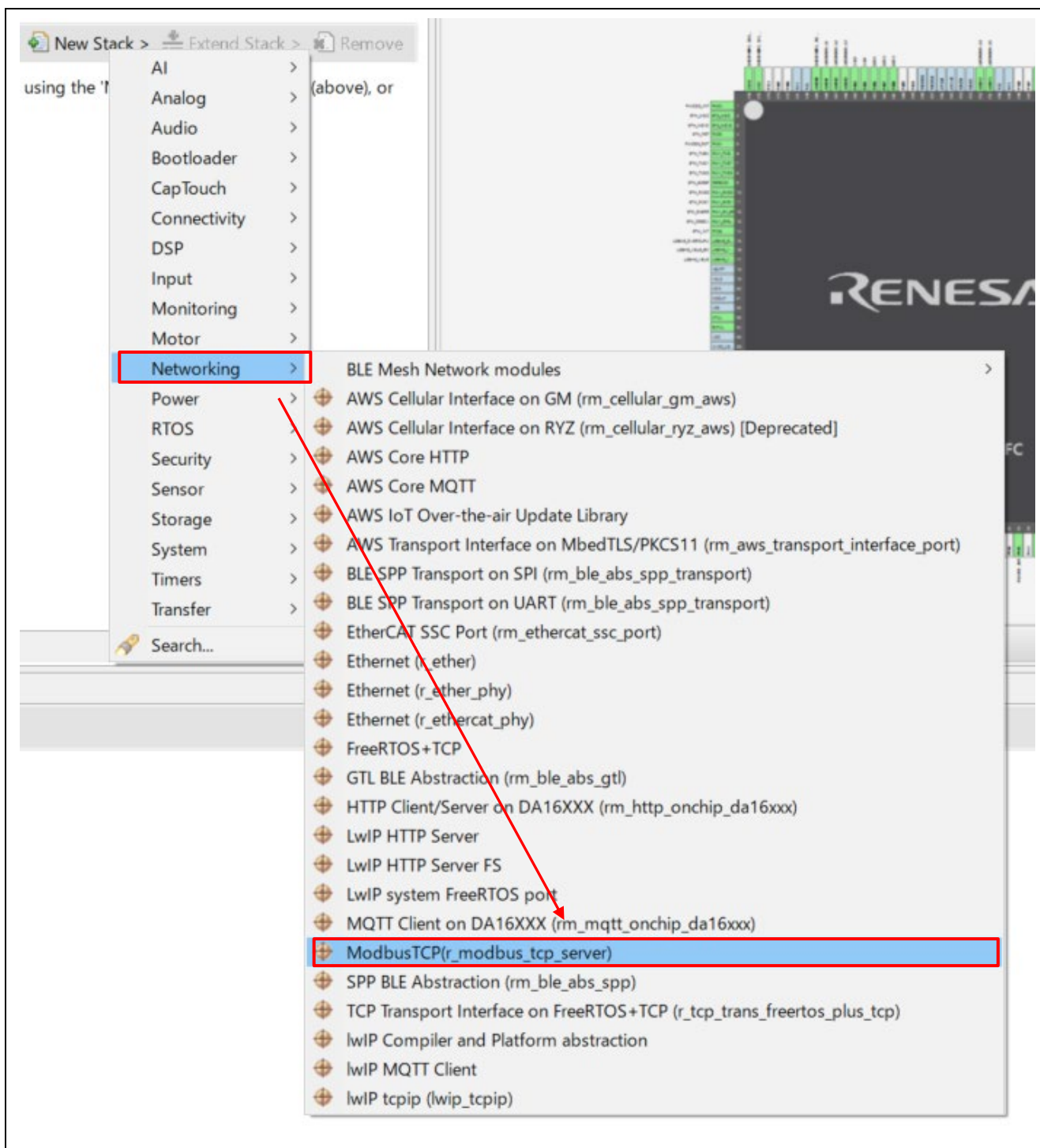
In the "FSP Configuration", click the "Stacks" tab, select "Threads" → "New Thread". A new thread appears.



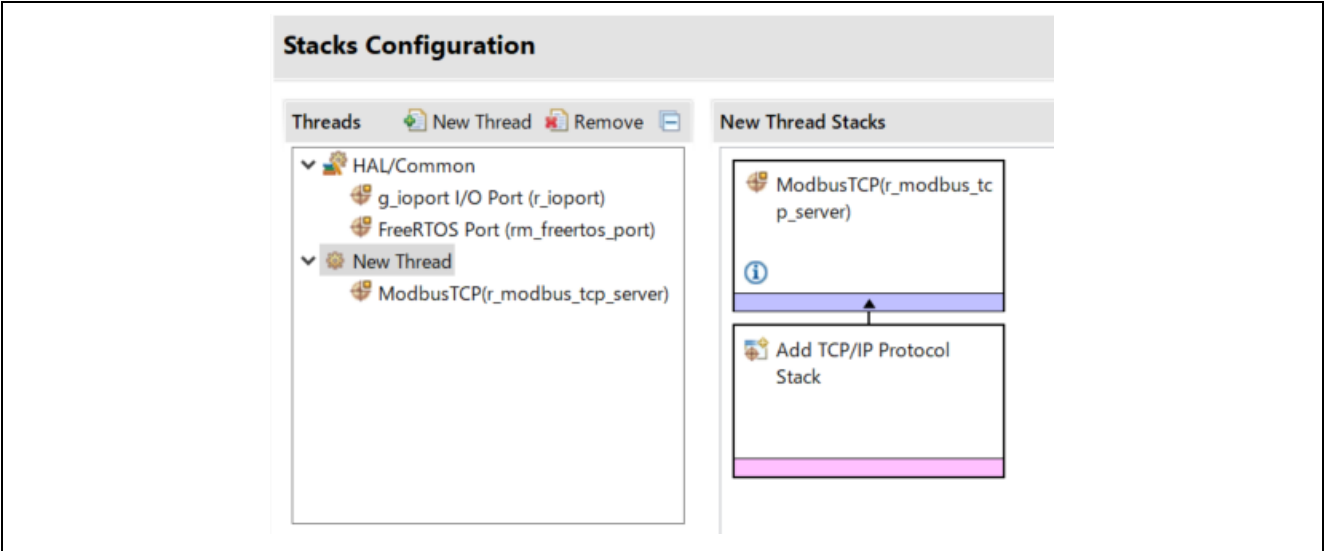
11. Create New Stack
Select "New Stacks" to be configured.



Select "Networking" → "ModbusTCP(r_modbus_tcp_server)".



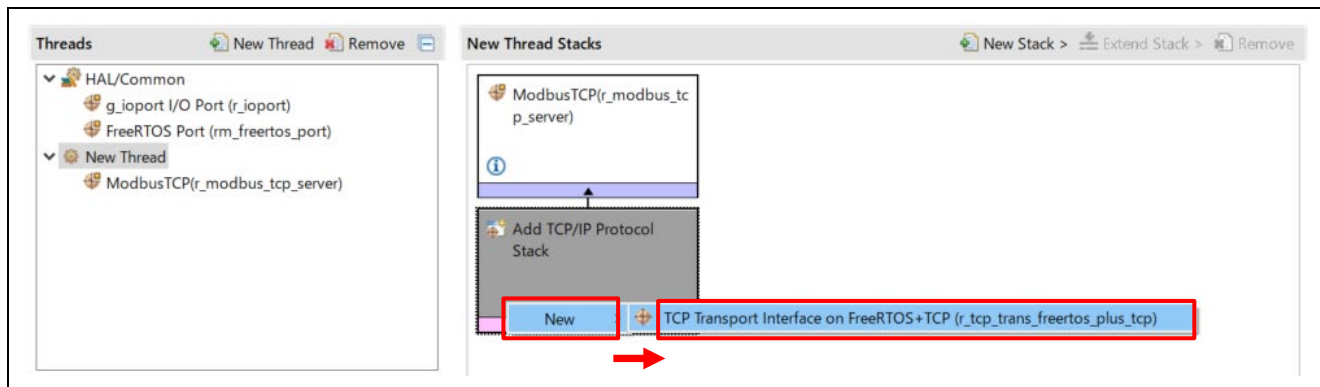
The stack is configured as follows:



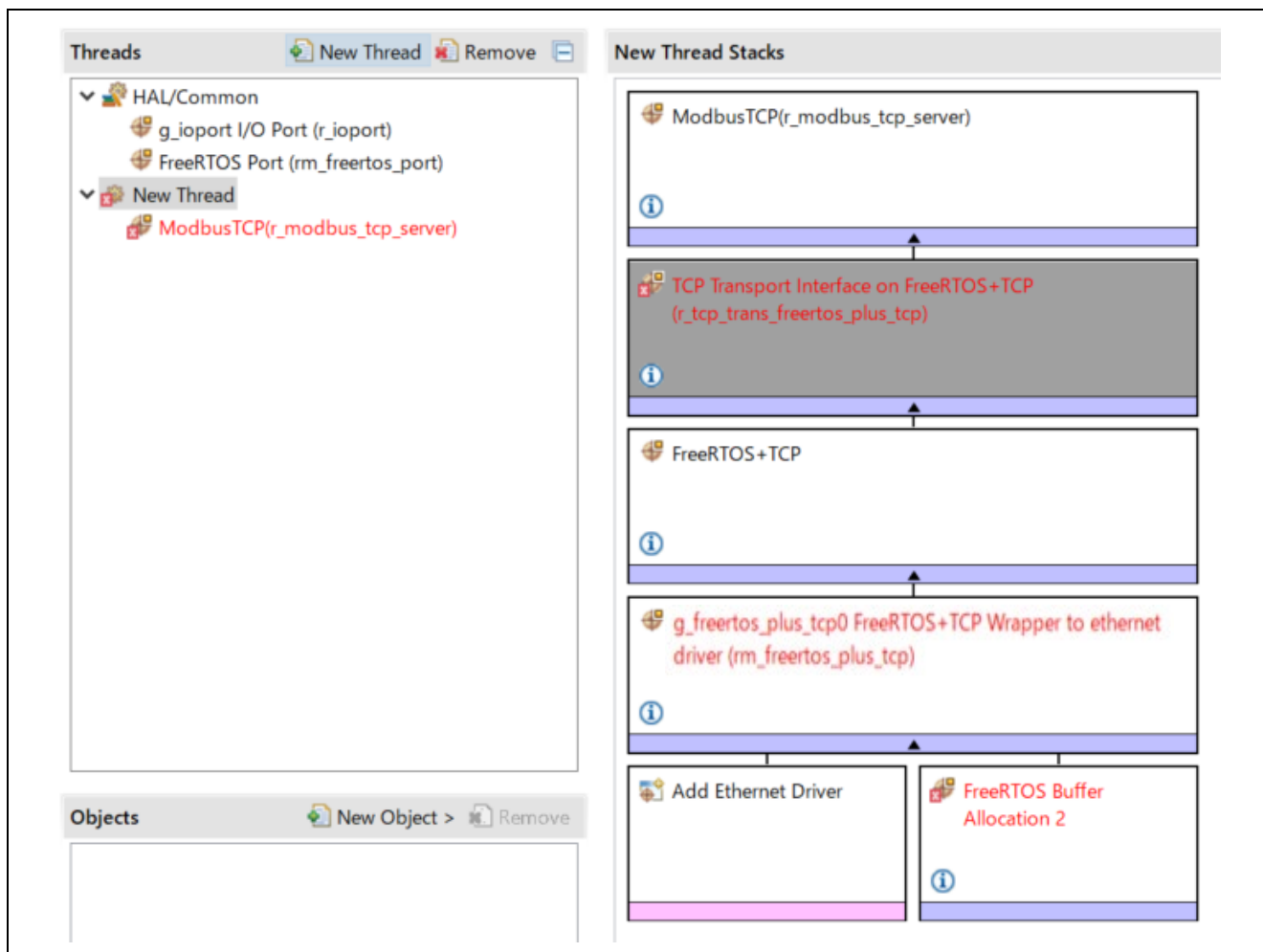
12. Add FreeRTOS+TCP

Click “New” → “TCP Transport Interface on FreeRTOS+TCP

(r_modbus_tcp_server_port_freertos_plus_tcp)” to “Add TCP/IP Protocol Stack”.

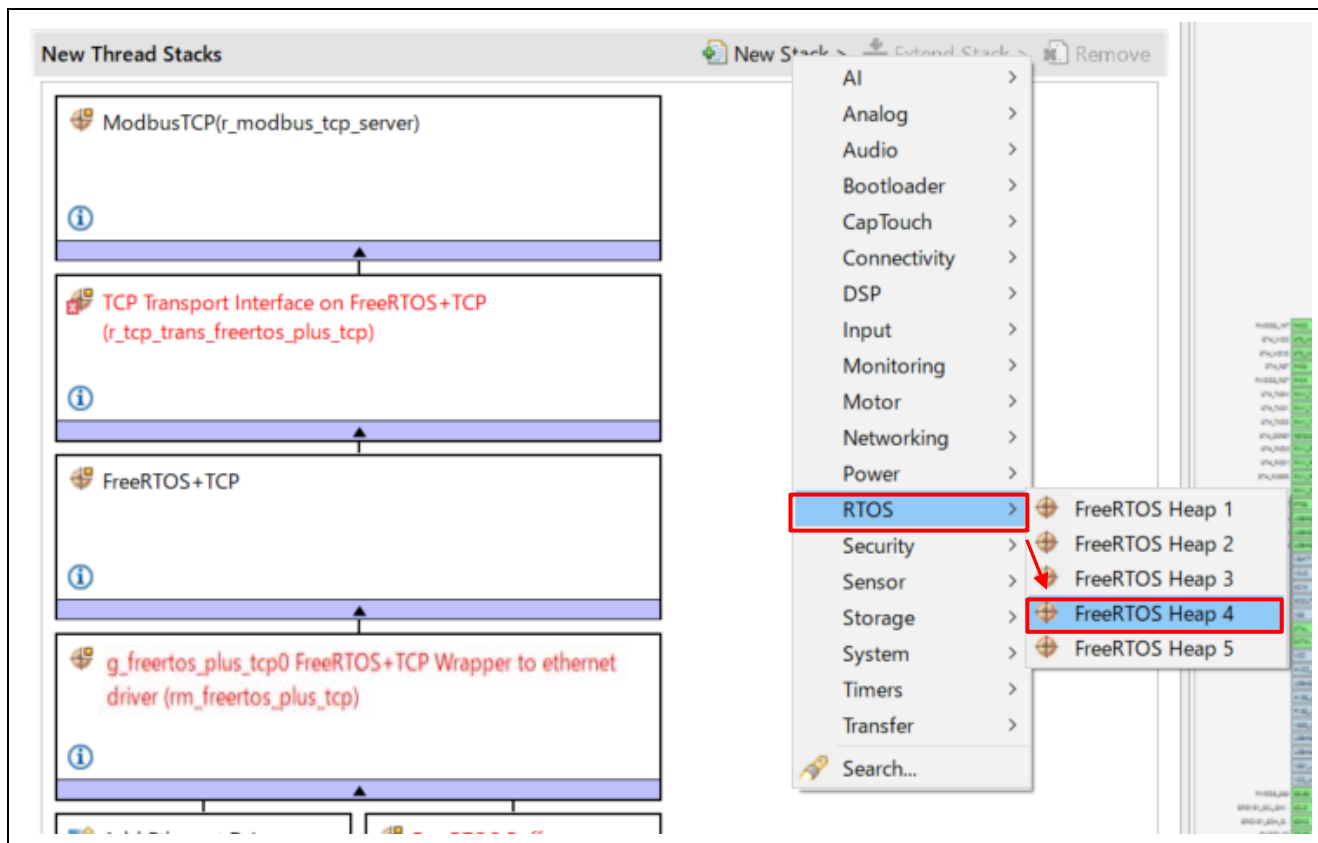


The stack is configured as follows:

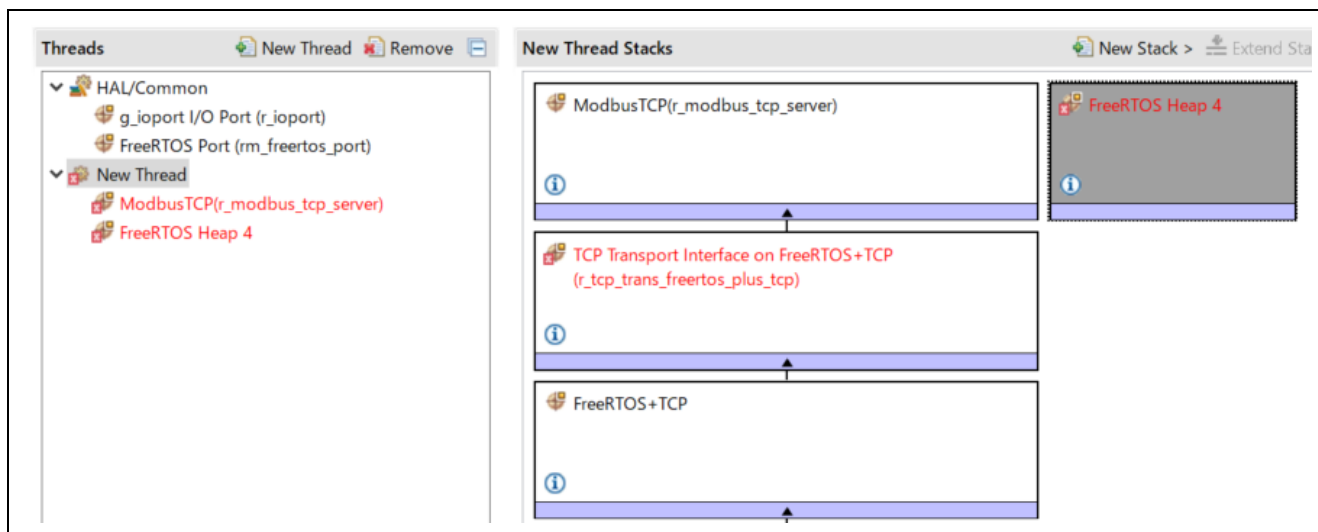


13. Add Heap

Select "New Stack" → "RTOS" → "FreeRTOS Heap 4".

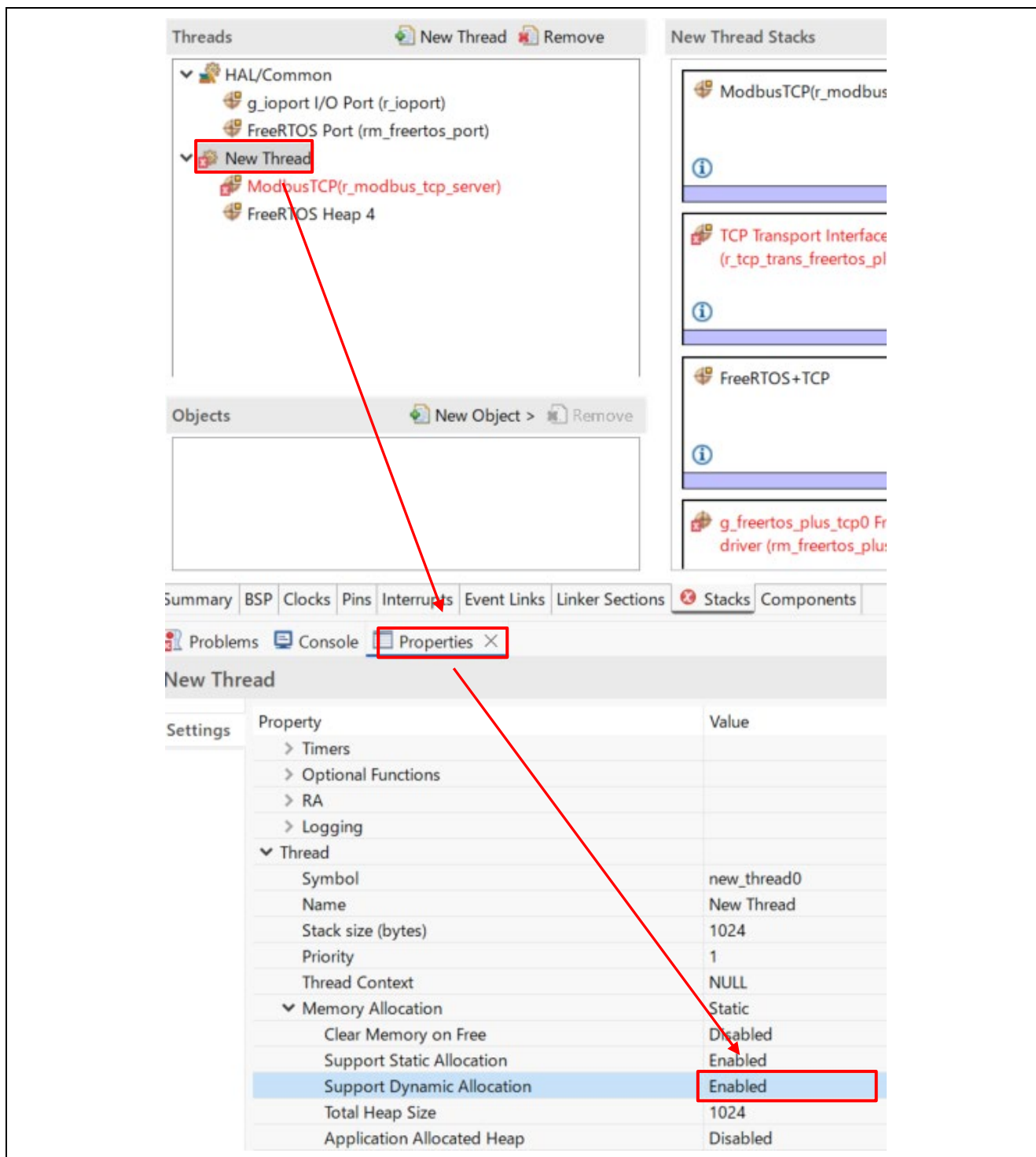


The stack will be configured as follows:



14. Set Support Dynamic Allocation

Open “Properties” of “New Thread” in “Stacks” and change “Support Dynamic Allocation” inside “Memory Allocation” to “Enabled”.



“FreeRTOS Heap4” stack configuration errors are resolved.

15. Set FreeRTOS+TCP

Open "Properties" of "FreeRTOS+TCP" in "Stacks" and change "Network Events call vApplicationIPNetworkEventHook", "Use DHCP", Total number of available network buffers, and "FreeRTOS_Select() (and associated) API function is available" in "Common" to the following values.

Network Events call vApplicationIPNetworkEventHook : **Disable**

Use DHCP : **Disable**

Total number of available network buffers : **30**

FreeRTOS_Select() (and associated) API function is available : **Enable**

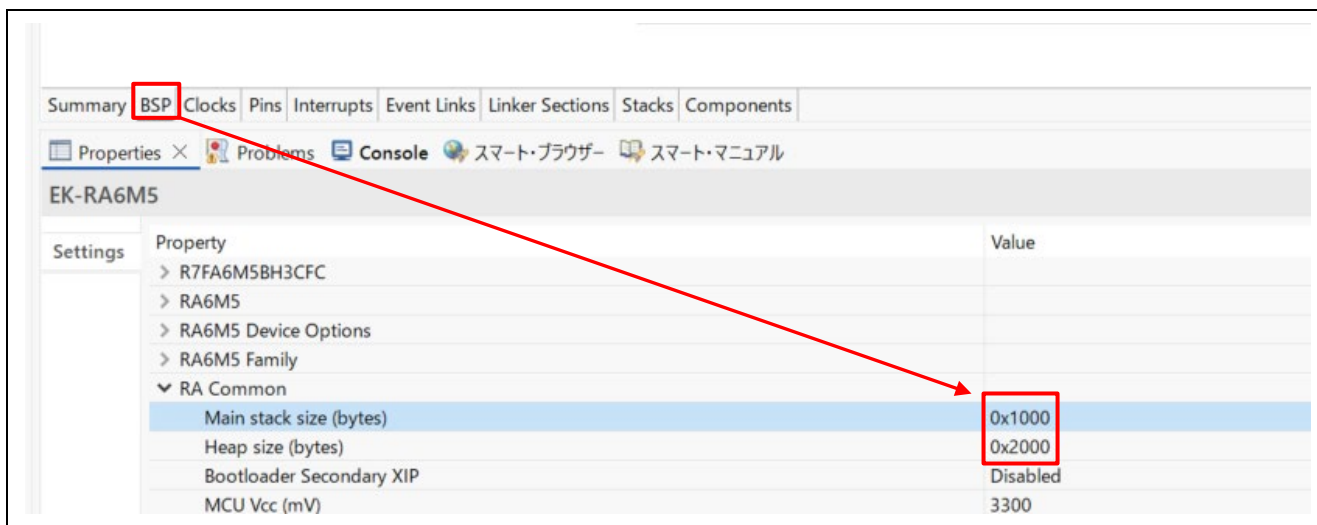
The screenshot shows the IDE interface for configuring the FreeRTOS+TCP stack. The 'Stacks' tab is active, and the 'FreeRTOS+TCP' stack is selected. The 'Properties' window for this stack is open, displaying a list of configuration properties. The following properties are highlighted with red boxes in the original image:

Property	Value
Network Events call vApplicationIPNetworkEventHook	Disable
Use DHCP	Disable
Total number of available network buffers	30
FreeRTOS_Select() (and associated) API function is available	Enable

16. Set Stack size

Open “Properties” of “BSP” and change “Main stack size” and “Heap size” in “RA Common” to the following values.

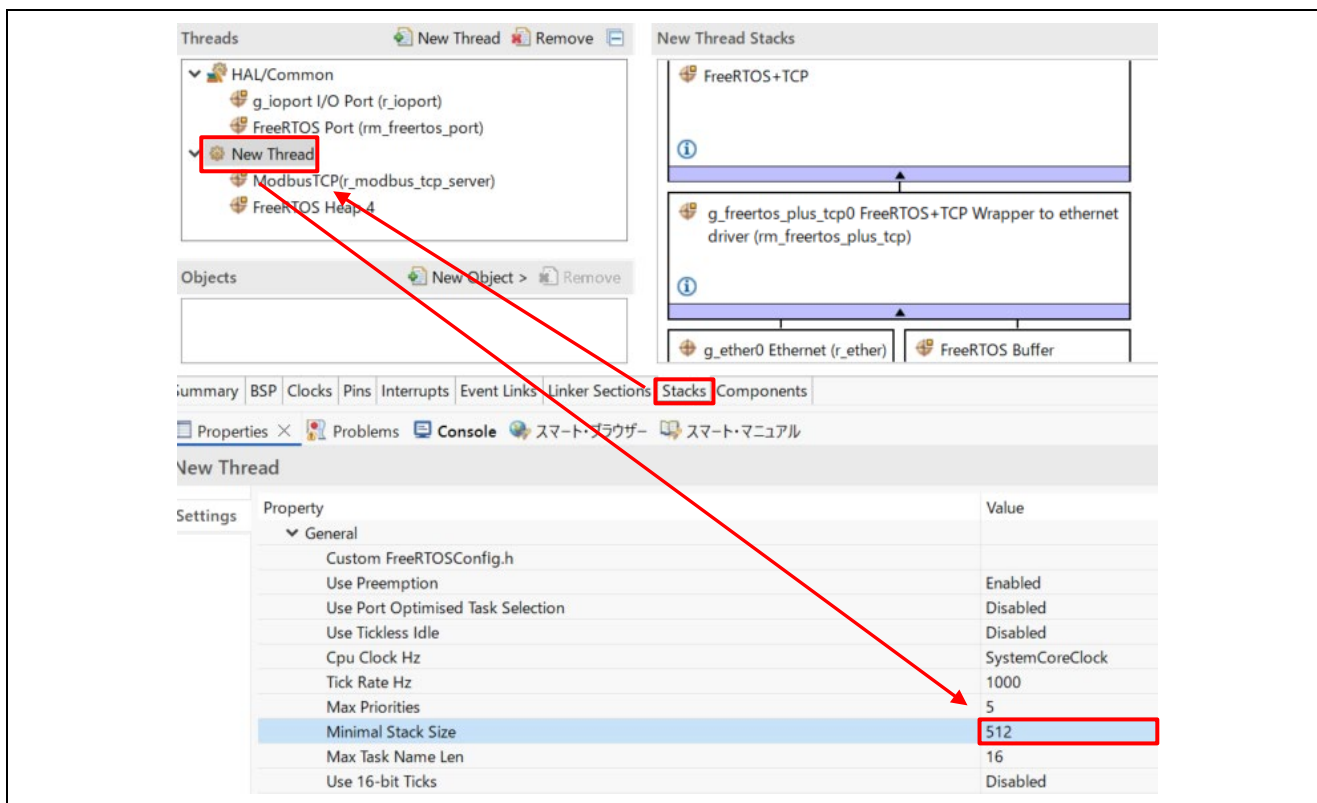
Main stack size : **0x1000**, Heap size : **0x2000**



The stack configuration error for "ModbusTCP (r_modbus_tcp_server)" is now resolved.

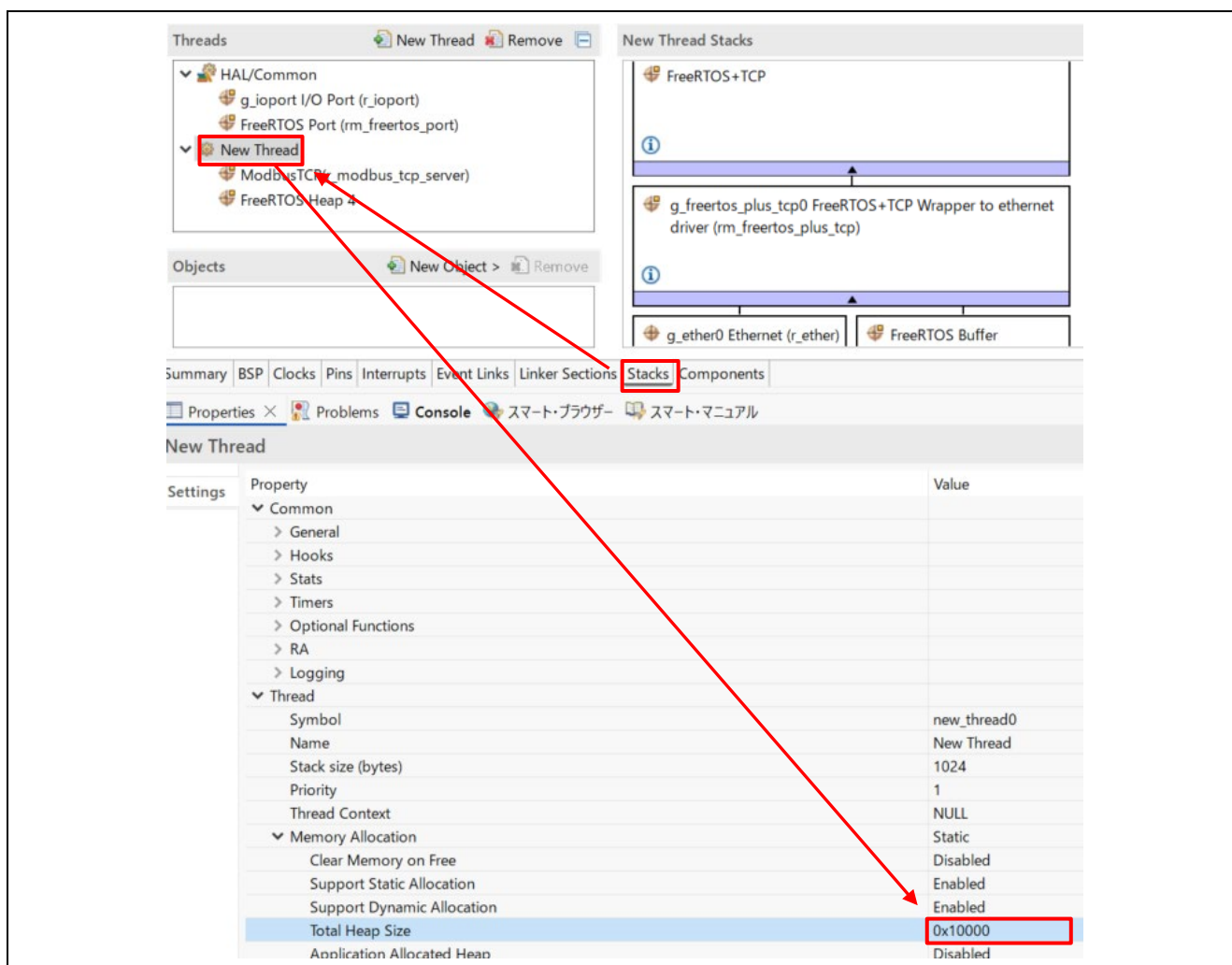
Open “Properties” of “New Thread”, then click on “Stacks” and change “Minimal Stack size” in “General” to the following values.

Minimal Stack size : **512**



Open “Properties” of “New Thread”, then click on “Stacks” and change “Total Heap Size” in “Memory Allocation” to the following values.

Total Heap Size : **0x10000**



17. Perform the steps for each evaluation board.

Refer to the following information for using the evaluation board and follow the steps:

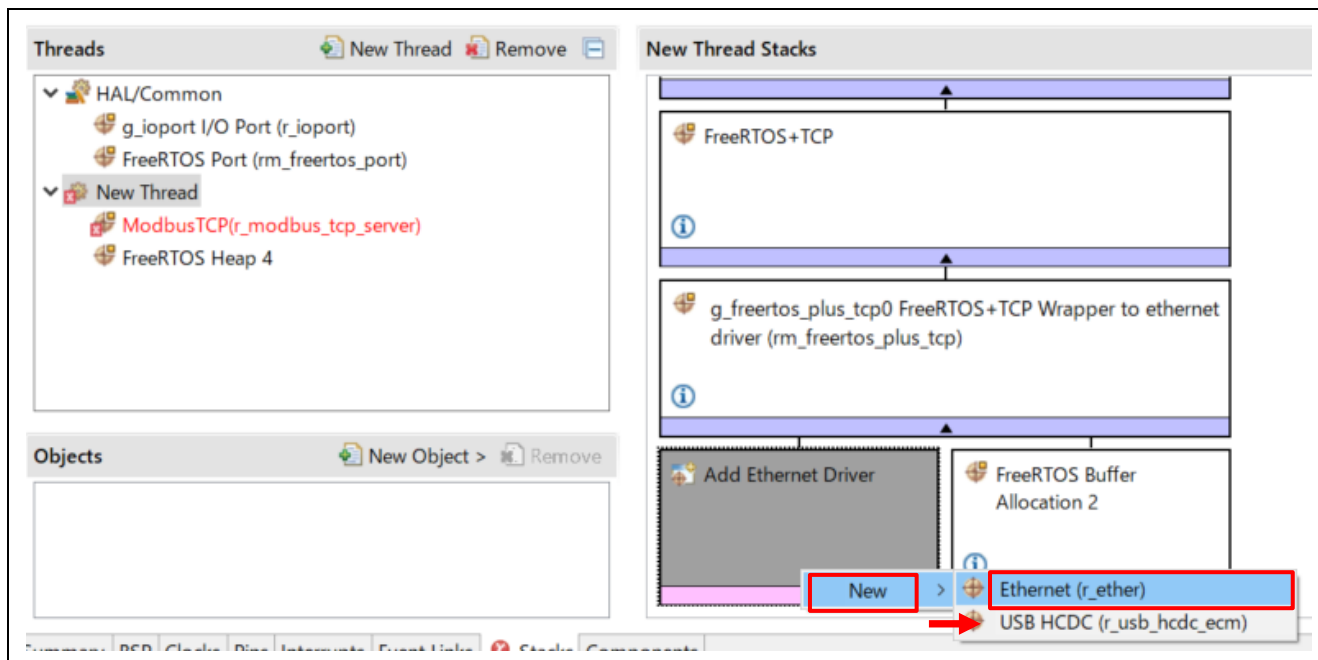
- EK-RA6M3, EK-RA6M4, EK-RA6M5, EK-RA8D1, EK-RA8M1, MCK-RA8T1 :
[6.2.2 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Creating Procedures](#)
- MCK-RA8T2 : [6.2.3. MCK-RA8T2 Creating Procedures](#)
- EK-RA8D2 : [6.2.4. EK-RA8D2 Creating Procedures](#)
- EK-RA8P1 : [6.2.5. EK-RA8P1 Creating Procedures](#)
- EK-RA8M2 : [6.2.6. EK-RA8M2 Creating Procedures](#)
- EK-RA8T2 : [6.2.7. EK-RA8T2 Creating Procedures](#)

6.2.2 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Creating Procedures

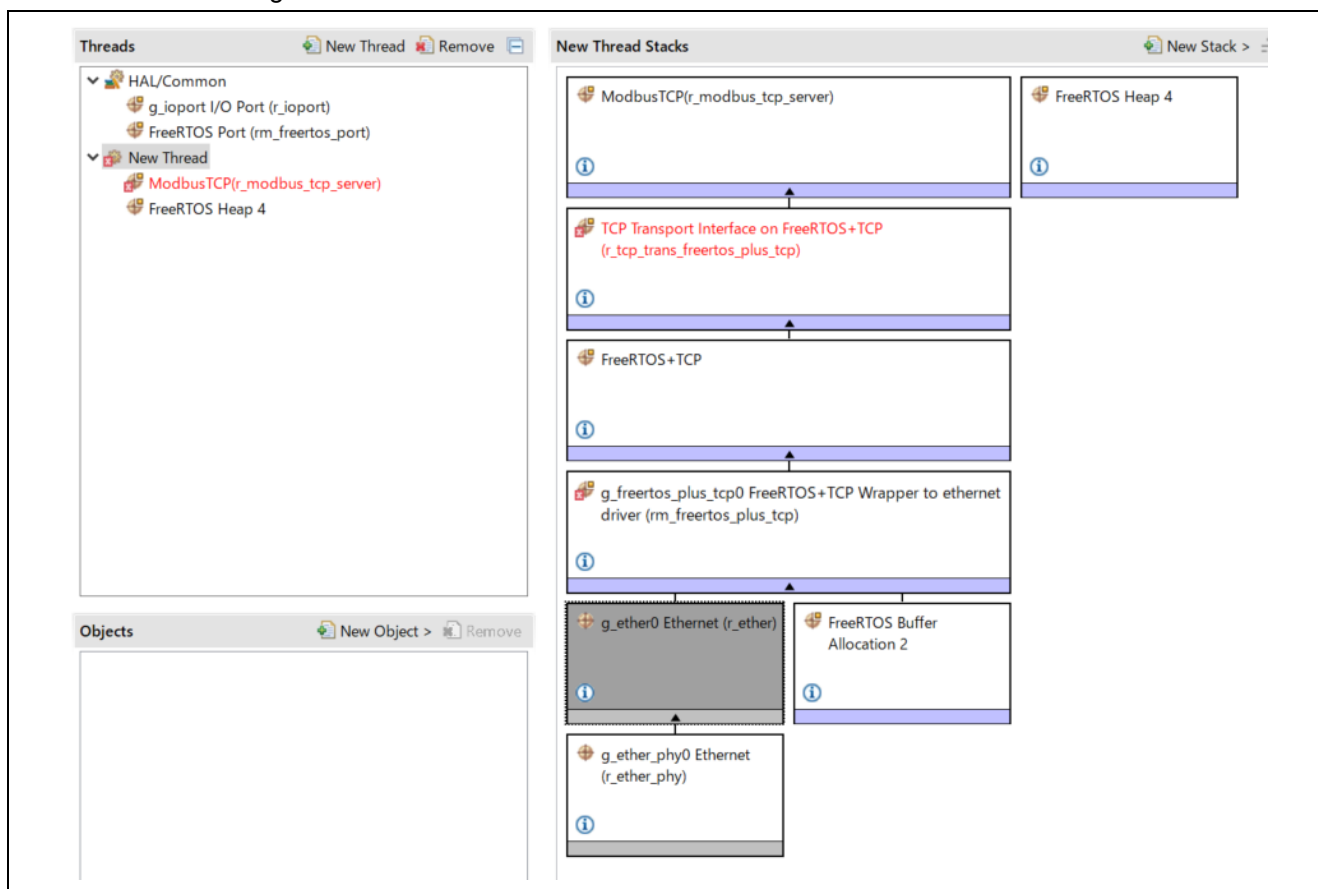
This section describes the procedures for creating EK-RA6Mx (EK-RA6M3, EK-RA6M4, EK-RA6M5) / EK-RA8x1 (EK-RA8D1, EK-RA8M1) / MCK-RA8T1.

1. Add Ethernet Driver

Click “New” → “Ethernet (r_ether)” to “Add Ethernet Driver”.



The stack is configured as follows:



2. Set PHY Address

Open “Properties” of “g_ether_phy0 Ethernet (r_ether_phy)” in “Stacks” and change “PHY-LSI Address” in “Module g_ether_phy0 Ethernet (r_ether_phy)” to the port0 address configured on the evaluation board.

RA6xx : PHY Address : 0, RA8xx : PHY Address : 5

The screenshot displays the Renesas RA IDE interface. The 'Stacks' tab is selected, showing a stack of components. The 'g_ether_phy0 Ethernet (r_ether_phy)' component is highlighted with a red box. The 'Properties' window for this component is open, showing the 'PHY-LSI Address' set to 0. A red arrow points from the component in the stack to the 'PHY-LSI Address' field in the properties window.

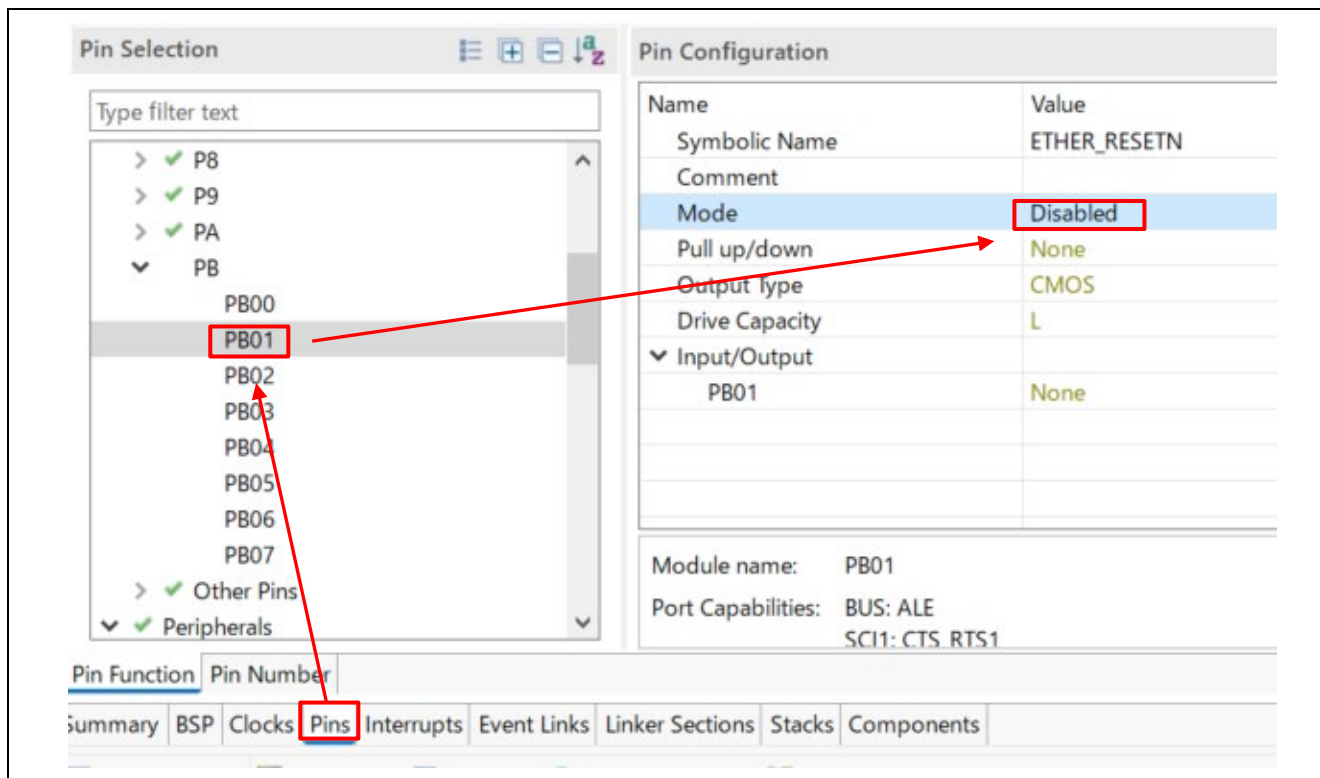
Property	Value
Name	g_ether_phy0
Channel	0
PHY-LSI Address	0
PHY-LSI Reset Completion Timeout	0x00020000

3. Set PHY Reset pin

The setting of the PHY Reset terminal on the evaluation boards RA8D1 and RA8T1 must be changed. Open "Pin Configuration" of [Reset pin for evaluation board] (see below) in "Pins" and change "Mode" to "Disabled".

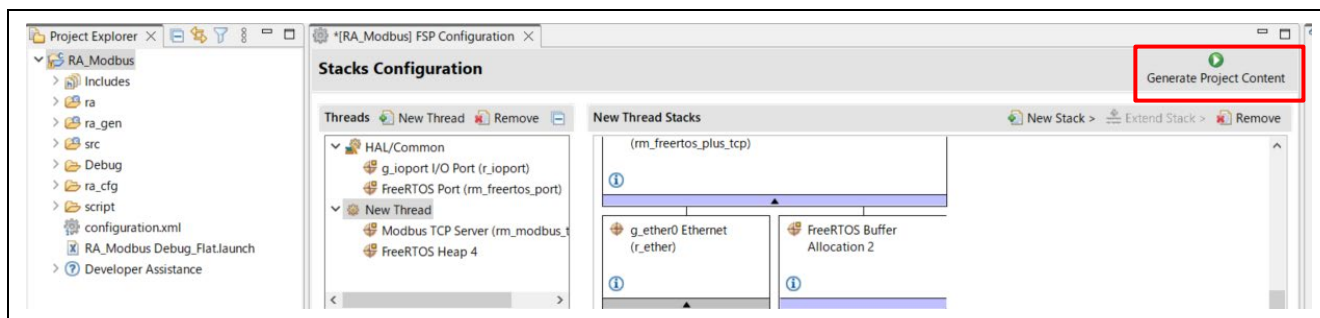
RA8D1 : **P706**, RA8T1 : **PB01**

Note: This procedure is not required for evaluation boards other than those listed above.



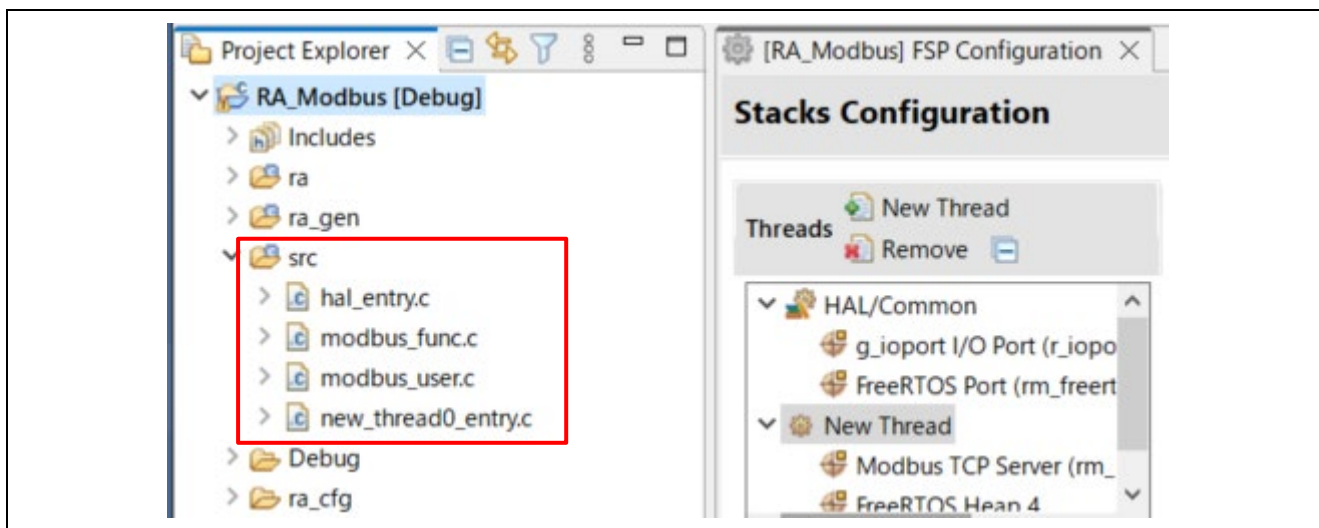
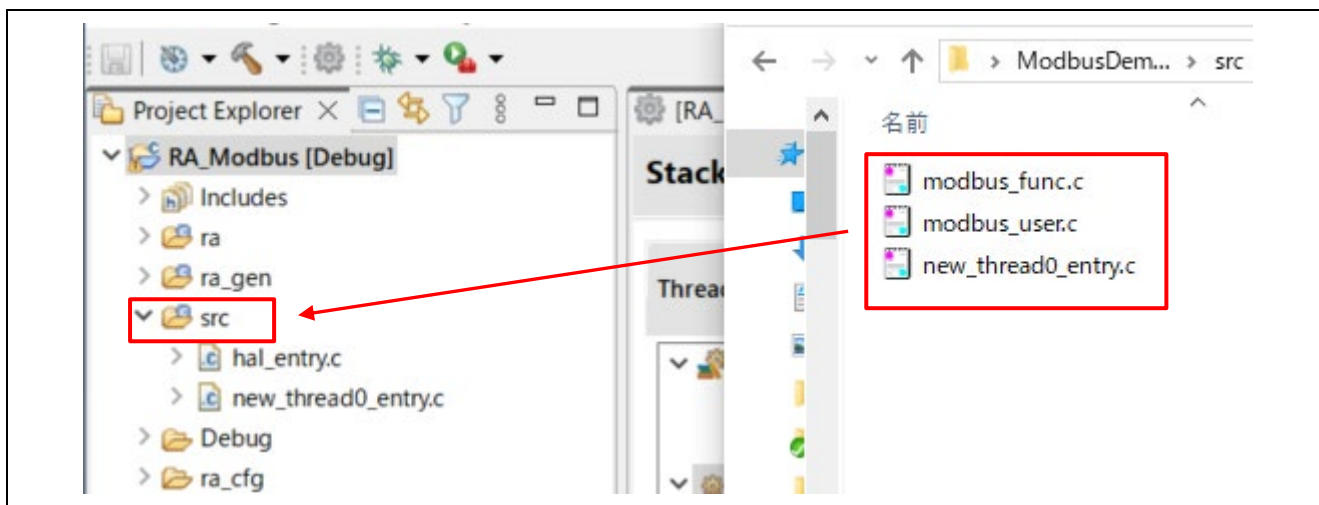
4. Generate the code

Generate the code with "Generate Project Content".



5. Add Modbus Sample Application

Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

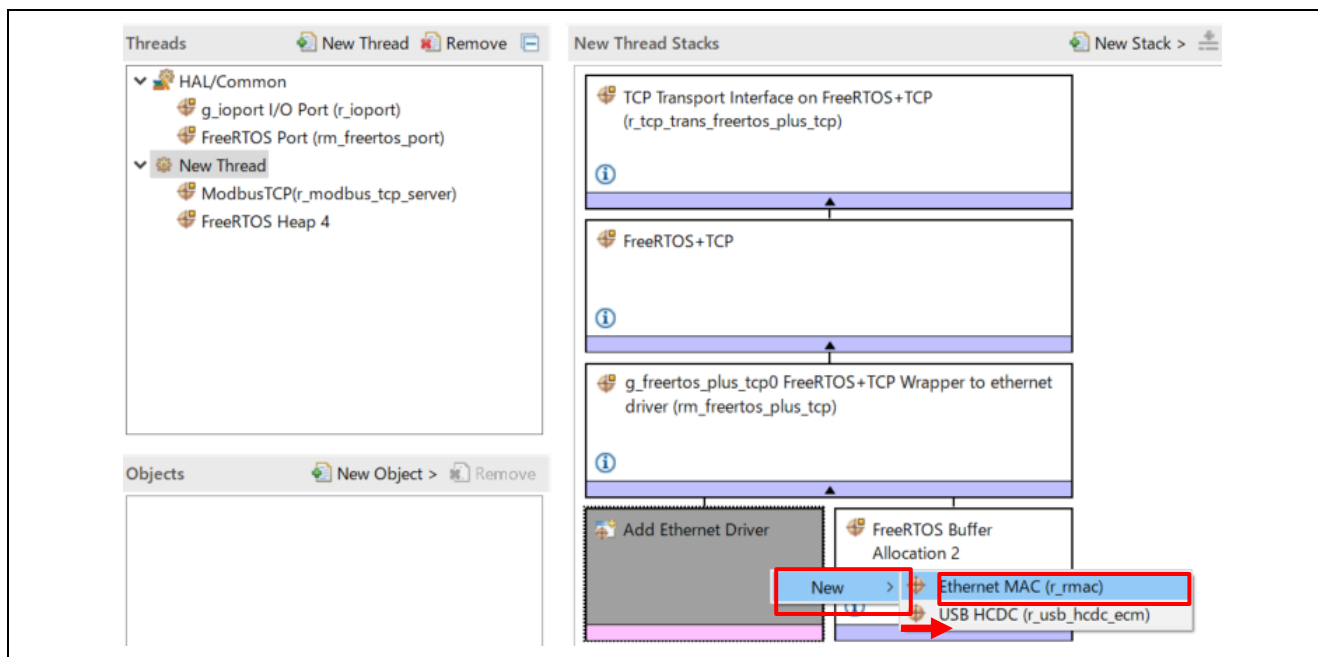


6.2.3 MCK-RA8T2 Creating Procedures

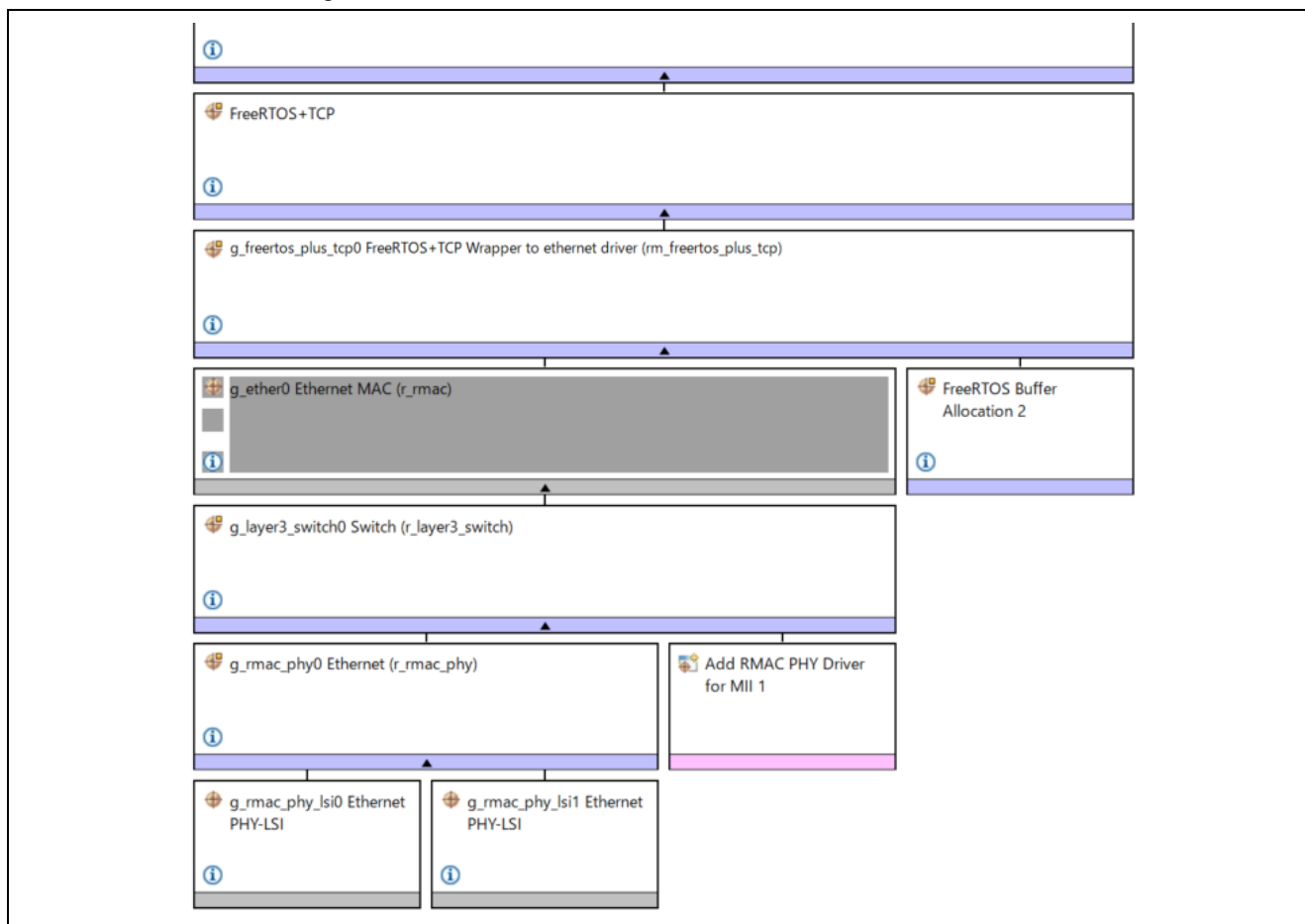
This section describes the procedures for creating MCK-RA8T2.

1. Add Ethernet Driver

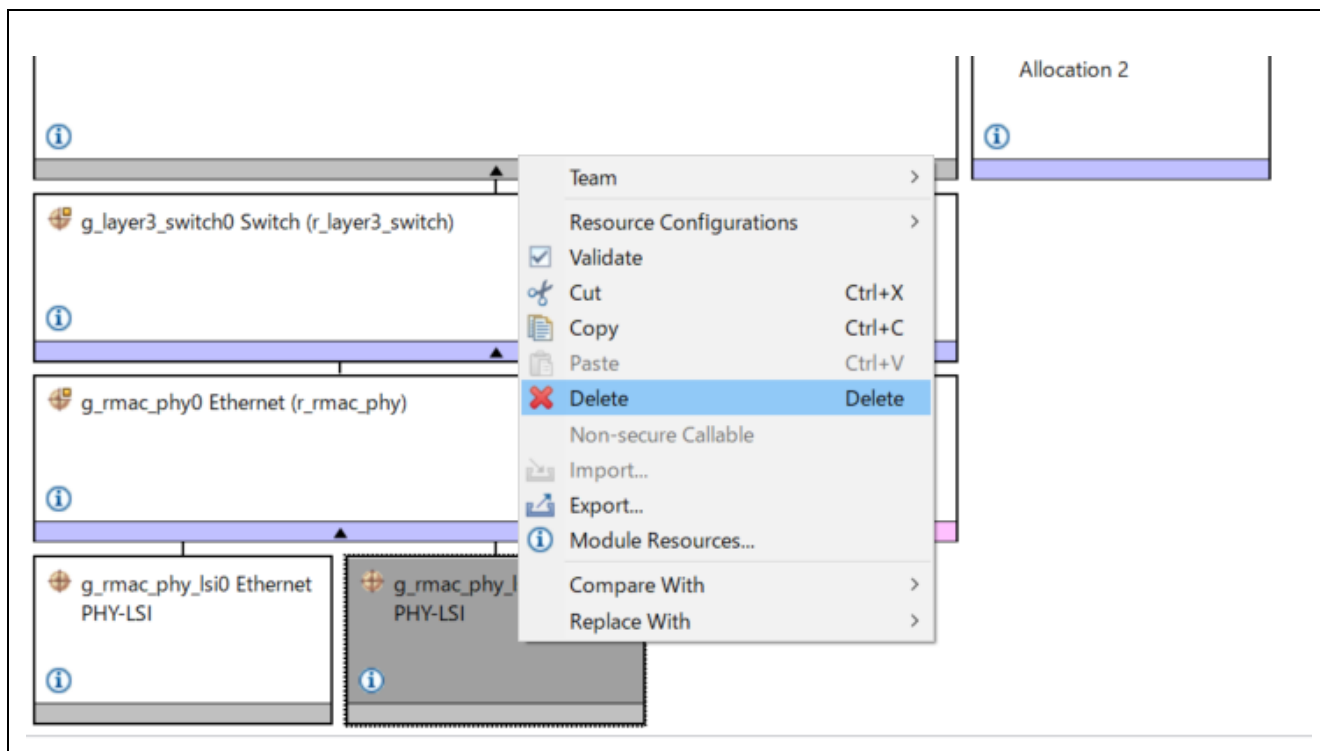
Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



The stack will be configured as follows:

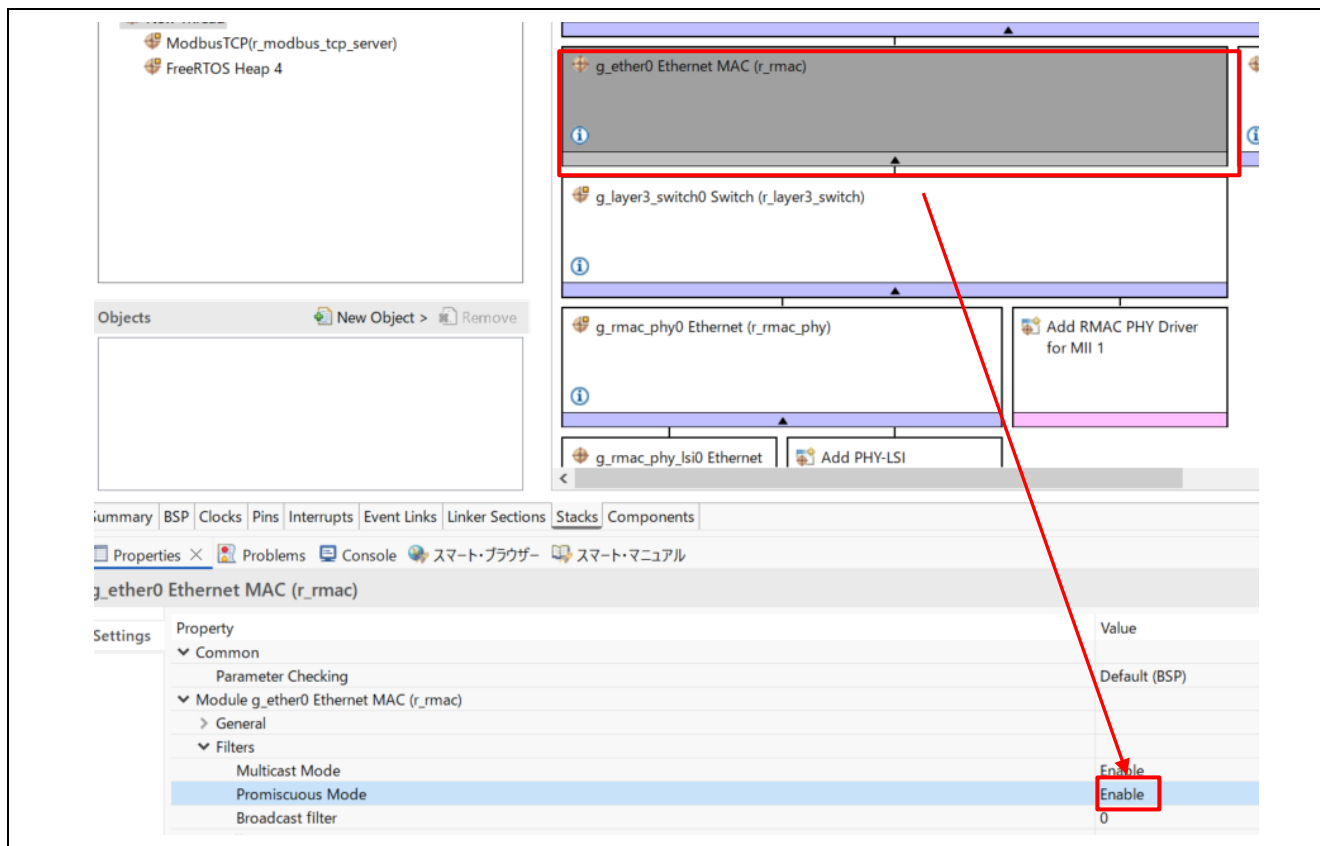


Right-click on “g_rmac_phy_Isi1 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



2. Set r_rmac

Change the “Stacks” → “g_ether0 Ethernet MAC (r_rmac)” → “Module g_ether0 Ethernet MAC (r_rmac)” → “Promiscuous Mode” in “Filters” to “Enable”.



3. Set r_layer3_switch
Open "Properties" of "g_layer3_switch0 Switch (r_layer3_switch)" in "Stacks" and change "gPTP enable" to the following values.

gPTP enable : **Disabled**

The screenshot displays the development environment interface. On the left, a tree view shows the project structure under 'HAL/Common', including 'g_ioport I/O Port (r_ioport)', 'FreeRTOS Port (rm_freertos_port)', 'New Thread', 'ModbusTCP(r_modbus_tcp_server)', and 'FreeRTOS Heap 4'. The 'Objects' panel below it shows 'New Object' and 'Remove' buttons. The main workspace shows a stack diagram with components: 'g_layer3_switch0 Switch (r_layer3_switch)', 'g_rmac_phy0 Ethernet (r_rmac_phy)', 'g_rmac_phy_0 Ethernet PHY-LSI', and 'Add PHY-LSI configuration for MII 1'. A red box highlights the 'g_layer3_switch0 Switch (r_layer3_switch)' component, and a red arrow points from it to the 'Properties' tab of the 'g_layer3_switch0 Switch (r_layer3_switch)' component.

The 'Properties' tab for 'g_layer3_switch0 Switch (r_layer3_switch)' is shown below. It has a table with 'Property' and 'Value' columns.

Property	Value
Common	
Parameter Checking	Default (BSP)
Available queue num	4
gPTP enable	Disabled
Time Aware Shaper	Disabled
Module g_layer3_switch0 Switch (r_layer3_switch)	

4. Set `r_rmac_phy`
 Open "Properties" of "`g_mac_phy0 Ethernet (r_rmac_phy)`" in "Stacks" and change "Select MII type", "MDIO hold timing adjustment", and "MDIO capture timing adjustment" in "Module `g_mac_phy0 Ethernet (r_rmac_phy)`" to the following values.

Select MII type : **MI**

MDIO hold timing adjustment : **7**

MDIO capture timing adjustment : **7**

Property	Value
Name	<code>g_mac_phy0</code>
Channel	0
Default PHY-LSI port	0
PHY-LSI Reset Completion Timeout	0x00020000
Select MII type	MI
Port Custom Init Function	NULL
Port Custom Link Partner Ability Get Function	NULL
Flow Control	Disable
Management frame format	Clause 22 frame format
MDC clock rate (Hz)	2500000
MDIO hold timing adjustment	7
MDIO capture timing adjustment	7

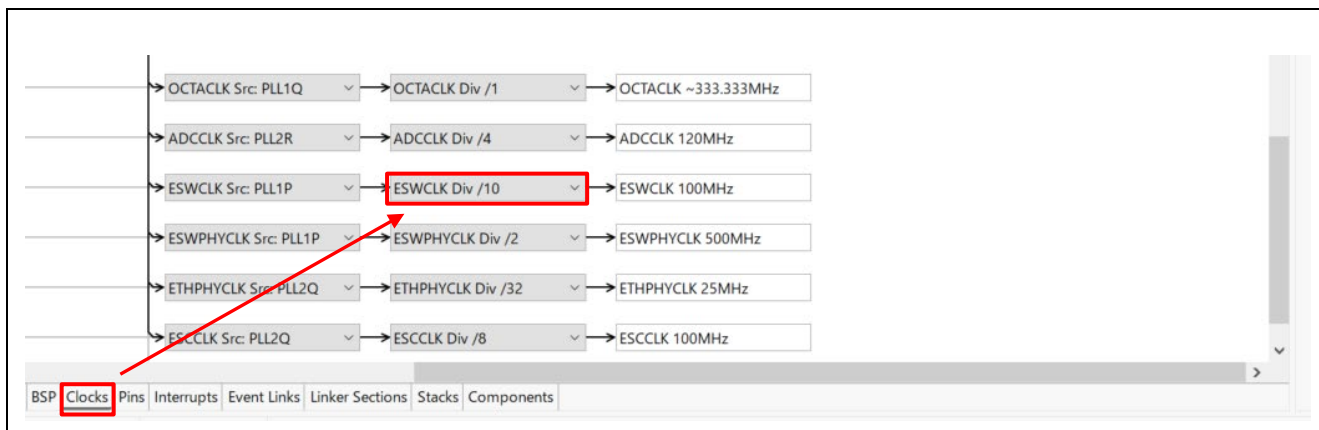
5. Set PHY Address

Open "Properties" of "`g_mac_phy_1si0 Ethernet PHY LSI`" in "Stacks" and change "PHY-LSI Address" in "Module `g_mac_phy_1si0 Ethernet PHY LSI`" to "1".

Property	Value
Name	<code>g_mac_phy_1si0</code>
PHY-LSI Address	1
PHY-LSI type	Kit Component

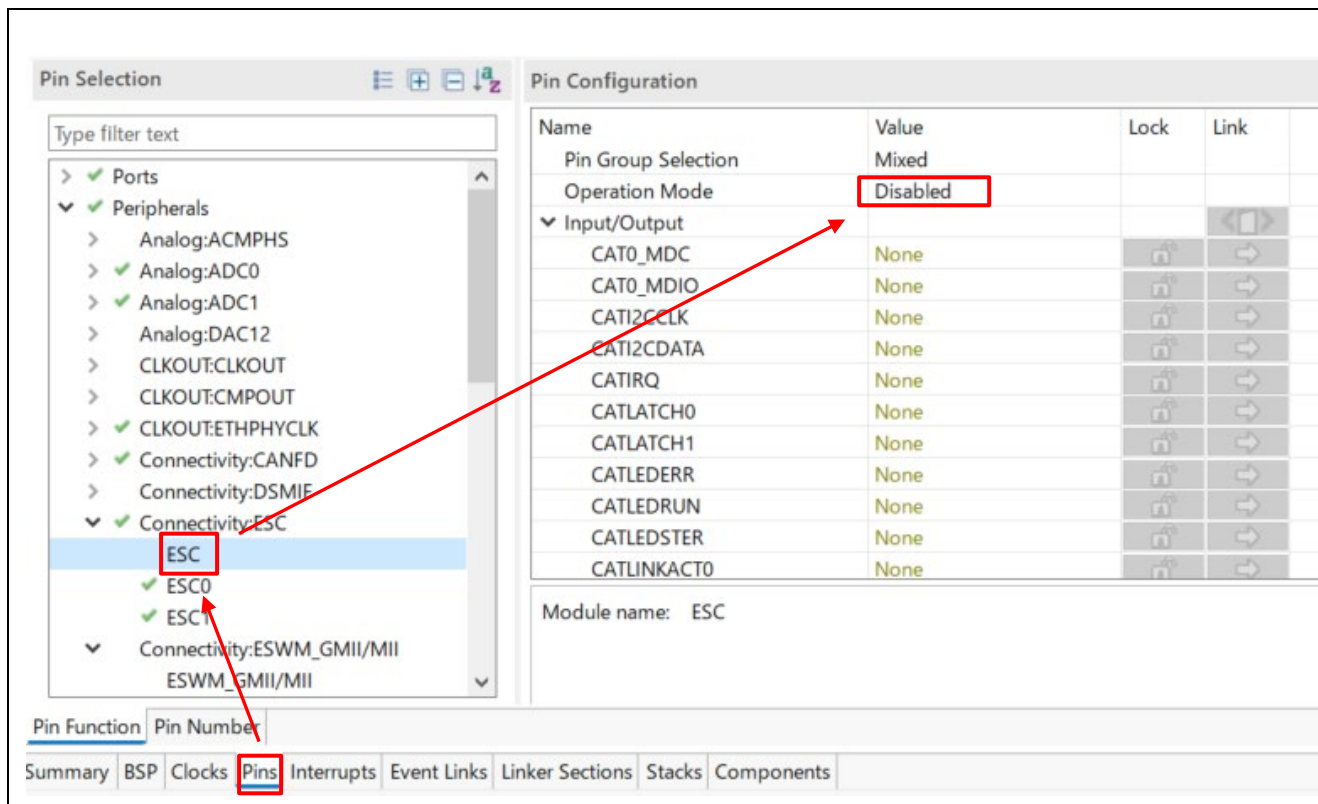
6. Set ESW Clock

Change the "ESWCLK Div" at the bottom of "Clocks" to "ESWCLK Div /10".



7. Set ESWM Terminals

Change the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESC" → "ESC" to "Disabled".
Change the "Operation Mode" of "ESC0" and "ESC1" to "Disabled" as well.



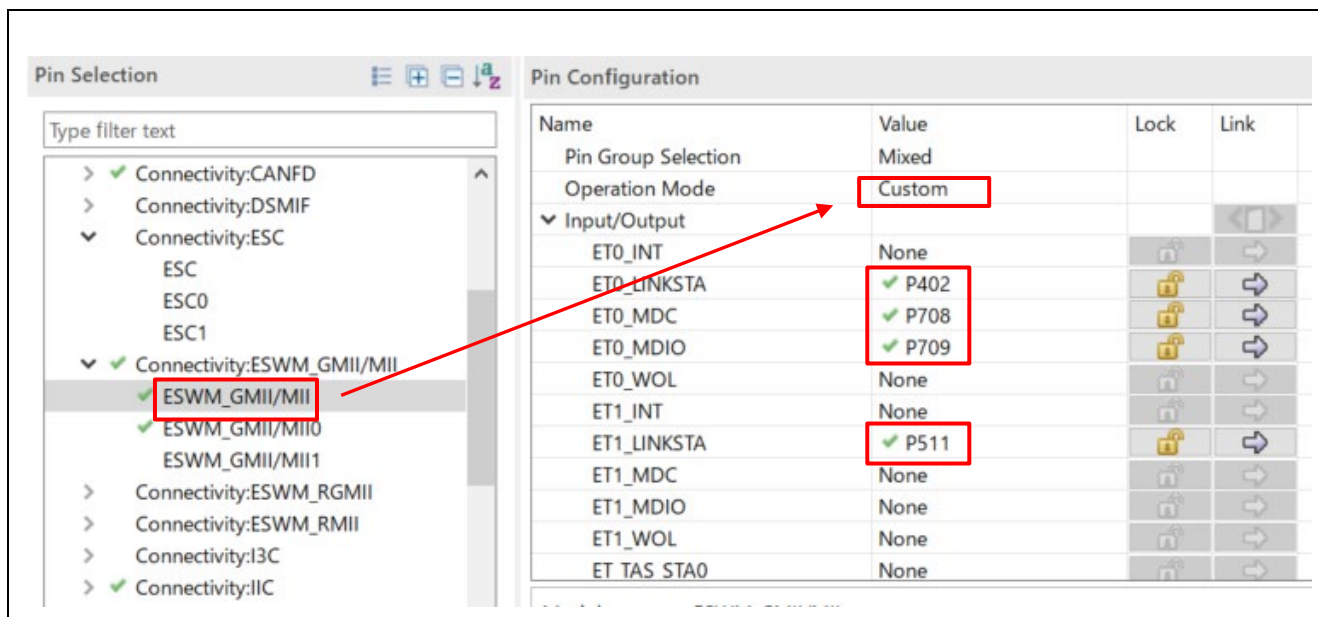
After changing the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESWM_GMII/MII" → "ESWM_GMII/MII" to "Custom", change the pins as follows:

ET0_LINKSTA : **P402**

ET0_MDC : **P708**

ET0_MDIO : **P709**

ET1_LINKSTA : **P511**



After changing the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESWM_GMII/MII" → "ESWM_GMII/MII0" to "Custom", change the pins as follows:

ET0_RXD0 : **P702**
 ET0_RXD1 : **P701**
 ET0_RXD2 : **P700**
 ET0_RXD3 : **P406**
 ET0_RX_CLK : **P703**
 ET0_RX_DV : **P405**
 ET0_RX_ER : **P704**
 ET0_TXD0 : **PB00**
 ET0_TXD1 : **PB02**
 ET0_TXD2 : **PB03**
 ET0_TXD3 : **PB04**
 ET0_TX_CLK : **PB01**
 ET0_TX_EN : **P705**

Name	Value	Lock	Link
Pin Group Selection	Mixed		
Operation Mode	Custom		
▼ Input/Output			
ET0_GTX_CLK	None		
ET0_RXD0	✓ P702		
ET0_RXD1	✓ P701		
ET0_RXD2	✓ P700		
ET0_RXD3	✓ P406		
ET0_RXD4	None		
ET0_RXD5	None		
ET0_RXD6	None		
ET0_RXD7	None		
ET0_RX_CLK	✓ P703		
ET0_RX_DV	✓ P405		
ET0_RX_ER	✓ P704		
ET0_TXD0	✓ PB00		
ET0_TXD1	✓ PB02		
ET0_TXD2	✓ PB03		
ET0_TXD3	✓ PB04		
ET0_TXD4	None		
ET0_TXD5	None		
ET0_TXD6	None		
ET0_TXD7	None		
ET0_TX_CLK	✓ PB01		
ET0_TX_EN	✓ P705		
ET0_TX_ER	None		

8. Set PHY Reset Terminals

Change the "Mode" of "Pins" → "Ports" → "P7" → "P711" to "Output mode (Initial High)".

The screenshot shows the Pin Configuration tool interface. On the left, the 'Pin Selection' pane lists pins P5 through P13 under port P7. Pin P711 is selected and highlighted with a red box. A red arrow points from this box to the 'Mode' field in the 'Pin Configuration' table on the right, which is also highlighted with a red box and contains the text 'Output mode (Initial High)'. Below the table, the 'Module name' is P711 and 'Port Capabilities' include AGT0: AGTEE0 and ESC: CATRESETOUT. At the bottom, the 'Pins' tab is selected in the navigation bar.

Name	Value	Link
Symbolic Name		
Comment		
Mode	Output mode (Initial High)	
Pull up/down	None	
IRQ	None	
Output Type	CMOS	
Drive Capacity	L	
Input Latch	None	
Input/Output	P711	GPIO

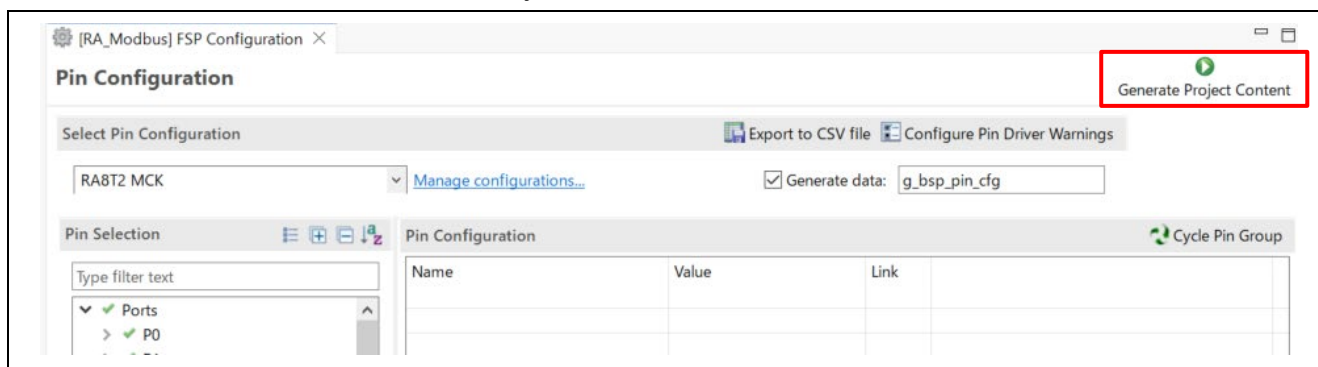
9. Set PHY CLK

Change the "Drive Capacity" of "Pins" → "Ports" → "P7" → "P706" to "H".

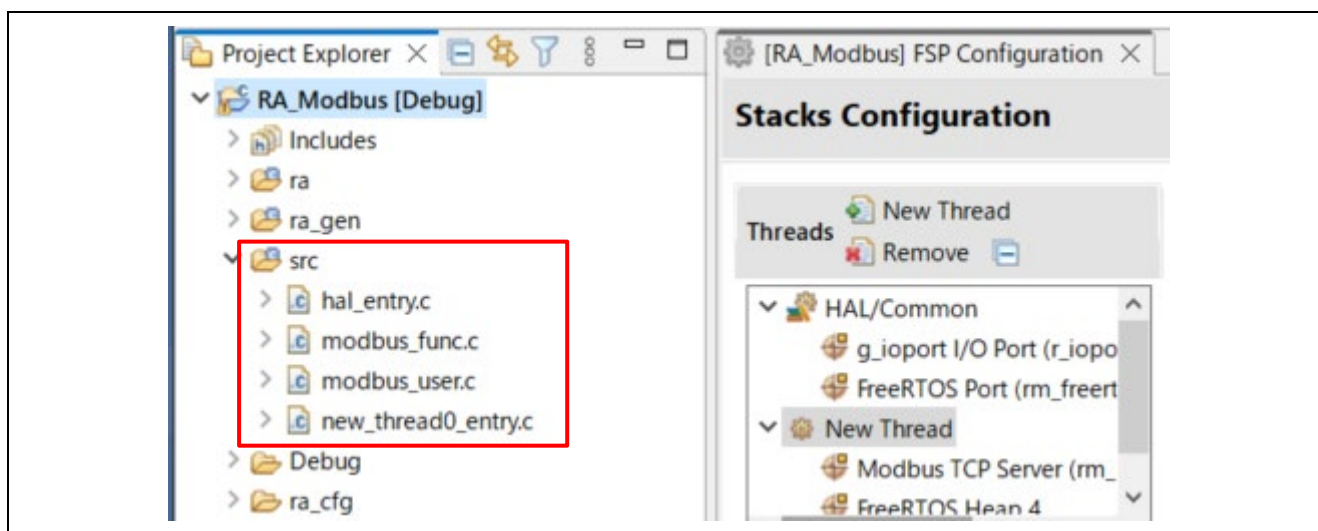
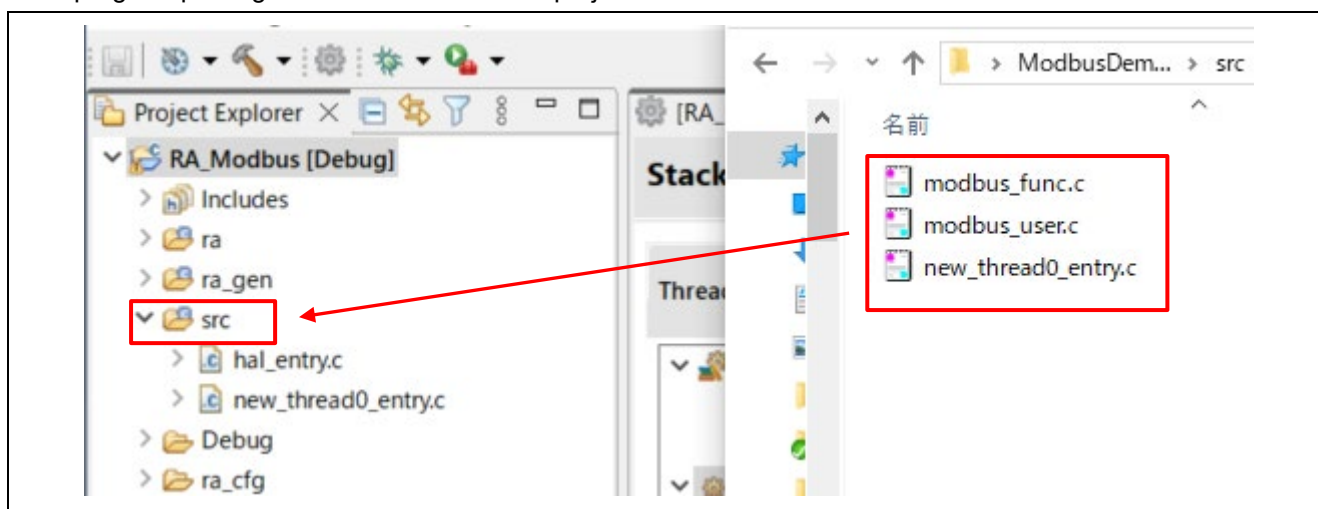
The screenshot shows the Pin Configuration tool interface. On the left, the 'Pin Selection' pane lists pins P5 through P13 under port P7. Pin P706 is selected and highlighted with a red box. A red arrow points from this box to the 'Drive Capacity' field in the 'Pin Configuration' table on the right, which is also highlighted with a red box and contains the letter 'H'. Below the table, the 'Module name' is P706 and 'Port Capabilities' include AGT0: AGTIO0 and DSMIF1: DSM1CLK0. At the bottom, the 'Pins' tab is selected in the navigation bar.

Name	Value	Link
Symbolic Name		
Comment		
Mode	Peripheral mode	
Pull up/down	None	
IRQ	None	
Output Type	CMOS	
Drive Capacity	H	
Input Latch	None	
Input/Output	P706	ETHPHYCLK_ETHPHYCLK

10. Generate the code
Generate the code with "Generate Project Content".



11. Add Modbus Sample Application
Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

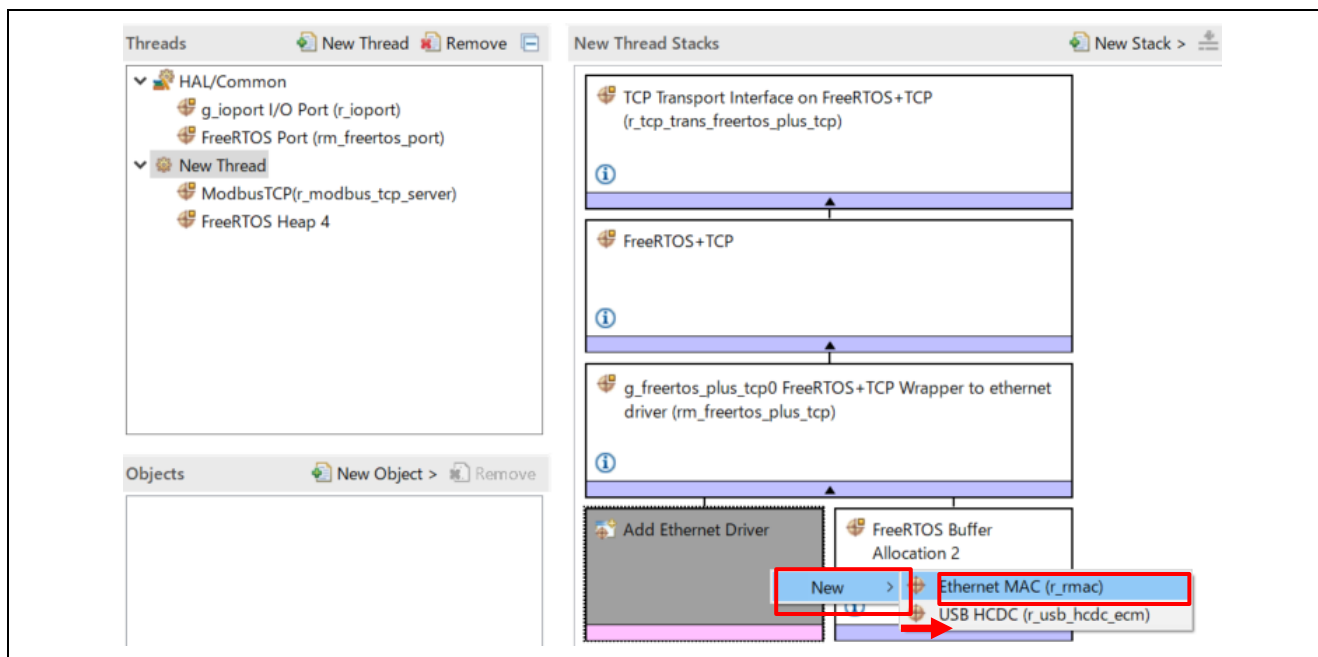


6.2.4 EK-RA8D2 Creating Procedures

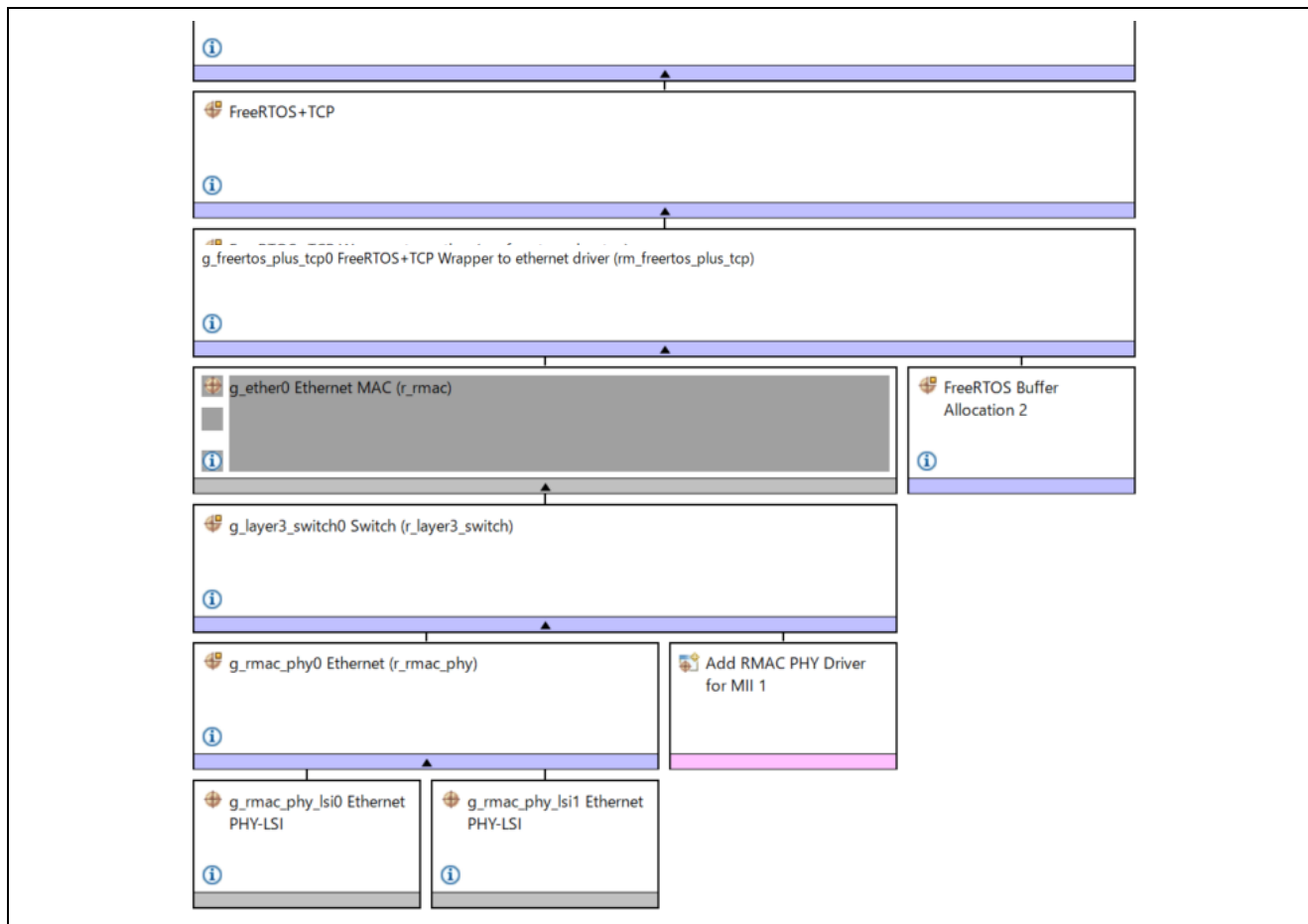
This section describes the procedures for creating EK-RA8D2.

1. Add Ethernet Driver

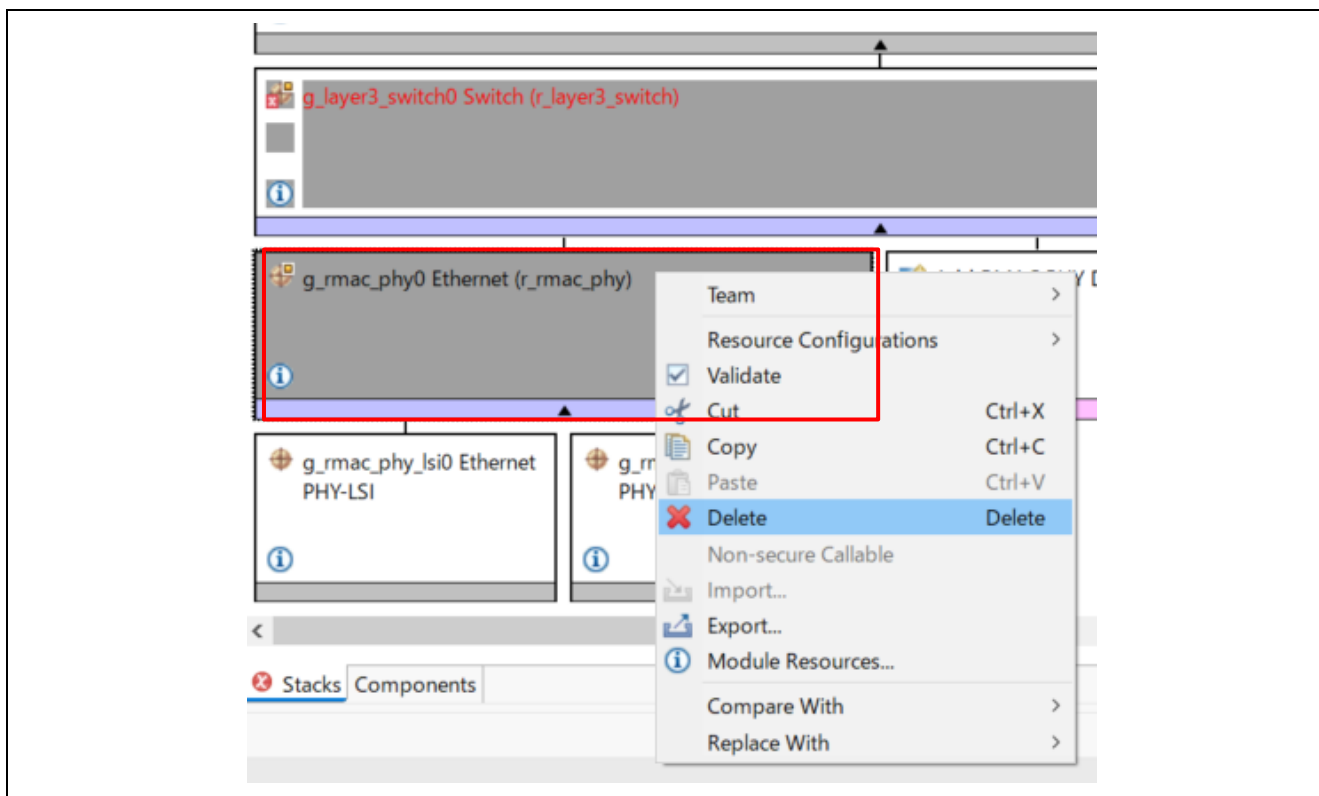
Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



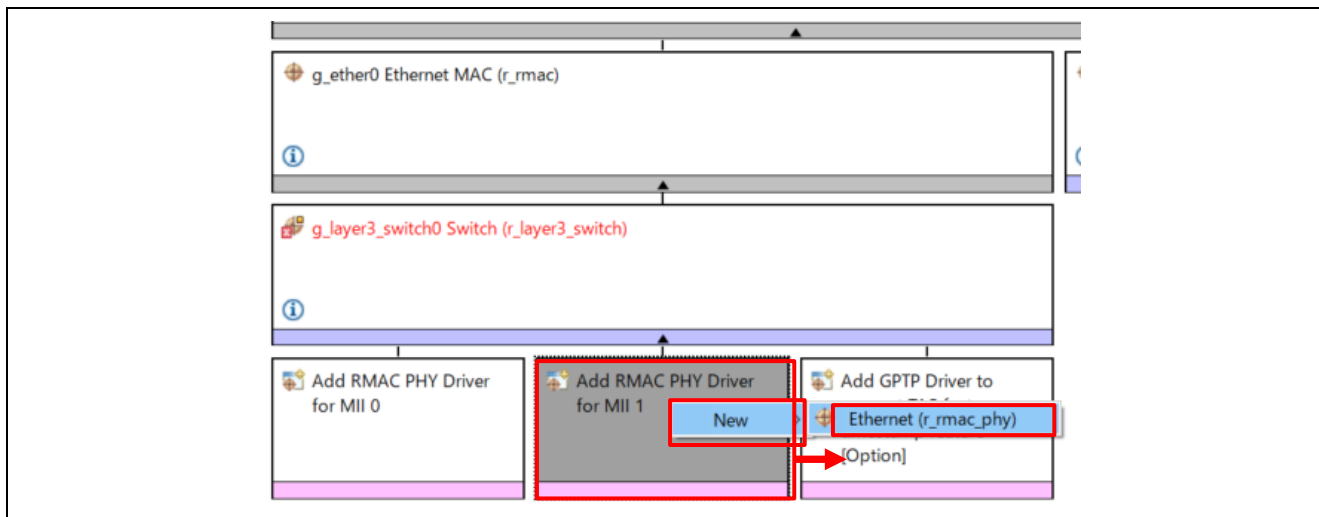
The stack is configured as follows:



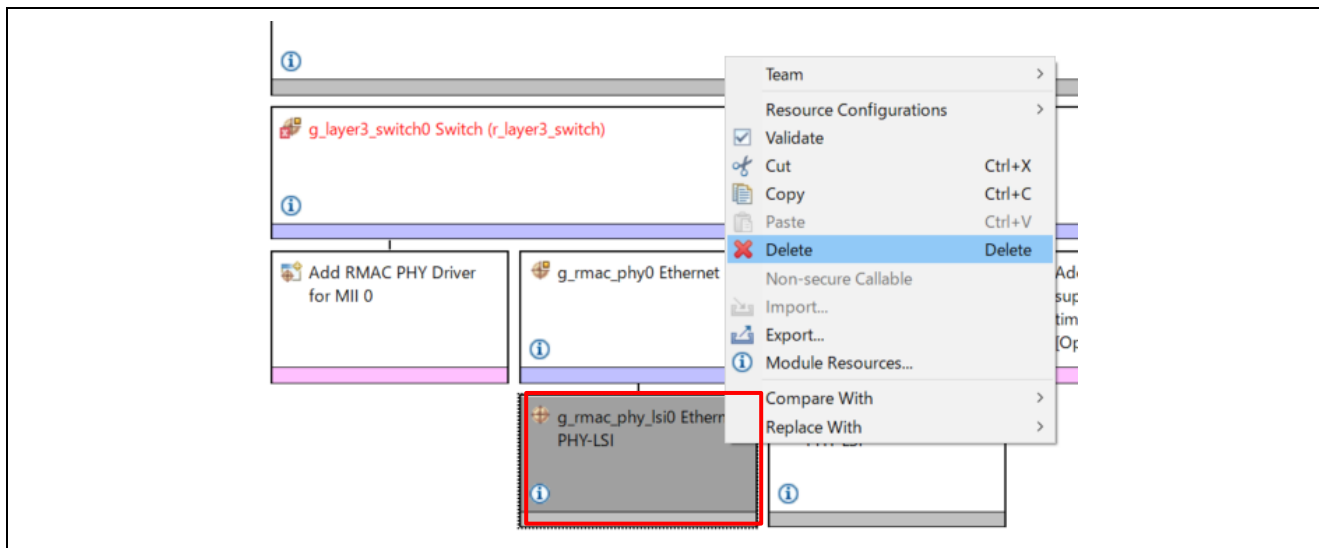
Right-click on “g_mac_phy0 Ethernet (r_rmac_phy)”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



Click “New” → “Ethernet (r_rmac_phy)” to “Add RMAC PHY Driver for MII1”.



Right-click on “g_rmac_phy_Isi0 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



2. Set r_rmac

Open "Properties" of "g_ether0 Ethernet MAC (r_rmac)" in "Stacks" and change "Channel" and "Promiscuous Mode" in "Module g_ether0 Ethernet MAC (r_rmac)" to the following values.

Channel : 1

Promiscuous Mode : **Enable**

The screenshot displays the development environment interface. On the left, a project tree shows components like 'FreeRTOS Port (rm_freertos_port)', 'New Thread', 'ModbusTCP(r_modbus_tcp_server)', and 'FreeRTOS Heap 4'. Below this is a 'Stacks' tab with a 'Properties' sub-tab. The 'Properties' window is open for the 'g_ether0 Ethernet MAC (r_rmac)' module. The 'Channel' property is set to '1' and the 'Promiscuous Mode' is set to 'Enable'. A red box highlights the 'Channel' and 'Promiscuous Mode' properties, and a red arrow points from the 'g_ether0 Ethernet MAC (r_rmac)' module in the 'Stacks' tab to the 'Channel' property in the 'Properties' window.

Property	Value
Parameter Checking	Default (BSP)
Module g_ether0 Ethernet MAC (r_rmac)	
General	
Name	g_ether0
Channel	1
Zero-copy Mode	Disable
Flow control functionality	Disable
MAC address	00:11:22:33:44:55
Filters	
Multicast Mode	Enable
Promiscuous Mode	Enable
Broadcast filter	0

3. Set r_layer3_switch

Open "Properties" of "g_layer3_switch0 Switch (r_layer3_switch)" in "Stacks" and change "gPTP enable" to the following values.

gPTP enable : **Disabled**

The screenshot shows the development environment interface. The top pane displays a stack of components: "g_layer3_switch0 Switch (r_layer3_switch)" (highlighted with a red box), "Add RMAC PHY Driver for MII 0", and "g_rmac_phy0 Ethernet (r_rmac_phy)". The bottom pane shows the "Properties" window for "g_layer3_switch0 Switch (r_layer3_switch)". The "Settings" tab is active, displaying a table of properties.

Property	Value
▼ Common	
Parameter Checking	Default (BSP)
Available queue num	4
gPTP enable	Disabled
Time Aware Shaper	Disabled
▼ Module g_layer3_switch0 Switch (r_layer3_switch)	
> Interrupts	

4. Set r_rmac_phy

Open "Properties" of "g_mac_phy0 Ethernet (r_rmac_phy)" in "Stacks" and change "Channel", "Default PHY-LSI port", and "Select MII type" in "Module g_mac_phy0 Ethernet (r_rmac_phy)" to the following values.

Channel : 1

Default PHY-LSI port : 1

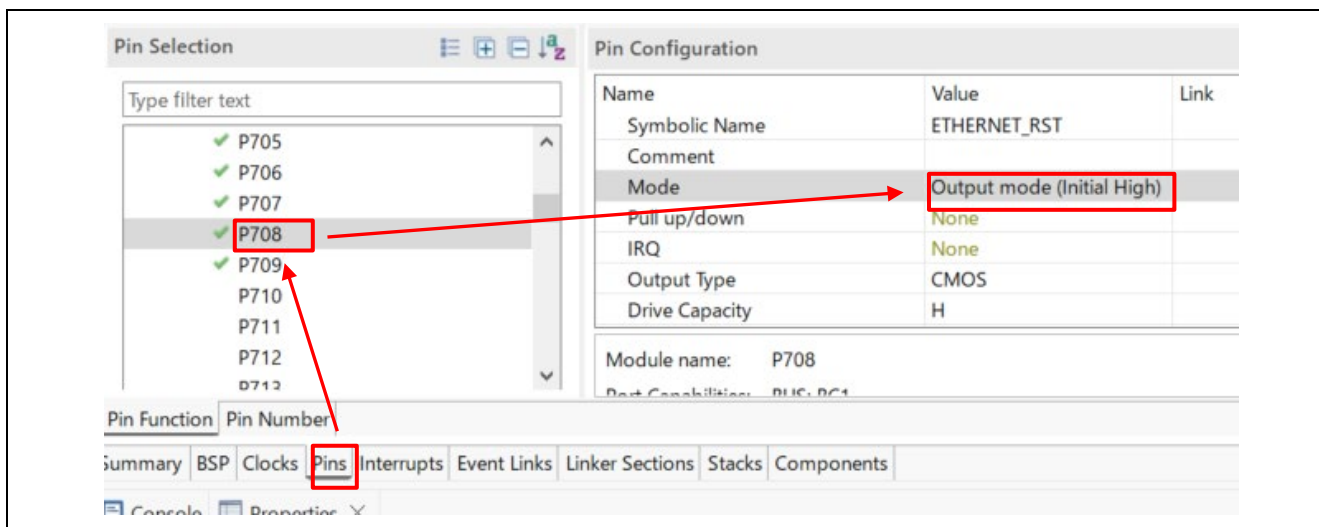
Select MII type : **RGMII**

The screenshot shows the IDE interface with the 'Stacks' tab selected. The 'g_mac_phy0 Ethernet (r_rmac_phy)' component is highlighted in the 'Stacks' view. The 'Properties' window for this component is open, showing the 'Module g_mac_phy0 Ethernet (r_rmac_phy)' section. The 'Select MII type' property is set to 'RGMII'. A red arrow points from the component in the 'Stacks' view to the 'Select MII type' property in the 'Properties' window.

Property	Value
Parameter Checking	Default (BSP)
KSZ8091RNB Target	Disabled
KSZ8041 Target	Disabled
DP83620 Target	Disabled
ICS1894 Target	Disabled
GPY111 Target	Disabled
User Own Target	Disabled
Reference Clock	Default
Module g_mac_phy0 Ethernet (r_rmac_phy)	
Name	g_rmac_phy0
Channel	1
Default PHY-LSI port	1
PHY-LSI Reset Completion Timeout	0x00020000
Select MII type	RGMII
Port Custom Init Function	NULL
Port Custom Link Partner Ability Get Function	NULL
Flow Control	Disable
Management frame format	Clause 22 frame format
MDC clock rate (Hz)	2500000
MDIO hold timing adjustment	0
MDIO capture timing adjustment	0

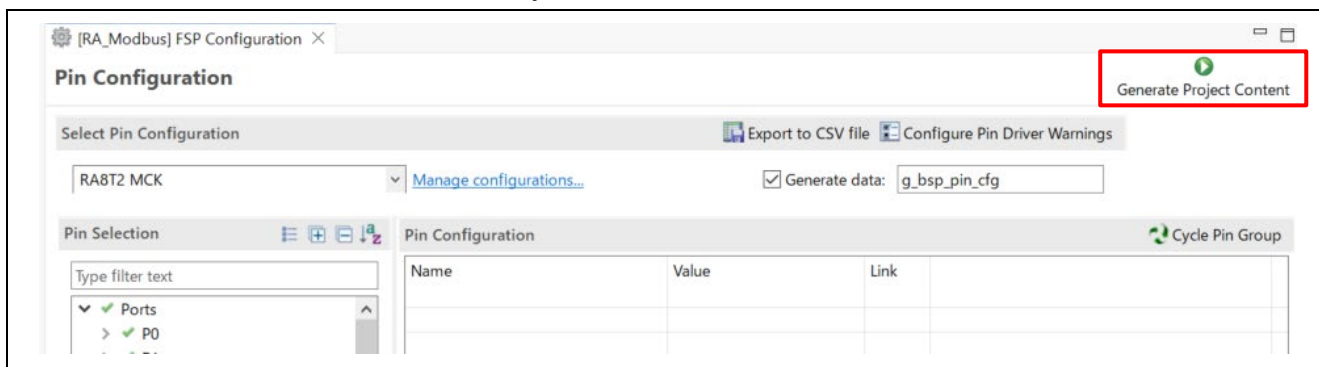
5. Set PHY Reset Terminals

Change the "Mode" of "Pins" → "Ports" → "P7" → "P708" to "Output mode (Initial High)".



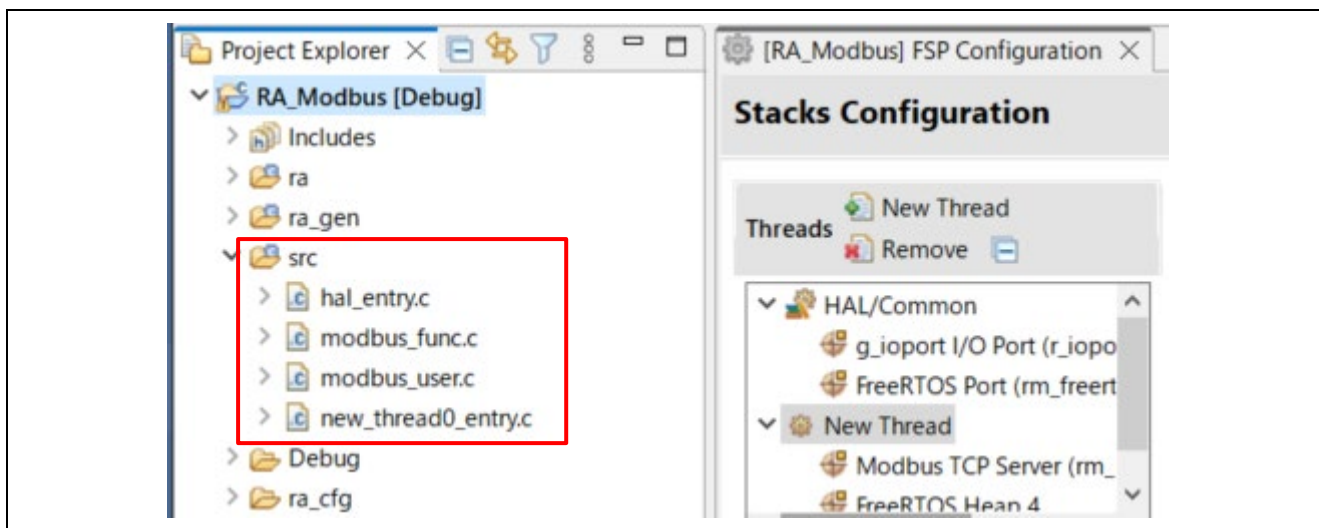
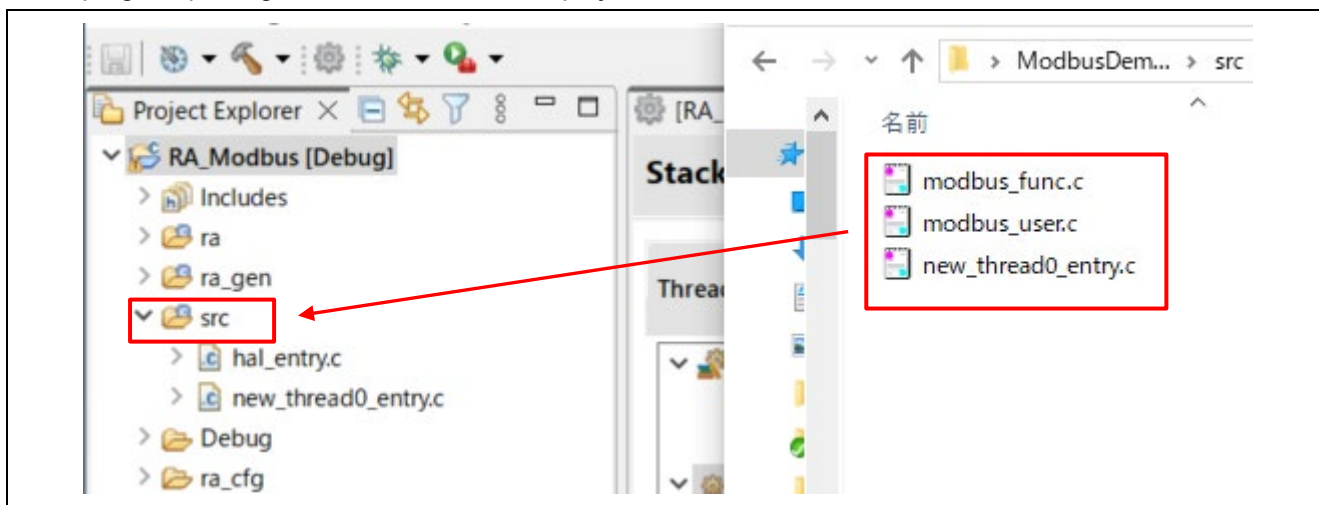
6. Generate the code

Generate the code with "Generate Project Content".



7. Add Modbus Sample Application

Copy `modbus_func.c`, `modbus_user.c`, and `new_thread0_entry.c` from the `src` folder of the sample program package to the `src` folder of the project and overwrite them.

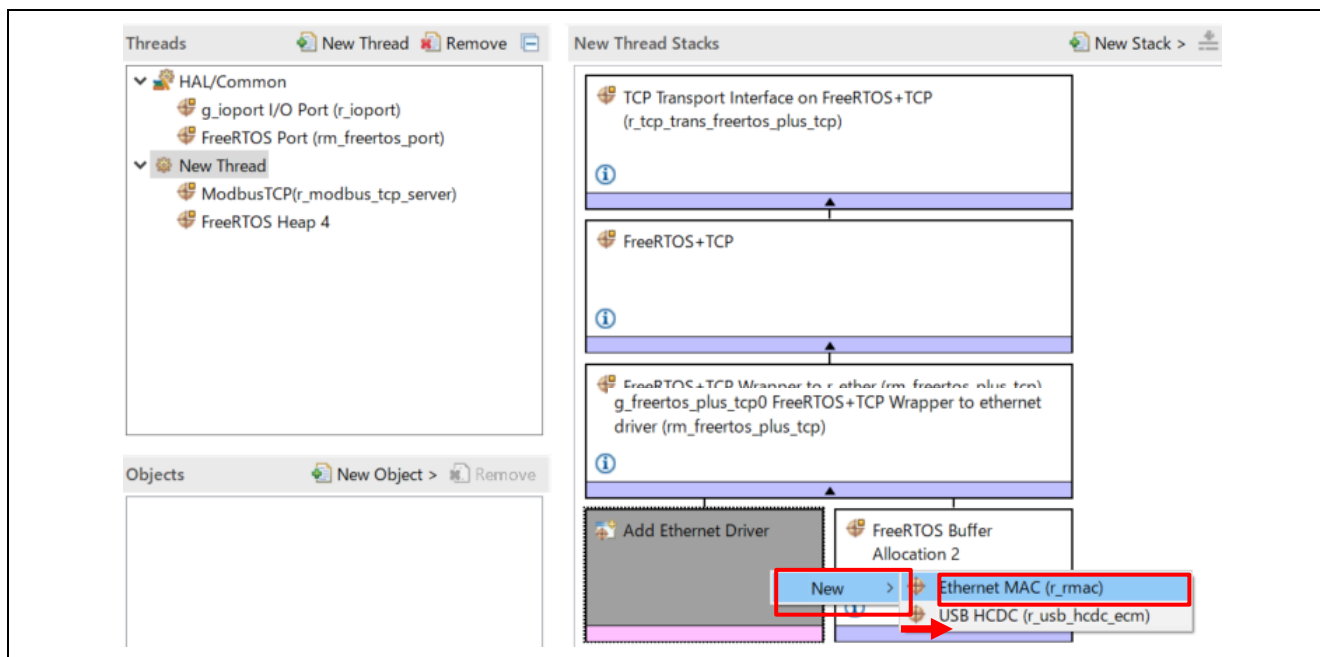


6.2.5 EK-RA8P1 Creating Procedures

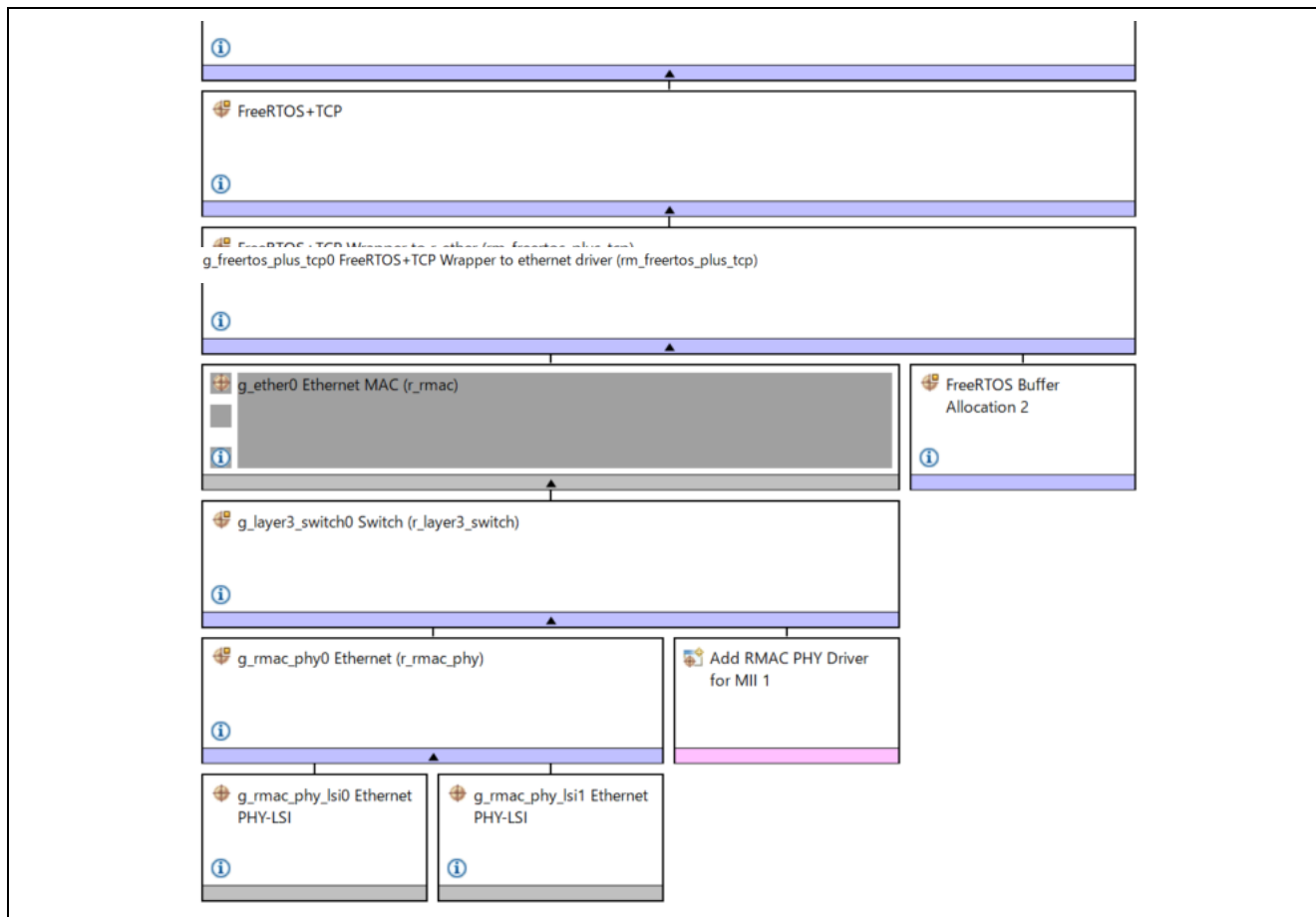
This section describes the procedures for creating EK-RA8P1.

1. Add Ethernet Driver

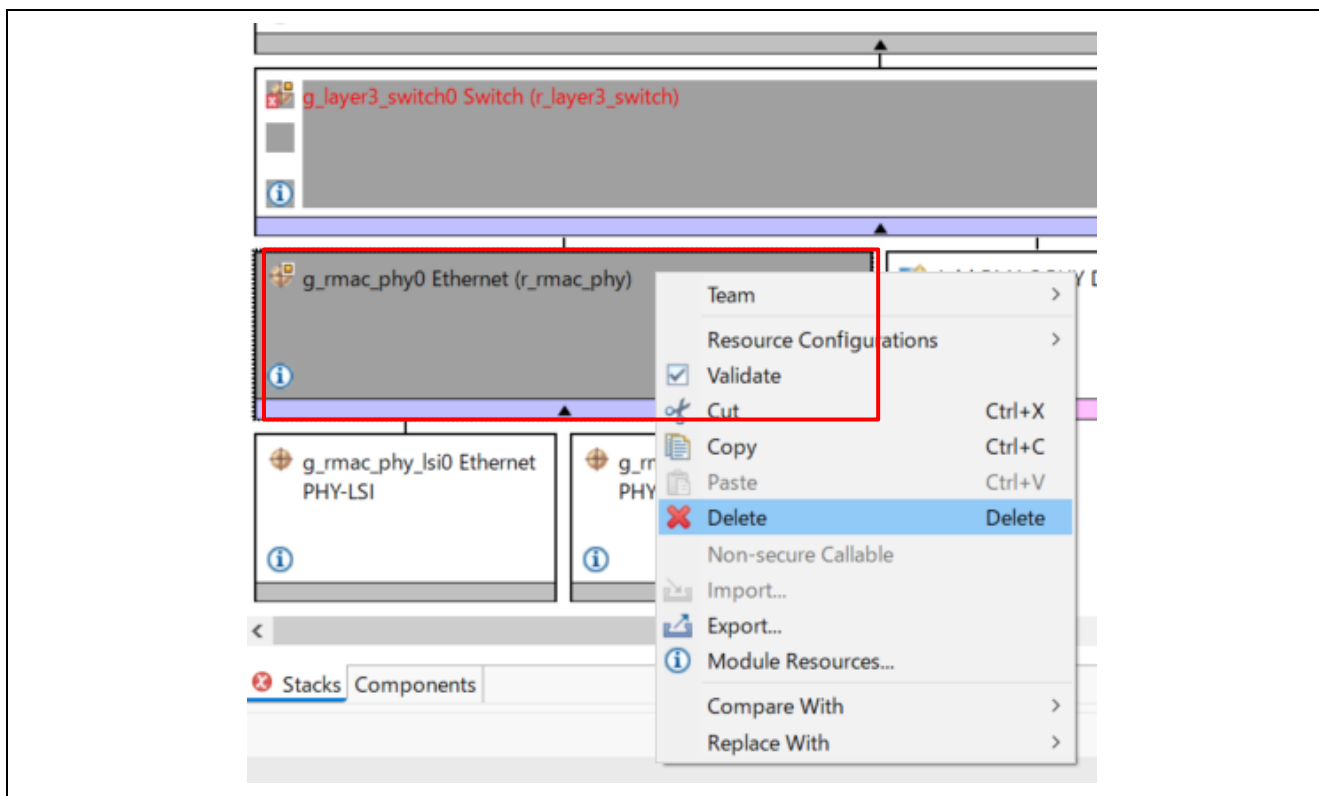
Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



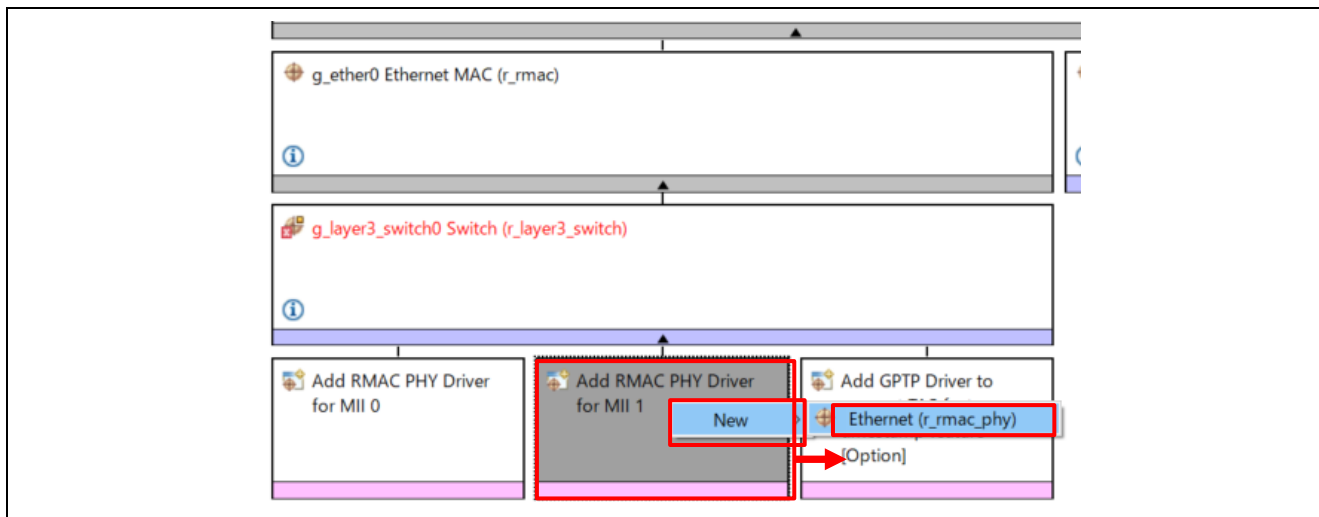
The stack is configured as follows:



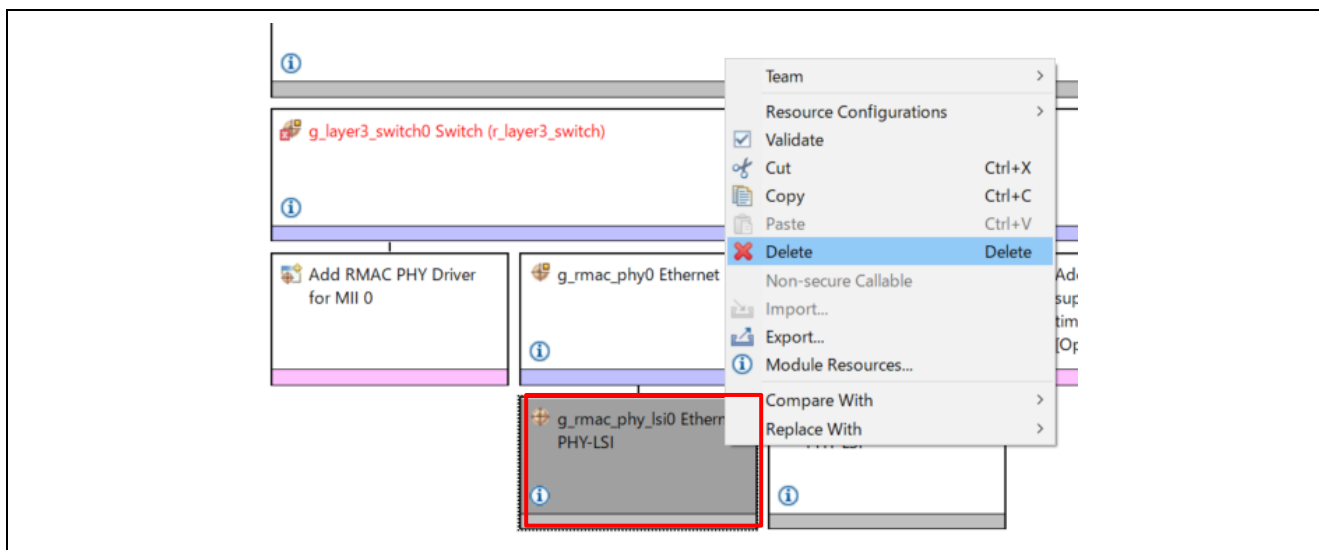
Right-click on “g_mac_phy0 Ethernet (r_rmac_phy)”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



Click “New” → “Ethernet (r_rmac_phy)” to “Add RMAC PHY Driver for MII1”.



Right-click on “g_rmac_phy_Isi0 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



2. Set r_rmac

Open "Properties" of "g_ether0 Ethernet MAC (r_rmac)" in "Stacks" and change "Channel" and "Promiscuous Mode" in "Module g_ether0 Ethernet MAC (r_rmac)" to the following values.

Channel : **1**

Promiscuous Mode : **Enable**

The screenshot shows the Renesas RA Family IDE interface. On the left, the 'Objects' pane lists components: FreeRTOS Port (rm_freertos_port), New Thread, ModbusTCP(r_modbus_tcp_server), and FreeRTOS Heap 4. The main workspace displays a stack diagram with components: g_ether0 Ethernet MAC (r_rmac) and g_layer3_switch0 Switch (r_layer3_switch). The 'g_ether0 Ethernet MAC (r_rmac)' component is highlighted with a red box. A red arrow points from this component to the 'Properties' window below. The 'Properties' window shows the configuration for 'g_ether0 Ethernet MAC (r_rmac)'. The 'Channel' property is set to '1' and 'Promiscuous Mode' is set to 'Enable'. Both values are highlighted with red boxes.

Property	Value
▼ Common	
Parameter Checking	Default (BSP)
▼ Module g_ether0 Ethernet MAC (r_rmac)	
▼ General	
Name	g_ether0
Channel	1
Zero-copy Mode	Disable
Flow control functionality	Disable
MAC address	00:11:22:33:44:55
▼ Filters	
Multicast Mode	Enable
Promiscuous Mode	Enable
Broadcast filter	0

3. Set r_layer3_switch
Open "Properties" of "g_layer3_switch0 Switch (r_layer3_switch)" in "Stacks" and change "gPTP enable" to the following values.

gPTP enable : **Disabled**

The screenshot displays the development environment interface. At the top, the "Stacks" tab is selected, showing a hierarchical view of the system components. A red box highlights the "g_layer3_switch0 Switch (r_layer3_switch)" component. Below this, the "Properties" tab is active, showing a table of properties for the selected component. A red arrow points from the "gPTP enable" property in the table to the "Disabled" value.

Property	Value
Parameter Checking	Default (BSP)
Available queue num	4
gPTP enable	Disabled
Time Aware Shaper	Disabled
Module g_layer3_switch0 Switch (r_layer3_switch)	
> Interrupts	

4. Set r_rmac_phy

Open "Properties" of "g_mac_phy0 Ethernet (r_rmac_phy)" in "Stacks" and change "Channel", "Default PHY-LSI port", and "Select MII type" in "Module g_mac_phy0 Ethernet (r_rmac_phy)" to the following values.

Channel : 1

Default PHY-LSI port : 1

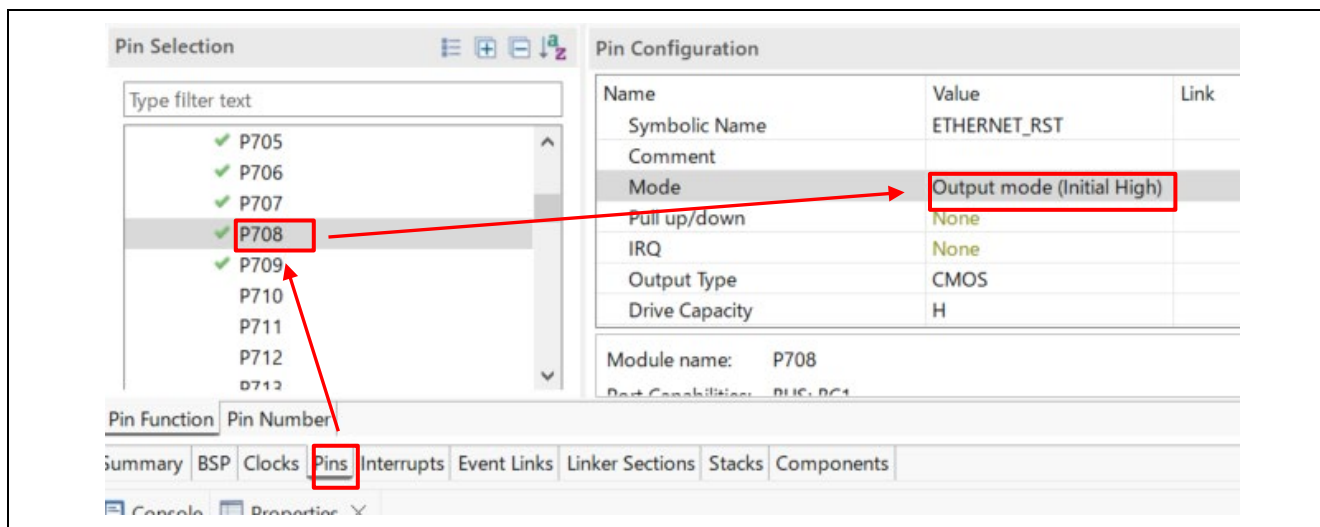
Select MII type : **RGMII**

The screenshot shows the development environment with the 'Properties' window for the 'g_mac_phy0 Ethernet (r_rmac_phy)' module. The 'Module g_mac_phy0 Ethernet (r_rmac_phy)' section is expanded, showing various properties. The 'Channel' and 'Default PHY-LSI port' are both set to '1'. The 'Select MII type' is set to 'RGMII'. A red arrow points from the 'g_mac_phy0 Ethernet (r_rmac_phy)' component in the 'Stacks' view to the 'Properties' window.

Property	Value
Parameter Checking	Default (BSP)
KSZ8091RNB Target	Disabled
KSZ8041 Target	Disabled
DP83620 Target	Disabled
ICS1894 Target	Disabled
GPY111 Target	Disabled
User Own Target	Disabled
Reference Clock	Default
Module g_mac_phy0 Ethernet (r_rmac_phy)	
Name	g_mac_phy0
Channel	1
Default PHY-LSI port	1
PHY-LSI Reset Completion Timeout	0x00020000
Select MII type	RGMII
Port Custom Init Function	NULL
Port Custom Link Partner Ability Get Function	NULL
Flow Control	Disable
Management frame format	Clause 22 frame format
MDC clock rate (Hz)	2500000
MDIO hold timing adjustment	0
MDIO capture timing adjustment	0

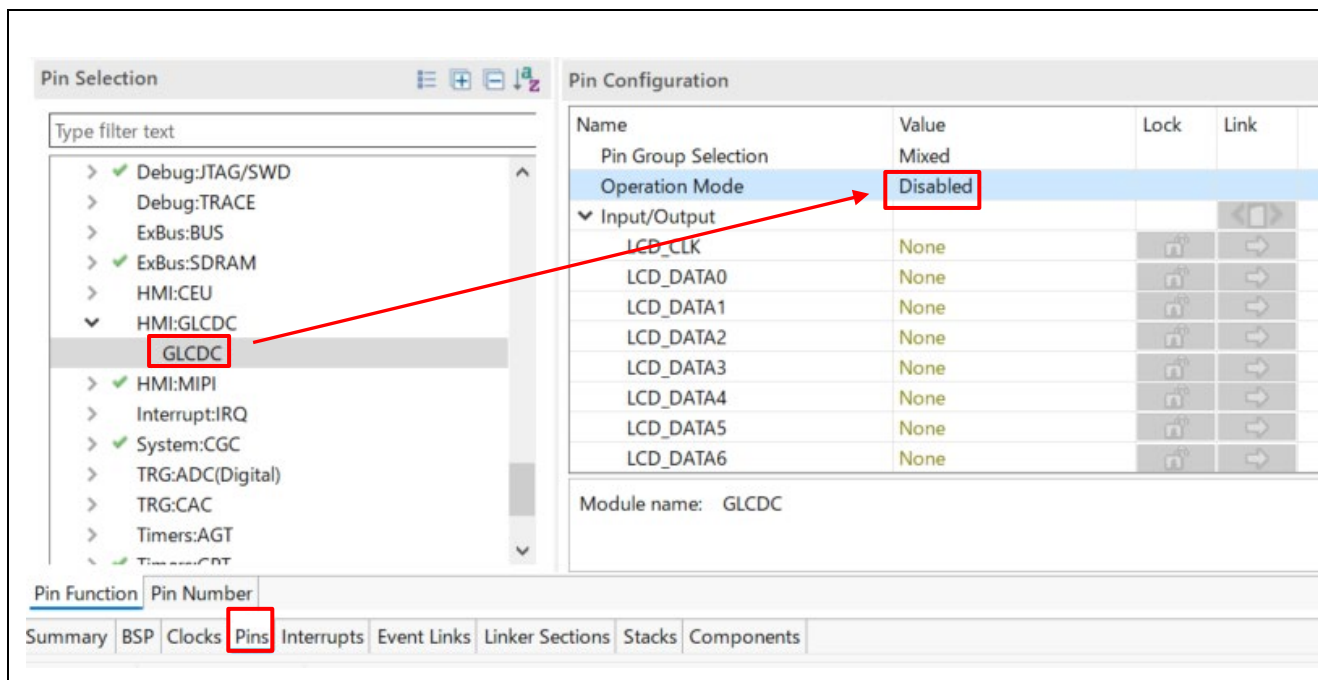
5. Set PHY Reset Terminals

Change the "Mode" of "Pins" → "Ports" → "P7" → "P708" to "Output mode (Initial High)".



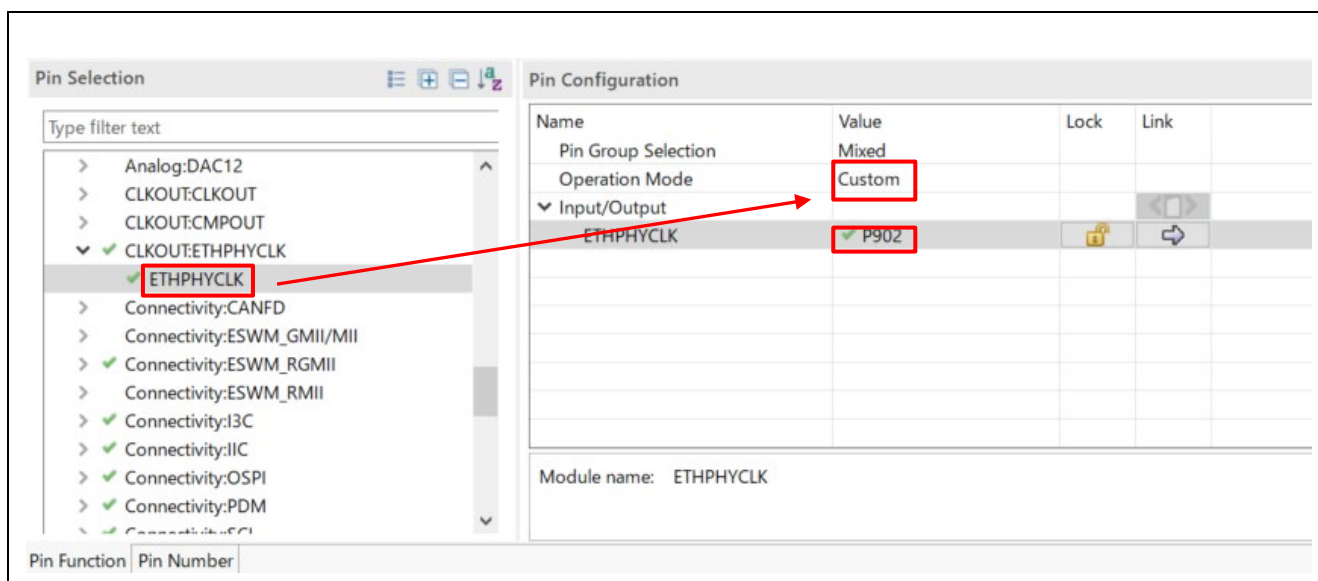
6. Set PHY Clock Terminals

Change the "Operation Mode" of "Pins" → "Peripherals" → " HMI: GLCDC " → " GLCDC " to "Disabled".

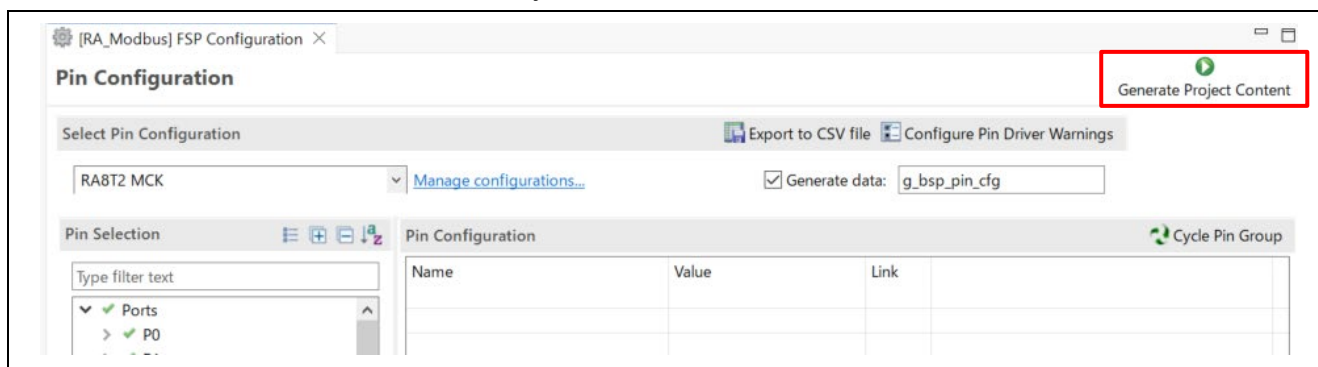


After changing the "Operation Mode" of "Pins" → "Peripherals" → " CLKOUT: ETHPHYCLK " → " ETHPHYCLK " to "Custom", change the pins as follows:

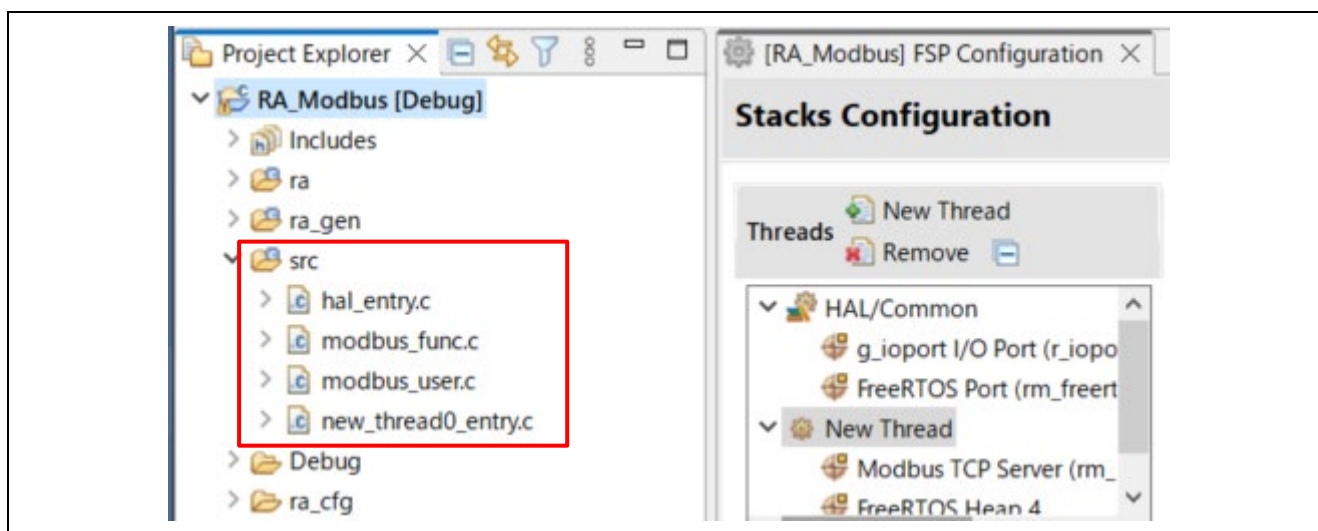
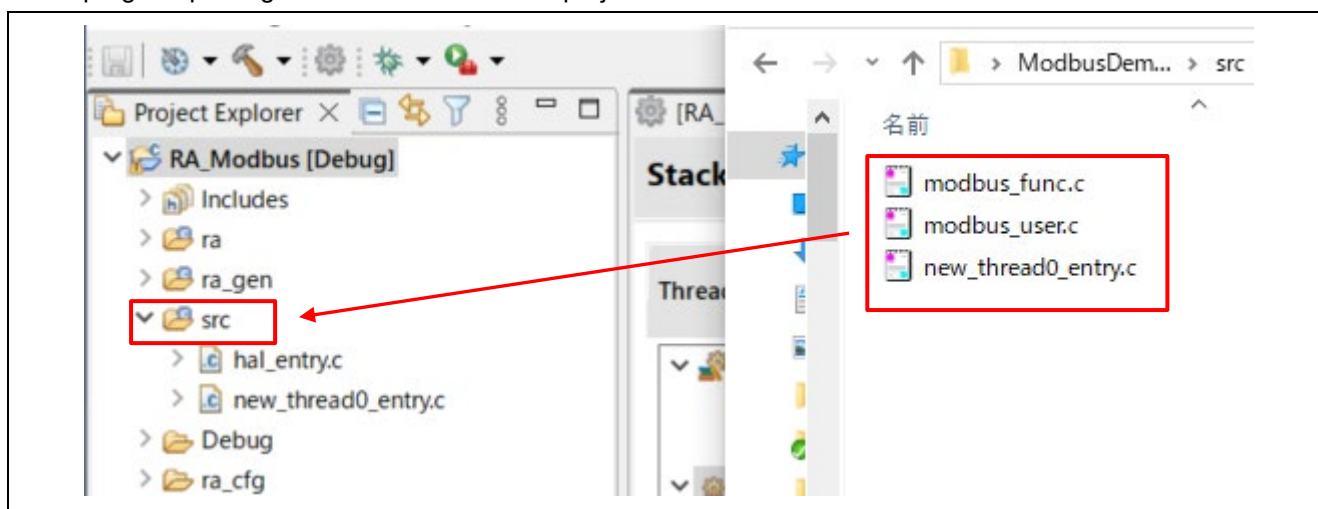
ETHPHYCLK : **P902**



7. Generate the code
Generate the code with "Generate Project Content".



8. Add Modbus Sample Application
Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

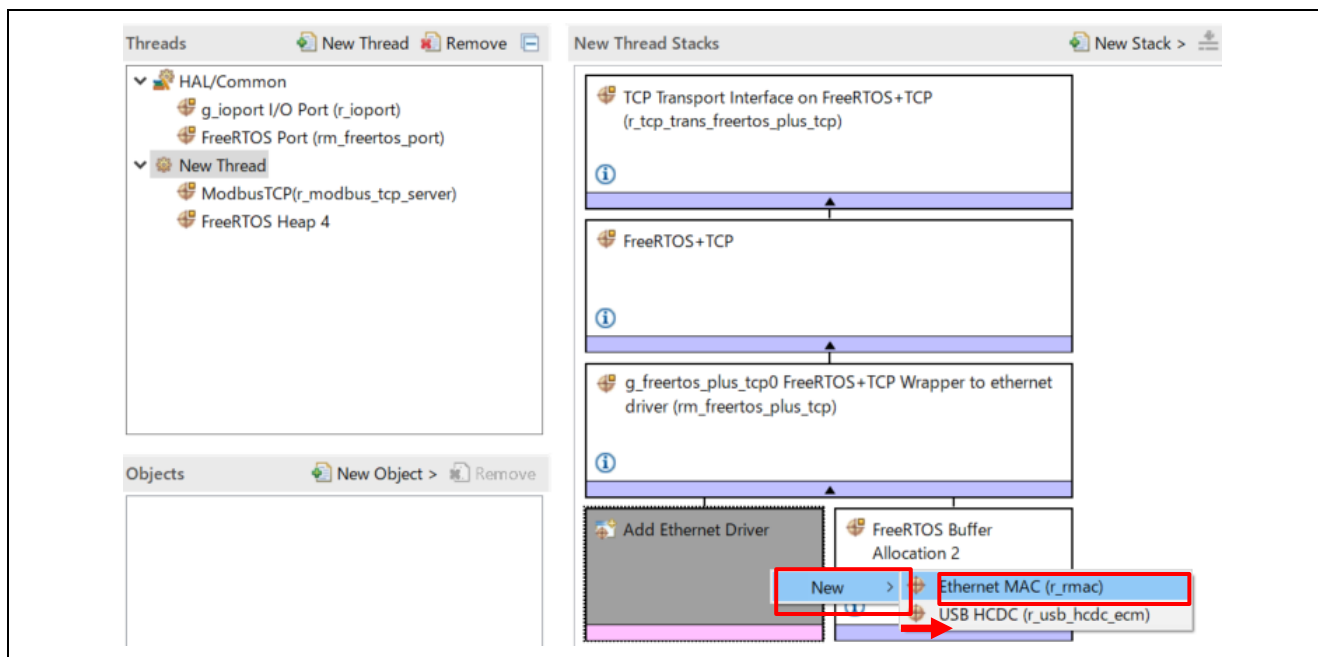


6.2.6 EK-RA8M2 Creating Procedures

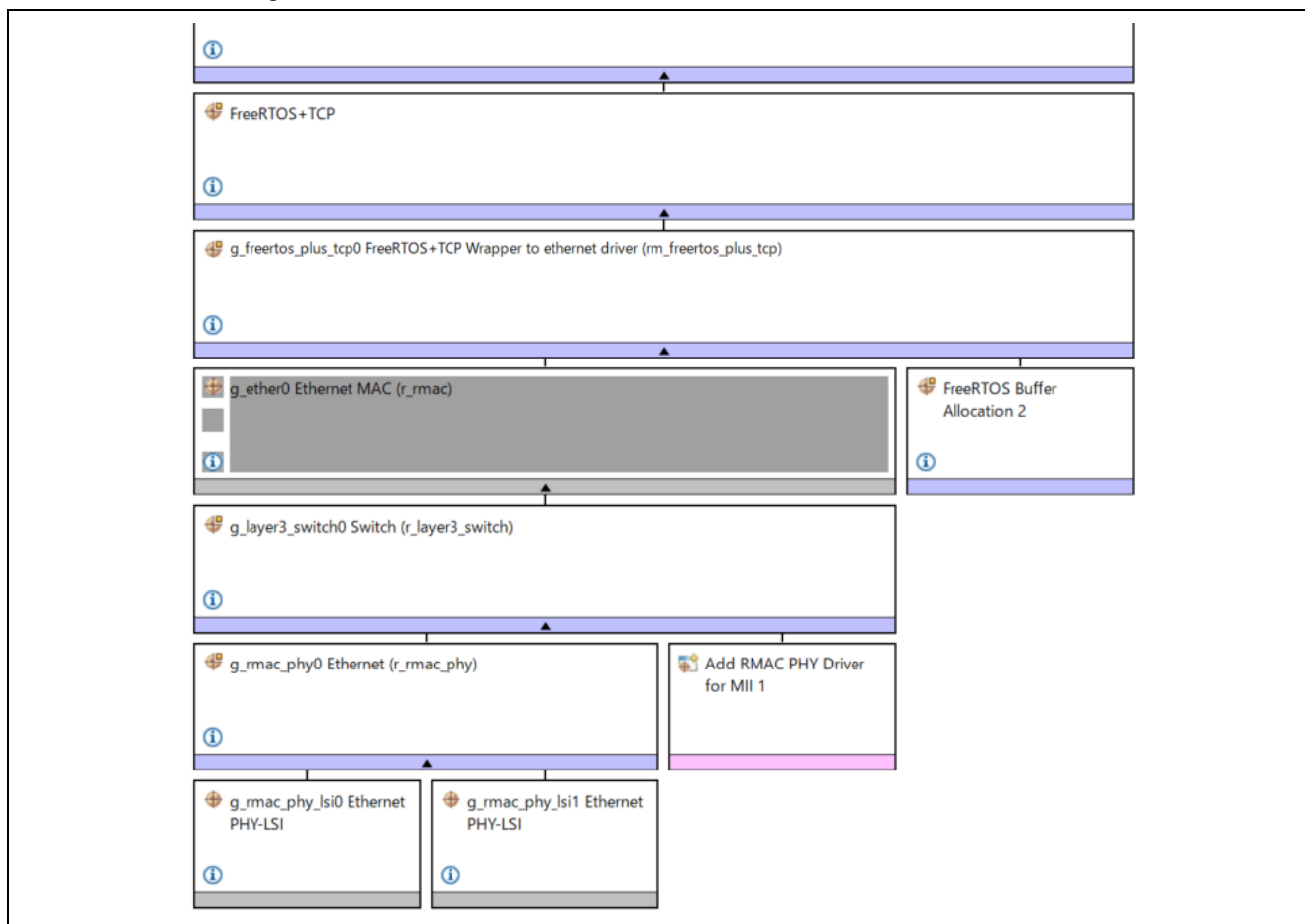
This section describes the procedures for creating EK-RA8M2.

1. Add Ethernet Driver

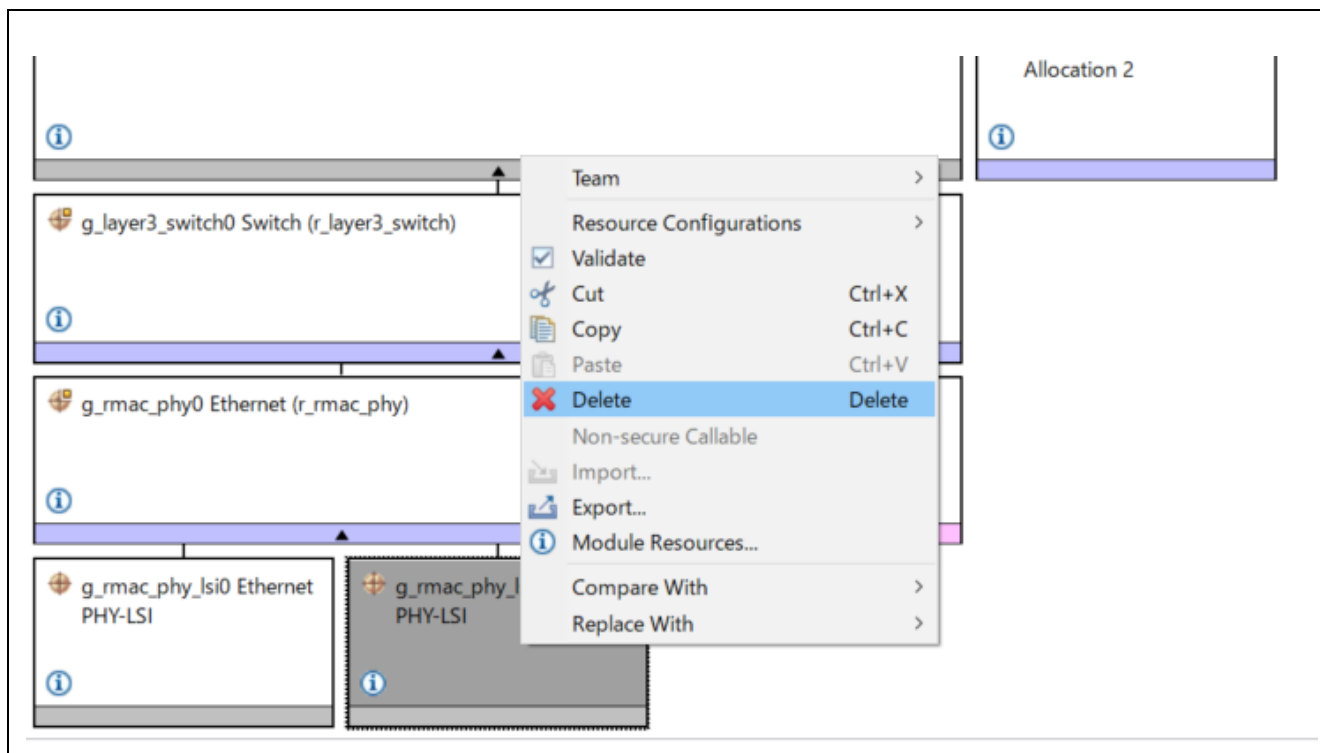
Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



The stack is configured as follows:

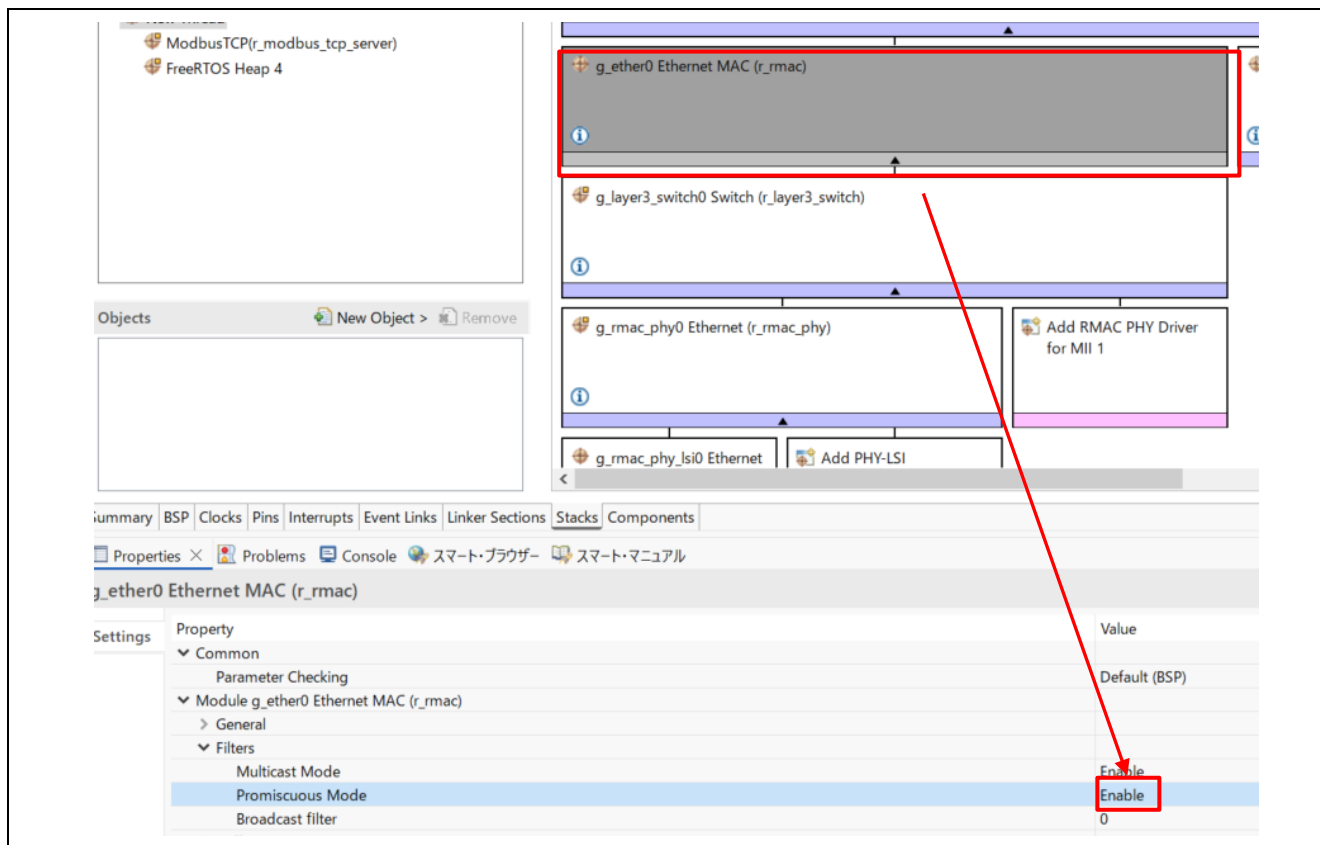


Right-click on “g_rmac_phy_Isi0 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



2. Set r_rmac

Change the “Stacks” → “g_ether0 Ethernet MAC (r_rmac)” → “Module g_ether0 Ethernet MAC (r_rmac)” → “Promiscuous Mode” in “Filters” to “Enable”.



3. Set `r_layer3_switch`
Open "Properties" of "g_layer3_switch0 Switch (r_layer3_switch)" in "Stacks" and change "gPTP enable" to the following values.

gPTP enable: **Disabled**

The screenshot displays the development environment interface. On the left, a tree view shows the project structure with components like HAL/Common, g_ioport I/O Port (r_ioport), FreeRTOS Port (rm_freertos_port), New Thread, ModbusTCP(r_modbus_tcp_server), and FreeRTOS Heap 4. The main area shows a block diagram with components: g_layer3_switch0 Switch (r_layer3_switch), g_rmac_phy0 Ethernet (r_rmac_phy), g_rmac_phy_si0 Ethernet PHY-LSI, and Add PHY-LSI configuration for MII 1. A red box highlights the g_layer3_switch0 Switch (r_layer3_switch) component, and a red arrow points from it to the Properties window.

The Properties window is open, showing the "Stacks" tab. The title bar indicates "layer3_switch0 Switch (r_layer3_switch)". The "Settings" section is expanded, showing a table of properties:

Property	Value
Common	
Parameter Checking	Default (BSP)
Available queue num	4
gPTP enable	Disabled
Time Aware Shaper	Disabled
Module g_layer3_switch0 Switch (r_layer3_switch)	

4. Set r_rmac_phy

Open “Properties” of “g_mac_phy0 Ethernet (r_rmac_phy)” in “Stacks” and change “Select MII type”, “MDIO hold timing adjustment”, and “MDIO capture timing adjustment” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

Select MII type : **RGMII**

MDIO hold timing adjustment : **1**

MDIO capture timing adjustment : **1**

ModbusTCP(r_modbus_tcp_server)
FreeRTOS Heap 4

Objects: New Object > Remove

Summary: BSP | Clocks | Pins | Interrupts | Event Links | Linker Sections | **Stacks** | Components

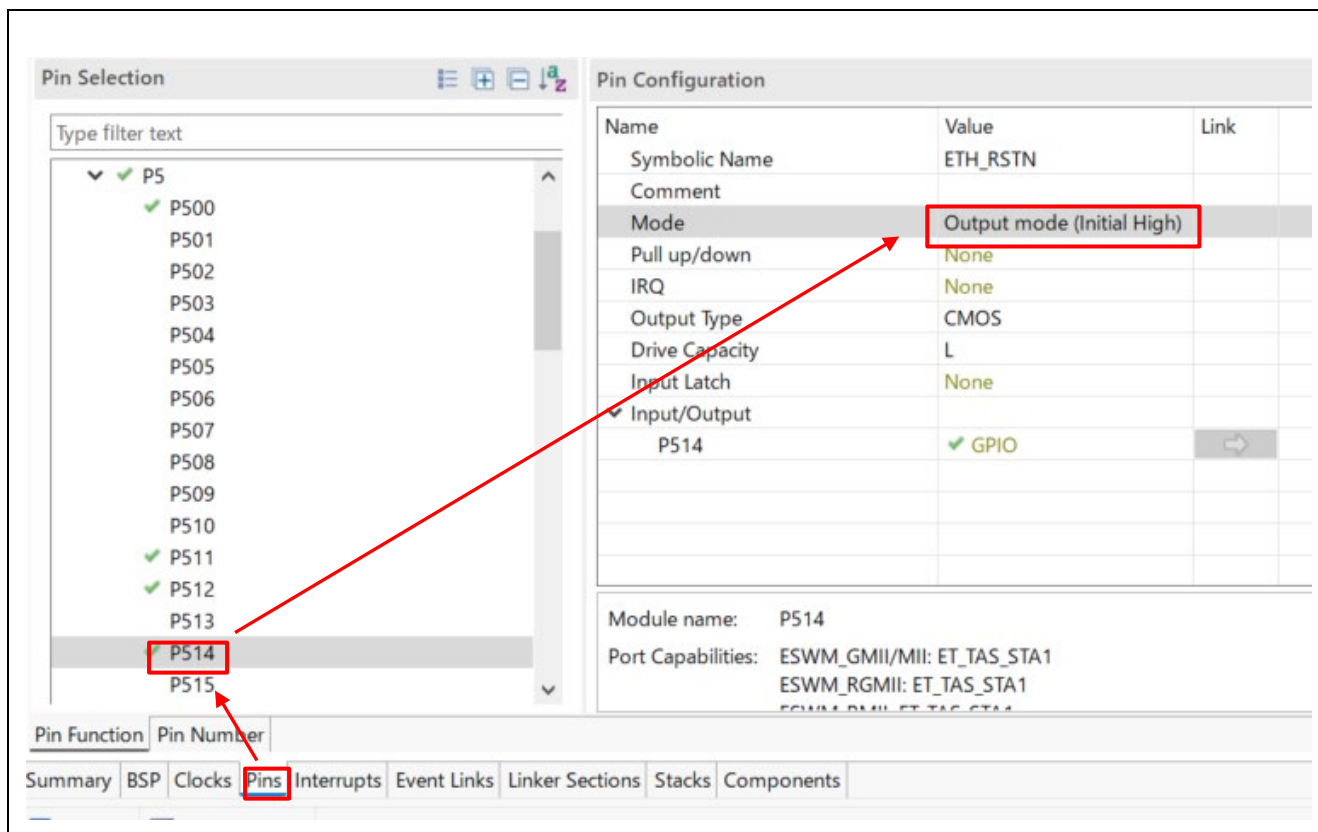
問題 | コンソール | **プロパティ** | スマート・ブラウザー | スマート・マニュアル

g_mac_phy0 Ethernet (r_rmac_phy)

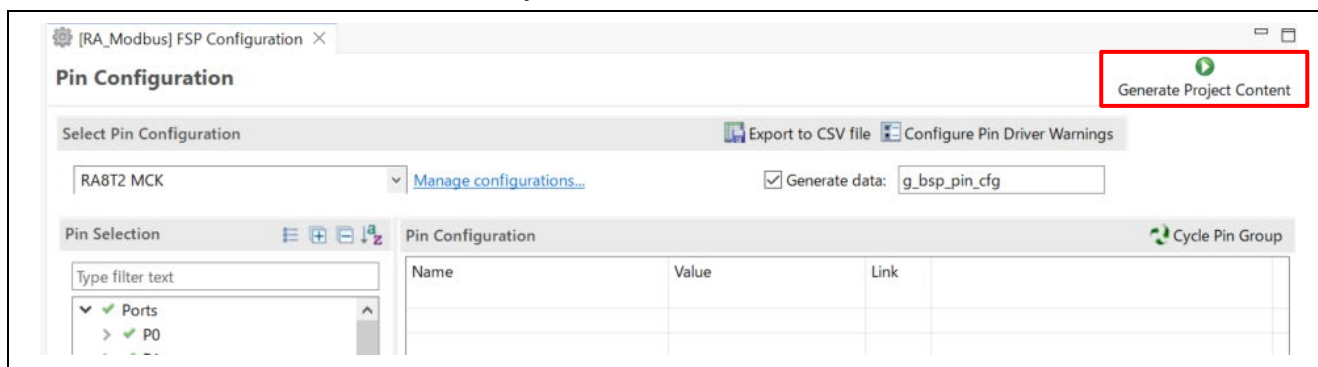
Settings	Value
KSZ8091RNB Target	Disabled
KSZ8041 Target	Disabled
DP83620 Target	Disabled
ICS1894 Target	Disabled
GPY111 Target	Disabled
User Own Target	Disabled
Reference Clock	Default
Module g_mac_phy0 Ethernet (r_rmac_phy)	
Name	g_mac_phy0
Channel	0
Default PHY-LSI port	0
PHY-LSI Reset Completion Timeout	0x00020000
Select MII type	RGMII
Port Custom Init Function	NULL
Port Custom Link Partner Ability Get Function	NULL
Flow Control	Disable
Management frame format	Clause 22 frame format
MDC clock rate (Hz)	2500000
MDIO hold timing adjustment	1
MDIO capture timing adjustment	1

5. Set PHY Reset Terminals

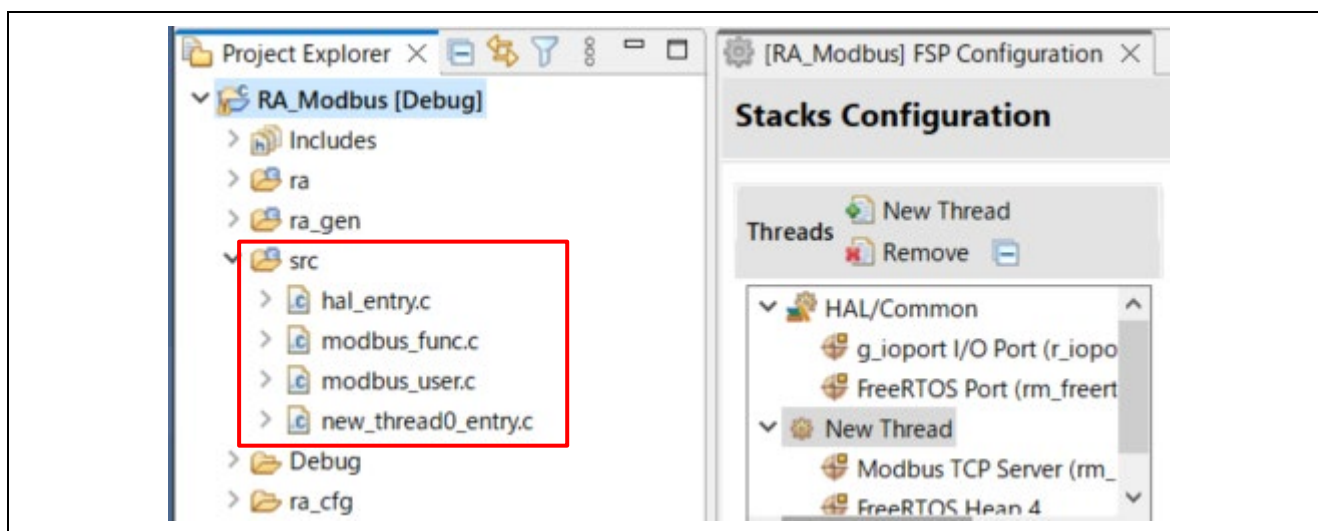
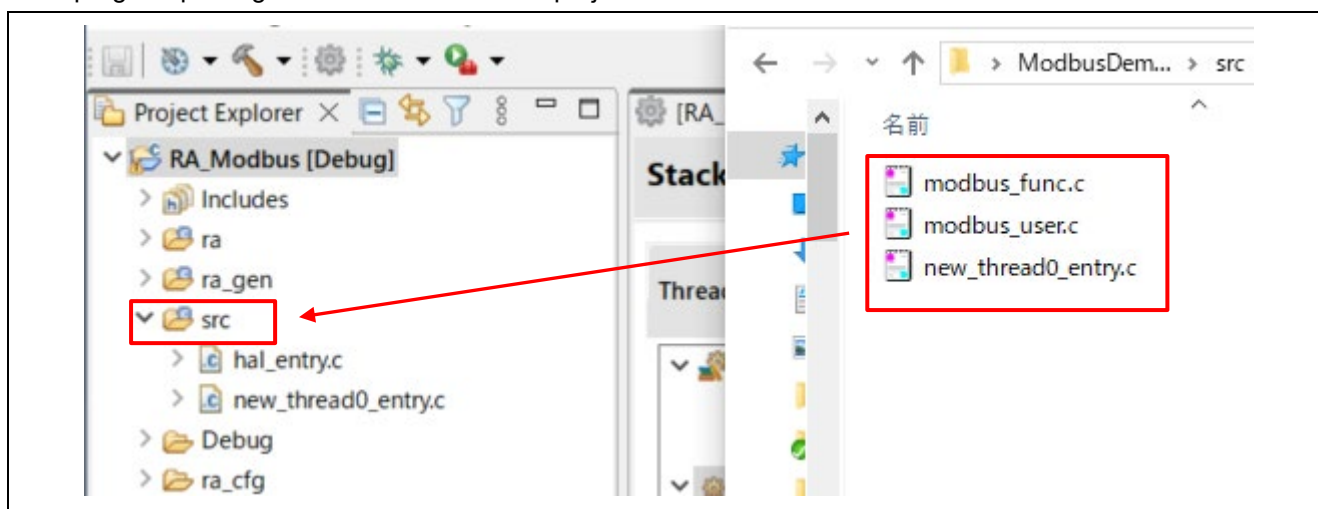
Change the "Mode" of "Pins" → "Ports" → "P5" → "P514" to "Output mode (Initial High)".



6. Generate the code
Generate the code with "Generate Project Content".



7. Add Modbus Sample Application
Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

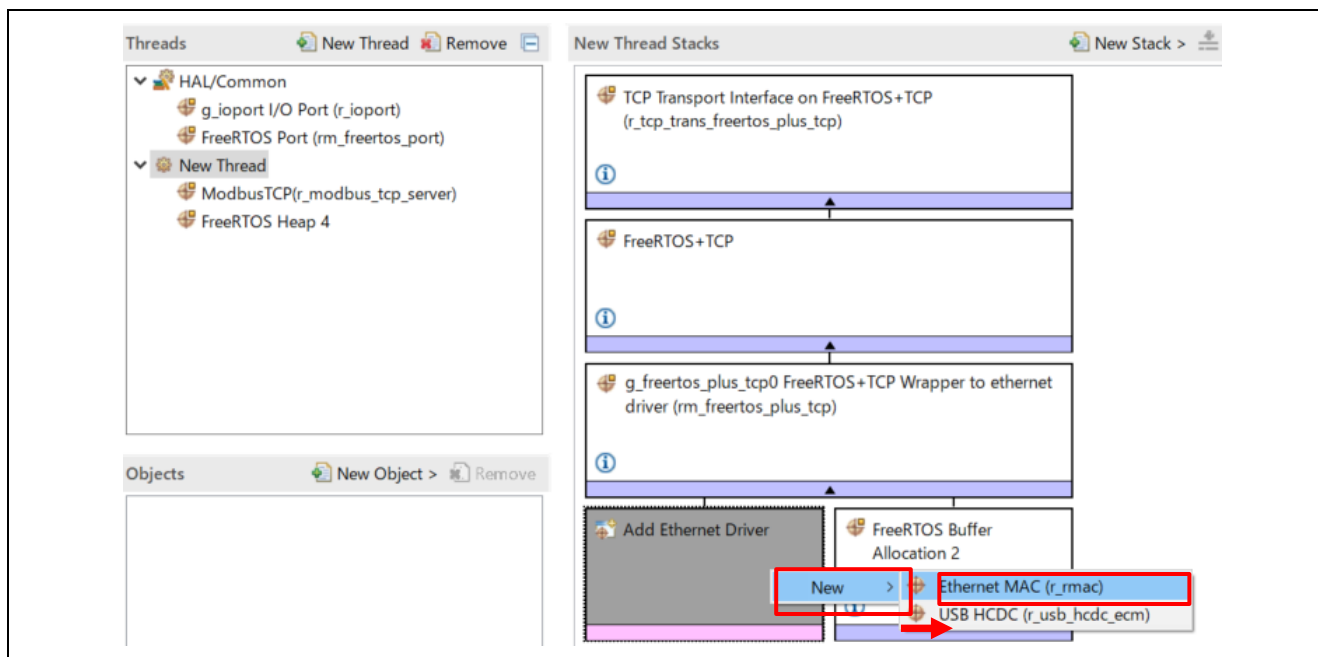


6.2.7 EK-RA8T2 Creating Procedures

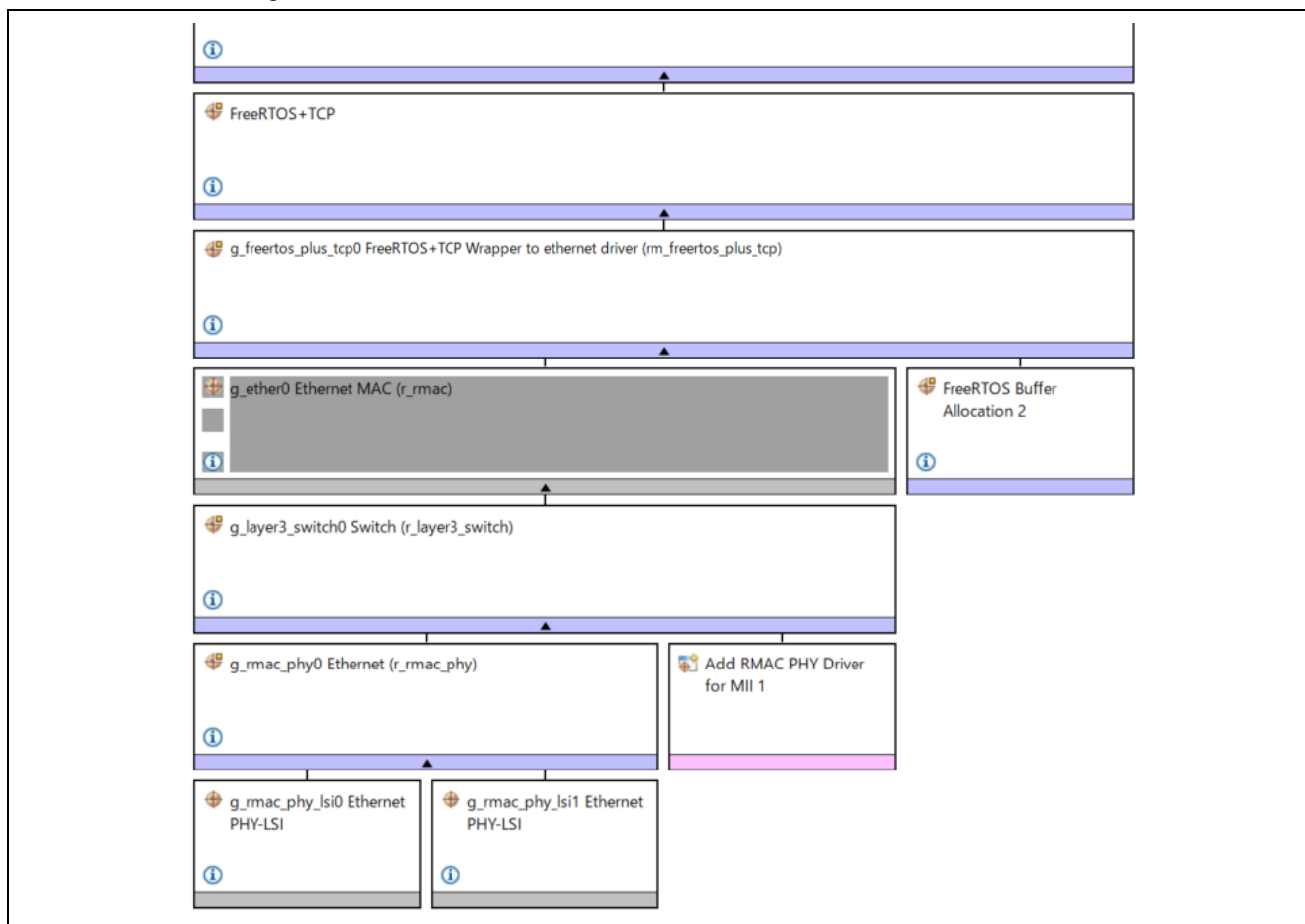
This section describes the procedures for creating EK-RA8T2.

1. Add Ethernet Driver

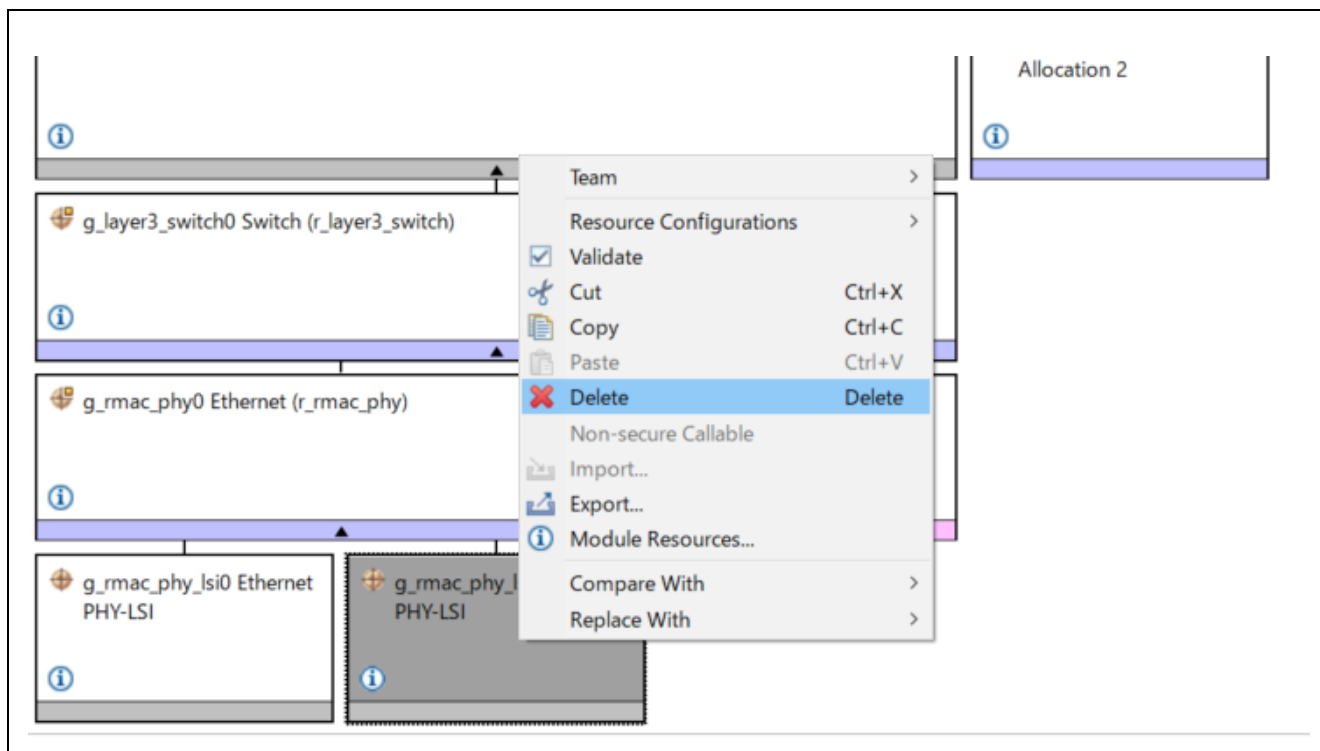
Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



The stack is configured as follows:

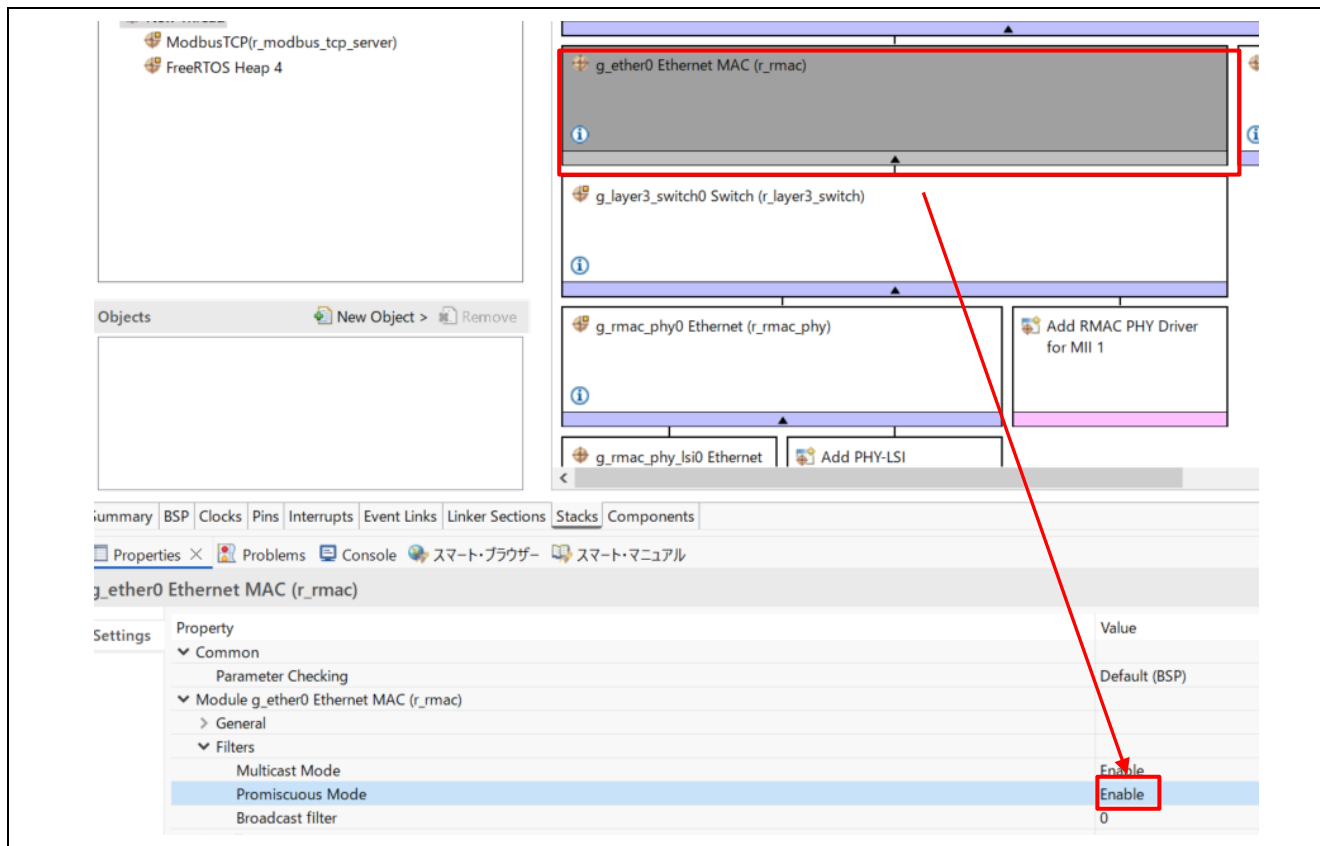


Right-click on “g_rmac_phy_Isi1 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



2. Set r_rmac

Change the “Stacks” → “g_ether0 Ethernet MAC (r_rmac)” → “Module g_ether0 Ethernet MAC (r_rmac)” → “Promiscuous Mode” in “Filters” to “Enable”.



3. Set `r_layer3_switch`
Open "Properties" of "g_layer3_switch0 Switch (r_layer3_switch)" in "Stacks" and change "gPTP enable" to the following values.

gPTP enable : **Disabled**

The screenshot displays the development environment interface. On the left, a tree view shows the project structure under "HAL/Common", including "g_ioport I/O Port (r_ioport)", "FreeRTOS Port (rm_freertos_port)", "New Thread", "ModbusTCP(r_modbus_tcp_server)", and "FreeRTOS Heap 4". The "Objects" panel below it shows "New Object" and "Remove" buttons. The main area shows a stack diagram with components: "g_layer3_switch0 Switch (r_layer3_switch)", "g_rmac_phy0 Ethernet (r_rmac_phy)", "g_rmac_phy_0 Ethernet PHY-LSI", and "Add PHY-LSI configuration for MII 1". A red box highlights the "g_layer3_switch0 Switch (r_layer3_switch)" component, and a red arrow points from it to the "Properties" tab of the "layer3_switch0 Switch (r_layer3_switch)".

The "Properties" tab shows the following settings:

Property	Value
Common	
Parameter Checking	Default (BSP)
Available queue num	4
gPTP enable	Disabled
Time Aware Shaper	Disabled
Module g_layer3_switch0 Switch (r_layer3_switch)	

4. Set r_rmac_phy

Open "Properties" of "g_mac_phy0 Ethernet (r_rmac_phy)" in "Stacks" and change "Select MII type", "MDIO hold timing adjustment", and "MDIO capture timing adjustment" in "Module g_mac_phy0 Ethernet (r_rmac_phy)" to the following values.

Select MII type: **GMII**

MDIO hold timing adjustment : **7**

MDIO capture timing adjustment : **7**

The screenshot shows the IDE interface with the 'Properties' window for the 'g_mac_phy0 Ethernet (r_rmac_phy)' module. The 'Stacks' view on the right shows the component hierarchy. The 'Properties' window on the left lists various settings. The 'Select MII type' field is highlighted with a red box and set to 'GMII'. The 'MDIO hold timing adjustment' field is highlighted with a red box and set to '7'. The 'MDIO capture timing adjustment' field is highlighted with a red box and set to '7'. A red arrow points from the component in the 'Stacks' view to the 'Select MII type' field in the 'Properties' window.

Property	Value
KSZ8091RNB Target	Disabled
KSZ8041 Target	Disabled
DP83620 Target	Disabled
ICS1894 Target	Disabled
GPY111 Target	Disabled
VSC8541 Target	Disabled
User Own Target	Disabled
Reference Clock	Default
Module g_mac_phy0 Ethernet (r_rmac_phy)	
Interrupts	
Name	g_mac_phy0
Channel	0
Default PHY-LSI port	0
PHY-LSI Reset Completion Timeout	0x00020000
Select MII type	GMII
Port Custom Init Function	NULL
Port Custom Link Partner Ability Get Function	NULL
Flow Control	Disable
Management frame format	Clause 22 frame format
MDC clock rate (Hz)	2500000
MDIO hold timing adjustment	7
MDIO capture timing adjustment	7
Frame preemption	Disable
Verify frame interval	10

5. Set ESWM Terminals

Change the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESC" → "ESC" to "Disabled".

The screenshot shows the Pin Configuration tool interface. On the left, the 'Pin Selection' pane shows a tree view with 'Peripherals' expanded, then 'Connectivity: ESC', and 'ESC' selected. A red box highlights 'ESC'. A red arrow points from this box to the 'Operation Mode' field in the 'Pin Configuration' pane, which is also highlighted with a red box and set to 'Disabled'. Below the panes, the 'Pins' tab is selected in the bottom navigation bar, also highlighted with a red box.

Name	Value	Lock	Link
Pin Group Selection	Mixed		
Operation Mode	Disabled		
Input/Output			
CAT0_MDC	None		
CAT0_MDIO	None		
CAT12CCLK	None		
CAT12CDATA	None		
CATIRQ	None		
CATLATCH0	None		
CATLATCH1	None		
CATLEDERR	None		
CATLEDRUN	None		
CATLEDSTER	None		
CATLINKACT0	None		

Module name: ESC

Pin Function: Pin Number

Summary | BSP | Clocks | Pins | Interrupts | Event Links | Linker Sections | Stacks | Components

6. Set PHY Reset Terminals

Change the "Mode" of "Pins" → "Ports" → "P7" → "P711" to "Output mode (Initial High)".

The screenshot shows the Pin Configuration tool interface. On the left, the 'Pin Selection' pane shows a tree view with 'Ports' expanded, then 'P7', and 'P711' selected. A red box highlights 'P711'. A red arrow points from this box to the 'Mode' field in the 'Pin Configuration' pane, which is also highlighted with a red box and set to 'Output mode (Initial High)'. Below the panes, the 'Pins' tab is selected in the bottom navigation bar, also highlighted with a red box.

Name	Value	Link
Symbolic Name		
Comment		
Mode	Output mode (Initial High)	
Pull up/down	None	
IRQ	None	
Output Type	CMOS	
Drive Capacity	L	
Input Latch	None	
Input/Output		
P711	GPIO	

Module name: P711

Port Capabilities: AGT0: AGTEE0
ESC: CATRESETOUT
ESW0: GMII/MII, CAT0: DM

Pin Function: Pin Number

Summary | BSP | Clocks | Pins | Interrupts | Event Links | Linker Sections | Stacks | Components

7. Set Clocks

Change " ESWCLK Src", " ESWPHYCLK Src", "ETHPHYCLK Src", and " ETHPHYCLK Div" in "Clocks" to the following values.

ESWCLK Src : **PLL1P**

ESWPHYCLK Src : **PLL1P**

ETHPHYCLK Src : **PLL1R**

ETHPHYCLK Div : / **16**

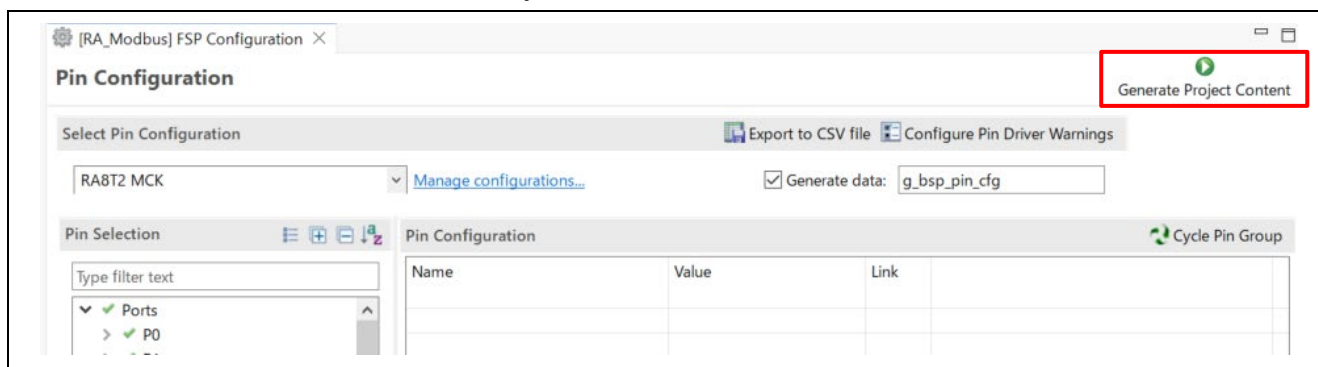
Clocks Configuration

Generate Project Content
Restore Defaults

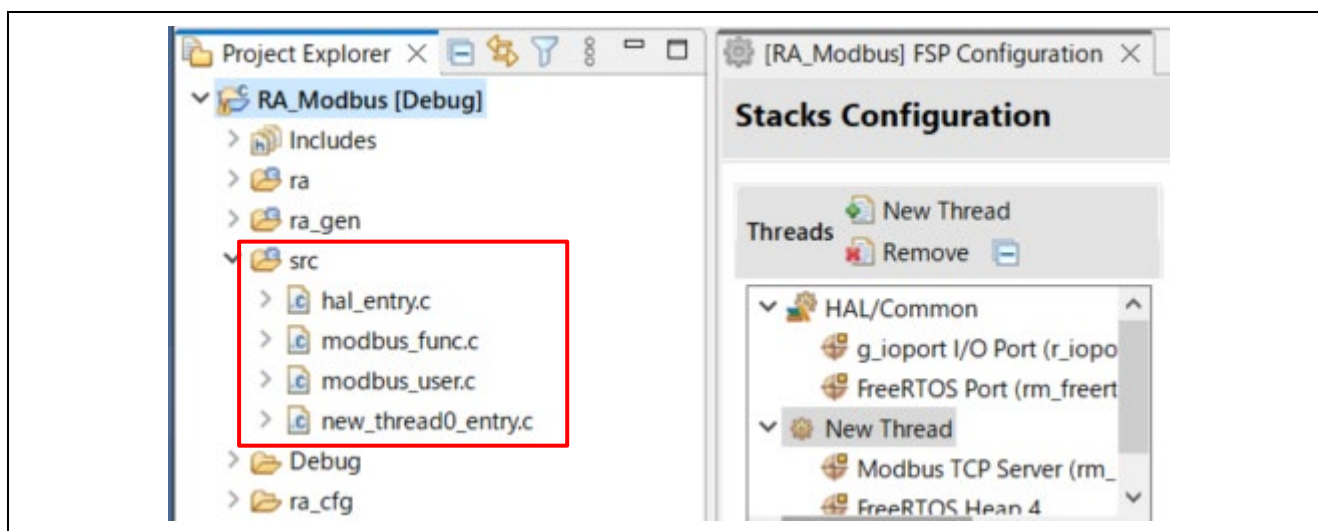
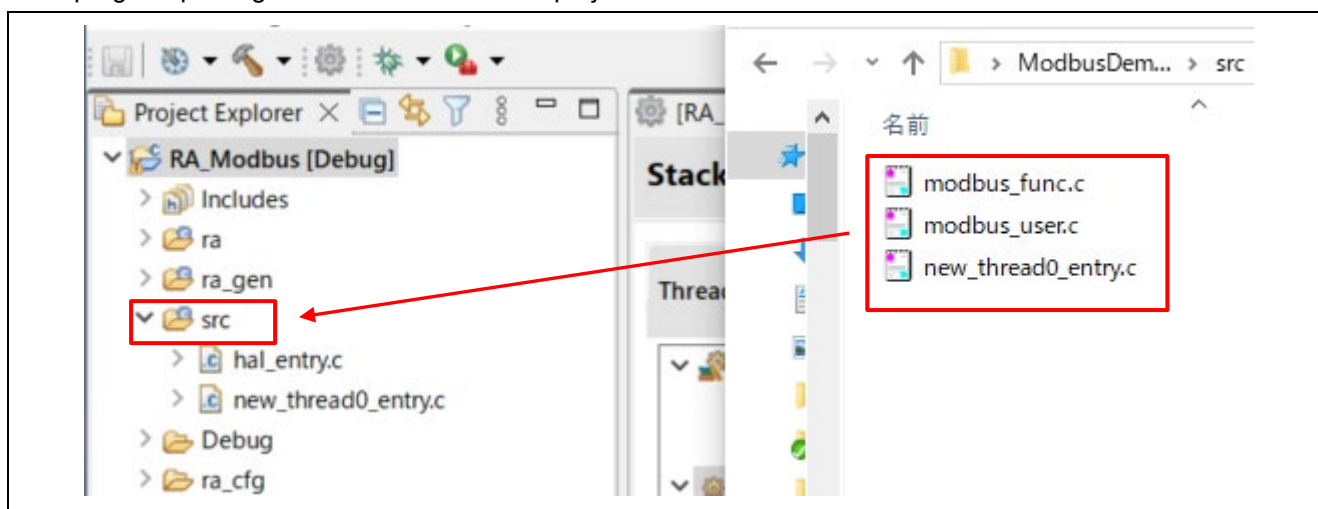
I3CCLK Disabled	I3CCLK Div /4	I3CCLK 0Hz	
USBCLK Disabled	USBCLK Div /10	USBCLK 0Hz	
OCTACLK Disabled	OCTACLK Div /1	OCTACLK 0Hz	OM_n_SCLK 0Hz
ADCCLK Disabled	ADCCLK Div /4	ADCCLK 0Hz	
ESWCLK Src: PLL1P	ESWCLK Div /10	ESWCLK 100MHz	
ESWPHYCLK Src: PLL1	ESWPHYCLK Div /2	ESWPHYCLK 500MHz	
ETHPHYCLK Src: PLL1	ETHPHYCLK Div /16	ETHPHYCLK 25MHz	
ESCCLK Disabled	ESCCLK Div /8	ESCCLK 0Hz	
DSMIFCLK Disabled	DSMIFCLK Div /2	DSMIFCLK 0Hz	

Summary | **BSP** | **Clocks** | Pins | Interrupts | Event Links | Linker Sections | Stacks | Components

8. Generate the code
Generate the code with "Generate Project Content".



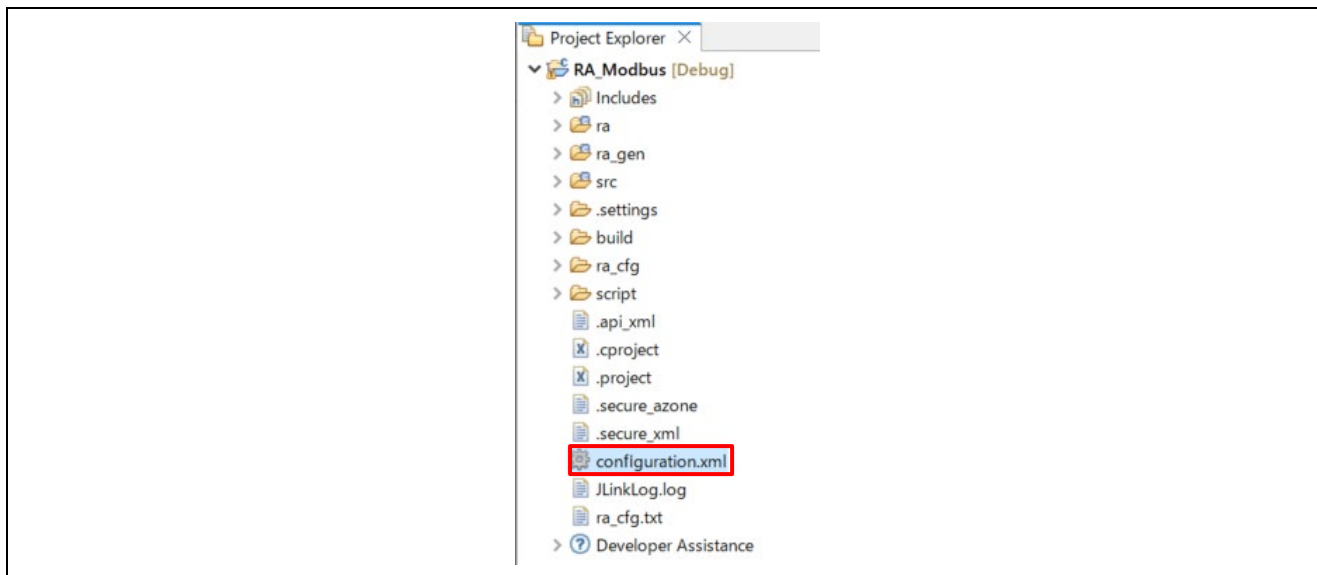
9. Add Modbus Sample Application
Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.



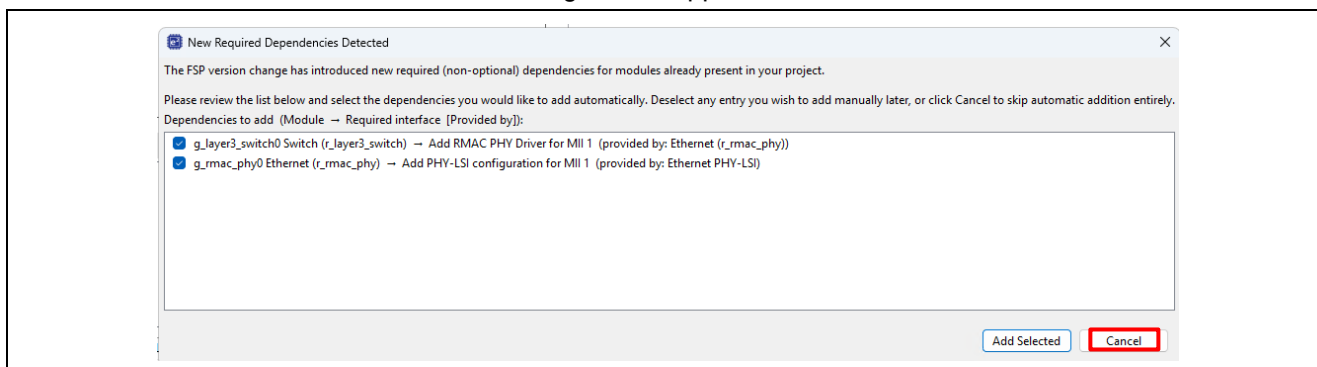
6.3 Changing of Configuration

Describes the procedure for changing the FSP configuration in a Modbus sample project.

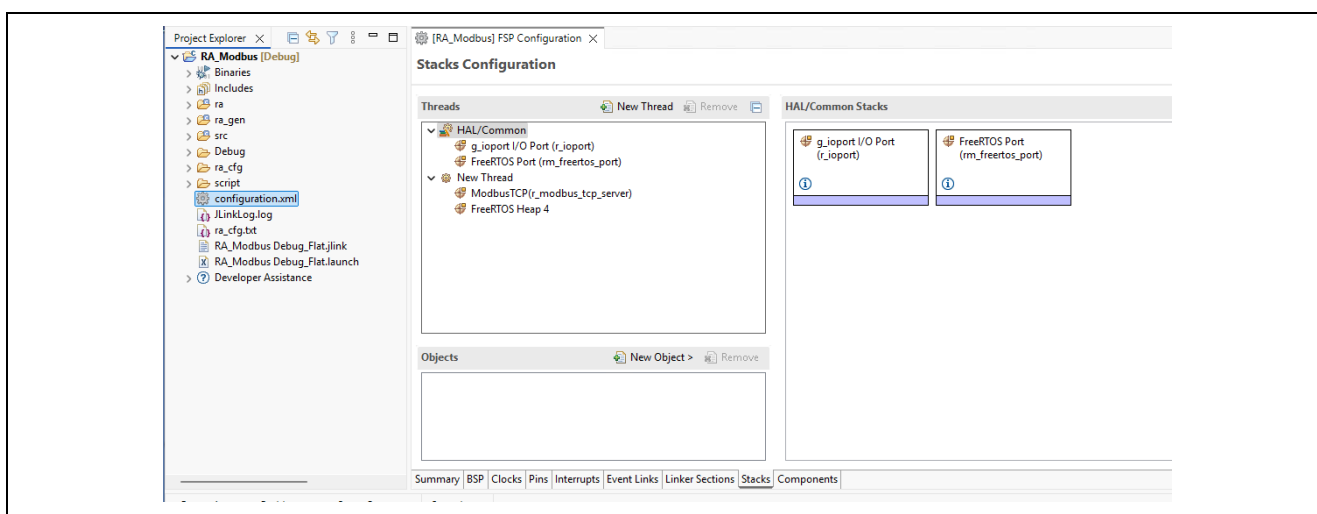
1. Open “configuration.xml” of the “RA_Modbus” project.



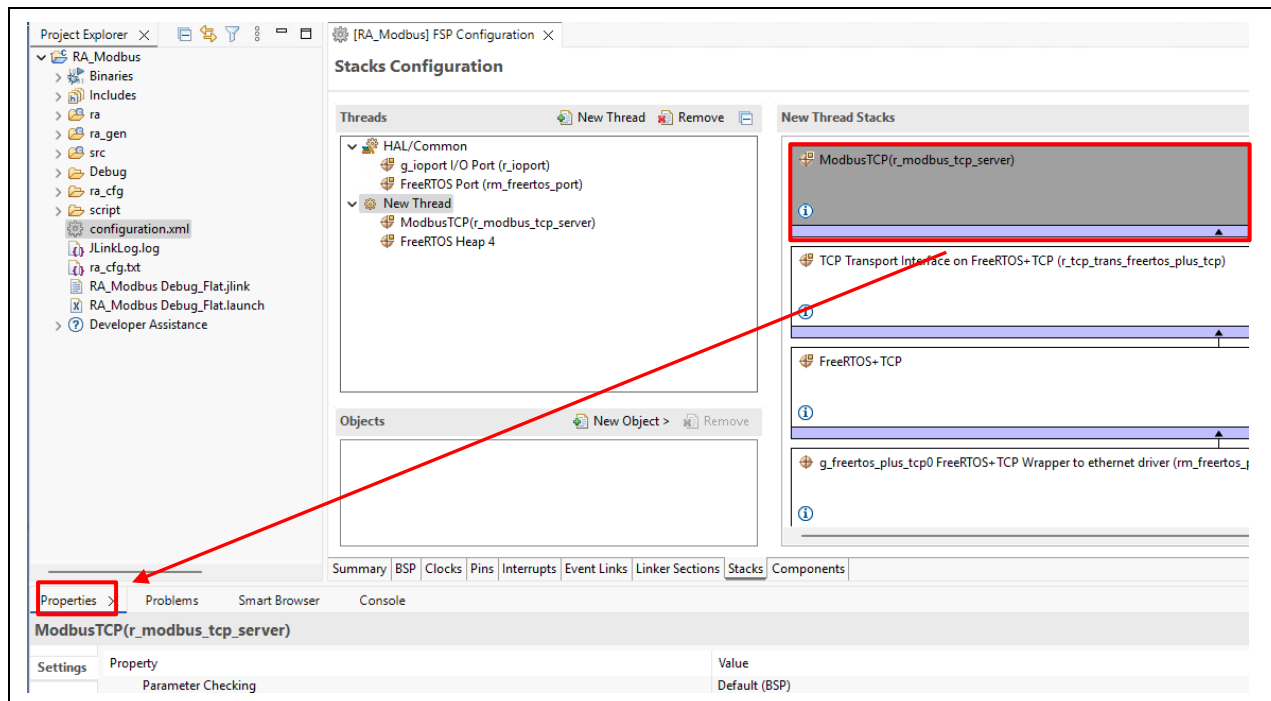
Note: When the user opens the "configuration.xml" of the sample project for MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2, the following screen appears; click "Cancel."



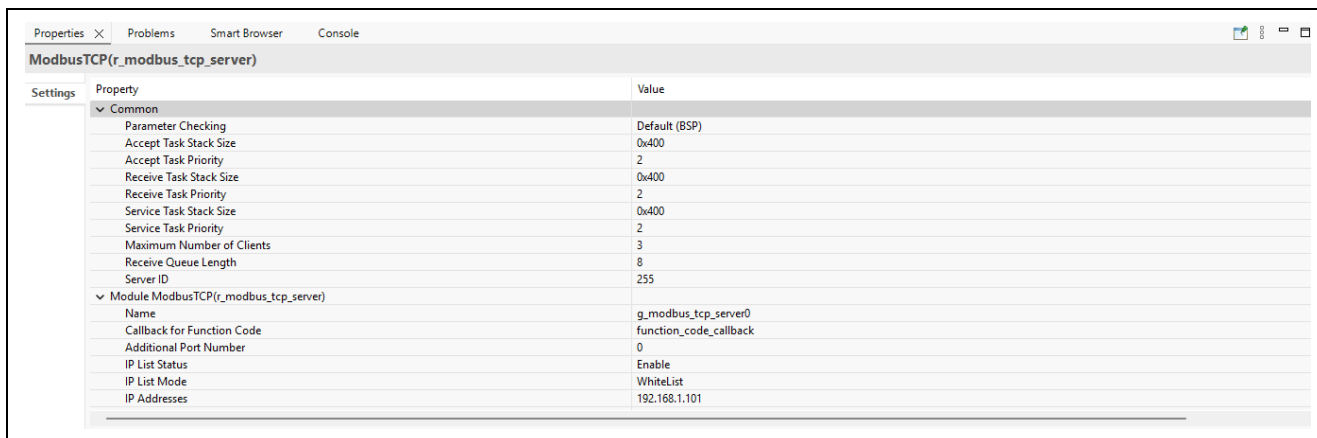
The “FSP Configuration” tab is displayed.



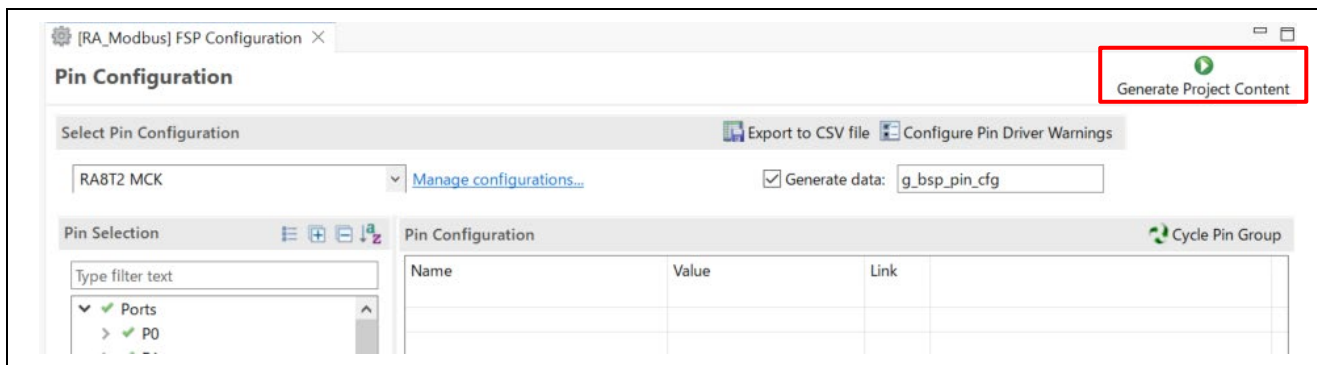
2. After selecting the stack to be changed, select "Properties."



3. Change the FSP configuration for the Modbus protocol stack, refer to "9.1 Appendix A: Modbus Protocol Stack Configuration".



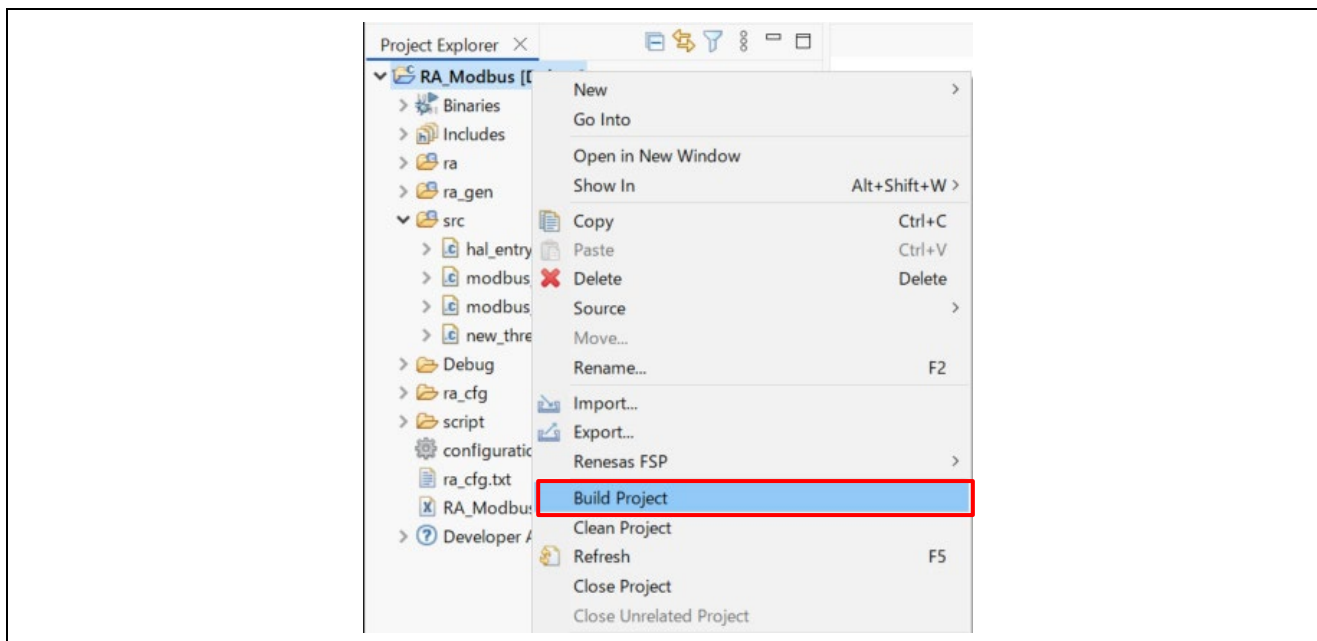
4. After changing the FSP configuration, generate the code with "Generate Project Content".



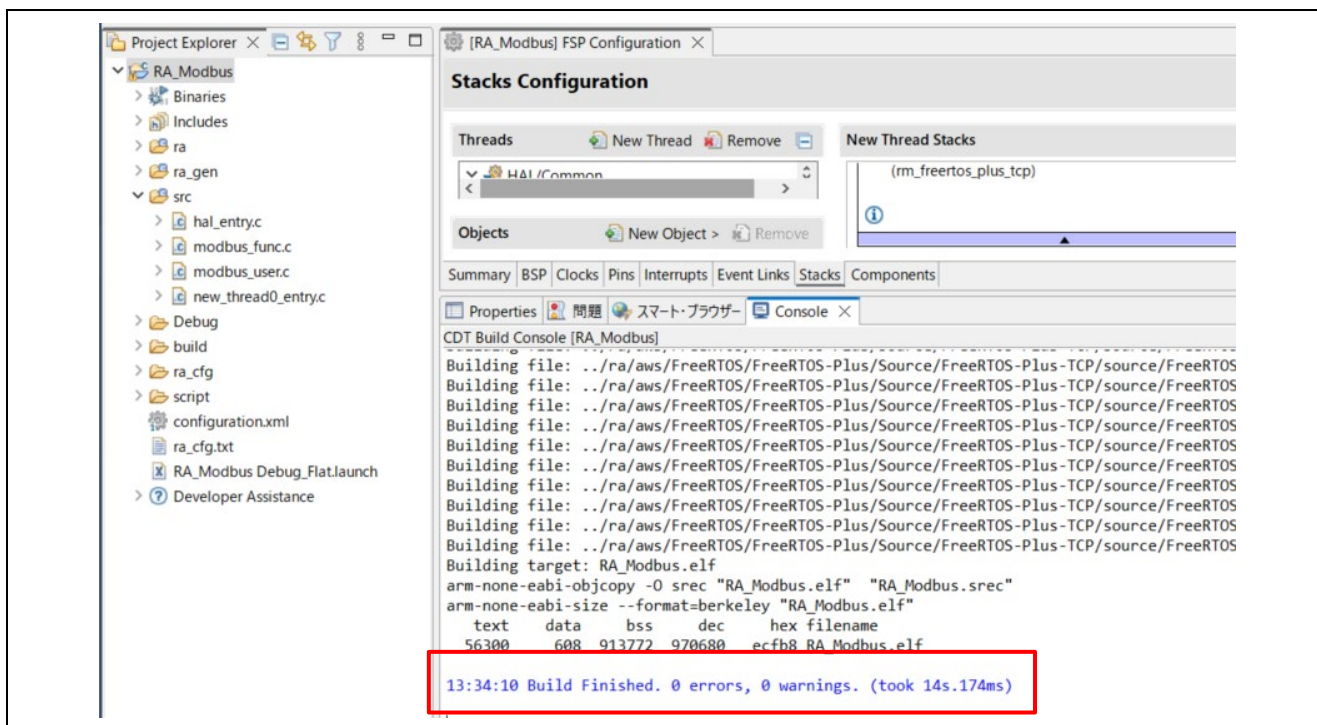
7. Execution of Modbus Sample Project

Before proceeding, read "[5. Evaluation Board Connection Setup](#)" to complete the hardware connections. Then, read "[6. Setting Up the Modbus Sample Project](#)" to prepare the Modbus sample project.

- To build the project, right-click on the project in the "Project Explorer", then select "Build Project".



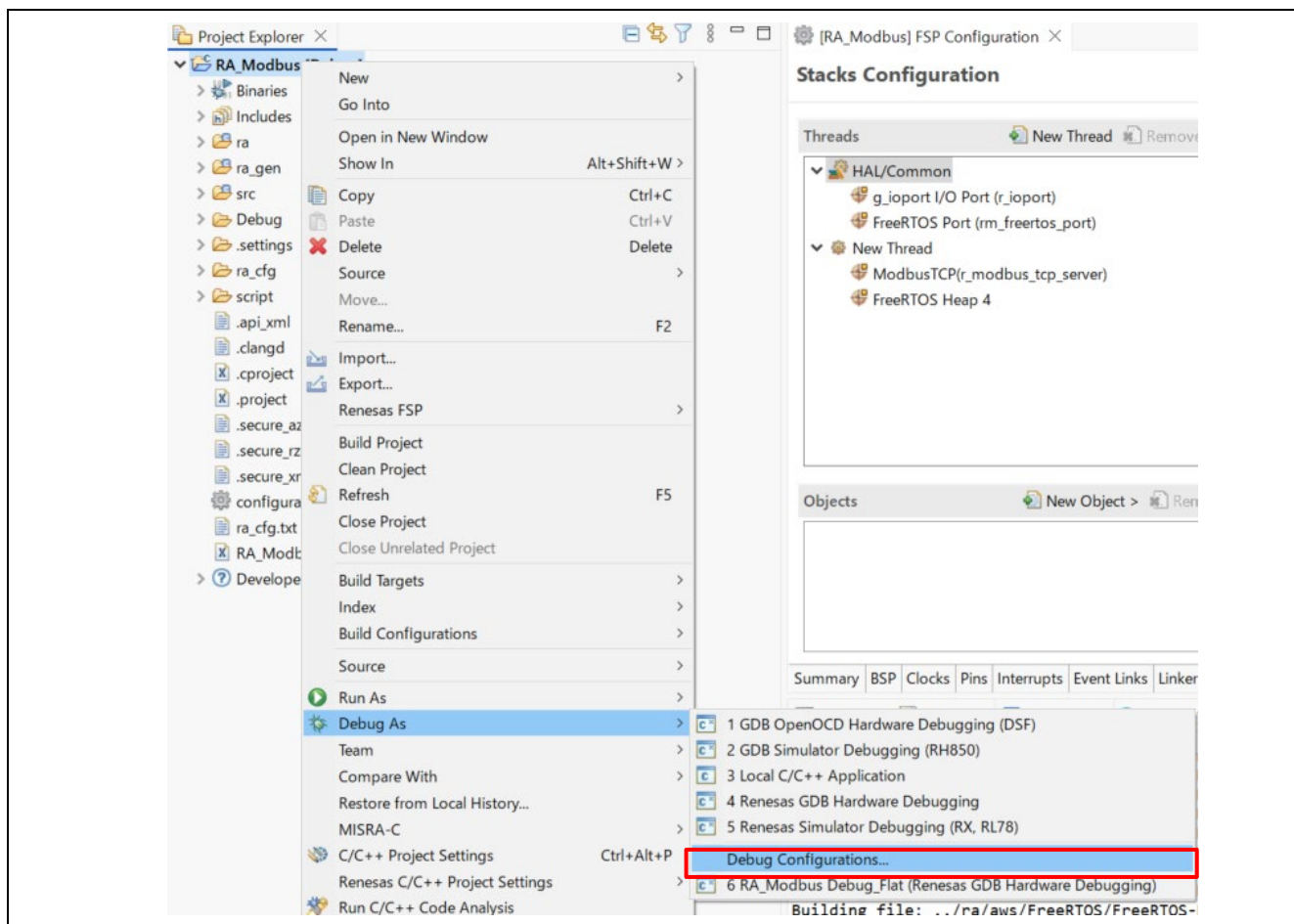
At this time, make sure there are no build errors.



- Download the application and run the debugger.

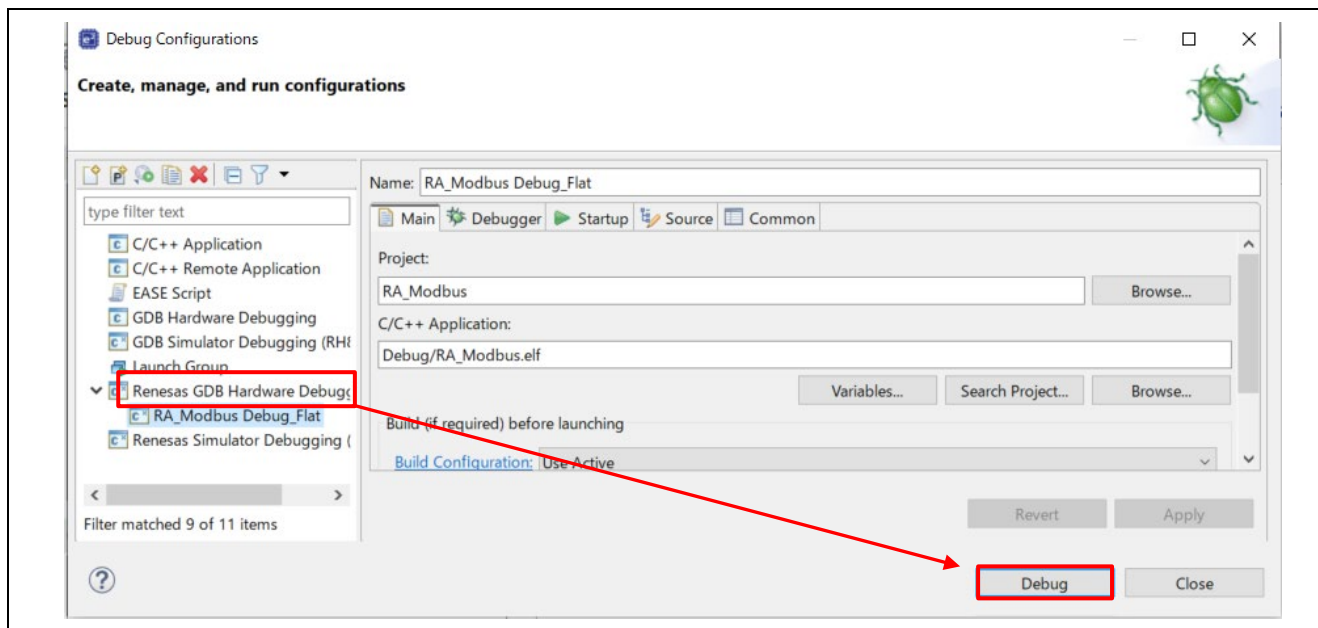
To start debugging, follow the steps below:

In “Project Explorer” view, right-click on the project to be debugged and select “Debug As” → “Debug Configurations”.

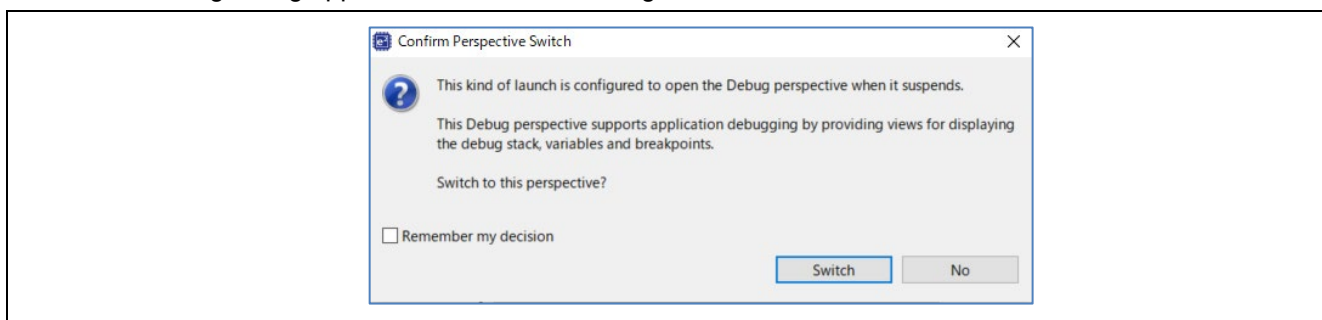


3. Program download.

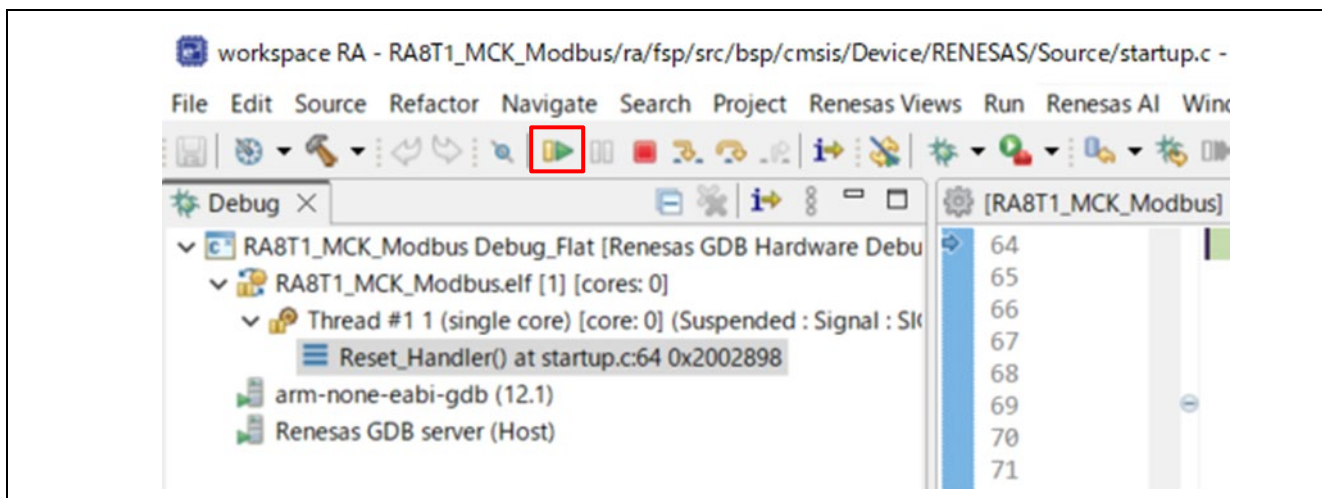
Select “Renesas GDB Hardware Debugging” → “RA_Modbus Debug_Flat”, then click “Debug”.



The following dialog appears. Switch to the debug screen.



4. The program starts.
Click the **Resume** button.
When debugging starts, the program is suspended at main.c.
Click the **Resume** button again to run the program.



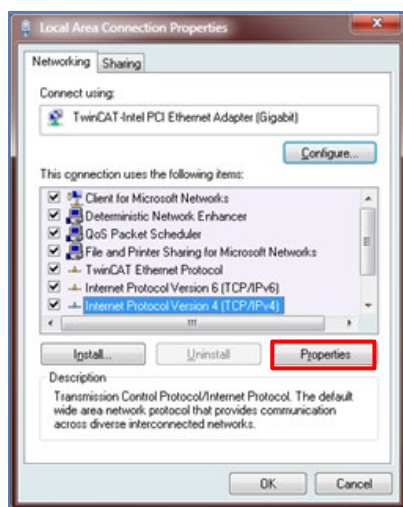
8. Modbus Communication Using Modbus Demo Application

This section describes the procedure for verifying the operation of the Modbus sample application using the Modbus demo application. For information on configuring the Modbus protocol stack, see "[9.1. Appendix A: Modbus Protocol Stack Configuration](#)".

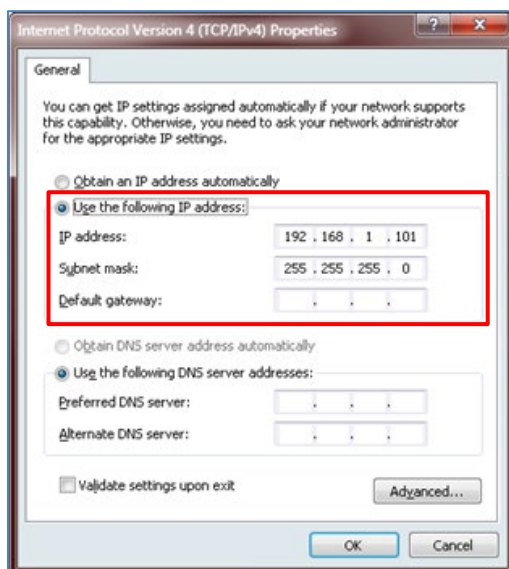
8.1 IP Address Setting

To run the Modbus sample application, it is necessary to set the IP address of the PC running the evaluation tool, which is the client, to the same domain as the evaluation board.

1. Open the network connections list.
Control Panel → **Network and Sharing Center** → **Change adapter settings**.
Double-click (or right-click) on the Local Area Connection, then select **Properties**.
Select TCP/IPv4 and click the **Properties** button.



2. Set the IP address and subnet mask
IP Address : 192.168.1.101, subnet mask : 255.255.255.0.



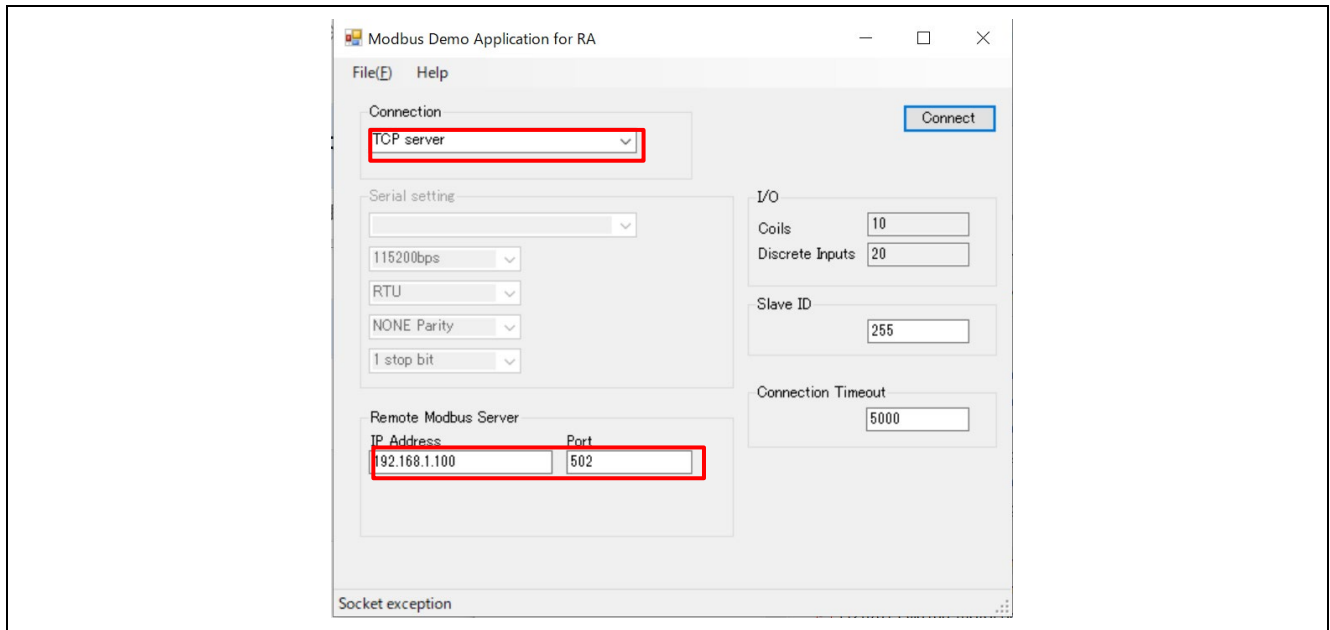
8.2 Setting up the Modbus Demo Application

Launch "ModbusDemoApplication.exe" included in this package and configure the following settings.

Connection: **TCP server**

IP Address: Modbus TCP server IP address (example: "192.168.1.100")

Port: Port number (example: "502")



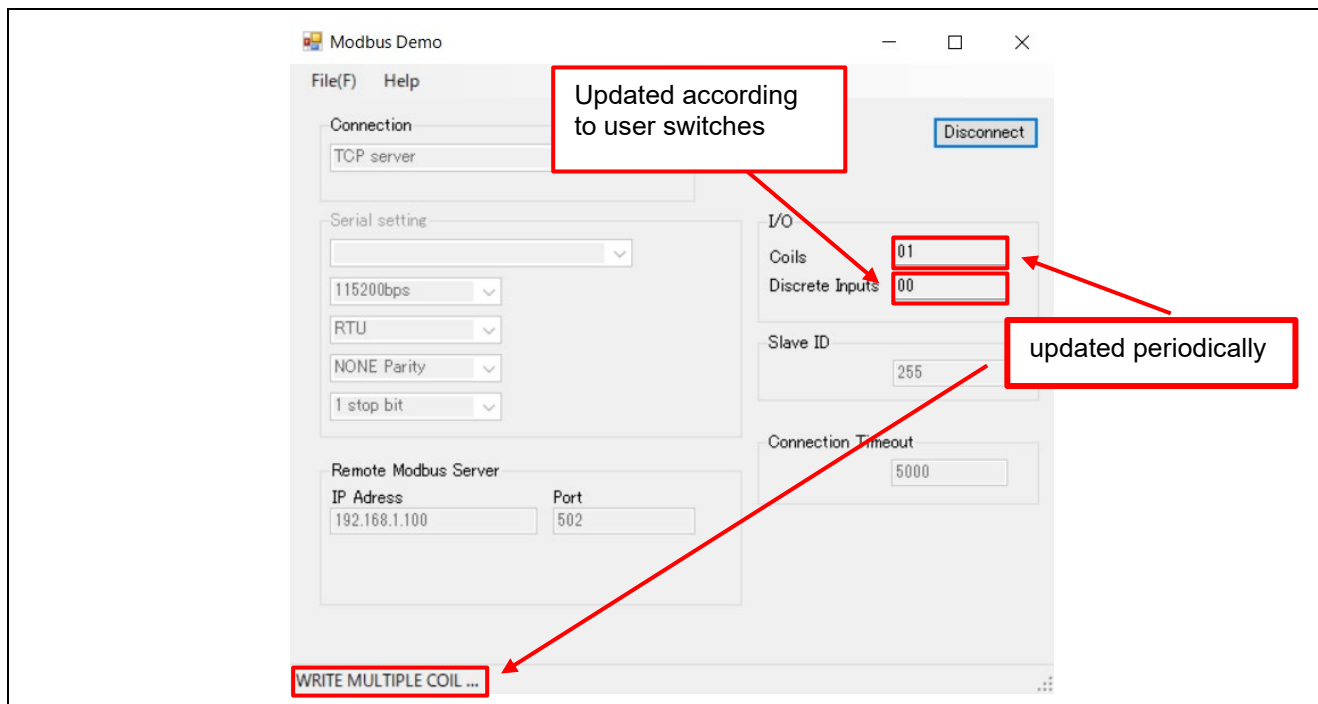
8.3 Modbus Demo Application Specification

The LED blinking can be controlled dynamically by communicating with a PC through the Modbus TCP protocol.

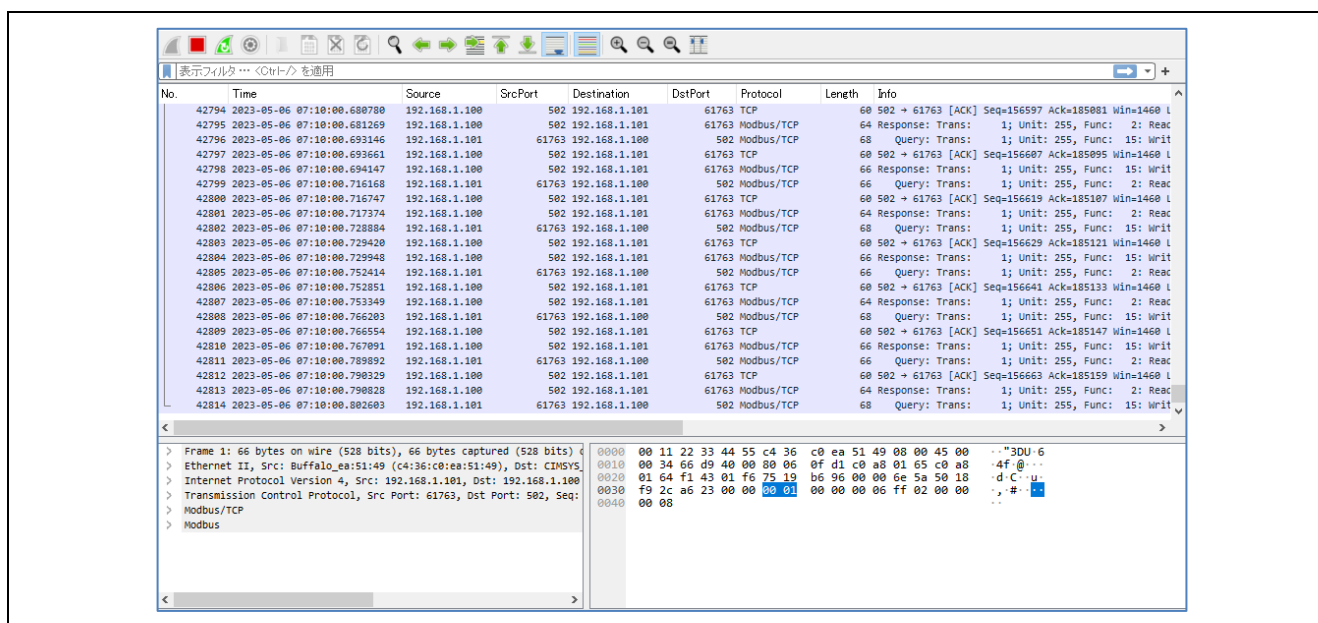
For this control, Read Coil and Write Coil function codes are used.

After clicking the "Connect" button, LEDs 1 to 3 will blink periodically.

"Discrete Inputs" are updated depending on whether the user switches listed in **"9.4.2 User-defined Functions"** is ON or OFF.



Check the Modbus communication status by using a packet analysis tool such as Wireshark.



9 Appendix

9.1 Appendix A: Modbus Protocol Stack Configuration

The Modbus protocol stack configuration is described below.

Configuration	Options	Default	Description
Common			
Parameter Checking	• Default (BSP) • Enable • Disable	Default (BSP)	If the selected code for parameter checking is included in the build.
Accept Task Stack Size	Legal values are 0x400 or higher	0x400	Accept task stack size. Legal values are 0x400 or higher.
Accept Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Accept task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Receive Task Stack Size	Legal values are 0x400 or higher	0x400	Receive task stack size. Legal values are 0x400 or higher.
Receive Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Receive task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Service Task Stack Size	Legal values are 0x400 or higher.	0x400	Service task stack size. Legal values are 0x400 or higher.
Service Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Service task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Maximum Number of Clients	Legal values are from 1 to 3	3	Maximum number of connection sockets for clients. Valid values are 1 to 3.
Receive Queue Length	Legal values are 8 or higher	8	The length of the queue that passes data between tasks. Legal values are 8 or higher.
Server ID	Legal values are from 1 to 255	255	Modbus server ID. Legal values are from 1 to 255.

Configuration	Options	Default	Description
Module Modbus TCP(r_modbus_tcp_server)			
Name	Name Must Be a Valid C Symbol	g_modbus_tcp_server0	Module name
Callback for Function Code	Name Must Be a Valid C Symbol	function_code_callback	Enter the user callback function name for "Function Code"
Additional Port Number	Legal values range from 0 to 65535. Also, do not specify the Modbus TCP Server default port number "502" with this property.	0	Modbus TCP Server listens on both the default port "502" and the port number specified in this property. Enter 0 if user do not want to use a port other than the default. Also, do not specify the Modbus TCP Server default port number "502" with this property.
IP List Status	• Disable • Enable	Enable	IP list enabled/disabled. If user want to use the IP list, select Enable. *1
IP List Mode	• BlackList • WhiteList	WhiteList	The selected option takes effect if the "IP List Status" is "Enable". *1
IP Addresses	Enter Valid IP addresses between 0.0.0.0 ~ 255.255.255.255 and each IP addresses should be separated with commas.	192.168.1.101	Enter one or more IP addresses to register in the IP List. *1

*1: For details, see " [9.2. Appendix B: IP List Related Parameters](#)" section.

9.2 Appendix B: IP List Related Parameters

IP List Status, IP List Mode, and IP Addresses are IP list-related parameters.

The settings for the "When not using an IP list", "When using an IP list as a WhiteList", and "When using an IP list as a BlackList" configurations are described below.

- When not using an IP list
 - IP List Status: Disable
 - IP List Mode: Any value (this configuration is invalid because IP List Status is "Disable")
 - IP Addresses: Any value (this configuration is invalid because IP List Status is "Disable")
- When using an IP list as a WhiteList
 - IP List Status: Enable
 - IP List Mode: WhiteList
 - IP Addresses: Any IP address to register as a WhiteList
When following the procedure explained in section ["8. Modbus Communication Using Modbus Demo Application"](#) to confirm the operation, set the IP address to "192.168.1.101".
- When using an IP list as a BlackList
 - IP List Status: Enable
 - IP List Mode: BlackList
 - IP Addresses: Any IP address to register as a BlackList
When following the procedure explained in section ["8. Modbus Communication Using Modbus Demo Application"](#) to confirm the operation, set an IP address other than "192.168.1.101".

If multiple IP addresses must be registered in the IP Addresses parameter, separate the IP addresses with a comma (",") as shown below.

Example: 192.168.1.101,192.168.1.102,192.168.1.103

IP addresses registered in the IP Addresses parameter are output as an array

g_modbus_tcp_server_ip_address_list within the project's ra_gen\common_data.c as shown below.

Example:

```
char *g_modbus_tcp_server_ip_address_list[] =  
{ "192.168.1.101", "192.168.1.102", "192.168.1.103" };
```

When using an IP list, call the API function `R_MODBUS_TCP_SERVER_RegisterIpList` (see ["9.6. Appendix F. Application Programming Interface"](#)) to register each IP address of the `g_modbus_tcp_server_ip_address_list` to the IP list.

When confirming the WhiteList functionality, configure the settings as shown below and confirm that **Modbus communication is possible** with the client (Modbus demo application) having the IP address "192.168.1.101".

IP List Status: Enable
IP List Mode: WhiteList
IP Addresses: "192.168.1.101"

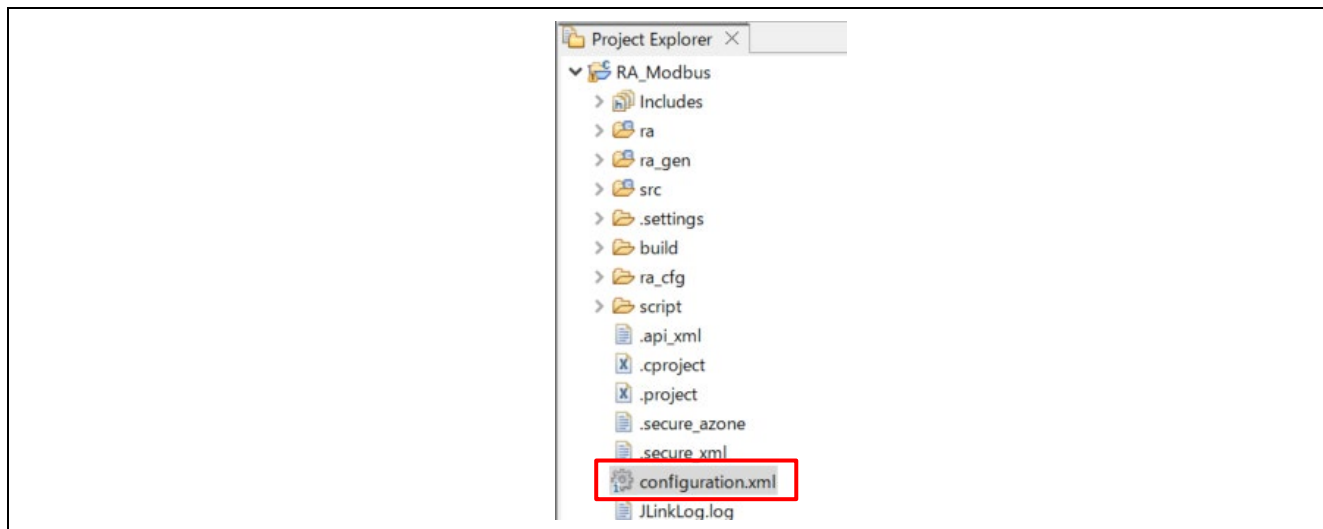
When confirming the BlackList functionality, configure the settings as shown below and confirm that **Modbus communication is not possible** with the client (Modbus demo application) having the IP address "192.168.1.101".

IP List Status: Enable
IP List Mode: BlackList
IP Addresses: "192.168.1.101"

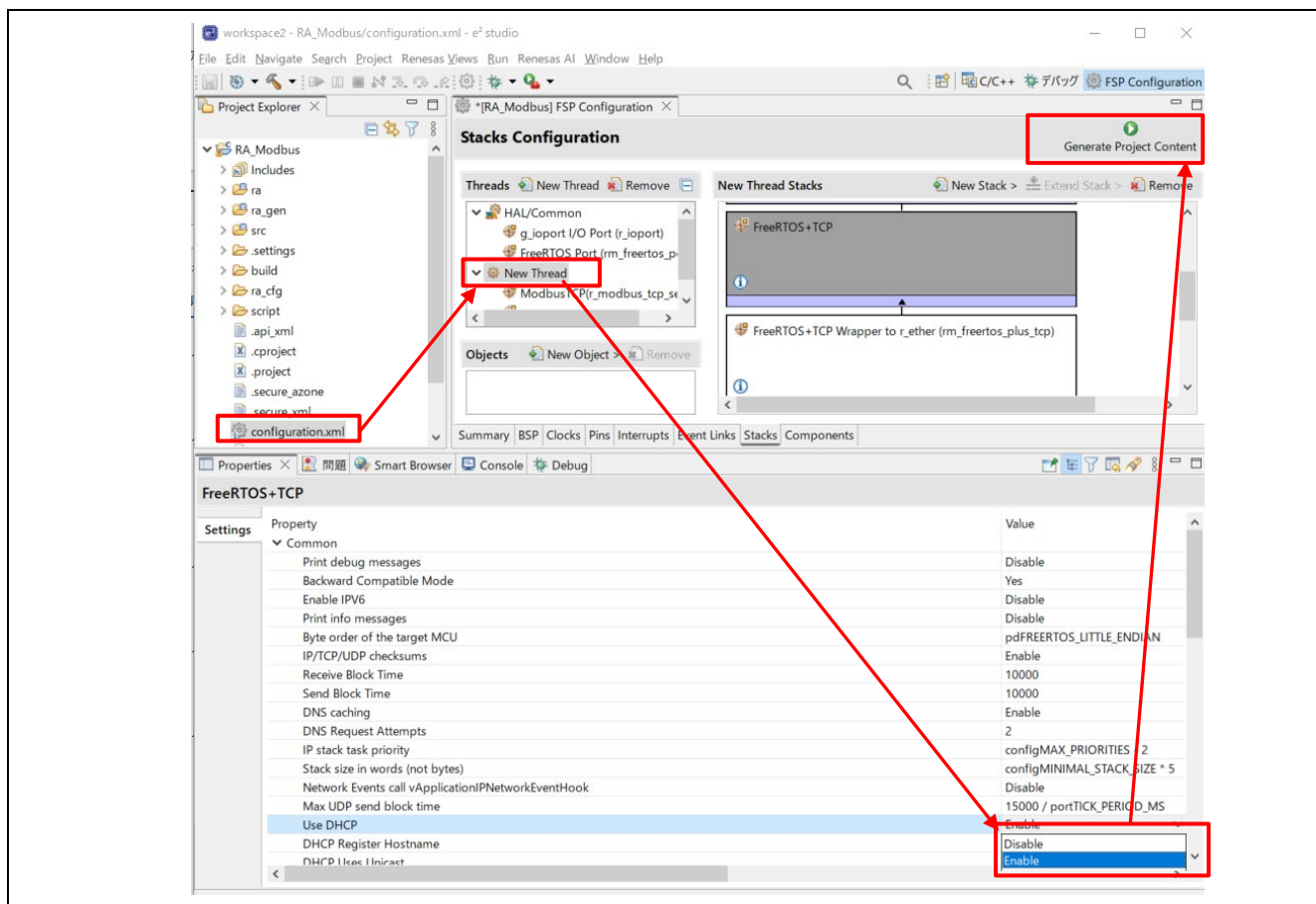
9.3 Appendix C: DHCP Mode

When the operation must be confirmed using DHCP mode, execute the procedure below:

1. Open configuration.xml from user's Modbus project.



2. Click the "Stacks" tab to open the stack configuration screen and select "FreeRTOS + TCP" in the thread window on the left.
3. Open the properties, change "Use DHCP" to "Enable", and click the "Generate Project Content" button.



9.4 Appendix D: User-defined Function

This section describes the Modbus sample application. Users can register their own implementation of the Modbus function code with the Modbus protocol stack.

9.4.1 Register Function Code

Definition file: src/modbus_func.c

Define the function to be registered in the callback function of the Modbus protocol stack.

9.4.2 User-defined Functions

User-defined functions are defined in src/modbus_user.c

User-defined read/write functions are used to process each function.

Some functions access the following components of the evaluation board.

Evaluation Boards	User LED			User-Switch	
	①	②	③	①	②
EK-RA6M3 EK-RA6M4 EK-RA6M5 EK-RA8D1 EK-RA8M1 EK-RA8D2 EK-RA8P1 EK-RA8M2 EK-RA8T2	User LED1	User LED2	User LED3	User-Switch SW1	User-Switch SW2
MCK-RA8T1 MCK-RA8T2	LED1	LED2	LED3	-	-

Read/Write functions and tables are provided that correspond to each address of the coil/discrete input/holding register/input register.

The evaluation board parts and global variables accessed by the functions in each table are as follows.

【Read Coils】	
address	Access
0001	Above table User - LED①, g_coils_area
0002	Above table User - LED②, g_coils_area
0003	Above table User - LED③, g_coils_area
0004	g_coils_area
0005	g_coils_area
0006	g_coils_area
0007	g_coils_area
0008	g_coils_area

【Write_Single_Coils】	
address	access
0001	Above table User - LED①, g_coils_area *1
0002	Above table User - LED②, g_coils_area *1
0003	Above table User - LED③, g_coils_area *1
0004	g_coils_area
0005	g_coils_area
0006	g_coils_area
0007	g_coils_area
0008	g_coils_area

*1 : Each LED on the evaluation board will turn on/off depending on the value written by this function.

【Read_Discrete_Inputs】	
address	access
1001	g_discrete_input_area
1002	g_discrete_input_area
1003	g_discrete_input_area
1004	g_discrete_input_area
1005	g_discrete_input_area
1006	Above table User-Switch①, g_discrete_input_area *2
1007	Above table User-Switch②, g_discrete_input_area *2
1008	g_discrete_input_area
1009	g_discrete_input_area
10010	ILLEGAL DATA ADDRESS
10011	g_discrete_input_area
10012	g_discrete_input_area

*2 : The value read by this function is displayed in “Discrete Inputs” of the Modbus demo application.

【Read_Discrete_Inputs】	
address	access
3001	g_input_reg_area
3002	g_input_reg_area
3003	g_input_reg_area
3004	ILLEGAL DATA ADDRESS
3005	ILLEGAL DATA ADDRESS
3006	ILLEGAL DATA ADDRESS
3007	ILLEGAL DATA ADDRESS
3008	g_input_reg_area

【READ_HOLDING_REGISTERS】	
address	access
4001	g_holding_reg_area
4002	g_holding_reg_area
4003	g_holding_reg_area
4004	ILLEGAL DATA ADDRESS
4005	ILLEGAL DATA ADDRESS
4006	ILLEGAL DATA ADDRESS
4007	g_holding_reg_area

【WRITE_SINGLE_REGISTER】	
address	access
4001	g_holding_reg_area
4002	g_holding_reg_area
4003	g_holding_reg_area
4004	ILLEGAL DATA ADDRESS
4005	ILLEGAL DATA ADDRESS
4006	ILLEGAL DATA ADDRESS
4007	g_holding_reg_area

9.5 Appendix E: Multiple Client Communication

The Modbus protocol stack can connect up to three clients using Modbus communication.

When the project is created using the steps in "[6. Setting Up the Modbus Sample Project](#)", continuous communication with the client is possible under the following conditions.

- Number of clients: 1 - 3
- Communication interval: 1000 ms
- Communication Timeout Time : 3000 ms
- No devices unrelated to Modbus communication are connected to the same network.

To use a communication interval or timeout period shorter than the values described above, or to connect additional devices to the same network, modify the following settings:

- Go to "Total number of available network buffers" property of "Stacks" → "FreeRTOS+TCP" → "Common" and increase its value.
Note that the total number of available network buffers uses 56 bytes of RAM, and the amount of RAM used will increase proportionally to this value.
(If the total number of available network buffers is 30, the amount of RAM used will be $56 \times 30 = 1680$ bytes)
- Below is the settings for EK-RA6M3 / EK-RA6M4 / EK-RA6M5 / EK-RA8D1 / EK-RA8M1 / MCK-RA8T1.
Go to "Stacks" → "g_ether0 Ethernet" → "Module g_ether0 Ethernet (r_ether)" → "Buffers" → "Number of RX buffer" and change its value to be greater than or equal to 1 (default value). The recommended value is [Number of clients].
(The RAM size used by the number of RX buffers is 1536 bytes.)
(If the Number of RX buffers is changed to 3, the total is $1536 \times 3 = 4608$ bytes)

Note:

If you intend to use the Modbus sample project in your system, thoroughly evaluate its suitability for your application.

When performing Modbus communication with multiple connected clients, modify the IP list-related parameters described in "[9.2. Appendix B: IP List Related Parameters](#)".

If the IP address of each client is other than "192.168.1.101", set the IP List Status, IP List Mode, and IP Addresses parameters as follows:

- When using an IP list as a WhiteList (Recommendation)
IP List Status: Enable
IP List Mode: WhiteList
IP Addresses: [Each client's IP address] (Example: 192.168.1.101, 192.168.1.102, 192.168.1.103)
- When not using an IP list
IP List Status: Disable
IP List Mode: Any value (this configuration is invalid because IP List Status is "Disable")
IP Addresses: Any value (this configuration is invalid because IP List Status is "Disable")
- When using an IP list as a BlackList
IP List Status: Enable
IP List Mode: BlackList
IP Addresses: IP addresses other than [Each client's IP address]

9.6 Appendix F. Application Programming Interface

The API specification for the Modbus protocol stack is described below.

· R_MODBUS_TCP_SERVER_Open

Description	MODBUS TCP server stack is initialized with this function.	
Format	<pre>modbus_err_t R_MODBUS_TCP_SERVER_Open (modbus_tcp_server_handle_t * const p_handle, modbus_tcp_server_cfg_t const * const p_cfg)</pre>	
Parameter	modbus_tcp_server_handle_t * const p_handle	Control block. Allocate an instance specific control block to pass into the Modbus TCP API calls.
	modbus_tcp_server_cfg_t const * const p_cfg	Modbus TCP Configuration parameters.
Return value	Error code	
Error code	MODBUS_ERR_OK	MODBUS TCP Server stack initialized successfully.
	MODBUS_ERR_ALREADY_OPEN	MODBUS TCP Server stack has already been initialized.
Example	<pre>R_MODBUS_TCP_SERVER_Open(&g_modbus_tcp_server0_ctrl, &g_modbus_tcp_server0_cfg);</pre>	

· R_MODBUS_TCP_SERVER_Close

Description	MODBUS TCP server stack is terminated with this function.	
Format	<pre>modbus_err_t R_MODBUS_TCP_SERVER_Close (modbus_tcp_server_handle_t * const p_handle)</pre>	
Parameter	modbus_tcp_server_handle_t * const p_handle	Control block. Allocate an instance specific control block to pass into the Modbus TCP API calls.
Return value	Error code	
Error code	MODBUS_ERR_OK	MODBUS TCP Server stack terminated successfully.
	MODBUS_ERR_STACK_TERM	MODBUS TCP Server stack terminated failure.
	MODBUS_ERR_NOT_OPEN	MODBUS TCP Server stack is not initialized.
Example	<pre>R_MODBUS_TCP_SERVER_Close(&g_modbus_tcp_server0_ctrl);</pre>	

· R_MODBUS_TCP_SERVER_Start

Description	MODBUS TCP server stack is started with this function.	
Format	<pre>modbus_err_t R_MODBUS_TCP_SERVER_Start (modbus_tcp_server_handle_t * const p_handle)</pre>	
Parameter	modbus_tcp_server_handle_t * const p_handle	Control block. Allocate an instance specific control block to pass into the Modbus TCP API calls.
Return value	Error code	
Error code	MODBUS_ERR_OK	MODBUS TCP Server stack started successfully.
	MODBUS_ERR_NOT_OPEN	MODBUS TCP Server stack is not initialized.
	MODBUS_ERR_STACK_INIT	Task creation or mailbox creation failed.
Example	R_MODBUS_TCP_SERVER_Start(&g_modbus_tcp_server0_ctrl);	

· R_MODBUS_TCP_SERVER_RegisterIpList

Description	Register the IP address in the IP list.	
Format	<pre>modbus_err_t R_MODBUS_TCP_SERVER_RegisterIpList (modbus_tcp_server_handle_t * const p_handle , char * const p_ip_address)</pre>	
Parameter	modbus_tcp_server_handle_t * const p_handle	Control block. Allocate an instance specific control block to pass into the Modbus TCP API calls.
	char * const p_ip_address	The IP address to be registered in the IP list.
Return value	Error code	
Error code	MODBUS_ERR_OK	The IP address has been successfully registered.
	MODBUS_ERR_NOT_OPEN	MODBUS TCP Server stack is not initialized.
	MODBUS_ERR_TCP_IP_TABLE_IP_ALREADY_PRESENT	The IP address is already registered.
	MODBUS_ERR_TCP_IP_TABLE_MAX_CLIENT	The maximum number of registerable IP addresses has been exceeded.
Example	R_MODBUS_TCP_SERVER_RegisterIpList (&g_modbus_tcp_server0_ctrl, g_modbus_tcp_server_ip_address_list[1]);	

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jun 27, 2025	-	First version
1.10	Sep 25, 2025	Page 4 Page 6 Page 9 Page 18 - 43	Added MCK-RA8T2 Updated Table 1.2 Updated 3.1 Modbus Sample Project Added Figure 5.3 6.2 Creation of a new Modbus Project is divided into common procedures for all evaluation boards, procedures for each evaluation board, and procedures for MCK-RA8T2 are added
		Page 7	Updated Operating Environment Requirements Updated Table 4.1
		Page 56	Added description of multi-client communication Added 9.5 Appendix E: Multiple Client Communication
1.20	Oct 31, 2025	Page 4 Page 6 Page 9 Page 16 Page 45 - 67 Page 80	Added EK-RA8D2, EK-RA8P1 and EK-RA8M2 Updated Table 1.2 Updated 3.1 Modbus Sample Project Updated Figure 5.3 Changed the title of 6.1.2 to MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 Importing Procedures Added 6.2.4 EK-RA8D2 Importing Procedures, 6.2.5 EK-RA8P1 Importing Procedures and 6.2.6 EK-RA8M2 Importing Procedures Changed 9.5 Appendix E: Multiple Client Communication
		Page 7	Updated Operating Environment Requirements Updated Table 4.1
		Page 38	Additional Procedures for Upgrading FSP Version Added step 3 to 6.2.3
1.30	Mar 31, 2026	Page 5 Page 7 Page 11 Page 17, 19 Page 18 Page 70 - 76	Added EK-RA8T2 Added Table 1.3 Updated 3.1 Modbus Sample Project Added Figure 5.4 Added steps to execute the Modbus sample project Changed the title of 6.1.2 to MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 / EK-RA8T2 Importing Procedures Added 6.2.7 EK-RA8T2 Importing Procedures
		Page 52, 61, 67	Removed ESW Clock setting
		Page 8	Updated from FSP6.2.0 to FSP6.4.0 Updated Table 4.1
		Page 82	Changed the description of Setting up the Modbus Demo Application
		Page 83	Changed chapter number from 8.2.1 to 8.3 Added information about Discrete Inputs Moving the description of Wireshark from Chapter 8.2
		Page 88	Updated the User LED and User Switch tables
		Page 91	Changed Number of clients: 1 to Number of clients: 1-3 Added a description about setting IP list-related parameters

Rev.	Date	Description	
		Page	Summary
1.40	Aug 31, 2026	Page 1	Changed the document title
		Page 9	Updated Table 4.1
		Page 78, 79	Added 6.3 Changing of configuration
		Page 88	Added an explanation regarding the IP Addresses parameter
		Page 94, 95	Added 9.6 Appendix F. Application Programming Interface

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.

- Arm® and Cortex® are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.
- Ethernet is a registered trademark of Fuji Xerox Co., Ltd.
- Modbus is a registered trademark of Schneider Electric, licensed to the Modbus Organization, Inc.
- Additionally all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners. a trademark or a registered trademark which belongs to the respective owners.

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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. 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.
6. 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.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. 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.5.0-1 October 2020)

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/.