

---

# RA Family, RX Family, RL78 Family

## ZMOD4XXX Sample Software Manual

---

### Introduction

This Application Note describes sample software for ZMOD4XXX gas sensor that operates on RA Family, RX Family and RL78 Family.

### Target Devices

RA2E1 Group

RA0E1 Group

RX140 Group

RL78/G23 Group

### Target Sensor Board

TVOC and Indoor Air Quality Sensor Pmod Board (US082-ZMOD4410EVZ)

Refrigeration Air Quality Sensor Pmod Board (US082-ZMOD4450EVZ)

Outdoor Air Quality Sensor Pmod Board (US082-ZMOD4510EVZ)

The setting example described in this application note is an example when using the sensor board mentioned above.

Therefore, you will need to review the following settings according to the target circuit.

- Interrupt Signal Circuit: Refer to “8.5 Notes for Interrupt Signal Circuits”.
- RESET Signal Circuit: Refer to “8.6 Notes for RESET Signal Circuits”.

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

## Contents

1. Overview .....	5
1.1 Terms/Abbreviations .....	5
2. Environment for Confirming Operation .....	6
2.1 Environment for Confirming Operation on RA Family MCU .....	6
2.2 Environment for Confirming Operation on RX Family MCU .....	9
2.3 Environment for Confirming Operation on RL78/G23 Group MCU .....	11
3. ZMOD4410 Sensor Specifications .....	13
3.1 Sensor Specifications Overview .....	13
3.2 Sensor Function and Methods.....	13
3.2.1 Conversion of Output Data – Firmware / API / Algorithms .....	13
3.2.2 Typical Hardware Requirements .....	13
4. ZMOD4450 Sensor Specifications .....	14
4.1 Sensor Specifications Overview .....	14
4.2 Sensor Function and Methods.....	14
4.2.1 Conversion of Output Data – Firmware / API / Algorithms .....	14
4.2.2 Typical Hardware Requirements .....	14
5. ZMOD4510 Sensor Specifications .....	15
5.1 Sensor Specifications Overview .....	15
5.2 Sensor Function and Methods.....	15
5.2.1 Conversion of Output Data – Firmware / API / Algorithms .....	15
5.2.2 Typical Hardware Requirements .....	15
6. Specification of Sample Software .....	16
6.1 Sample Software Structure.....	16
6.2 Specification of Sensor Control Module API Functions .....	17
6.2.1 List of Sensor Control Module API Functions .....	17
6.2.2 API Usage Guide.....	18
6.3 Flowchart of Main Processing in Non-OS Version of Sample Software .....	20
6.3.1 ZMOD4410, ZMOD4450 .....	20
6.3.2 ZMOD4510.....	23
6.4 Flowchart of OS Version of Sample Software .....	26
6.4.1 ZMOD4410, ZMOD4450 .....	26
6.4.2 ZMOD4510.....	29
6.5 Control Sequence when Interrupt Control .....	32
7. Configuration Settings .....	33
7.1 ZMOD4XXX Sensor Control Module Settings .....	33

7.1.1	RA Family.....	33
7.1.2	RX Family.....	34
7.1.3	RL78 Family .....	35
7.2	I2C Communication Middleware (COMMS_I2C) Settings .....	36
7.2.1	RA Family.....	36
7.2.2	RX Family.....	37
7.2.3	RL78 Family .....	38
7.3	I2C Driver Settings .....	39
7.3.1	RA Family.....	39
7.3.2	RX Family.....	43
7.3.3	RL78 Family .....	46
7.4	IRQ Driver Settings .....	47
7.4.1	RA Family.....	47
7.4.2	RX Family.....	48
7.4.3	RL78 Family .....	48
8.	Guide for Changing Target Device .....	49
8.1	Importing Sample Project .....	49
8.2	RA Sample Project.....	51
8.2.1	Modifying Settings of FSP Configurator .....	51
8.2.2	Changing Sample Code .....	61
8.2.3	Changing when not using IRQ.....	61
8.2.4	Changing Toolchain Setting .....	61
8.3	RX Sample Project.....	62
8.3.1	Modifying Settings of Smart Configurator.....	62
8.3.2	Changing Sample Code .....	71
8.3.3	Changing Toolchain Setting .....	71
8.3.4	Notes for Build on GCC.....	71
8.3.5	When using IAR Integrated Development Environment "IAR Embedded Workbench" .....	71
8.4	RL78 Sample Project .....	72
8.4.1	Modifying Settings of Smart Configurator.....	72
8.4.2	Modifying Generated Code .....	82
8.4.3	Changing Sample Code .....	83
8.4.4	Changing Toolchain Setting .....	84
8.4.5	Notes for Build on LLVM .....	86
8.5	Notes for Interrupt Signal Circuits.....	87
8.6	Notes for RESET Signal Circuits.....	88
8.7	Pull-up Resistor Circuit Configuration when Daisy Chain Connections of Renesas Sensor Pmod Boards.....	88
9.	Viewing Gas Data.....	89

Revision History .....91

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products ..92

Notice .....93

**Trademarks**

FreeRTOS™ is a trademark of Amazon Web Services, Inc.

Pmod™ is a trademark of Digilent Inc.

## 1. Overview

This sample software acquires data from the ZMOD4XXX gas sensor and calculates the result values. In combination with the I2C driver of the FSP/FIT or the Code Generator, the sample software controls the ZMOD4XXX through the I2C in the MCU to measure gas environment, acquire ADC data, convert and calculate the acquired results.

### 1.1 Terms/Abbreviations

The terms and their abbreviations are listed below.

**Table 1-1 List of Terms/Abbreviations**

Terms	Abbreviation
ZMOD4XXX Sensor Control Module	Sensor Control Module When MCU is RA Family, "rm_zmod4xxx" When MCU is RX Family, "r_zmod4xxx_rx" When MCU is RL78 Family, "rm_zmod4xxx"
I2C Communication Middleware	COMMS_I2C When MCU is RA Family, "rm_comms_i2c" When MCU is RX Family, "r_comms_i2c_rx" When MCU is RL78 Family, "r_comms_i2c"
I2C Driver	When MCU is RA Family, "r_iic_master", "r_sci_i2c", "r_iica_master", "r_sau_i2c" When MCU is RX Family, "r_riic_rx", "r_sci_iic_rx" When MCU is RL78 Family, "r_iica_master"
IRQ Driver	When MCU is RA Family, "r_icu" When MCU is RX Family, "r_irq_rx" When MCU is RL78 Family, "Interrupt Controller"
Serial Communications Interface	When MCU is RA Family, "SCI", "SCI I/F" When MCU is RX Family, "SCI", "SCI I/F"
Serial Array Unit	When MCU is RA Family, "SAU", "SAU I/F" When MCU is RL78 Family, "SAU", "SAU I/F"
I2C Bus Interface	When MCU is RA Family, "IIC", "IIC I/F" When MCU is RX Family, "RIIC", "RIIC I/F"
I2C Bus Interface (IICA)	When MCU is RA Family, "IICA", "IICA I/F"
Serial Interface IICA	When MCU is RL78 Family, "IICA", "IICA I/F"
General Term for I2C Bus Interface, I2C Bus Interface (IICA), Serial Interface (IICA)	"I2C I/F"
General Term for Interrupt Controller	"ICU I/F" (Interrupt Controller Unit)
General purpose I/O Port	"GPIO", "GPIO I/F"
Pin No.1 (#1) of Renesas Pmod Type 6A Sensor Board	"IRQ#" (L output when an interrupt occurs)
ZMOD4XXX Sensor Pmod Board	ZMOD4410 Sensor Pmod Board, ZMOD4450 Sensor Pmod Board, ZMOD4510 Sensor Pmod Board

## 2. Environment for Confirming Operation

### 2.1 Environment for Confirming Operation on RA Family MCU

The operation of this software has been confirmed on RA family MCU in the following environment.

#### (1) Evaluation Kit for RA2E1 (EK-RA2E1)

**Table 2-1 Confirming Operating Environment for EK-RA2E1**

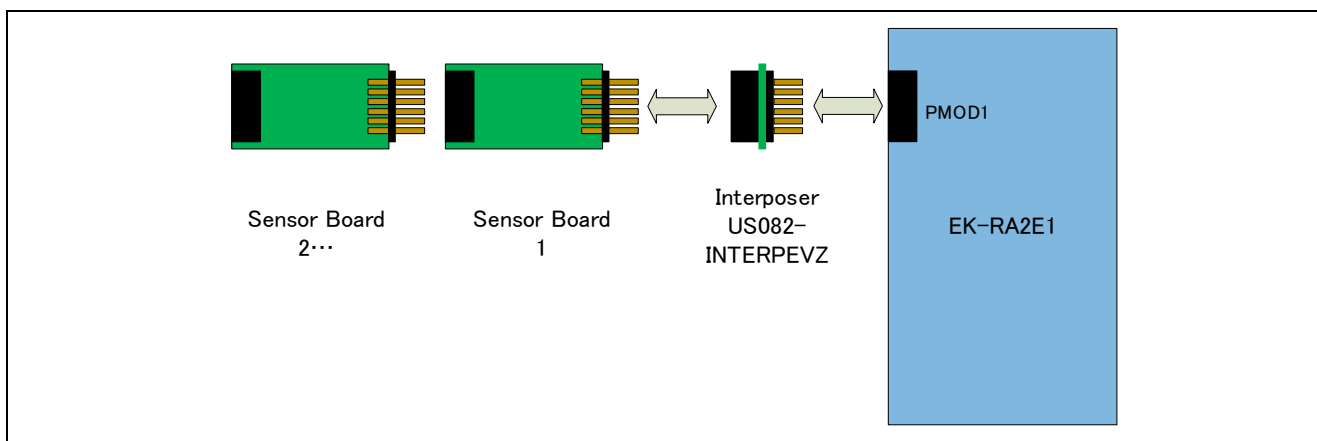
Item	Description
Demonstration board	RTK7EKA2E1S00001BE (EK-RA2E1)
Microcontroller	RA2E1 (R7FA2E1A92DFM:64pin)
Operating frequency	48MHz
Operating voltage	5V
Integrated development environment	Renesas Electronics e <sup>2</sup> studio 2024-07
C compiler	GNU ARM Embedded 13.2.1.arm-13-7
Configuration options	Add the following settings to the compiler default settings: ISO C99 (-std = c99), Optimization Level: Default settings (-O2)
FSP	v5.5.0
RTOS	FreeRTOS v10.6.1
Emulator	On board (J-LINK)
Interposer	Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ)
Sensor board	TVOC and Indoor Air Quality Sensor Pmod Board (US082-ZMOD4410EVZ) Refrigeration Air Quality Sensor Pmod Board (US082-ZMOD4450EVZ) Outdoor Air Quality Sensor Pmod Board (US082-ZMOD4510EVZ)

**Table 2-2 Memory Size Used in RA2E1 (When an Operation Mode with Maximum ROM Size Is Set)**

Sensor (Operation Mode)	Area	Size (Non-OS) [Bytes]	Size (FreeRTOS) [Bytes]
ZMOD4410 (IAQ 2nd Gen)	ROM	9,108	11,072 (Note 1)
	RAM	968	1,080
ZMOD4450 (RAQ)	ROM	4,376	6,300 (Note 1)
	RAM	508	620
ZMOD4510 (NO2 O3)	ROM	14,788	16,624 (Note 1)
	RAM	680	792

Note Memory size is calculated for sample code, ZMOD4XXX sensor control module, and COMMS\_I2C. In RTOS, memory size does not include memory size of the thread.

Note 1 This includes an increase of 1,554 bytes due to the Relax function.



**Figure 2-1 Hardware Connections for EK-RA2E1**

(2) RA0E1 Fast Prototyping Board (FPB-RA0E1)

Table 2-3 Confirming Operating Environment for FPB-RA0E1

Item	Description
Demonstration board	RTK7FPA0E1S00001BJ (FPB-RA0E1)
Microcontroller	RA0E1 (R7FA0E1073CFJ:32pin)
Operating frequency	32MHz
Operating voltage	5V
Integrated development environment	Renesas Electronics e <sup>2</sup> studio 2024-07
C compiler	GNU ARM Embedded 13.2.1.arm-13-7
Configuration options	Add the following settings to the compiler default settings: ISO C99 (-std = c99), Optimization Level: Default settings (-Oz)
FSP	v5.5.0
Emulator	On board (J-LINK)
Interposer	Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ)
Sensor board	TVOC and Indoor Air Quality Sensor Pmod Board (US082-ZMOD4410EVZ) Refrigeration Air Quality Sensor Pmod Board (US082-ZMOD4450EVZ) Outdoor Air Quality Sensor Pmod Board (US082-ZMOD4510EVZ)

Table 2-4 Memory Size Used in RA0E1 (When an Operation Mode with Maximum ROM Size Is Set)

Sensor (Operation Mode)	Area	Size (Non-OS) [Bytes]
ZMOD4410 (IAQ 2nd Gen)	ROM	8,224
	RAM	956
ZMOD4450 (RAQ)	ROM	3,824
	RAM	496
ZMOD4510 (NO2 O3)	ROM	14,180
	RAM	664

Note Memory size is calculated for sample code, ZMOD4XXX sensor control module, and COMMS\_I2C.

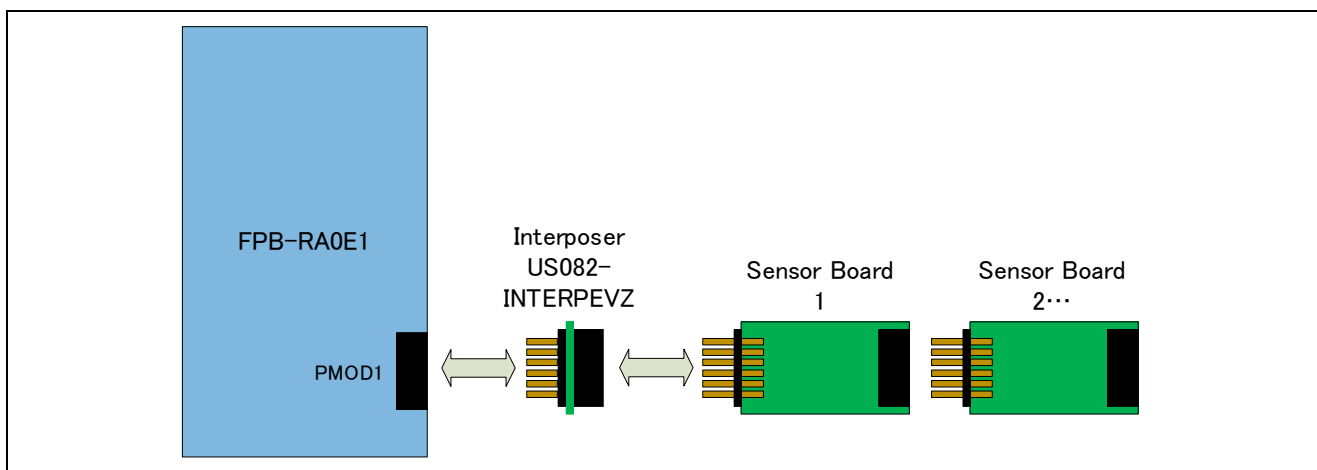


Figure 2-2 Hardware Connections for FPB-RA0E1

**(3) Use of Interposer Board**

The Interposer Board is an I/F conversion board for connecting Pmod Type 6A sensors by switching the Pmod Type 2A/Type 3A connector of the SCI I/F to the Simple IIC function.

Therefore, it cannot be used with the Pmod Type 2A/Type 3A connector of the SAU I/F. However, it may be usable by switching to the IICA I/F. Refer to the MCU hardware manual.

**Table 2-5 Operational Feasibility Depending on Pmod I/F, Serial I/F, and Presence or Absence of Interposer Board**

Pmod I/F	Destination MCU Serial I/F	Operational Feasibility
Type 2A, Type 3A	SCI I/F, IICA I/F (Note 1)	It works when using an Interposer Board. (Note 2)
	SAU I/F	It does not work regardless of whether the Interposer Board is present or not.
Type 6, Type 6A	SCI I/F, IIC I/F, SAU I/F, IICA I/F	It works without an Interposer Board. (Note 2)

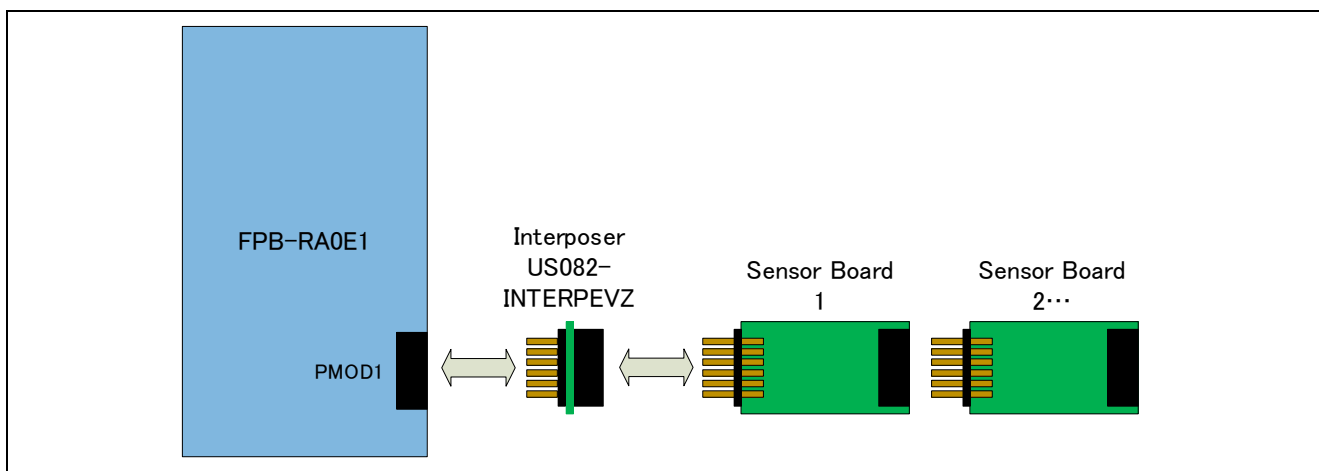
Note 1: These pins are provided for SAU I/F but can be used when it is switchable to IICA pins by multi-function pins assignment. The signal connections when switchable are shown below.

Pmod Pin	Type 2A /Type 3A	Destination SAU I/F ICU I/F GPIO I/F	Switching to Multi-Function IICA I/F	Interposer Board	Renesas Pmod Type 6A Sensor Board
#1	CS/CTS	GPIO		↔	IRQ# (Note 3)
#2	MOSI/TXD	SAU TXD	SDAA	↔	RESET#
#3	MISO/RXD	SAU RXD	SCLA	↔	IIC_SCL
#4	SCK/RTS	GPIO		↔	IIC_SDA
#7	INT	IRQ#		↔	BUSY#
#8	RESET	GPIO		↔	ENABLE
#9	CS2/GPIO	GPIO		↔	POWER_ON
#10	CS3/GPIO	GPIO		↔	GPIO

Note 2: If an IRQ signal is used, make sure that the IRQ signal on MCU is connected to Pmod #1.

Note 3: For an interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

Application example: FPB-RA0E1 Pmod1 is applicable.



**Figure 2-3 Hardware Connections for using IICA at Pmod1 Type 2A, Type 3A on FPB-RA0E1**



## 2.2 Environment for Confirming Operation on RX Family MCU

The operation of this software has been confirmed on RX family MCU in the following environment.

### (1) RX140 Fast Prototyping Board (FPB-RX140)

**Table 2-6 Confirming Operating Environment for FPB-RX140**

Item	Description
Demonstration board	RTK5FP1400S00001BE (FPB-RX140)
Microcontroller	RX140 (R5F51406BGFN: 80pin)
Operating frequency	48MHz
Operating voltage	5V
Integrated development environment	Renesas Electronics e <sup>2</sup> studio 2024-10
C compiler	Renesas Electronics CC-RX V.3.06.00 GCC for Renesas RX 8.3.0.202405
Configuration options	Add the following settings to the compiler default settings. CC-RX: C99 (-lang = c99), Optimization Level: Default settings (Level 2) GCC: ISO C99 (-std = c99), Optimization Level: Default settings (-Og)
FIT (RX Driver Package v1.45)	Board Support Packages (r_bsp) v7.51 GPIO Driver (r_gpio_rx) v5.10 ZMOD4XXX Sensor Control Module (r_zmod4xxx_rx) v1.40 IIC Communication Middleware (r_comms_i2c_rx) v1.22 RIIC Multi Master I2C Driver (r_riic_rx) v3.00 Simple IIC Driver (r_sci_iic_rx) v2.80 IRQ Driver (r_irq_rx) v4.50 CMT Driver (r_cmt_rx) v5.70
RTOS	FreeRTOS Kernal 10.4.3-rx-1.0.9、FreeRTOS Object 10.4.3-rx-1.0.9
Emulator	On board (E2OB)
Interposer	Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ)
Sensor board	TVOC and Indoor Air Quality Sensor Pmod Board (US082-ZMOD4410EVZ) Refrigeration Air Quality Sensor Pmod Board (US082-ZMOD4450EVZ) Outdoor Air Quality Sensor Pmod Board (US082-ZMOD4510EVZ)

**Table 2-7 Memory Size Used in RX140 (When an Operation Mode with Maximum ROM Size Is Set)**

Sensor (Operation Mode)	Area	Size (Non-OS) [Bytes] (CC-RX)	Size (FreeRTOS) [Bytes] (CC-RX)
ZMOD4410 (IAQ 2nd Gen)	ROM	8,348	8,506
	RAM	994	990
ZMOD4450 (RAQ)	ROM	4,759	4,894
	RAM	519	515
ZMOD4510 (NO2 O3)	ROM	13,695	13,830
	RAM	691	711

Note Memory size is calculated for sample code, ZMOD4XXX sensor control module, and COMMS\_I2C.  
In RTOS, memory size does not include memory size of the thread.

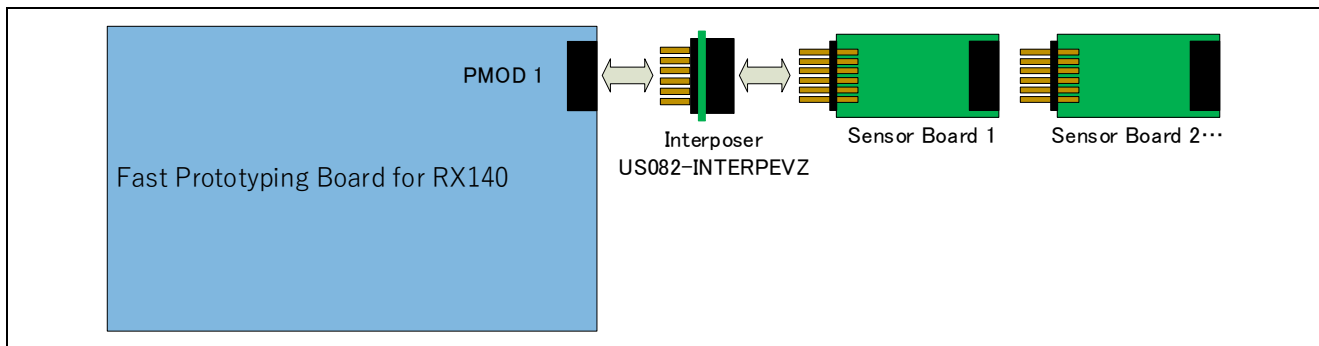


Figure 2-4 Hardware Connections for FPB-RX140 Group

**(2) Use of Interposer Board**

If you add an Interposer Board to the Pmod Type 2A/Type 3A connector to which the SCI I/F is connected, you can use the Pmod Type 6A Sensor Pmod Board.

2.3 Environment for Confirming Operation on RL78/G23 Group MCU

The operation of this software has been confirmed on RL78/G23 group MCU in the following environment.

(1) RL78/G23-128p Fast Prototyping Board (RL78/G23-128p FPB)

Table 2-8 Confirming Operating Environment for RL78/G23-128p FPB

Item	Description
Demonstration board	RTK7RLG230CSN000BJ (RL78/G23-128p FPB)
Microcontroller	RL78/G23 (R7F100GSN2DFB: 128pin)
Operating frequency	32MHz
Operating voltage	3.3V
Integrated development environment	Renesas Electronics e <sup>2</sup> studio 2024-07
C compiler	Renesas Electronics CC-RL V1.14.00 LLVM for RL78 17.0.1.202409
Configuration options	Add the following settings to the compiler default settings. CC-RL: C99 (-lang = c99), Optimization Level: Default settings (-Odefault) LLVM: GNU ISO C99 (-std = gnu99), Optimization Level: Default settings (-Og)
SIS / CG	Board Support Packages (r_bsp) v1.70 Ports v1.5.0 ZMOD4XXX Sensor Middleware (r_zmod4xxx) v1.40 IIC Communication Driver Interface Middleware (r_comms_i2c) v1.11 IIC Communication (Master mode) v1.6.0 Interrupt Controller v1.5.0 Interval Timer v1.5.0
Emulator	On board (COM Port)
Sensor board	TVOC and Indoor Air Quality Sensor Pmod Board (US082-ZMOD4410EVZ) Refrigeration Air Quality Sensor Pmod Board (US082-ZMOD4450EVZ) Outdoor Air Quality Sensor Pmod Board (US082-ZMOD4510EVZ)

Table 2-9 Memory Size Used in RL78/G23 (When an Operation Mode with Maximum ROM Size Is Set)

Sensor (Operation Mode)	Area	Size (Non-OS) [Bytes] (CC-RL)
ZMOD4410 (IAQ 2nd Gen)	ROM	12,863
	RAM	849
ZMOD4450 (RAQ)	ROM	6,116
	RAM	382
ZMOD4510 (NO2 O3)	ROM	17,140
	RAM	556

Note Memory size is calculated for sample code, ZMOD4XXX sensor control module, and COMMS\_I2C.

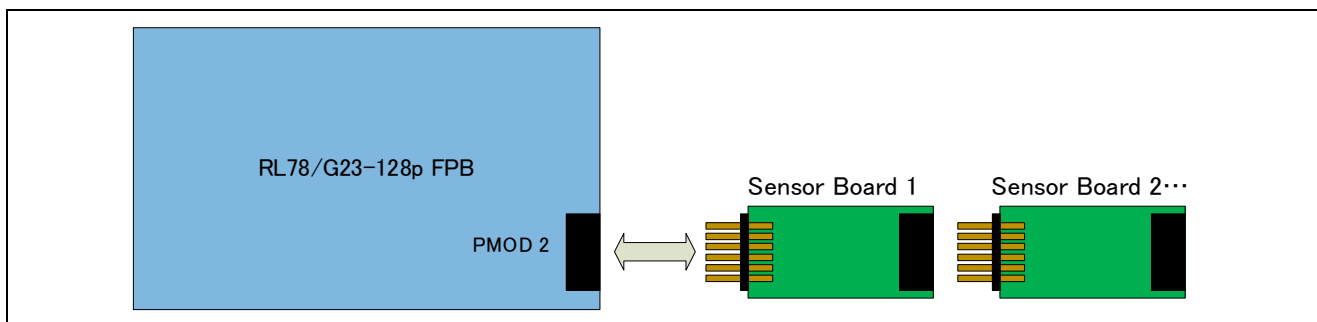


Figure 2-5 Hardware Connections for RL78/G23-128p FPB

**(2) Use of Interposer Board**

The Interposer Board is an I/F conversion board for connecting a Pmod Type 6A sensor by switching the Pmod Type 2A/Type 3A connector of an SCI I/F for RA/RX to the Simple IIC function.

Therefore, even if the Interposer Board is added to a Pmod Type 2A/Type 3A connector to which an SAU I/F is connected, the Pmod Type 6A Sensor Pmod Board **cannot be used**.

### 3. ZMOD4410 Sensor Specifications

#### 3.1 Sensor Specifications Overview

The ZMOD4410 Gas Sensor Module is designed to detect typical TVOC contaminations based on studies and international standards for indoor air quality. Please refer to the [ZMOD4410](#) datasheet for more information about the sensor module, including parameters that describe the module's characteristics.

#### 3.2 Sensor Function and Methods

The ZMOD4XXX architecture leverages different “Methods of Operation” which use time, temperature, and signatures from gases that enable unique signals from a highly trained machine learning system and makes use of embedded artificial intelligence (AI) technology. This section discusses the different operation modes of the ZMOD4410. At present, the following operation modes are released.

Operation Mode 1:	IAQ 1 <sup>st</sup> Generation (Continuous)	; Measurement of UBA levels for IAQ and eCO2
Operation Mode 2:	IAQ 1 <sup>st</sup> Generation (Low Power)	; Measurement of UBA levels for IAQ and eCO2
Operation Mode 3:	IAQ 2 <sup>nd</sup> Generation (Functions improved by AI and recommended for new designs)	; TVOC [ppm], IAQ and eCO2 functionality
Operation Mode 4:	Odor	; Control signal based on Air Quality Changes
Operation Mode 5:	Sulfur-based Odor Discrimination	
Operation Mode 6:	Relative IAQ	; Relative IAQ Measurement based on Air Quality Changes
Operation Mode 7:	PBAQ	; TVOC measurement to meet PBAQ standards

The IAQ 2<sup>nd</sup> Generation (Mode 3) operation should be used for new designs. In case of the need for a slightly faster sample rate and a larger VOC range (up to 30ppm), it is recommended to use the IAQ 1Gen algorithms (Operation Mode 1 or Operation Mode 2).

Additional technical information on sensitivity, selectivity, and stability for all operation modes is available in Renesas' *ZMOD4410 Application Note – TVOC Sensing*. For more information, including application notes, white papers, blog, and manuals, visit the [ZMOD4410](#) webpage.

##### 3.2.1 Conversion of Output Data – Firmware / API / Algorithms

To operate the ZMOD4410, a firmware provided by Renesas containing an API, algorithm libraries, and an example should be used. For implementing the sensor module in a customer-specific application, detailed information on the programming is available. For downloading these documents, please visit the [ZMOD4410](#) webpage

##### 3.2.2 Typical Hardware Requirements

To operate the ZMOD4410, customer-specific hardware with a microcontroller unit (MCU) is needed. Depending on the sensor configuration and on the hardware itself, the requirements differ, and the following minimum requirements are provided as an orientation only:

- 12kB to 20kB program flash for ZMOD4410-related firmware code (MCU architecture and compiler dependent)
- 1kB RAM for ZMOD4410-related operations
- Capability to perform I2C communication, timing functions, and floating-point instructions
- The algorithm functions work with variables saved in background and need memory retention between each call

## 4. ZMOD4450 Sensor Specifications

### 4.1 Sensor Specifications Overview

The ZMOD4450 Gas Sensor Module is designed to detect typical gases inside refrigeration applications associated with food ripening or rotting. Please refer to the [ZMOD4450](#) datasheet for more information about the sensor module, including parameters that describe the module's characteristics.

### 4.2 Sensor Function and Methods

The ZMOD architecture leverages different "Methods of Operation" which use time, temperature, and signatures from gases that enable unique signals from a highly trained machine learning system and makes use of embedded artificial intelligence (AI) technology. This section discusses the different operation modes of the ZMOD4450. At present, the following operation mode is released.

Operation Mode 1:     RAQ     ;Algorithmic calculation of the refrigeration air quality (RAQ) change

In addition, details for sensitivity, reliability, sample rates, and sensor module influences are explained in detail in the following sections. All graphs and information show the typical responses that are to be expected from the sensor module upon exposure to a variety of test conditions. For more information, including application notes, white papers, blog, and manuals, visit the [ZMOD4450](#) product page.

#### 4.2.1 Conversion of Output Data – Firmware / API / Algorithms

To operate the ZMOD4450, a firmware provided by Renesas containing an API, algorithm libraries, and an example should be used. For implementing the sensor module in a customer-specific application, detailed information on the programming is available. For downloading these documents, please visit the [ZMOD4450](#) webpage

#### 4.2.2 Typical Hardware Requirements

To operate the ZMOD4450, customer-specific hardware with a microcontroller unit (MCU) is needed. Depending on the sensor configuration and on the hardware itself, the requirements differ, and the following minimum requirements are provided as an orientation only:

- 10kB to 20kB program flash for ZMOD4450-related firmware code (MCU architecture and compiler dependent)
- 1kB RAM for ZMOD4450-related operations
- Capability to perform I2C communication, timing functions, and floating-point instructions
- The algorithm functions work with variables saved in background and need memory retention between each call

## 5. ZMOD4510 Sensor Specifications

### 5.1 Sensor Specifications Overview

The ZMOD4510 Gas Sensor Module detects typical gases based on studies and international standards for outdoor air quality. Please refer to the [ZMOD4510](#) datasheet for more information about the sensor module, including parameters that describe the module's characteristics.

### 5.2 Sensor Function and Methods

The ZMOD architecture leverages different “Methods of Operation” which use time, temperature, and signatures from gases that enable unique signals from a highly trained machine learning system and makes use of embedded artificial intelligence (AI) technology. This section discusses the different operation modes of the ZMOD4510. At present, the following operation modes are released.

Operation Mode 1:	NO2 O3	; Selective measurement of Nitrogen dioxide (NO <sub>2</sub> ) and Ozone (O <sub>3</sub> )
Operation Mode 2:	OAQ 2nd Gen.	; Selective Ozone (O <sub>3</sub> ) measurement using Ultra-Low Power

In addition, details for sensitivity, reliability, sample rates, and sensor module influences are explained in detail in the following sections. All graphs and information show the typical responses that are to be expected from the sensor module upon exposure to a variety of test conditions. For more information, including application notes, white papers, blog, and manuals, visit the [ZMOD4510](#) product page.

#### 5.2.1 Conversion of Output Data – Firmware / API / Algorithms

To operate the ZMOD4510, a firmware provided by Renesas containing an API, algorithm libraries, and an example should be used. For implementing the sensor module in a customer-specific application, detailed information on the programming is available. More information and guidance on the firmware integration, architecture, and supported platforms are available in the ZMOD4510 Programming Manual – Read Me. Code Examples in C and additional firmware descriptions for API, HAL, libraries, etc., are included at no cost in the downloadable firmware package from the ZMOD4510 product page.

#### 5.2.2 Typical Hardware Requirements

To operate the ZMOD4510, customer-specific hardware with a microcontroller unit (MCU) is needed. Depending on the sensor configuration and on the hardware itself, the requirements differ, and the following minimum requirements are provided as an orientation only:

- 12kB to 20kB program flash for ZMOD4510-related firmware code (MCU architecture and compiler dependent)
- 1kB RAM for ZMOD4510-related operations
- Capability to perform I2C communication, timing functions, and floating-point instructions
- The algorithm functions work with variables saved in background and need memory retention between each call

## 6. Specification of Sample Software

The sample software package includes the following 9 projects for each sensor (total 27 projects):

- RA2E1 group: Non-OS projects and OS (FreeRTOS) projects
- RA0E1 group: Non-OS (IICA / SAU) projects
- RX140 group: Non-OS (CC-RX / GCC) projects and OS (FreeRTOS) projects
- RL78/G23 group: Non-OS (CC-RL / LLVM) projects

In addition, the following 3 projects are included as combination projects of ZMOD4410 and ZMOD4510.

- RA2E1 group: OS (FreeRTOS) project
- RX140 group: OS (FreeRTOS) project
- RL78/G23 group: Non-OS project

For the FreeRTOS settings for RX family, refer to the [FAQ](#).

### 6.1 Sample Software Structure

The following shows the layer diagram of the sample software.

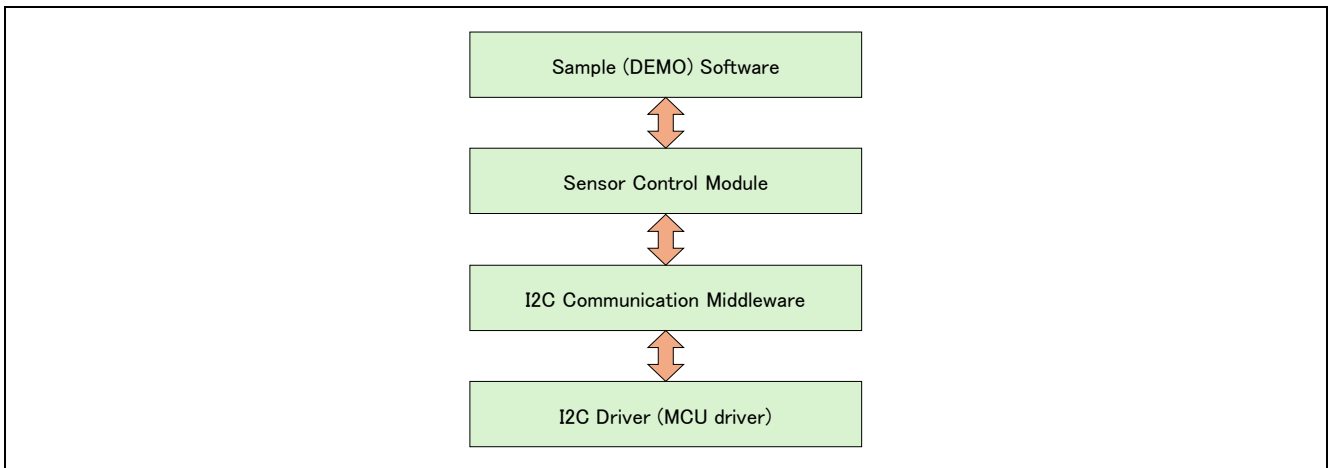


Figure 6-1 Layer Diagram of Sample Software



## 6.2 Specification of Sensor Control Module API Functions

### 6.2.1 List of Sensor Control Module API Functions

The Sensor Control Module API includes the following functions.

For details on the Function API, see below.

RA Flexible Software Package Documentation

Renesas Sensor Control Modules Firmware Integration Technology (R01AN5892)

Renesas Sensor Control Modules Software Integration System (R01AN6192)

**Table 6-1 List of Sensor Control Module API Functions**

Function	Feature
RM_ZMOD4XXX_Open()	Opens a sensor.
RM_ZMOD4XXX_Close()	Closes a sensor.
RM_ZMOD4XXX_MeasurementStart()	Starts measurement.
RM_ZMOD4XXX_MeasurementStop()	Stops measurement.
RM_ZMOD4XXX_StatusCheck()	Reads status of the sensor.
RM_ZMOD4XXX_Read()	Reads ADC data.
RM_ZMOD4XXX_TemperatureAndHumiditySet()	Sets temperature and humidity. (enabled in IAQ 2nd Gen, IAQ 2nd Gen ULP, OAQ 2nd Gen, PBAQ, and NO2 O3)
RM_ZMOD4XXX_DeviceErrorCheck()	Checks ADC Data Validity.
RM_ZMOD4XXX_Iaq1stGenDataCalculate()	Calculates IAQ 1st Gen. values from ADC data.
RM_ZMOD4XXX_Iaq2ndGenDataCalculate()	Calculates IAQ 2nd Gen. values from ADC data.
RM_ZMOD4XXX_OdorDataCalculate()	Calculates Odor values from ADC data.
RM_ZMOD4XXX_SulfurOdorDataCalculate()	Calculates Sulfur Odor values from ADC data.
RM_ZMOD4XXX_Oaq2ndGenDataCalculate()	Calculates OAQ 2nd Gen. values from ADC data.
RM_ZMOD4XXX_RaqDataCalculate()	Calculates RAQ values from ADC data.
RM_ZMOD4XXX_RellaqDataCalculate()	Calculates Rel IAQ. values from ADC data.
RM_ZMOD4XXX_PbaqDataCalculate()	Calculates PBAQ values from ADC data.
RM_ZMOD4XXX_No2O3DataCalculate()	Calculates NO2 O3 values from ADC data.

6.2.2 API Usage Guide

Figure 6-2 Transition of API Functions shows the transition diagram of functions calling order as the usage condition of ZMOD4XXX API functions.

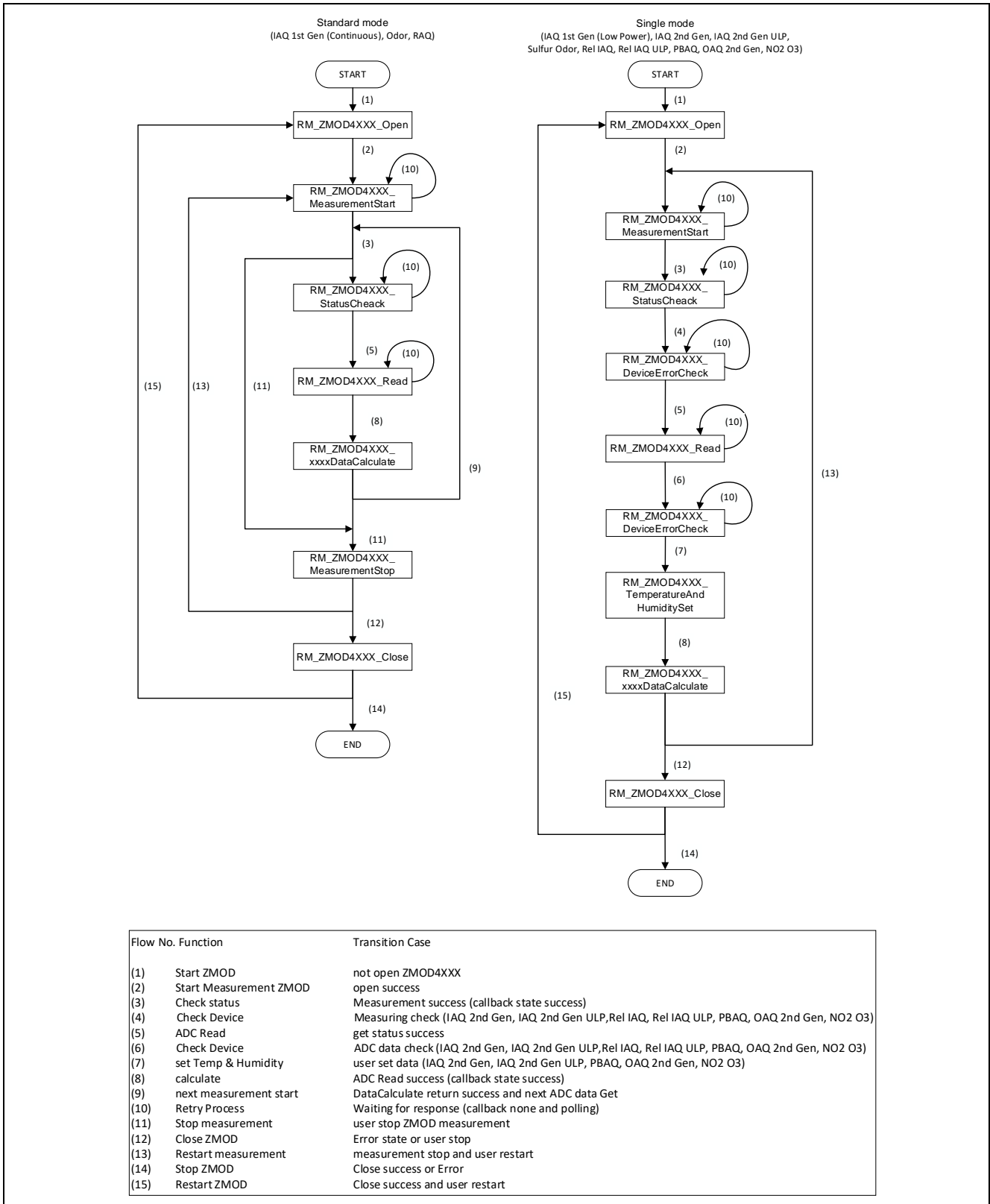


Figure 6-2 Transition of API Functions

The calling conditions for each function are as follows:

- RM\_ZMOD4XXX\_Open(): (1) When starting ZMOD4XXX  
(15) When re-starting after RM\_ZMOD4XXX\_Close()
- RM\_ZMOD4XXX\_Close(): (12) When each process ends successfully or  
with an error
- RM\_ZMOD4XXX\_MeasurementStart(): (2) When starting measurement after  
RM\_ZMOD4XXX\_Open()  
(13) When re-starting next measurement  
(10) When retrying by waiting for measurement start  
response
- RM\_ZMOD4XXX\_MeasurementStop(): (11) When stopping measurement
- RM\_ZMOD4XXX\_StatusCheck(): (3) When checking the status by polling
- RM\_ZMOD4XXX\_Read(): (5) When acquiring measurement data after  
RM\_ZMOD4XXX\_StatusCheck()  
(10) When retrying due to waiting for data acquisition  
response
- RM\_ZMOD4XXX\_TemperatureAndHumiditySet(): (7) Before RM\_ZMOD4XXX\_xxxxDataCalculate() call  
(IAQ 2nd Gen, IAQ 2nd Gen ULP,  
OAQ 2nd Gen, PBAQ, and NO2 O3,  
Improved calculation accuracy)
- RM\_ZMOD4XXX\_DeviceErrorCheck(): (4) When acquiring the measurement status  
(6) When checking ADC data
- RM\_ZMOD4XXX\_xxxxDataCalculate(): (8) When calculating data after  
RM\_ZMOD4XXX\_Read()  
“xxxx” is the name of each sensor function  
(Iaq1stGen, Iaq2ndGen, Odor, SulfurOdor,  
Oaq2ndGen, Raq, Rellaq, Pbaq, No2O3)

Note:

Since RM\_ZMOD4XXX\_Open() checks the state of the I2C driver, the I2C driver must be opened before the RM\_ZMOD4XXX\_Open() processing.

Regarding how to open the I2C driver of RA family and RX family, refer to the g\_comms\_i2c\_bus0\_quick\_setup() function in the sample software. For RL78 family, this is not necessary because the I2C driver will be opened in the startup processing.

When using this API functions in an RTOS system, bus controlling by using semaphore by user is required if controlling the sensors at the same time in multiple threads/tasks.

### 6.3 Flowchart of Main Processing in Non-OS Version of Sample Software

This sample software first starts the driver and then repeats the processing for acquiring data from the sensor and calculating values from the results of measurement.

#### 6.3.1 ZMOD4410, ZMOD4450

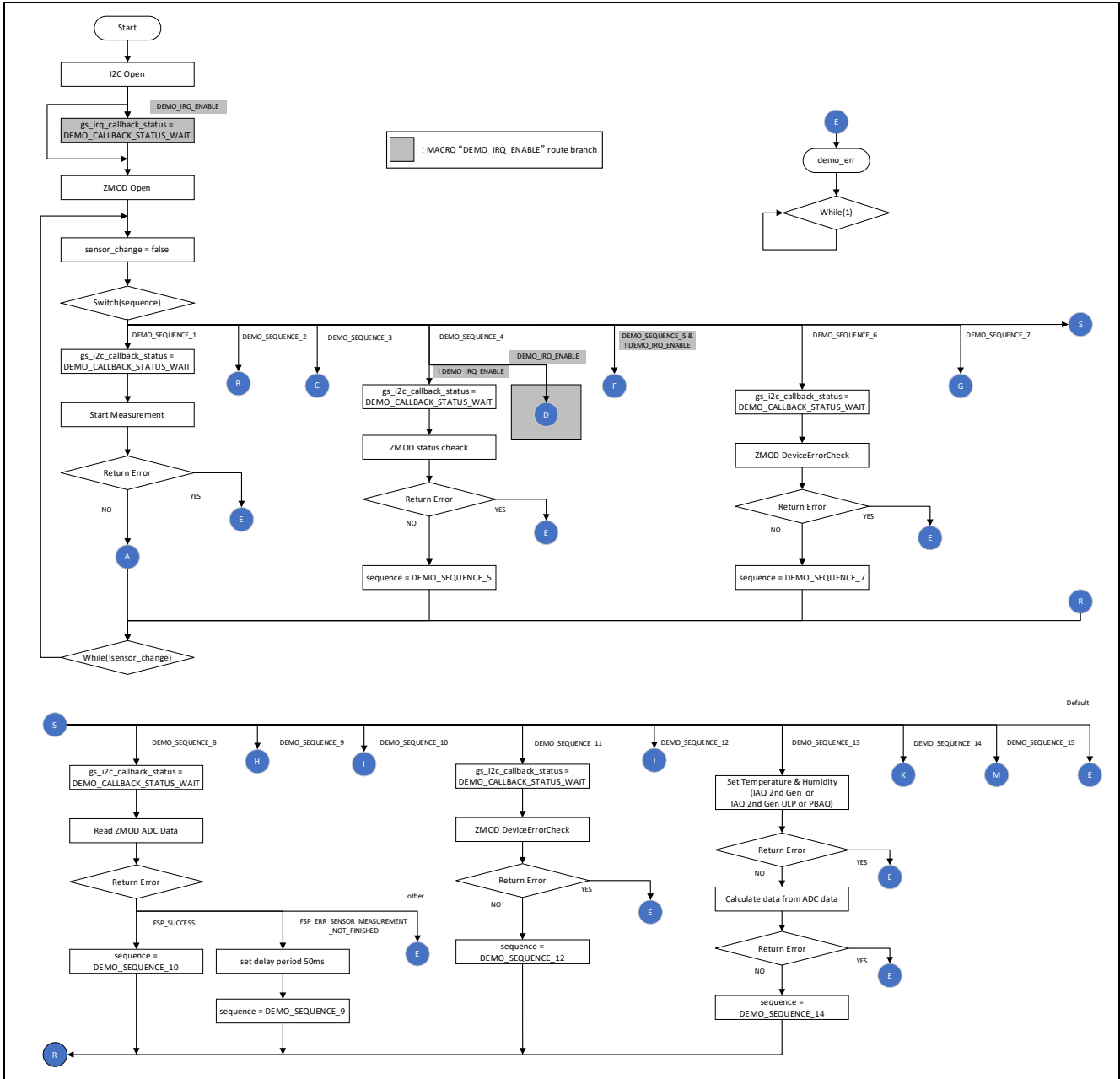


Figure 6-3 Flowchart of Main Processing in Non-OS Version of ZMOD4410, ZMOD4450 Sample Software (1)

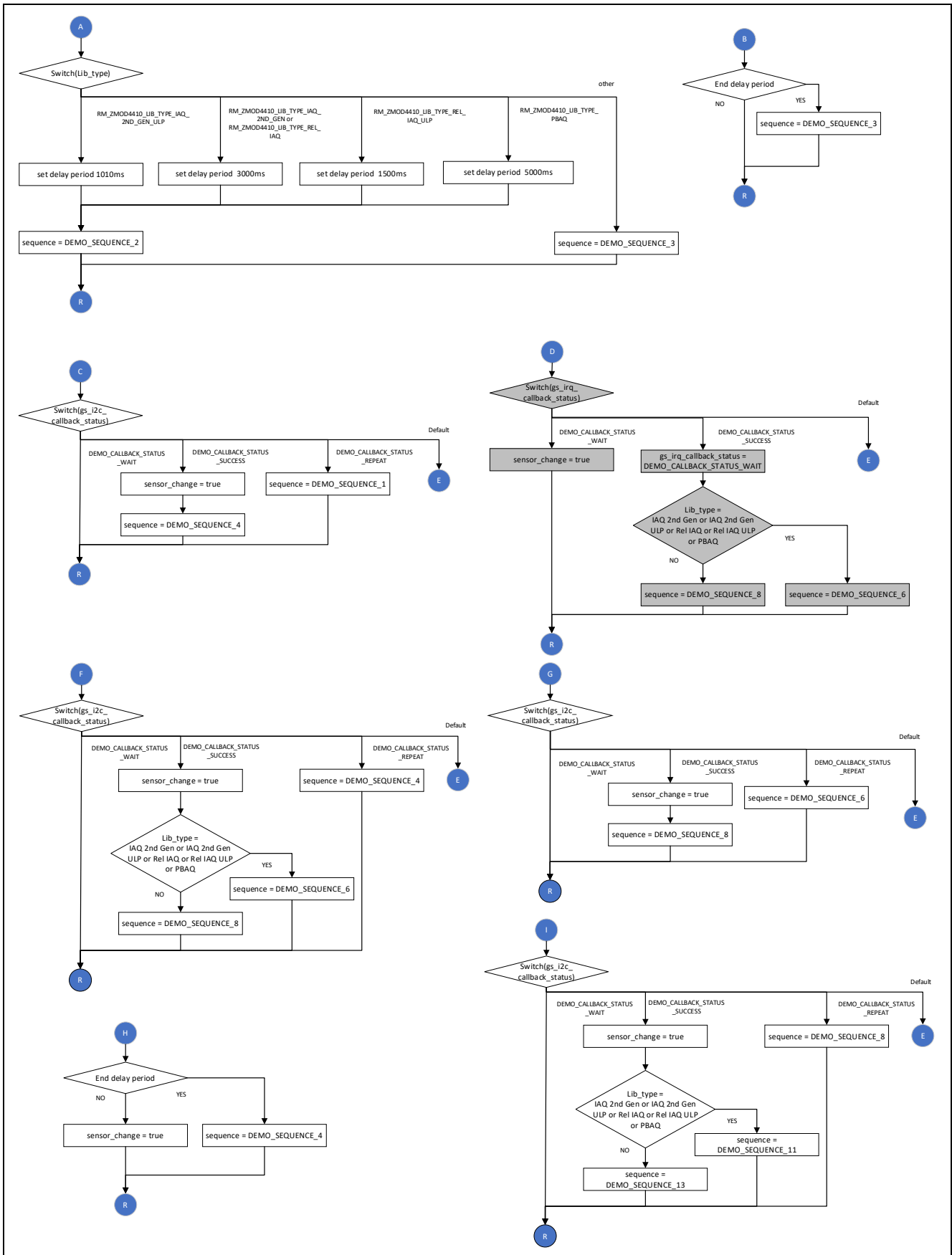


Figure 6-4 Flowchart of Main Processing in Non-OS Version of ZMOD4410, ZMOD4450 Sample Software (2)

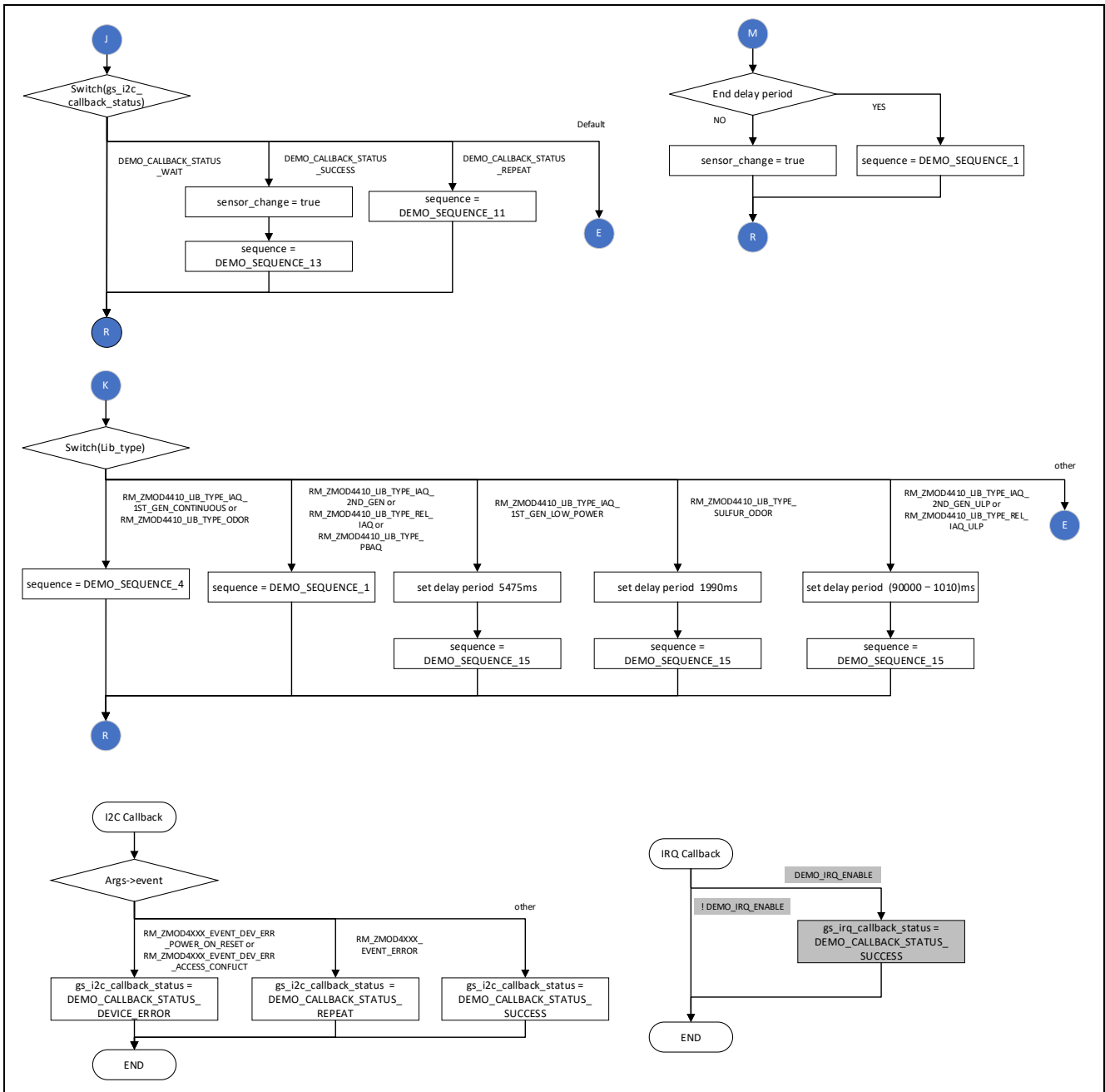


Figure 6-5 Flowchart of Main Processing in Non-OS Version of ZMOD4410, ZMOD4450 Sample Software (3)

6.3.2 ZMOD4510

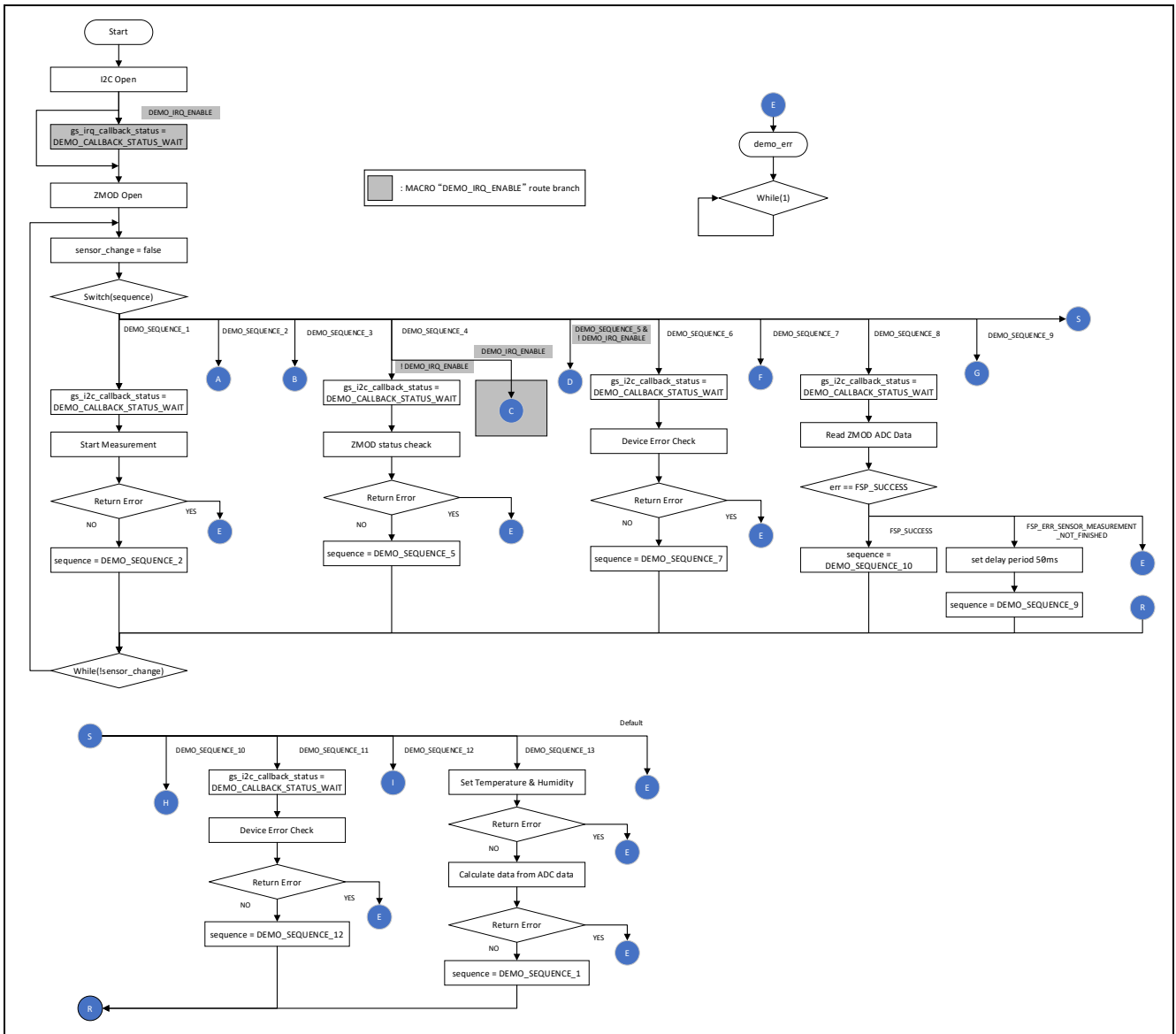


Figure 6-6 Flowchart of Main Processing in Non-OS Version of ZMOD4510 Sample Software (1)

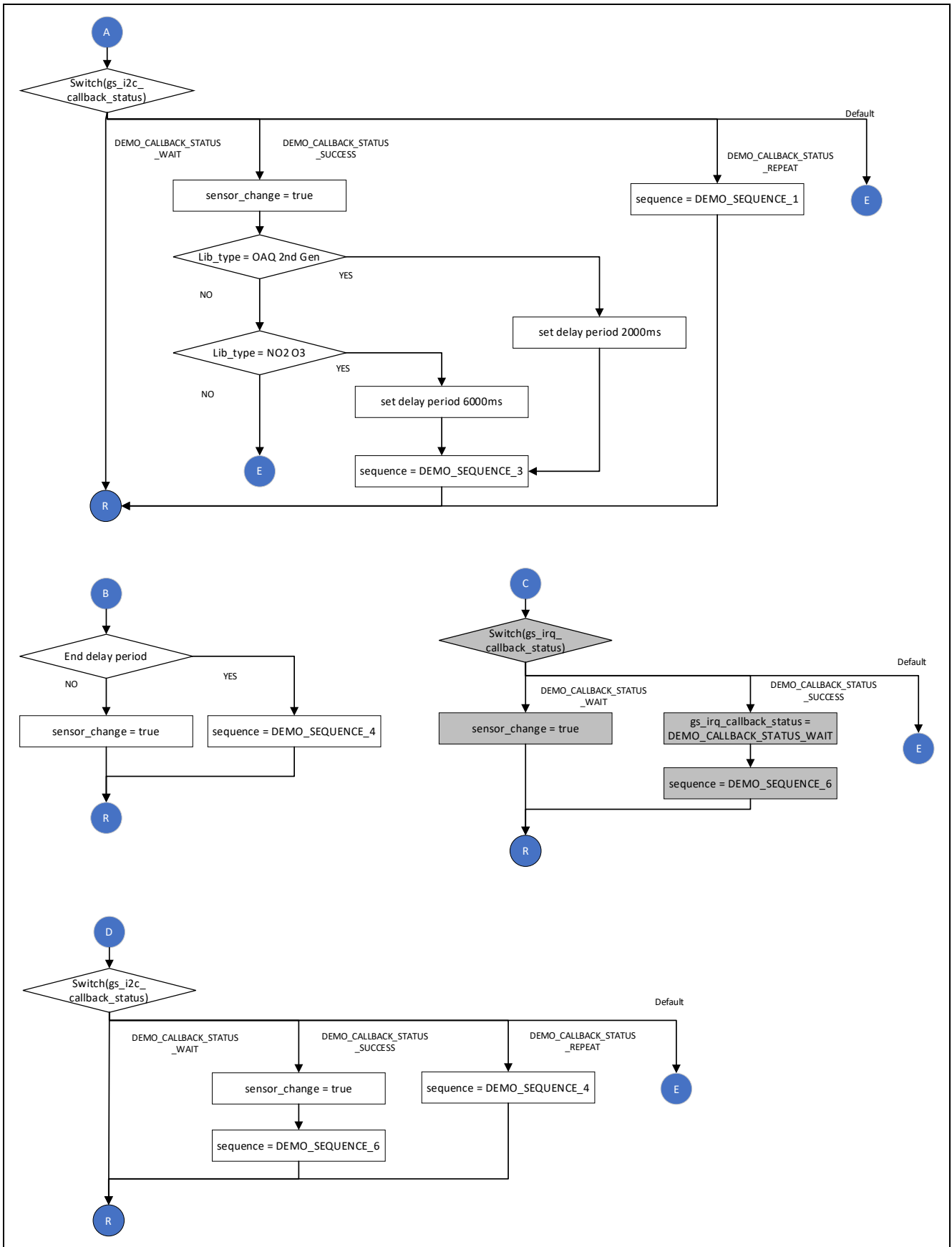


Figure 6-7 Flowchart of Main Processing in Non-OS Version of ZMOD4510 Sample Software (2)



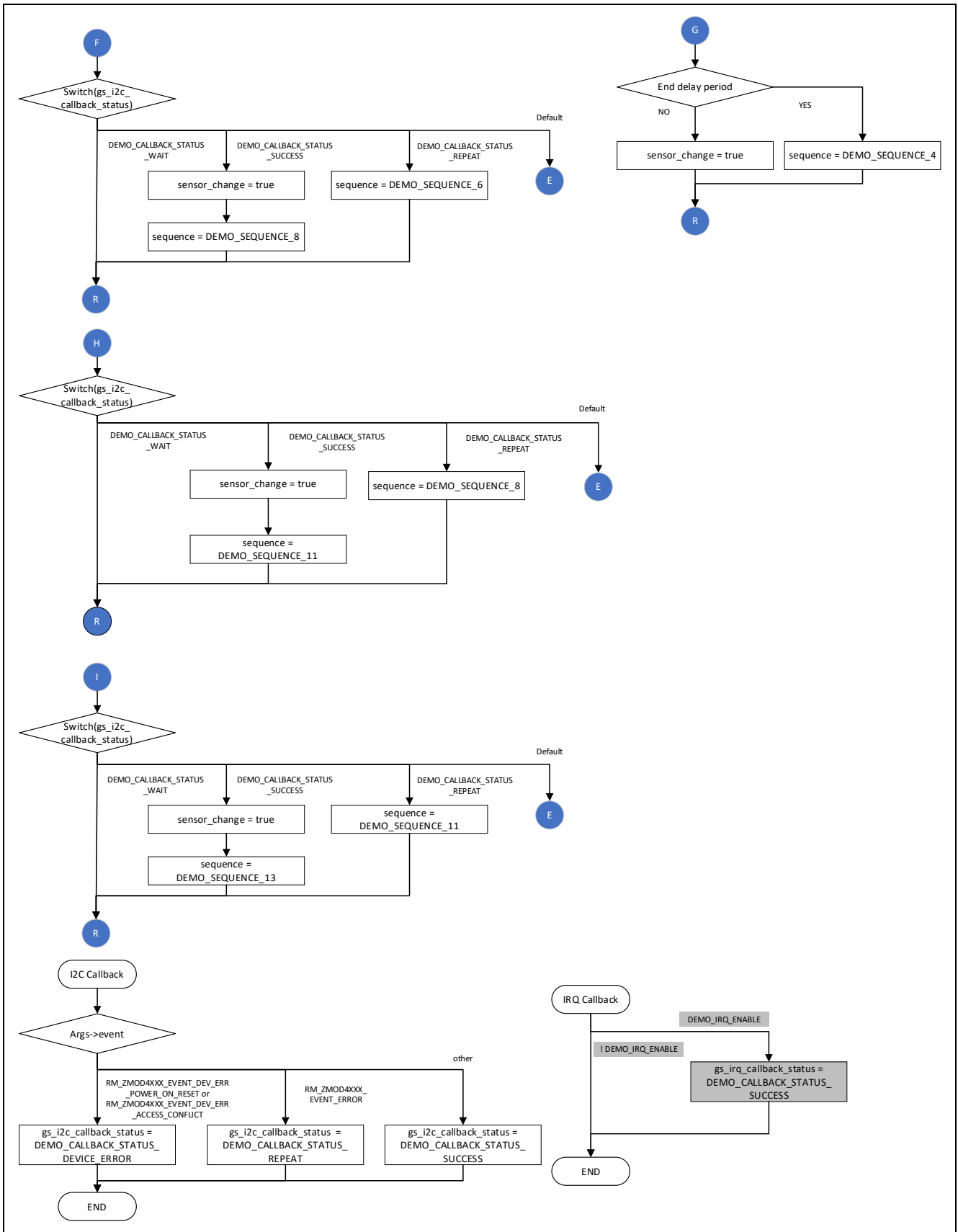


Figure 6-8 Flowchart of Main Processing in Non-OS Version of ZMOD4510 Sample Software (3)

### 6.4 Flowchart of OS Version of Sample Software

The OS version uses a semaphore in control of the sensor and operates a thread for controlling the sensor. The sensor control in thread first starts the driver and then repeats the processing for acquiring data from the sensor and calculating values from the results of measurement.

#### 6.4.1 ZMOD4410, ZMOD4450

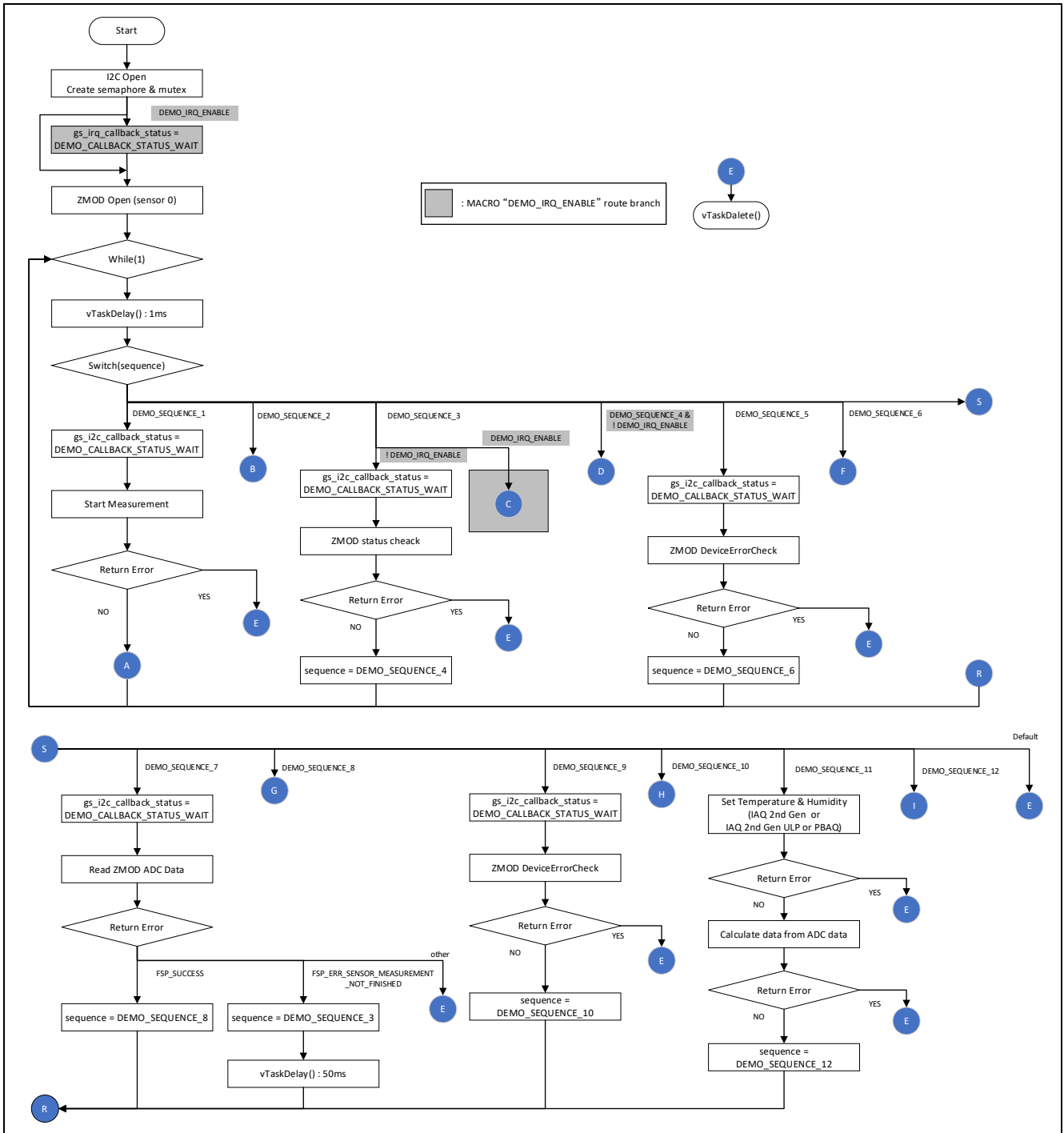


Figure 6-9 Flowchart of Main Processing in OS Version of ZMOD4410, ZMOD4450 Sample Software (1)

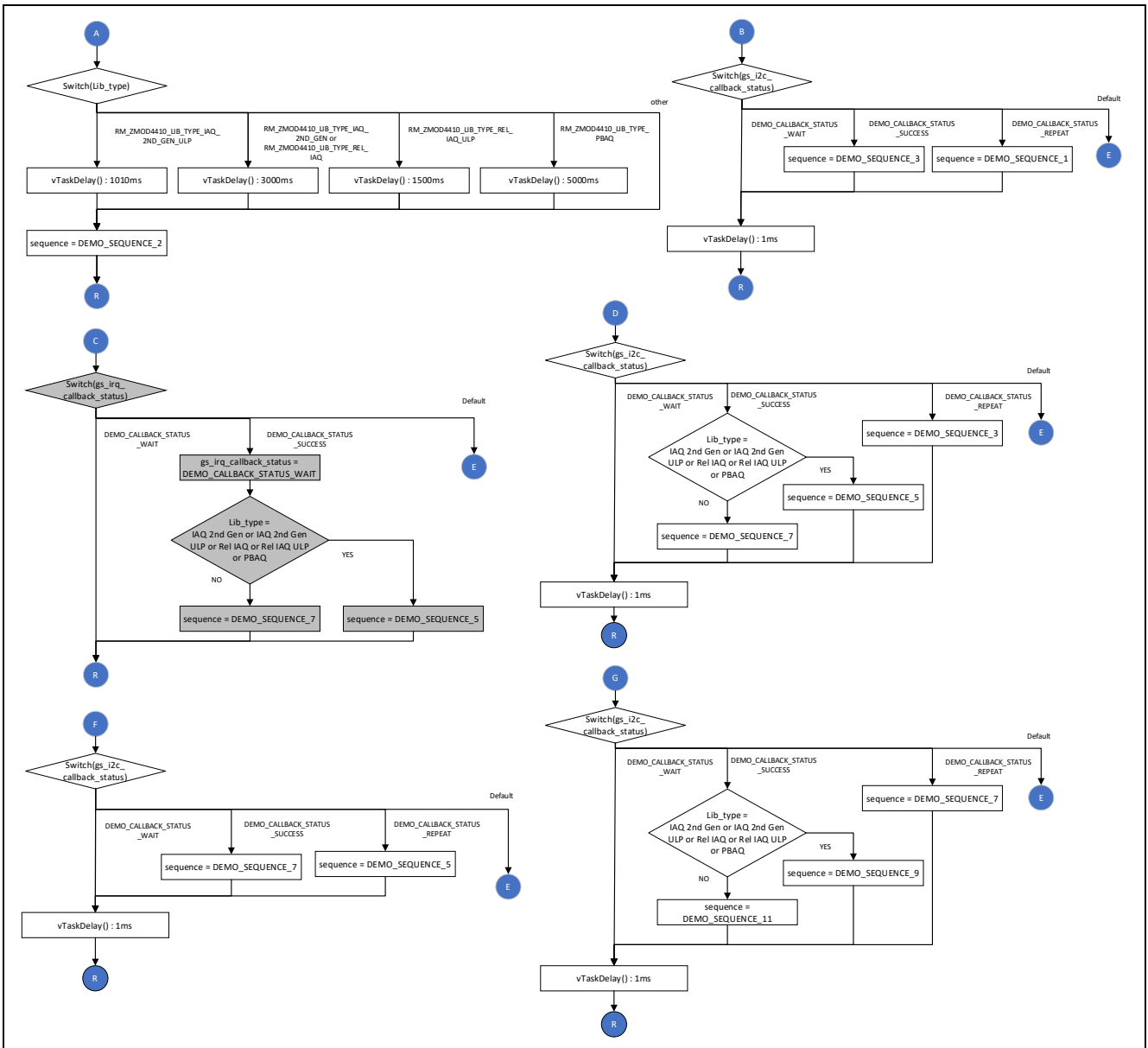


Figure 6-10 Flowchart of Main Processing in OS Version of ZMOD4410, ZMOD4450 Sample Software (2)

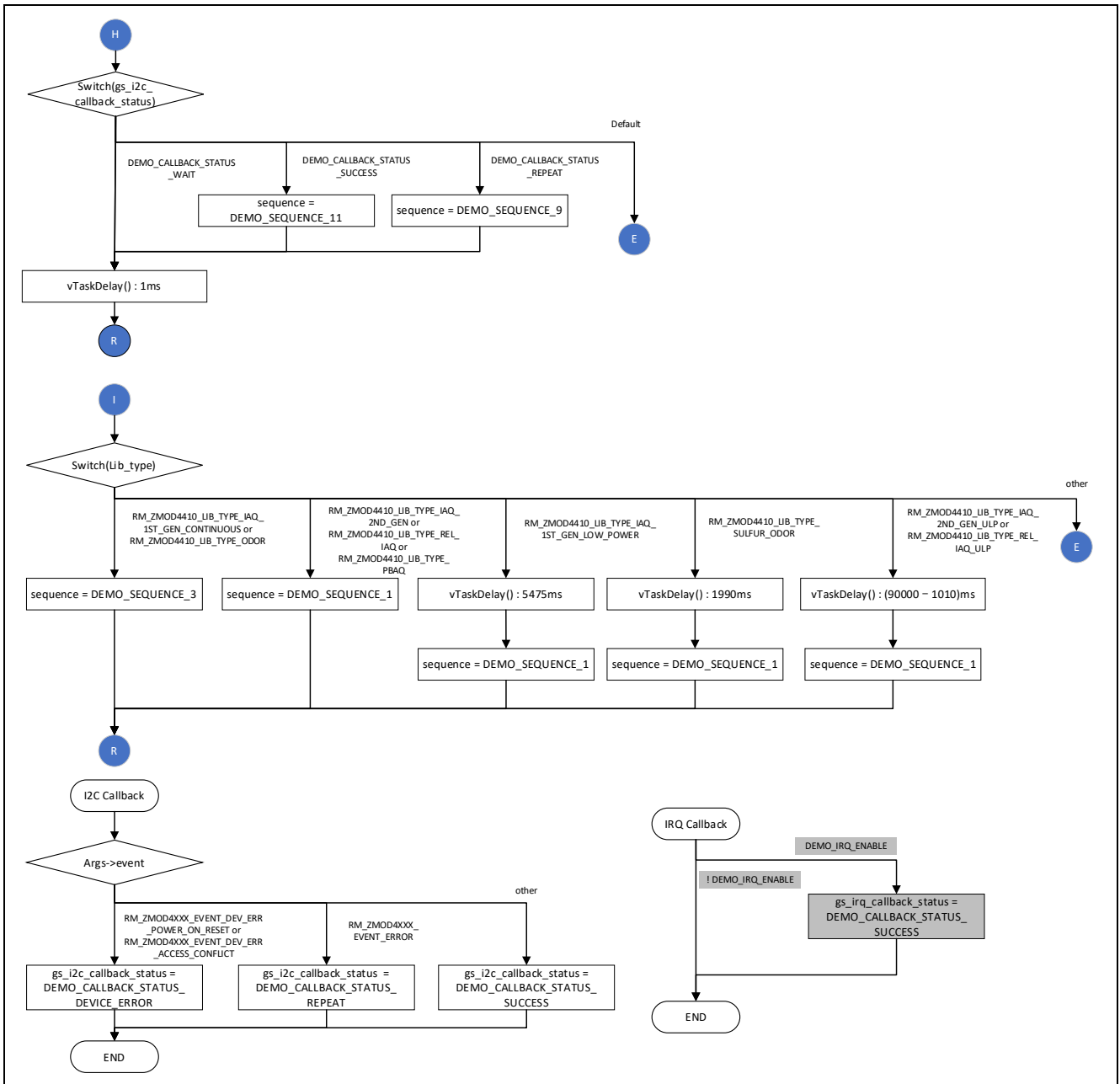


Figure 6-11 Flowchart of Main Processing in OS Version of ZMOD4410, ZMOD4450 Sample Software (3)

6.4.2 ZMOD4510

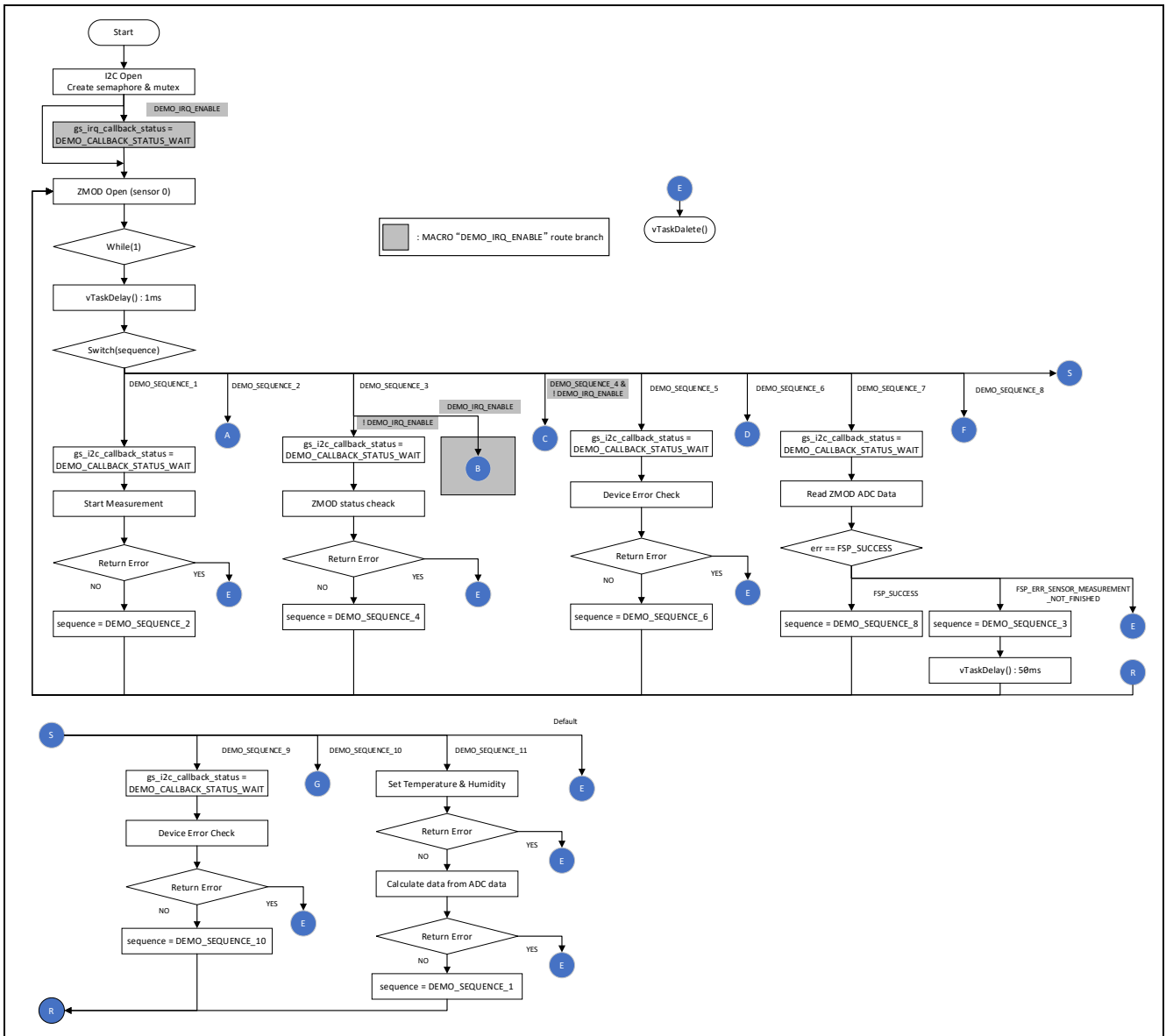


Figure 6-12 Flowchart of Main Processing in OS Version of ZMOD4510 Sample Software (1)

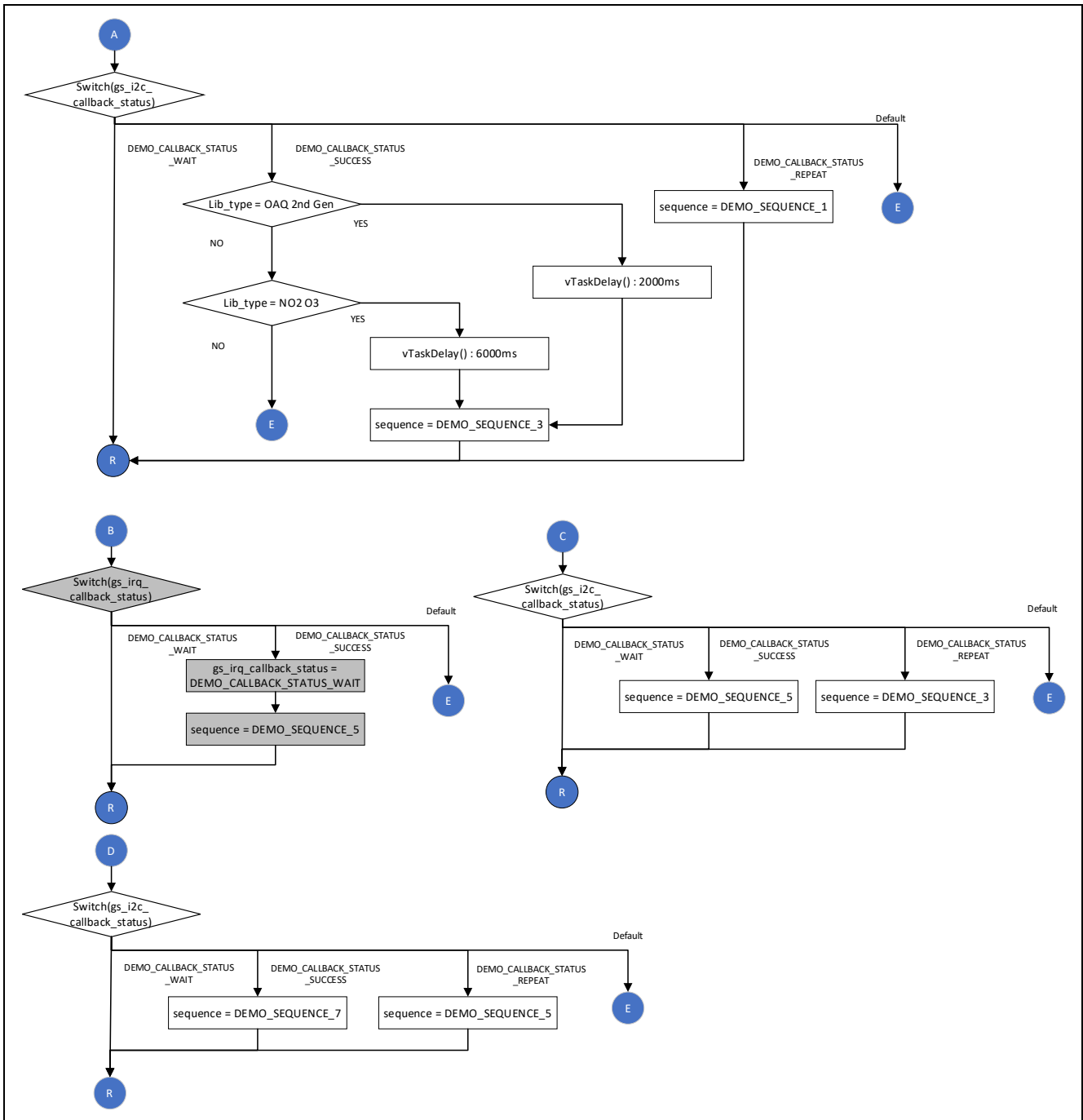


Figure 6-13 Flowchart of Main Processing in OS Version of ZMOD4510 Sample Software (2)

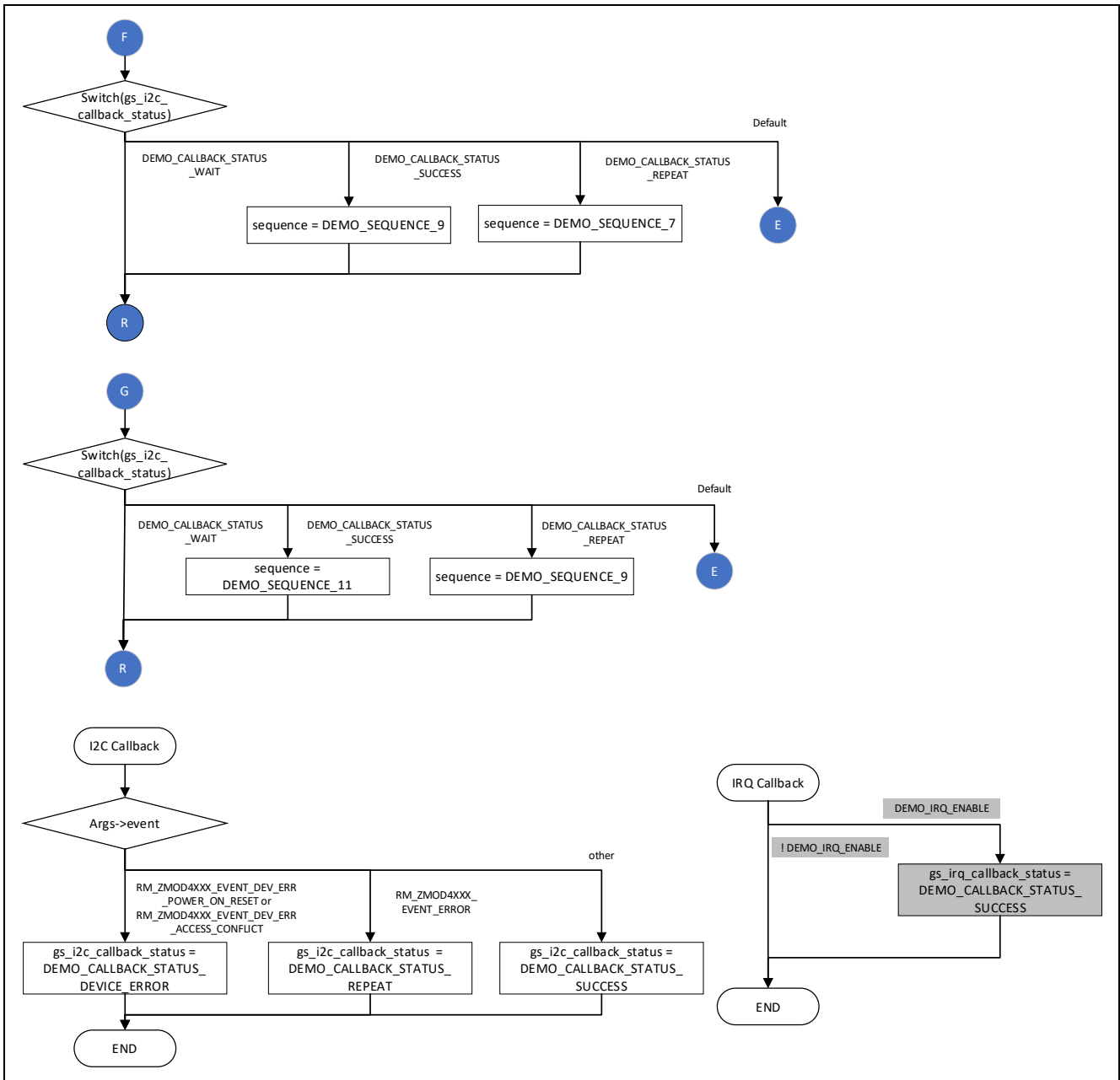


Figure 6-14 Flowchart of Main Processing in OS Version of ZMOD4510 Sample Software (3)

## 6.5 Control Sequence when Interrupt Control

In the following operation modes, measurement start communication is performed at regular intervals to keep the sensor accuracy. The execution timing of Measurement Start communication and data read communication under interrupt control is shown below.

The ZMOD4410 and ZMOD4510 combination project, each software is designed to run in parallel.

Therefore, even when combined, communication with each sensor is performed at the following timing. However, if there is a delay during communication processing, it may be executed at a different cycle than the one below.

**Table 6-2 List of Measurement Start Cycle and Read Timing**

Operation Mode	Measurement Start Cycle	Read Timing
IAQ 1st Gen (Low Power)	5475ms	After measurement starts, read data as soon as measurement is completed.
IAQ 2nd Gen	3sec	After measurement starts, read data 3sec later.
IAQ 2nd Gen ULP	90sec	After measurement starts, read data 1010ms later.
Sulfur Odor	1990ms	After measurement starts, read data as soon as measurement is completed.
Rel IAQ	3sec	After measurement starts, read data 3sec later.
Rel IAQ ULP	90sec	After measurement starts, read data 1500ms later.
PBAQ	5sec	After measurement starts, read data 5sec later.
OAQ 2nd Gen	2sec	After measurement starts, read data 2sec later.
NO2 O3	6sec	After measurement starts, read data 6sec later.

Note If sensor measurement is not completed, data read communication will not be performed.



## 7. Configuration Settings

The following items and values can be specified.

**Green** setting value is an item selected by default, and **Orange** setting value is an item that cannot be changed.

For module names and callback function names, specify names that conform to the C language standard.

When using the different module version, the settings items and values shown below may differ.

### 7.1 ZMOD4XXX Sensor Control Module Settings

#### 7.1.1 RA Family

Select the “**rm\_zmod4xxx**” stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

**Table 7-1 ZMOD4XXX Settings for RA Family**

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
Module g_zmod4xxxx_sensor ZMOD4XXX Gas Sensor (rm_zmod4xxx)		
Name	g_zmod4xxx_sensor0	Specify the name of the module.
Comms I2C Callback	zmod4xxx_comms_i2c_callback	Specify the name of the user callback function. When "NULL" is specified, no callback function is used.
IRQ Callback	zmod4xxx_irq_callback	Specify the IRQ user callback function name. When "NULL" is specified, no callback function is used.

7.1.2 RX Family

Select the “r\_zmod4xxxx\_rx” component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

Table 7-2 ZMOD4XXX Settings for RX Family

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
Number of ZMOD4XXX Sensors	1	Specify the number of ZMOD4XXX sensors.
	2	
Operation mode of ZMOD4XXX Sensor{x} (x = 0 or 1)	Not selected.	Specify ZMOD4XXX sensor operation mode.
	IAQ 1st Gen. (Continuous)	
	IAQ 1st Gen. (Low Power)	
	IAQ 2nd Gen.	
	Odor	
	Sulfur-based Odor	
	OAQ 2nd Gen.	
	IAQ 2nd Gen. Ultra-Low Power	
	RAQ	
	Rel IAQ.	
	Rel IAQ. Ultra-Low Power	
	PBAQ.	
NO2 O3		
I2C Communication device No. for ZMOD4XXX sensor device{x} (x = 0 or 1)	I2C Communication Device{x} (x = 0 - 15)	Specify the communications device number to be used by the sensor.
I2C callback function for ZMOD4XXX sensor device{x} (x = 0 or 1)	zmod4xxx_user_i2c_callback{x} (x = 0 or 1)	Specify the name of I2C callback function. When "NULL" is specified, no callback function is used.
Enable IRQ from ZMOD4XXX sensor device{x} (x = 0 or 1)	Disabled	Specify IRQ enable or disable.
	Enabled	
IRQ Callback function for ZMOD4XXX sensor device{x} (x = 0 or 1)	zmod4xxx_user_irq_callback{x} (x = 0 or 1)	Specify the name of IRQ callback function. When "NULL" is specified, no callback function is used.
IRQ number for ZMOD4XXX sensor device{x} (x = 0 or 1)	IRQ{x} (x = 0 - 15)	Specify valid IRQ number.
IRQ trigger for ZMOD4XXX sensor device {x} (x = 0 or 1)	Low Level	Specify IRQ trigger. When using ZMOD4XXX Sensor Pmod Board, specify Rising. (Note 1)
	Falling	
	Rising	
	Both Edges	
IRQ interrupt priority for ZMOD4XXX sensor device{x} (x = 0 or 1)	Priority{x} (x = 0 - 15)	Specify IRQ interrupt priority.

Note 1: The interrupt trigger setting depends on the interrupt signal circuit. For information of the interrupt circuit configuration, refer to "8.5 Notes for Interrupt Signal Circuits".

If the circuit configuration connects the ZMOD4XXX INT signal directly to the MCU interrupt terminal, set it to "Falling".

### 7.1.3 RL78 Family

Select the “r\_zmod4xxx” component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

**Table 7-3 ZMOD4XXX Settings for RL78 Family**

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
Number of ZMOD4XXX Sensors	1	Specify the number of ZMOD4XXX sensors.
	2	
Operation mode of ZMOD4XXX Sensor{x} (x = 0 or 1)	IAQ 1st Gen. (Continuous)	Specify ZMOD4XXX sensor operation mode.
	IAQ 1st Gen. (Low Power)	
	IAQ 2nd Gen.	
	Odor	
	Sulfur-based Odor	
	OAQ 2nd Gen.	
	IAQ 2nd Gen. Ultra-Low Power	
	RAQ	
	Rel IAQ.	
	Rel IAQ. Ultra-Low Power	
	PBAQ.	
	NO2 O3	
I2C Communication device No. for ZMOD4XXX sensor device{x} (x = 0 or 1)	I2C Communication Device{x} (x = 0 - 4)	Specify the communications device number to be used by the sensor.
I2C callback function for ZMOD4XXX sensor device{x} (x = 0 or 1)	zmod4xxx_user_i2c_callback{x} (x = 0 or 1)	Specify the name of I2C callback function. When "NULL" is specified, no callback function is used.
Enable INTC from ZMOD4XXX sensor device{x} (x = 0 or 1)	Disabled	Specify INTC enable or disable.
	Enabled	
INTC Callback function for ZMOD4XXX sensor device{x} (x = 0 or 1)	zmod4xxx_user_irq_callback{x} (x = 0 or 1)	Specify the name of INTC callback function. When "NULL" is specified, no callback function is used.
INTC Callback function for ZMOD4XXX sensor device{x} (x = 0 or 1)	INTP{x} (x = 0 - 15)	Specify the number of INTC.

Note: When using Code Generator or using Smart Configurator of e2 studio 2021-10 or later, library settings are not automatically set in the build settings. Please set the library settings manually after code generation.

## 7.2 I2C Communication Middleware (COMMS\_I2C) Settings

### 7.2.1 RA Family

Select the “`rm_comms_i2c`” stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

**Table 7-4 COMMS\_I2C Settings for RA Family**

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
Module <code>g_comms_i2c_device0</code> I2C Communication Device ( <code>rm_comms_i2c</code> )		
Name	<code>g_comms_i2c_device0</code>	Specify the name of the module.
Semaphore Timeout	<code>0xFFFFFFFF</code>	For an RTOS project, specify the time of semaphore timeout.
Slave Address	<code>0x32</code> or <code>0x33</code>	Specify the slave address. No setting is required as this will be overwritten by the Sensor Control module.
Address Mode	7-Bit	Specify the number of slave address bits. No setting is required as this will be overwritten by the Sensor Control module.
Callback	<code>rm_zmod4xxx_comms_i2c_callback</code>	Specify the name of the user callback function. No setting is required as this will be overwritten by the Sensor Control module.
Module <code>g_comms_i2c_bus0</code> I2C Shared Bus ( <code>rm_comms_i2c</code> )		
Name	<code>g_comms_i2c_bus0</code>	Specify the name of the I2C module.
Bus Timeout	<code>0xFFFFFFFF</code>	Specify the time of I2C bus timeout.
Semaphore for blocking	Unuse	For an RTOS project, enable or disable the blocking processing.
	Use	
Recursive Mutex for Bus	Unuse	For an RTOS project, enable or disable the recursive operation when blocking processing is enabled.
	Use	
Channel	0	Specify the channel number to be used. This setting is valid only when the I2C driver is “ <code>r_iic_master</code> ”. When using other I2C drivers, this setting is invalid.
Rate	Standard	Specify the bit rate. When using ZMOD4XXX, Standard or Fast-mode can be set. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices. When using “ <code>r_iica_master</code> ”, specify “Standard” due to the electrical characteristics of IICAx. When using “ <code>r_sci_i2c</code> ”, “ <code>r_sau_i2c</code> ” this setting is invalid.
	Fast-mode	
	Fast-mode plus	

## 7.2.2 RX Family

Select the “r\_comms\_i2c\_rx” component on the “Component” tabbed page of the Smart Configurator, and the configurable items are shown in the “Configure” panel.

**Table 7-5 COMMS\_I2C Settings for RX Family**

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
Number of I2C Shared Buses	Unused	Specify the number of communications bus lines that can be connected.
	1	
	2 - 16	
Number of I2C Communication Devices	Unused	Specify the number of I2C device that can be connected.
	1	
	2 - 16	
Blocking operation supporting with RTOS	Disabled	For an RTOS project, enable or disable the blocking operation.
	Enabled	
Bus lock operation supporting with RTOS	Disabled	For an RTOS project, enable or disable the bus lock operation.
	Enabled	
I2C Driver Type for I2C Shared bus{x} (x = 0 - 15)	RIIC	Specify the I2C bus type to be used for the communication bus. When using the “RIIC”, r_riic_rx is necessary. When using the “SCI IIC”, r_sci_iic_rx is necessary. If an unused FIT module is deleted, a warning message will appear, but this does not affect the operation.
	SCI IIC	
	Not selected	
Channel No. for I2C Shared bus{x} (x = 0 - 15)	0	Specify the I2C channel number to be used for the communication bus.
Timeout for the bus lock of I2C Shared Bus{x} (x = 0 - 15)	0xFFFFFFFF	Specify the time of I2C bus lock timeout.
I2C Shared Bus No. for I2C Communication Device{x} (x = 0 - 15)	I2C Shared Bus{x} (x = 0 - 15)	Specify the configuration of used communication bus.
Slave address for I2C Communication device{x} (x = 0 - 15)	0x00	Specify the slave address of the device to be connected to the communications bus. When using ZMOD4XXX, specify 0x32 or 0x33.
Address mode for I2C Communication device{x} (x = 0 - 15)	7 bit address mode	Specify the slave address mode. When using ZMOD4XXX, specify the 7-bit address mode.
Callback function for I2C Communication device{x} (x = 0 - 15)	comms_i2c_user_callback{x} (x = 0 - 15)	Specify the name of the user callback function. When using r_zmod4xxx_rx, specify rm_zmod4xxx_callback{y} (y = 0 or 1).
Timeout for the blocking bus of I2C Communication device{x} (x = 0 - 15)	0xFFFFFFFF	Specify the time of I2C bus blocking timeout.

### 7.2.3 RL78 Family

Select the "r\_comms\_i2c" component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the "Configure" panel.

**Table 7-6 COMMS\_I2C Settings for RL78 Family**

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
	Enabled	
	Disabled	
Number of I2C Shared Buses	Unused	Specify the number of communication bus lines that can be connected.
	1	
	2 - 5	
Number of I2C communication Devices	Unused	Specify the number of I2C devices can be connected.
	1	
	2 - 5	
I2C Driver Type for I2C Shared bus{x} (x = 0 - 4)	IICA	Specify the I2C type to be used for the communication bus. When using ZMOD4XXX, specify "IICA".
	SAU IIC	
	Not selected	
Component name for the I2C bus{x} (x = 0 - 4)	Config_IIC00	Specify the I2C bus component name to be used for the communication bus.
I2C Shared Bus No. for I2C Communication Device{x} (x = 0 - 4)	I2C bus0	Specify the I2C bus configuration to be used for the communication bus.
	I2C bus1	
	I2C bus2	
	I2C bus3	
	I2C bus4	
Slave address for I2C Communication device{x} (x = 0 - 4)	0x00	Specify the slave address of the device to be connected to the communications bus. When using ZMOD4XXX, specify 0x32 or 0x33.
Callback function for I2C Communication device{x} (x = 0 - 4)	comms_i2c_user_callback{x} (x = 0 - 4)	Specify the name of the user callback function. When using r_zmod4xxx specify rm_zmod4xxx_callback{y} (y = 0 or 1).

## 7.3 I2C Driver Settings

### 7.3.1 RA Family

Select the “r\_iic\_master”, “r\_sci\_i2c”, “r\_sau\_i2c” or “r\_iica\_master” stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

#### (1) r\_iic\_master

Table 7-7 r\_iic\_master Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code.
	Enabled	When “Disabled” is specified, excluding in the code.
	Disabled	When “Enabled” is specified, including in the code.
DTC on Transmission and Reception	Enabled	Specify whether to use the DTC for transmission and reception.
	Disabled	
10-bit slave addressing	Enabled	Specify whether to support 10-bit addressing for the slave address. No setting is required as this will be overwritten by COMMS_I2C.
	Disabled	
Module g_i2c_master0 I2C Master (r_iic_master)		
Name	g_i2c_master0	Specify the name of the module.
Channel	0	Specify the channel number to be used. No setting is required as this will be overwritten by COMMS_I2C.
Rate	Standard	Specify the bit rate.
	Fast-mode	No setting is required as this will be overwritten by COMMS_I2C.
	Fast-mode plus	
Custom Rate (bps)	0	Specify the custom bit rate. This setting is valid when the value is other than 0. Use this setting when you want to set the low bitrate within the “Rate” setting range.
Rise Time (ns)	120	Specify the SCL rise time according to the specifications of the target board to be used.
Fall Time (ns)	120	Specify the SCL fall time according to the specifications of the target board to be used.
Duty Cycle (%)	50	Specify the SCL duty cycle.
Slave Address	0x00	Specify the slave address for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
Address Mode	7-Bit	Specify the slave address mode for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
	10-Bit	
Timeout Mode	Short Mode	Specify the time of I2C bus timeout.
	Long Mode	
Timeout during SCL low	Enabled	Specify whether to timeout can occur when SCL is held low for a duration longer than what is set in the timeout mode.
	Disabled	
Callback	rm_comms_i2c_callback	Set the user callback function name. No setting is required as this will be overwritten by COMMS_I2C.
Interrupt Priority Level	Priority 0 (highest)	Specify the interrupt priority level of the I2C bus driver.
	Priority 1	
	Priority 2	
	Priority 3	
Pins		
SDA	Pxxx	The pin numbers to be used by the driver are displayed.
SCL	Pxxx	Use the "Pins" tabbed page to modify the pin configuration.

## (2) r\_sci\_i2c

Table 7-8 r\_sci\_i2c Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code.
	Enabled	When "Disabled" is specified, excluding in the code.
	Disabled	When "Enabled" is specified, including in the code.
DTC on Transmission and Reception	Enabled	Specify whether to use the DTC for transmission and reception.
	Disabled	
10-bit slave addressing	Enabled	Specify whether to support 10-bit addressing for the slave address. No setting is required as this will be overwritten by COMMS_I2C.
	Disabled	
Module g_i2c0 I2C Master (r_sci_i2c)		
Name	g_i2c0	Specify the name of the module.
Channel	0	Specify the channel number to be used.
Slave Address	0x00	Specify the slave address for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
Address Mode	7-Bit	Specify the salve address mode for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
	10-bit	
Rate	Standard	Specify the bit rate. When using ZMOD4XXX, Standard or Fast-mode can be set. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices.
	Fast-mode	
Custom Rate (bps)	0	Specify the custom bit rate. This setting is valid when the value is other than 0. Use this setting when you want to set the low bitrate within the "Rate" setting range.
SDA Output Delay (nano seconds)	300	Specify the SDA output delay time.
Noise filter setting	Use clock signal divided by 1 with noise filter	Specify the noise filter to be used for input signals.
	Use clock signal divided by 2 with noise filter	
	Use clock signal divided by 4 with noise filter	
	Use clock signal divided by 8 with noise filter	
Bit Rate Modulation	Enable	Enable or disable the bit rate modulation function.
	Disable	
Callback	rm_comms_i2c_callback	Set the user callback function name. No setting is required as this will be overwritten by COMMS_I2C.
Interrupt Priority Level	Priority 0 (highest)	Specify the interrupt priority level of the I2C bus driver.
	Priority 1	
	Priority 2	
	Priority 3	
RX Interrupt Priority Level [Only used when DTC is enabled]	Priority 0 (highest)	When using DTC, specify the priority level of the reception interrupt.
	Priority 1	
	Priority 2	
	Priority 3	
	Disabled	
Pins		
SDA	Pxxx	The pin numbers to be used by the driver are displayed.
SCL	Pxxx	Use the "Pins" tabbed page to modify the pin configuration.



## (3) r\_sau\_i2c

Table 7-9 r\_sau\_i2c Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code.
	Enabled	When "Disabled" is specified, excluding in the code.
	Disabled	When "Enabled" is specified, including in the code.
Enable Critical Section	Enabled	Set enable or disable of critical section.
	Disabled	When using multiple channels on the same SAU unit, specify Enabled.
Manual Start-Stop	Enabled	Specify whether the user calls the start condition and stop condition function.
	Disabled	When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
Enable Single Channel	00	Does not include processing other than the specified channel.
	20	When "Disabled" is selected, all channels are supported.
	11	
	Disabled	
I2C Restart	Enabled	Specify the include resuming condition processing in code.
	Disabled	When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
DTC Support	Enabled	Specify whether to support the DTC.
	Disabled	
Module g_i2c0 I2C Master (r_sau_i2c)		
Name	g_i2c0	Specify the name of the module.
Channel	00	Specify the channel number to be used.
	11	
	20	
Operation clock	CK0	Specify the I2C operation clock.
	CK1	
Slave Address	0x00	Specify the slave address for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
Rate	Standard	Specify the bit rate.
	Fast-mode	When using ZMOD4XXX, Standard or Fast-mode can be set. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices.
	Fast-mode plus	
Delay time (Microseconds)	5	Specify the SDA output delay time.
Callback	rm_comms_i2c_callback	Set the user callback function name. No setting is required as this will be overwritten by COMMS_I2C.
Transfer end interrupt priority	Priority 0 (highest)	Specify the priority level of the transmission end interrupt.
	Priority 1	
	Priority 2	
	Priority 3	
Custom Rate (bps)	0	Specify the custom bit rate. This setting is valid when the value is other than 0. Use this setting when you want to set the low bitrate within the "Rate" setting range.
Pins		
SCL	Pxxx	The pin numbers to be used by the driver are displayed.
SDA	Pxxx	Use the "Pins" tabbed page to modify the pin configuration.

(4) r\_iica\_master

When configuring IICA using FSP v5.4.0 or higher, set “SCLA Pin” and “SDAA Pin” in “Stacks” tabbed page to Pin numbers only.

Table 7-10 r\_iica\_master Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code.
	Enabled	When “Disabled” is specified, excluding in the code.
	Disabled	When “Enabled” is specified, including in the code.
10-bit slave addressing	Enabled	Specify whether to support 10-bit addressing for the slave address.
	Disabled	No setting is required as this will be overwritten by COMMS_I2C.
Module g_iica_master0 IICA Master (r_iica_master)		
Name	g_iica_master0	Specify the name of the module.
Rate	Standard	Specify the bit rate.
	Fast-mode	No setting is required as this will be overwritten by COMMS_I2C.
	Fast-mode plus	
Custom Rate (bps)	0	Specify the custom bit rate. This setting is valid when the value is other than 0. Use this setting when you want to set the low bitrate within the “Rate” setting range.
Signal Rising Times (us)	0	Specify the SCL rise time according to the specifications of the target board to be used.
Signal Falling Times (us)	0	Specify the SCL fall time according to the specifications of the target board to be used.
Duty Cycle (%)	53	Specify the SCL duty cycle.
Digital Filter	Enabled	Specify whether to use the digital filter.
	Disabled	
Address Mode	7-Bit	Specify the salve address mode for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
	10-Bit	
Slave Address	0x00	Specify the slave address for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
Communication reservation	Enabled	Specify whether to use the communication reservation.
	Disabled	
Callback	rm_comms_i2c_callback	Set the user callback function name. No setting is required as this will be overwritten by COMMS_I2C.
IICA0 communication interrupt priority	Priority 0 (highest)	Specify the interrupt priority level of the I2C bus driver.
	Priority 1	
	Priority 2	
	Priority 3	
SCLA Pin	Pxxx	Specify the pin numbers to be used.
SDAA Pin	Pxxx	No setting is required in “Pins” tabbed page.

g\_iica\_master0 IICA Master (r\_iica\_master)

Settings	Property	Value
Common		
Module g_iica_master0 IICA Master (r_iica_master)		
	Name	g_iica_master0
	Rate	Standard
	Signal Rising Time (us)	0
	Signal Falling Time (us)	0
	Duty Cycle (%)	53
	Digital Filter	Disabled
	Address Mode	7-Bit
	Slave Address	0x00
	Communication reservation	Disabled
	Callback	rm_comms_i2c_callback
	IICA0 communication interrupt priority	Priority 2
	SCLA Pin	P100
	SDAA Pin	P101

### 7.3.2 RX Family

Select the “r\_riic\_rx” or “r\_sci\_iic\_rx” component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

#### (1) r\_riic\_rx

**Table 7-11 r\_riic\_rx Settings for RX Family**

Configurable Item	Value	Description
Configurations		
Set parameter checking enable	System Default	Specify the include parameter check processing in code. When “Not” is specified, excluding in the code. When “Include” is specified, including in the code.
	Not	
	Include	
MCU supported channels for CH{x} (x = 0 - 2)	Not supported	Specify whether to support the operation of the channel. When “Not supported” is specified, excluding in the code. When “Supported” is specified, including in the code.
	Supported	
CH{x} RIIC bps(kbps) (x = 0 - 2)	400	Specify the bit rate. When using ZMOD4XXX, set it to 400kbps or less. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices.
Digital filter for CH{x} (x = 0 - 2)	Not	Specify the digital filter for input signals. When “Not” is specified, disable the digital filter.
	One IIC phi	
	Two IIC phi	
	Three IIC phi	
	Four IIC phi	
Setting port setting processing	Not include port setting	Specify whether to include the pin function settings in the code to be generated. When “Not include port setting” is specified, excluding in the code. When “Include port setting” is specified, including in the code.
	Include port setting	
Master arbitration lost detection function for CH{x} (x = 0 - 2)	Unused	Specify whether to use the master arbitration lost detection function. If using it in a multi-master environment, set it to "Used". When “Unused” is specified, disable. When “Used” is specified, enable.
	Used	
Address {y} format for CH{x} (x = 0 - 2, y = 0 - 2)	Not	Specify whether to support 7-bit addressing or 10-bit addressing for the slave address. When using ZMOD4XXX, select "7 bit address format". Do not connect devices with different address formats on the same bus.
	7 bit address format	
	10 bit address format	
Slave Address {y} for CH{x} (x = 0 - 2, y = 0 - 2)	0x0025	Specify the slave address of the designated device. No setting is required as this will be overwritten by COMMS_I2C.
General call address for CH{x}	Unused	Specify whether to use the general call function. When “Unused” is specified, disable. When “Used” is specified, enable.
	Used	
CH{x} RXI INT Priority Level (x = 0 - 2)	Level 1	Specify the priority level of the reception interrupt.
	Level 2	
	...	
	Level 14	
	Level 15 (highest)	
CH{x} TXI INT Priority Level (x = 0 - 2)	Level 1	Specify the priority level of the transmission interrupt.
	Level 2	
	...	
	Level 14	
	Level 15 (highest)	

CH{x} EEI INT Priority Level (x = 0 - 2)	Level 1	Specify the priority level of the error interrupt.
	Level 2	
	...	
	Level 14	
	Level 15 (highest)	
CH{x} TEI INT Priority Level (x = 0 - 2)	Level 1	Specify the priority level of the transmission end interrupt.
	Level 2	
	...	
	Level 14	
	Level 15 (highest)	
Timeout function for CH{x} (x = 0 - 2)	Unused	Specify whether to use the timeout function. When "Unused" is specified, disable. When "Used" is specified, enable.
	Used	
Timeout detection time for CH{x} (x = 0 - 2)	Long mode	Specify the time for timeout detection. When "Long mode" is specified, select the long mode. When "Short mode" is specified, select the short mode.
	Short mode	
Count up during low period of timeout detection for CH{x} (x = 0 - 2)	Unused	Specify whether to increment the counter for detecting a timeout while SCL is at the low level when the "Timeout function" for the specified channel is enabled. When "Unused" is specified, disable. When "Used" is specified, enable.
	Used	
Count up during high period of timeout detection for CH{x} (x = 0 - 2)	Unused	Specify whether to increment the counter for detecting a timeout while SCL is at the high level when the "Timeout function" for the specified channel is enabled. When "Unused" is specified, disable. When "Used" is specified, enable.
	Used	
Set Counter of checking bus busy	1000	Specify the counter value to be judged to represent the bus busy state.
<b>Resources</b>		
SCLx Pins	Checked	Specify the pins to be used. Set the pin to "Checked".
	Unchecked	
SDAx Pins	Checked	
	Unchecked	

(2) r\_sci\_iic\_rx

Table 7-12 r\_sci\_iic\_rx Settings for RX Family

Configurable Item	Value	Description
Configurations		
Set parameter checking enable	System Default	Specify the include parameter check processing in code.
	Not	When “Not” is specified, excluding in the code.
	Include	When “Include” is specified, including in the code.
MCU supported channels for CH{x} (x = 0 - 12)	Not supported	Specify whether to support the operation of channel.
	Supported	
SCI IIC bitrate (bps) for CH{x} (x = 0 - 12)	384000	Specify the bit rate. When using ZMOD4XXX, set it to 38400bps or less. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices.
Interrupt Priority for CH{x} (x = 0 - 12)	Level 1	Specify the interrupt priority level.
	Level 2	
	...	
	Level 14	
	Level 15 (highest)	
Digital noise filter (NFEN bit) for CH{x} (x = 0 - 12)	Disable	Specify whether to use the digital noise filter.
	Enable	
Noise Filter Setting Register (NFCS bit) for CH{x} (x = 0 - 12)	The clock divided by 1	Specify the sample clock of the digital noise filter.
	The clock divided by 2	
	The clock divided by 4	
	The clock divided by 8	
I2C Mode Register 1 (IICDL bit) for CH{x} (x = 0 - 12)	18	Specify the number of SDA output delay cycles relative to the falling edge of SSCL pin output. Set in the range of 1 to 31.
Software bus busy check counter	1000	Specify the counter value to be judged to represent the bus busy state.
Port Setting Processing	Not include port setting	Specify whether to include the pin function settings in the code for using the ports as SSCL and SSDA pins. Not include port setting: Omitted from the code. Include port setting: Included in the code
	Include port setting	
Resources		
SSCLx Pins	Checked	Specify the pins to be used. Set the pin to “Checked”.
	Unchecked	
SSDAx Pins	Checked	
	Unchecked	

### 7.3.3 RL78 Family

Select "IICAx" as resource the IIC Communication (Master mode) component in the Smart Configurator, and the configurable items will be shown in the "Configure" panel.

When using SCI I/F, the IIC communication (Master mode) driver (Ver.1.6.0) for SCI I/F generated by the Smart Configurator settings cannot be used as is. This will be supported in the next version.

The following will not provide examples of settings or instructions on how to change the code when using SCI I/F.

#### (1) IICAx

**Table 7-13 IICAx Settings for RL78 Family**

Configurable Item	Value	Description
Configurations		
Clock mode setting	fCLK	Specify the clock to drive counting.
	fCLK/2	
Address	16	Specify the local address.
Operation mode setting	Standard	Specify the operating mode.
	Fast mode	When using ZMOD4XXX, Standard or Fast-mode can be set. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices.
	Fast mode plus	
Digital filter on	Checked	Specify whether to use the digital filtering.
	Unchecked	
Transfer clock (fSCL)	100000	Specify the bit rate. Due to the electrical characteristics of IICAx, specify 100000bps or less.
Set tR and tF manually	Checked	Manually set the SDAAn and SCLAn signal rising / falling times.
	Unchecked	
tR	0	Specify the SDAAn and SCLAn signal rising times.
tF	0	Specify the SDAAn and SCLAn signal falling times.
Communication end interrupt priority (INTIICAx)	Level0 (high)	Specify the priority level of the communication end interrupt.
	Level1	
	Level2	
	Level3 (low)	
Master transmission end	Checked	Specify whether to use the callback function when master transmission ends.
	Unchecked	
Master reception end	Checked	Specify whether to use the callback function when master reception ends.
	Unchecked	
Master error	Checked	Specify whether to use the callback function when a communication error occurs.
	Unchecked	
Generated stop condition in master transmission / reception end callback function	Checked	Specify whether to generate a stop condition in the callback function. Set to "Unchecked".
	Unchecked	

## 7.4 IRQ Driver Settings

For information on matching the interrupt signal circuit, refer to "8.5 Notes for Interrupt Signal Circuits".

### 7.4.1 RA Family

Select the "r\_icu" stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

The following is a configuration example for the ZMOD4XXX Sensor Pmod Board.

**Table 7-14 r\_icu Settings for RA Family**

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
	Enabled	
	Disabled	
Module g_external_irq0 External IRQ (r_icu)		
Name	g_external_irq0	Specify the name of the module.
Channel	0	Specify the channel number to be used.
Trigger	Falling	Specify the trigger. When using ZMOD4XXX Sensor Board, select "Rising". (Note 1)
	Rising	
	Both Edges	
	Low Level	
Digital Filtering	Enabled	Specify whether to use the digital filtering.
	Disabled	
Digital Filtering Sample Clock (Only valid when Digital Filtering is Enabled)	PCLK / 1	Specify the sample clock of digital filtering.
	PCLK / 8	
	PCLK / 32	
	PCLK / 64	
Callback	rm_zmod4xxx_irq_callback	Set the user callback function name. The name of the user callback function is automatically specified by r_icu.
Pin Interrupt Priority	Priority 0 (highest)	Specify the interrupt priority level of the IRQ driver.
	Priority 1	
	Priority 2	
	Priority 3	
Pins		
IRQ	Pxxx	The pin numbers to be used by the driver are displayed. Use the "Pins" tabbed page to modify the pin configuration.

Note 1: The interrupt trigger setting depends on the interrupt signal circuit. For information of the interrupt circuit configuration, refer to "8.5 Notes for Interrupt Signal Circuits".

If the circuit configuration connects the ZMOD4XXX INT signal directly to the MCU interrupt terminal, set it to "Falling".

### 7.4.2 RX Family

Select the “**r\_irq\_rx**” component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

The following is a configuration example for the ZMOD4XXX Sensor Pmod Board.

**Table 7-15 r\_irq\_rx Settings for RX Family**

Configurable Item	Value	Description
Configurations		
Locking function for IRQ APIs	Enabled	Enable or disable the locking functions for the IRQ APIs.
	Disabled	
Set parameter checking enable	System Default	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Not	
	Include	
Filter for IRQ{x} (x = 0 - 15)	Enabled	Specify whether to use the digital filtering.
	Disabled	
Filter clock divisor for IRQ{x} (x = 0 - 15)	Divisor 1	Specify the sample clock of digital filtering.
	Divisor 8	
	Divisor 32	
	Divisor 64	
IRQx Pins	Checked	Specify the pins to be used. Set the pin to “Checked”.
	Unchecked	

### 7.4.3 RL78 Family

Select the “**Interrupt Controller**” component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

The following is a configuration example for the ZMOD4XXX Sensor Pmod Board.

**Table 7-16 Interrupt Controller Settings for RL78 Family**

Configurable Item	Value	Description
Configurations		
INTP{x} (x = 0 - 11)	Checked	Enable or disable INTP{x}. (x = 0 - 11)
	Unchecked	
Valid edge	Falling edge	Specify the trigger. When using ZMOD4XXX Sensor Board, select "Rising". (Note 1)
	Rising edge	
	Both edges	
Priority	Level 0 (high)	Specify the interrupt priority level of the INTP{x}. (x = 0 - 11)
	Level 1	
	Level 2	
	Level 3 (low)	

Note 1: The interrupt trigger setting depends on the interrupt signal circuit. For information of the interrupt circuit configuration, refer to "8.5 Notes for Interrupt Signal Circuits".

If the circuit configuration connects the ZMOD4XXX INT signal directly to the MCU interrupt terminal, set it to "Falling".



## 8. Guide for Changing Target Device

Use the following procedures to change the target device to a new one and run a sample project on the new device.

Therefore, you will need to review the following settings according to the user target circuit.

- Interrupt Signal Circuit: Refer to “8.5 Notes for Interrupt Signal Circuits”.
- RESET Signal Circuit: Refer to “8.6 Notes for RESET Signal Circuits”.

In the ZMOD4XXX sample projects, measurement start communication is performed at regular intervals to keep the sensor accuracy. Please refer to “6.5 Control Sequence when Interrupt Control” for design the timing.

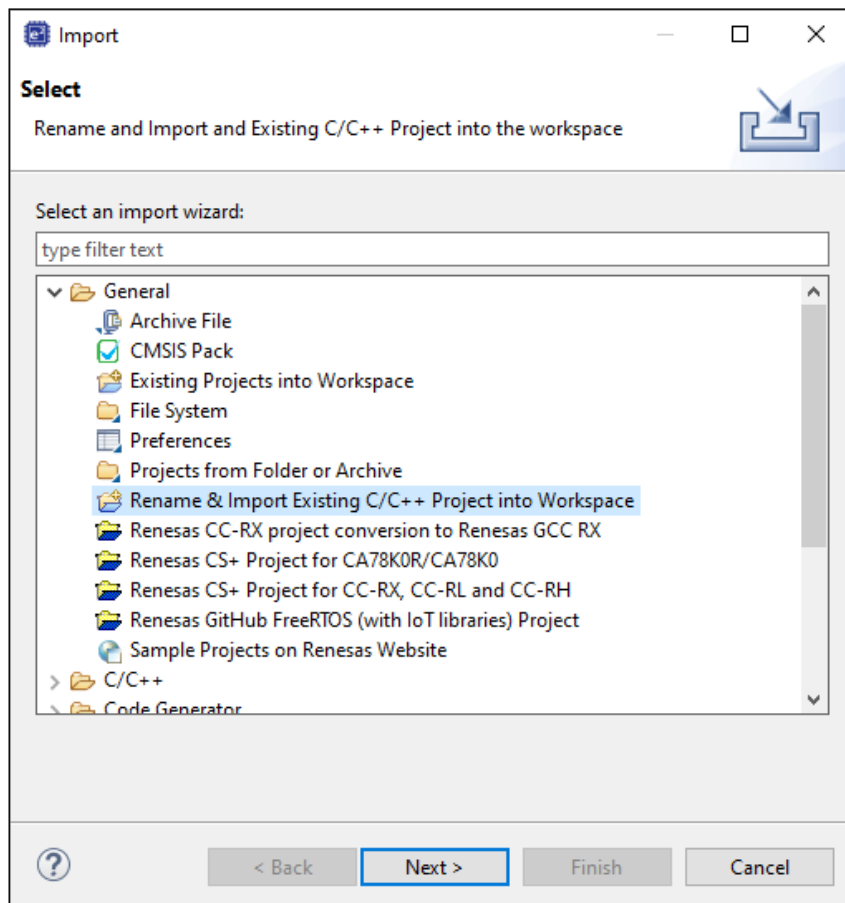
### 8.1 Importing Sample Project

To change a device in a sample project, need to import.

To import a sample project, follow the steps below.

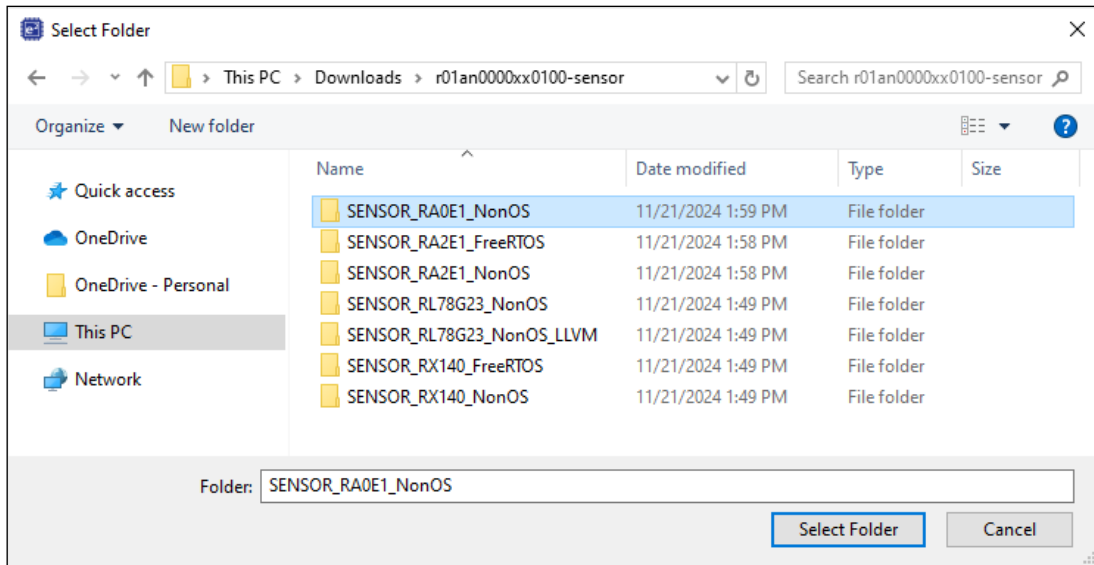
1. Select [Import] from the menu.

The "Import" window will appear. Select "Rename & Import Existing C/C++ Project into Workspace" in the window and press the [Next] button.

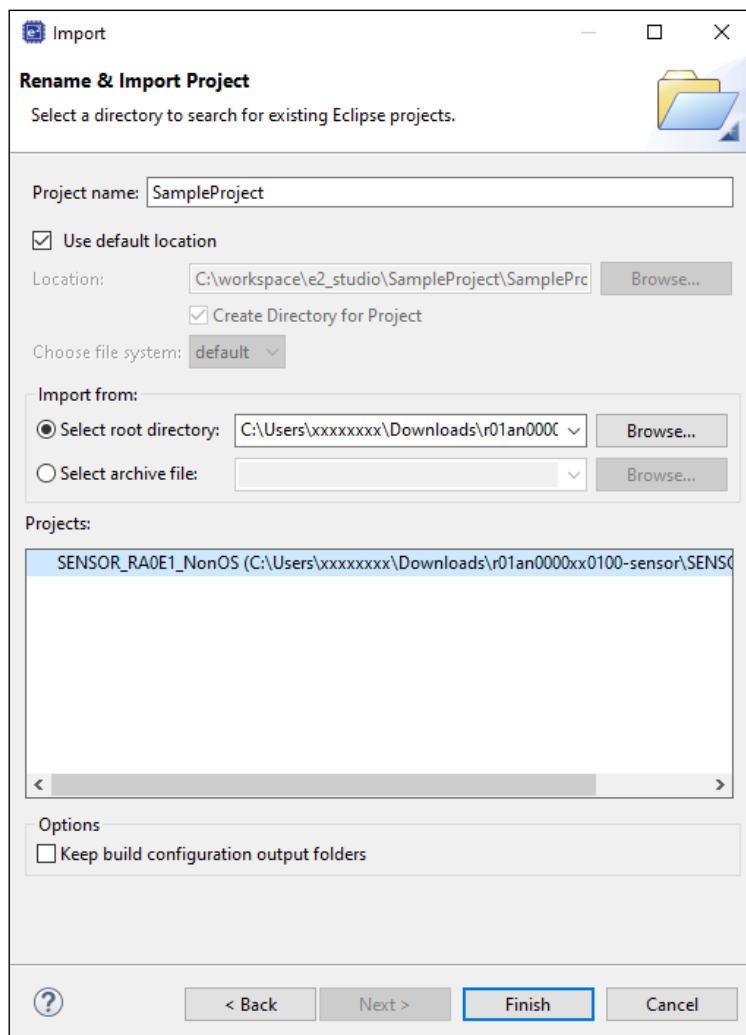


2. Press the [Browse] button to open the "Select Folder" window.

Select the folder of the original project for the current device from a list of imported sample projects and press the [Select Folder] button.



3. Enter the project name, select the original project for the current device, and press the [Finish] button.



## 8.2 RA Sample Project

After importing the sample projects, follow the steps below. Please refer to "8.1 Importing Sample Project" for importing instructions.

The following explains the change procedure for the following board change example. In addition, an Interposer Board is required when using a Pmod Type 2A/3A connector.

- Sample Project "ZMOD4XXX\_RA2E1\_NonOS":  
Pmod1 (Type 2A/3A: SCI0)  
→ Pmod1 (Option Type 6A: IIC1) or Pmod2 (Type 2A: SCI0) of the EK-RA6M4 board

### 8.2.1 Modifying Settings of FSP Configurator

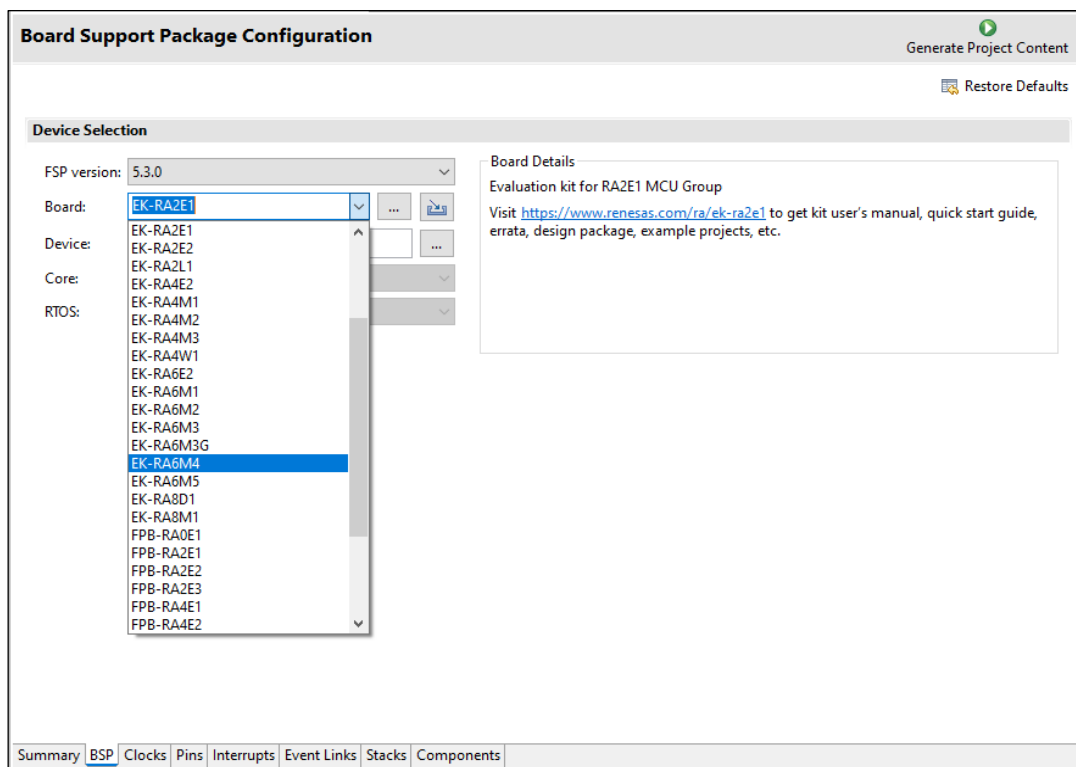
Double-click on "Configuration.xml" in the project tree to open the FSP Configurator.

#### (1) BSP

Change the settings of "Board" and "Device" in the "BSP" tabbed page.

When selecting a Renesas board, modify the "Board" setting only.

When selecting a board provided from other companies, change the "Board" setting to "Custom User Board (Any Device)" and then change the "Device" setting to the new device to be used.

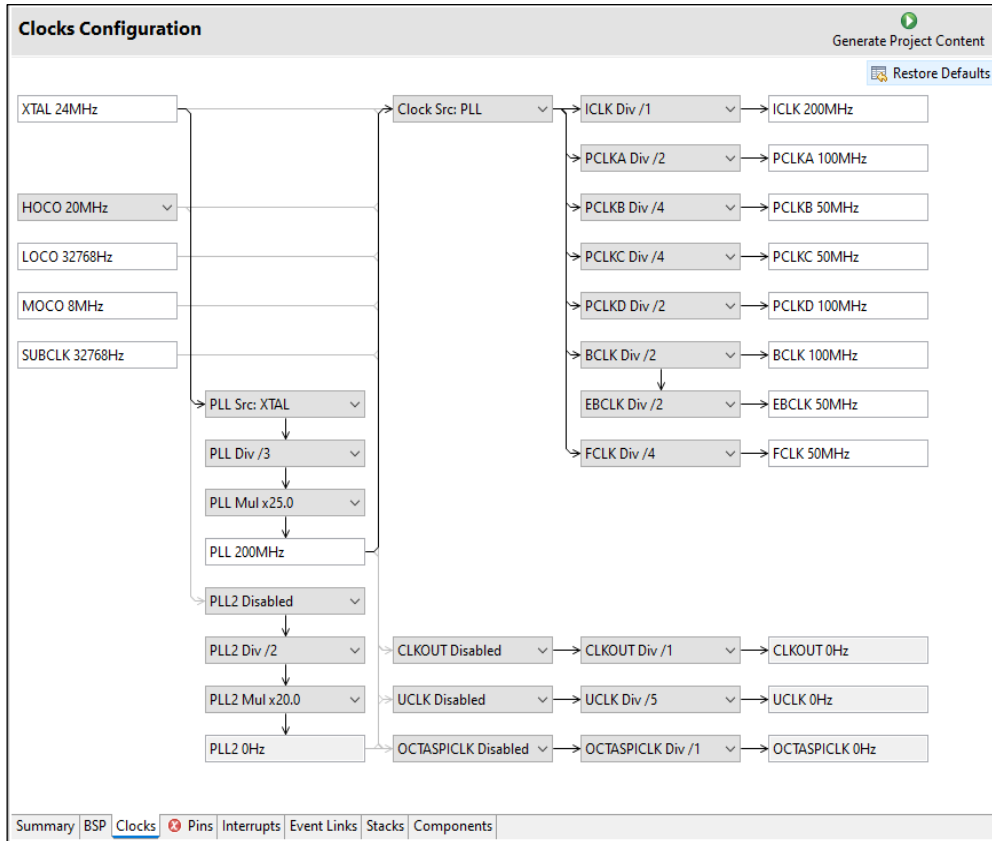


(2) Clocks

Set up the clocks in the "Clocks" tabbed page.

When "Custom User Board (Any Device)" is selected for "Board", set up the clocks according to the specifications of the target board to be used.

When a Renesas board is selected for "Board", the clocks are automatically set up.

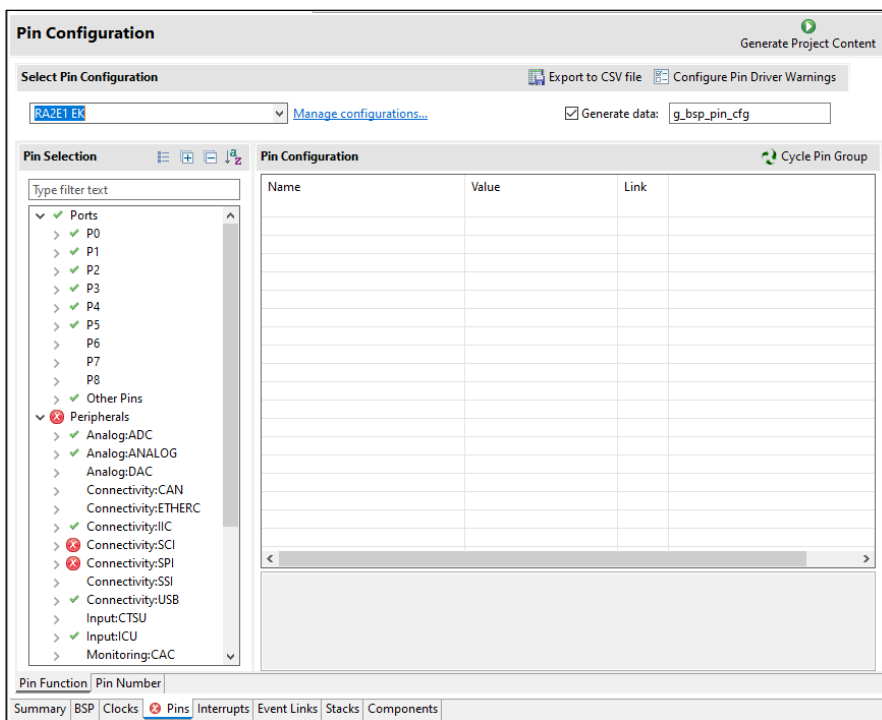


(3) Pins

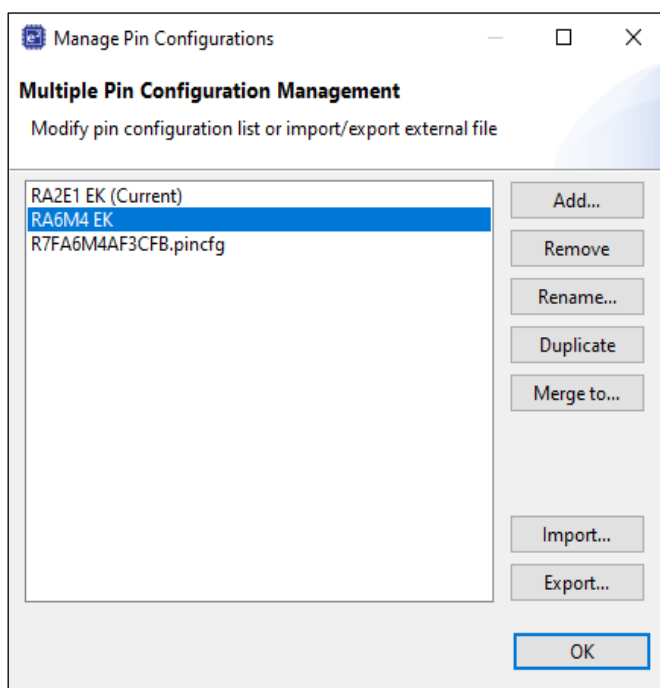
(a) Changing Board

In the “Pins” tabbed page, modify the pin configuration according to the specifications of the target board to be used.

When using a Renesas board, change the selection for "Select Pin Configuration" from "RA2E1 EK" to the target board; appropriate pins are automatically assigned.



If the desired board is not displayed in the drop-down list for "Select Pin Configuration", click on [Manage Configuration] to open the "Manage Pin Configurations" window and select the desired board in the window.



**(b) Changing I2C I/F Pins**

However, the assignment on the above "(a)Changing Board" will apply the SPI communication pin settings that support Pmod Type 2A on the EK-RA6M4 board.

This sample software uses Pmod Type 6A, therefore it is necessary to change the I2C communication pin settings that support Pmod Type 6A.

IIC1 (Pmod1 #3 P512 SCL1 and Pmod1 #4 P511 SDA1) is assigned to Pmod1 and SCI0 (Pmod2 #3 P410 SCL0 and Pmod2 #2 P411 SDA0) is assigned to Pmod2 on the EK-RA6M4 board.

Therefore, the pins used for I2C communication are as follows, so after automatic assignment of "Select Pin Configuration", reconfigure in "Pin Configuration":

- When using Pmod1 (Option Type 6A), set SCL1 to P512 and SDA1 to P511.
- When using Pmod2 (using the Interposer Board), set SCL0 to P410 and SDA0 to P411.

**Pin Configuration** Generate Project Content

Select Pin Configuration Export to CSV file Configure Pin Driver Warnings

RA6M4 EK [Manage configurations...](#)  Generate data:

**Pin Selection**

Type filter text

- > Other Pins
- ✓ Peripherals
  - > Analog:ADC
  - > Analog:ANALOG
  - > Analog:DAC
  - > Connectivity:CAN
  - > Connectivity:ETHERC
  - > Connectivity:IIC
  - ✓ Connectivity:SCI
    - ✓ SCI0
    - SCI1
    - SCI2
    - SCI3
    - SCI4
    - SCI5
    - ✓ SCI6
    - ✓ SCI7
    - SCI8
    - SCI9

**Pin Configuration** Cycle Pin Group

Name	Value	Lock	Link
Pin Group Selection	Mixed		
Operation Mode	Simple I2C		
Input/Output			
TXD0	None		
RXD0	None		
SCK0	None		
CTS0	None		
SDA0	✓ P411		
SCL0	✓ P410		
CTSRTS0	None		

Module name: SCI0

Usage: When using Simple I2C mode, ensure port pins output type is n-ch open drain.  
When switching between I2C and other modes, first disable.

Pin Function Pin Number

Summary BSP Clocks Pins Interrupts Event Links Stacks Components

**(c) Changing IRQ Pin**

For information on configuring the pin of the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

Set the IRQ pin to match the interrupt signal pin of the sensor.

P301 IRQ06 is assigned to Pmod1 #1 and P414 IRQ09 is assigned to Pmod2 #7 on the EK-RA6M4 board.

Therefore, set the IRQ pin as follows:

- When using Pmod1 (Option Type 6A), set IRQ06 pin to P301.
- When using Pmod2 (using the Interposer Board), set IRQ09 pin to P414.

**Pin Configuration** Generate Project Content

Select Pin Configuration Export to CSV file Configure Pin Driver Warnings

RA6M4 EK Manage configurations...  Generate data: g\_bsp\_pin\_cfg\_6m4

Pin Selection		Pin Configuration			
Type filter text		Name	Value	Lock	Link
> ✓ Other Pins		IRQ00	None		
✓ Peripherals		IRQ01	None		
> ✓ Analog:ADC		IRQ02	None		
> ✓ Analog:ANALOG		IRQ03	None		
> Analog:DAC		IRQ04	None		
> Connectivity:CAN		IRQ05	None		
> ✓ Connectivity:ETHERC		IRQ06	✓ P409		
> ✓ Connectivity:IIC		IRQ07	None		
> ✓ Connectivity:SCI		IRQ08	✓ P002		
> ✓ Connectivity:SPI		IRQ09	✓ P414		
> Connectivity:SSI		IRQ10	✓ P005		
> ✓ Connectivity:USB		IRQ11	✓ P006		
> Input:CTSU		IRQ12	✓ P008		
✓ Input:ICU		IRQ13	None		
✓ ICU0		IRQ14	None		
> Monitoring:CAC					
> ✓ Storage:OSPI					
> ✓ Storage:QSPI					
> Storage:SDHI					

Module name: ICU0  
Usage: To use IRQ function with output or peripheral modes, change directly in port dialog

Pin Function | Pin Number | Summary | BSP | Clocks | Pins | Interrupts | Event Links | Stacks | Components

**(d) Changing General Purpose I/O Port Pin: RESET**

For information on configuring the pin of the RESET signal circuit, refer to “8.6 Notes for RESET Signal Circuits”.

Set the RESET pin to match the RESET signal pin of the sensor.

P203 Port is assigned to Pmod1 #2 and P412 Port is assigned to Pmod2 #4 on the EK-RA6M4 board.

Therefore, set the RESET pin as follows:

- When using Pmod1 (Option Type 6A), set P203 to GPIO Port Output pin.
  - When using Pmod2 (using the Interposer Board), set P412 to GPIO Port Output pin.
- For both, select "Output mode (Initial High)" as "Mode".

The screenshot shows the 'Pin Configuration' interface for the RA6M4 EK device. The 'Pin Selection' pane on the left lists pins P2 through P415, with P412 selected. The 'Pin Configuration' pane on the right shows the following settings:

Name	Value	Link
Symbolic Name	PMOD2_CLK	
Comment		
Mode	Output mode (Initial High)	
Pull up	None	
Drive Capacity	Low	
Output type	CMOS	
Input/Output		
P412	GPIO	→

Below the table, the module name is P412 and port capabilities include AGT1: AGTEE1, CTSU0: TS08, and ETHERC\_MII0: ET0\_ETXD0. The 'Generate data' checkbox is checked, and the filename is g\_bsp\_pin\_cfg\_6m4.

When you change the device, "Generate data" will be disabled. The next page explains how to enable it.

This screenshot shows the same 'Pin Configuration' interface, but the 'Generate data' checkbox is now unchecked and highlighted with a red box. The rest of the configuration for P412 remains the same.





(4) Stacks

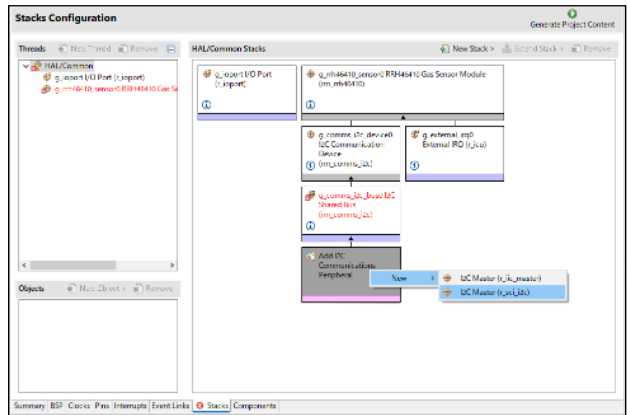
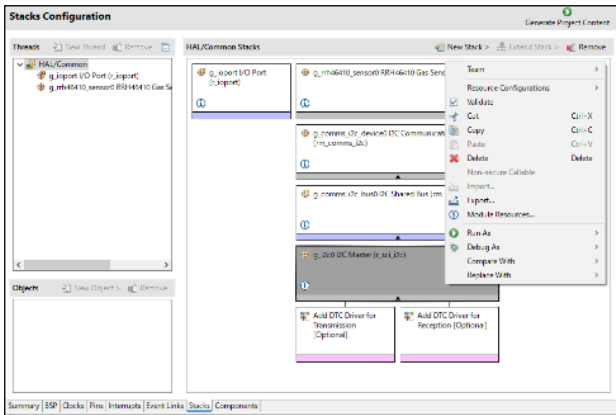
Modify the configuration of individual components in the "Stacks" tabbed page.

(a) Changing COMMS\_I2C Settings and I2C Driver Settings

Modify the settings of COMMS\_I2C and I2C driver according to the specifications of the target board. To use the pins of the I2C I/F, delete the unnecessary stack and add the new stack to use.

Table 8-1 Settings of I2C I/F and Channel for EK-RA6M4

EK-RA6M4	I2C I/F	g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)	g_i2c_master0 I2C Master
Pmod1 Option Type 6A	IIC1	Channel: 1	Check Pins
Pmod2 Type 2A	SCI0	Channel: 0	Check Pins



g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)		
Settings	Property	Value
API Info	Common	
	Parameter Checking	Default (BSP)
	Module g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)	
	Name	g_comms_i2c_bus0
	Bus Timeout	0xFFFFFFFF
	Semaphore for Blocking (RTOS only)	Use
	Recursive Mutex for Bus (RTOS only)	Use
Channel	1	
Rate	Standard	

g_i2c_master0 I2C Master (r_iic_master)		
Settings	Property	Value
API Info	Common	
	Parameter Checking	Default (BSP)
	DTC on Transmission and Reception	Disabled
	10-bit slave addressing	Disabled
	Module g_i2c_master0 I2C Master (r_iic_master)	
	Name	g_i2c_master0
	Channel	1
	Rate	Standard
	Custom Rate (bps)	0
	Rise Time (ns)	120
	Fall Time (ns)	120
	Duty Cycle (%)	50
	Slave Address	0x00
	Address Mode	7-Bit
	Timeout Mode	Short Mode
	Timeout during SCL Low	Enabled
	Callback	rm_comms_i2c_callback
	Interrupt Priority Level	Priority 12
Pins		
SDA1	P511	
SCL1	P512	

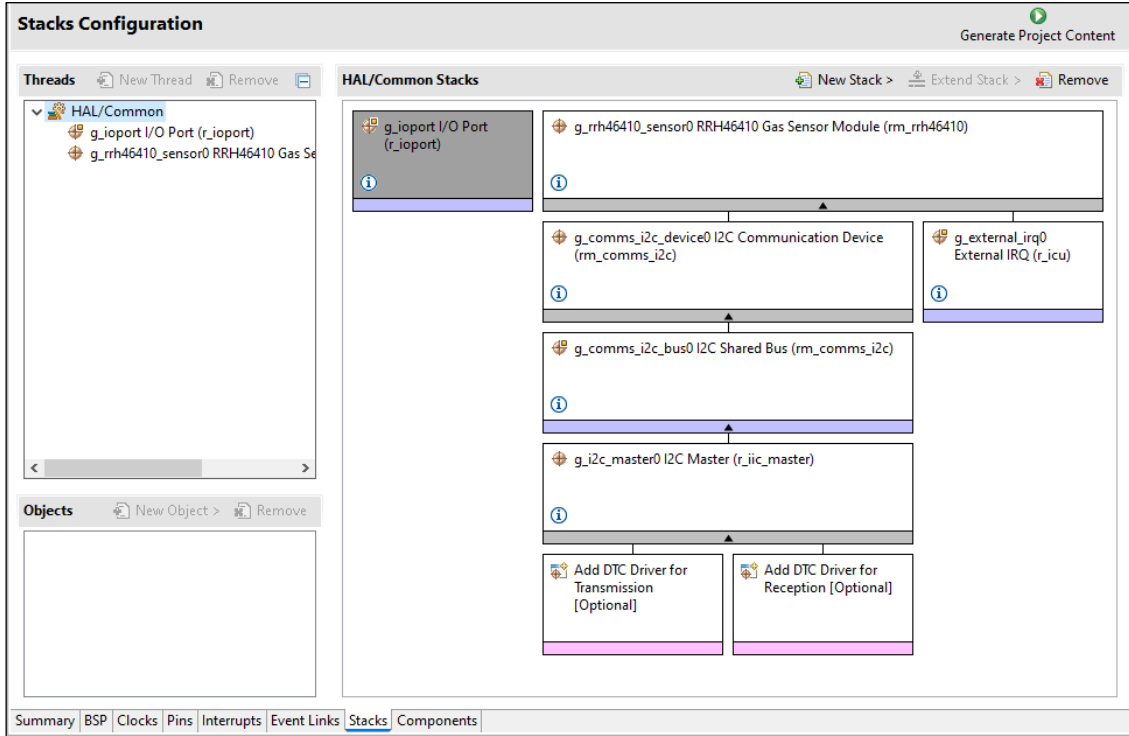
g_i2c0 I2C Master (r_sci_i2c)		
Settings	Property	Value
API Info	Common	
	Parameter Checking	Default (BSP)
	DTC on Transmission and Rec	Disabled
	10-bit slave addressing	Disabled
	Module g_i2c0 I2C Master (r_sci_i2c)	
	Name	g_i2c0
	Channel	0
	Slave Address	0x00
	Address Mode	7-Bit
	Rate	Standard
	Custom Rate (bps)	0
	SDA Output Delay (nano sec)	300
	Noise filter setting	Use clock signal divided by 1 with noise filter
	Bit Rate Modulation	Enable
	Callback	rm_comms_i2c_callback
	Interrupt Priority Level	Priority 2
	RX Interrupt Priority Level [O]	Disabled
	Pins	
SDA0	P411	
SCL0	P410	

**(b) Changing General Purpose I/O Port Driver Settings: RESET**

For information on how to modify the source of the RESET signal control, refer to “8.2.2(1) RESET Signal Control”.

Enter the pin configuration name to use in "Pin Configuration Name" of "g\_ioport I/O Port".

The following is an example named “g\_bsp\_pin\_cfg\_6m4”.



g_ioport I/O Port (r_ioport)		
Settings	Property	Value
API Info	Common	
	Parameter Checking	Default (BSP)
API Info	Module g_ioport I/O Port (r_ioport)	
	Name	g_ioport
	1st Port ELC Trigger Source	Disabled
	2nd Port ELC Trigger Source	Disabled
	3rd Port ELC Trigger Source	Disabled
	4th Port ELC Trigger Source	Disabled
	Pin Configuration Name	g_bsp_pin_cfg_6m4
	Pins	
	TCK	P300
	TDI	P110
	TDO	P109
	TMS	P108
	SWCLK	<unavailable>
	SWDIO	<unavailable>
	TRACESWO	<unavailable>
	TCLK	<unavailable>
	TDATA0	<unavailable>
	TDATA1	<unavailable>
	TDATA2	<unavailable>

**(c) Changing IRQ Driver Settings**

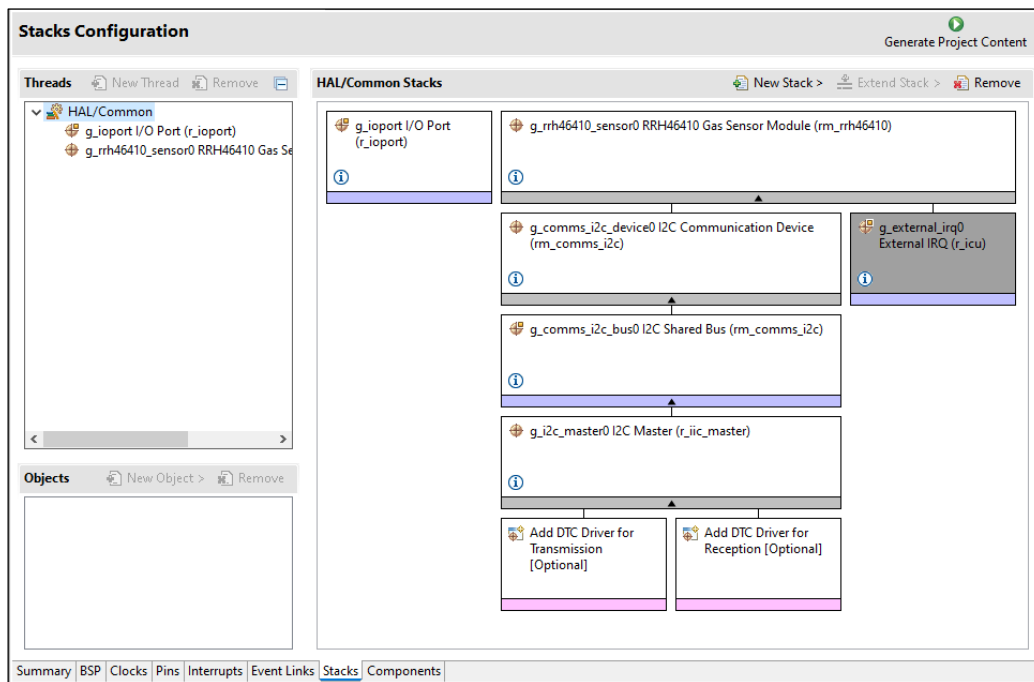
For information on matching the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

Modify the settings of r\_icu according to the specifications of the target board.

P301 IRQ6 is assigned to Pmod1 #1 and P414 IRQ9 is assigned to Pmod2 #7 on the EK-RA6M4 board.

Therefore, set IRQ channel as follows:

- When using Pmod1 (Option Type 6A), set “Channel” to 6.
- When using Pmod2 (using the Interposer Board), set “Channel” to 9.



g_external_irq0 External IRQ (r_icu)		
Settings	Property	Value
API Info	Common	
	Parameter Checking	Default (BSP)
	Module g_external_irq0 External IRQ (r_icu)	
	Name	g_external_irq0
	Channel	9
	Trigger	Falling
	Digital Filtering	Disabled
	Digital Filtering Sample Clock (Only valid when Digital Filtering is Enabled)	PCLK / 64
	Callback	rm_rrh46410_irq_callback
	Pin Interrupt Priority	Priority 2
	Pins	
	IRQ09	P414

If an error is displayed in other stacks, modify the specified item according to the displayed error.

**(5) Code Generation and Build**

After modifications are finished, press [Generate Project Content] to generate files.

Build the project after implementing “8.2.2 Changing Sample Code”

Select [Debug Configurations] from the menu and modify the debugger settings according to the specifications of the emulator to be connected to the target board.

## 8.2.2 Changing Sample Code

### (1) RESET Signal Control

For pin configuration, refer to "8.2.1(3)(d) Changing General Purpose I/O Port Pin: RESET"

Open "hal\_entry.c" (Non-OS) or "zmod4xxx\_sensor\_thread\_entry.c" (FreeRTOS) and change the reset operation when resetting the sensor. Also, change the reset control logic to an appropriate one.

Modify RESET pin designation according to the specifications of the target board.

With the above settings, the EK-RA6M4 board is configured as follows:

- When using Pmod1 (Option Type 6A), RESET pin is assigned to P203.
- When using Pmod2 (using the Interposer Board), RESET pin is assigned to P412.

```

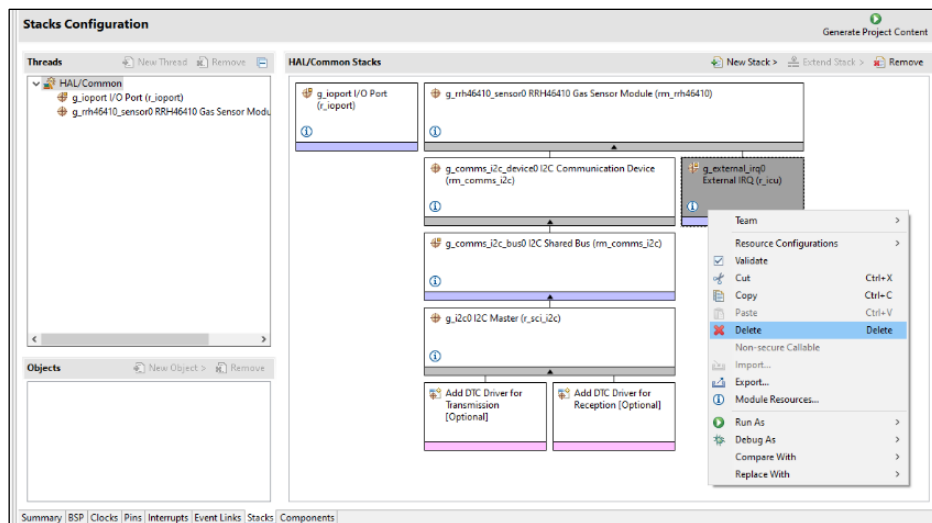
/* Reset ZMOD sensor (active low). Please change to the IO port connected to the
RES_N pin of the ZMOD sensor on the customer board. */
R_IOPORT_PinWrite(&g_ioport_ctrl, BSP_IO_PORT_04_PIN_12, BSP_IO_LEVEL_HIGH);
R_BSP_SoftwareDelay(10, BSP_DELAY_UNITS_MILLISECONDS);
R_IOPORT_PinWrite(&g_ioport_ctrl, BSP_IO_PORT_04_PIN_12, BSP_IO_LEVEL_LOW);
R_BSP_SoftwareDelay(10, BSP_DELAY_UNITS_MILLISECONDS);
R_IOPORT_PinWrite(&g_ioport_ctrl, BSP_IO_PORT_04_PIN_12, BSP_IO_LEVEL_HIGH);
R_BSP_SoftwareDelay(10, BSP_DELAY_UNITS_MILLISECONDS);

```

### 8.2.3 Changing when not using IRQ

If IRQ is not used, delete stack, and change the values of constants defined.

In the "Stacks" tabbed page, delete "g\_external\_irq0 External IRQ".



Open "RA\_ZMOD4XXX.c" (Non-OS) or "zmod4xxx\_sensor\_thread\_entry.c" (FreeRTOS) and change the value of "G\_ZMOD4XXX\_SENSOR0\_IRQ\_ENABLE" to "0".

```

/* TODO: Enable if you want to open ZMOD4XXX */
#define G_ZMOD4XXX_SENSOR0_IRQ_ENABLE (0)

```

### 8.2.4 Changing Toolchain Setting

If you want to use a toolchain other than the GCC ARM Embedded toolchain, copy RA\_ZMOD4XXX.c (Non-OS) or zmod4xxx\_sensor\_thread\_entry.c, sensor\_thread\_common.c and sensor\_thread\_common.h (FreeRTOS) from this project to create a new project.

### 8.3 RX Sample Project

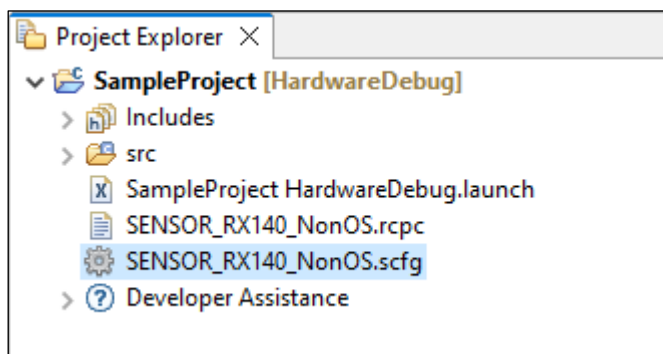
After importing the sample projects, follow the steps below. Please refer to “8.1 Importing Sample Project” for importing instructions.

The following explains the change procedure for the following board change example. In addition, an Interposer Board is required when using a Pmod Type 2A/3A connector.

- Sample project "ZMOD4XXX\_RX140\_NonOS":  
 Pmod1 (Type2A: SCI5)  
 → Pmod1 (Option Type 6A: RIIC0) or Pmod2 (Option Type6A: SCI11) of the EK-RX671 board

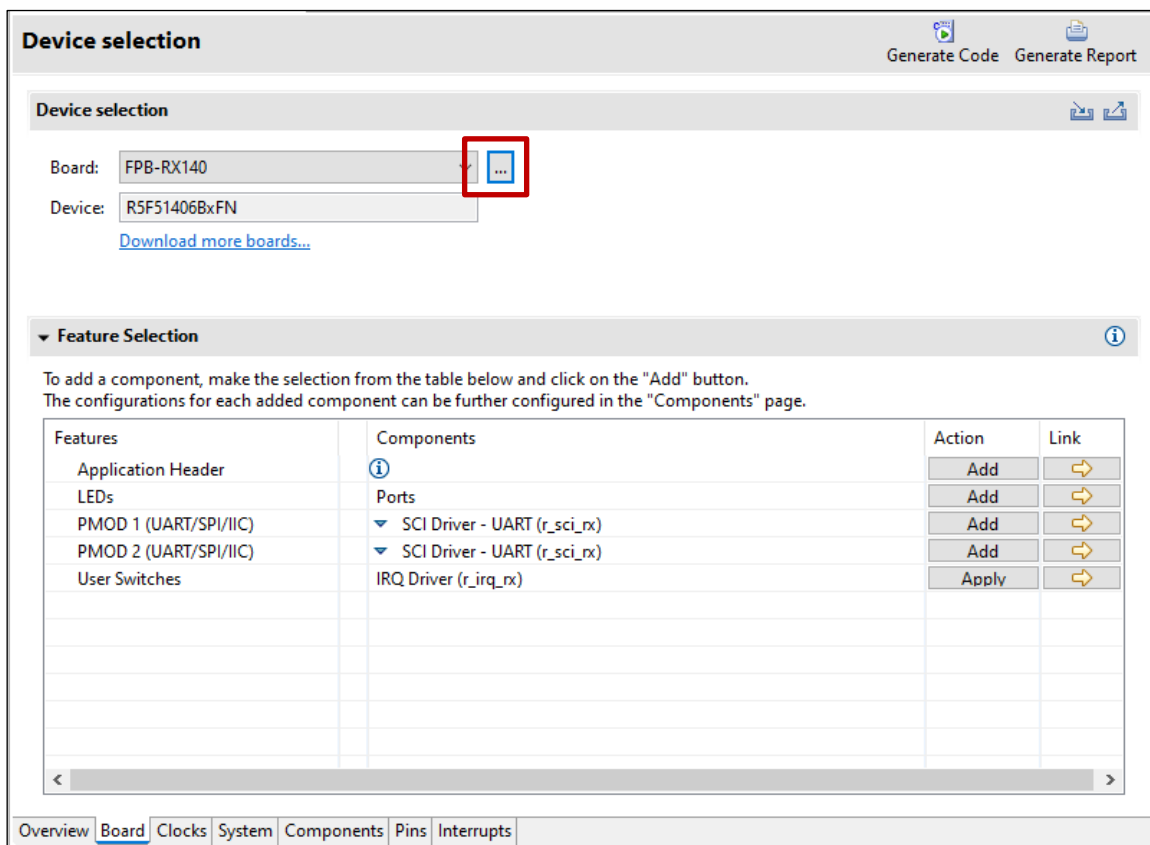
#### 8.3.1 Modifying Settings of Smart Configurator

On the project tree, double-click on the .scfg file of the imported project in the Smart Configurator window will open.

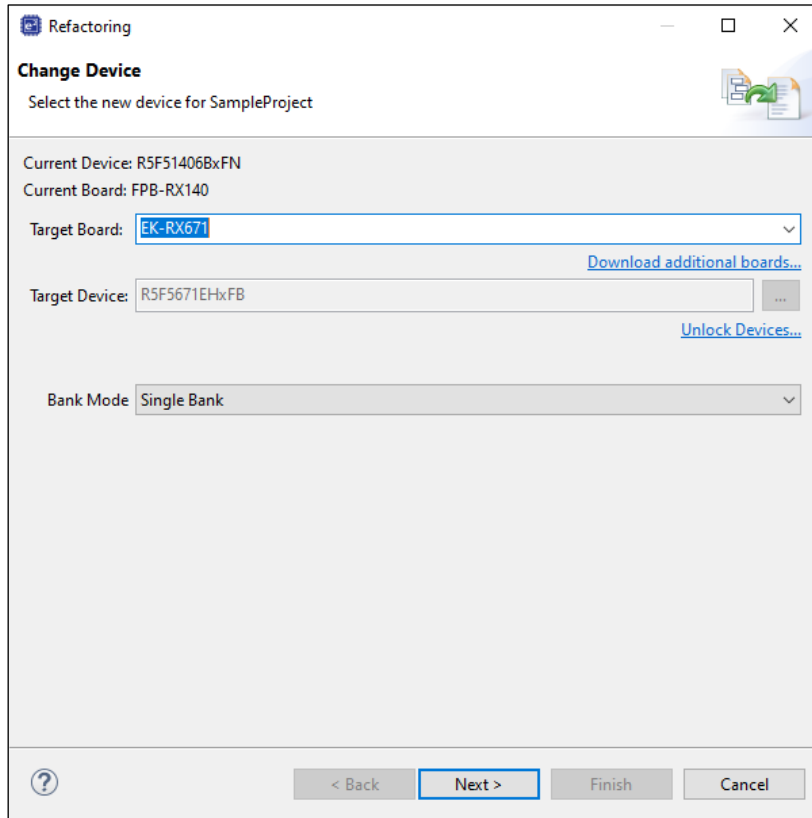


#### (1) Board

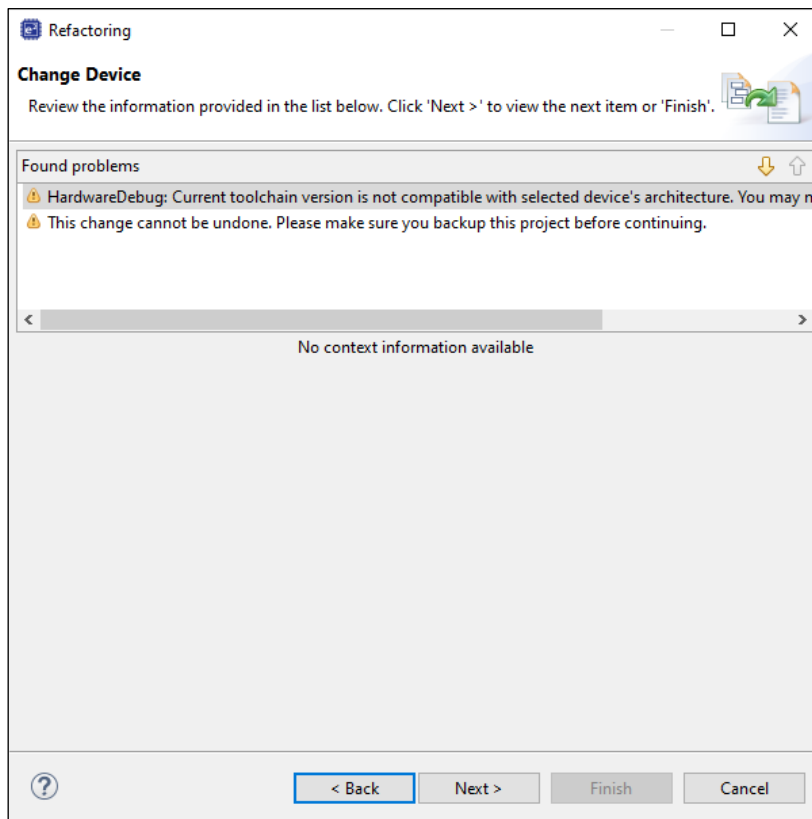
1. On the Board tab, click the [...] button.



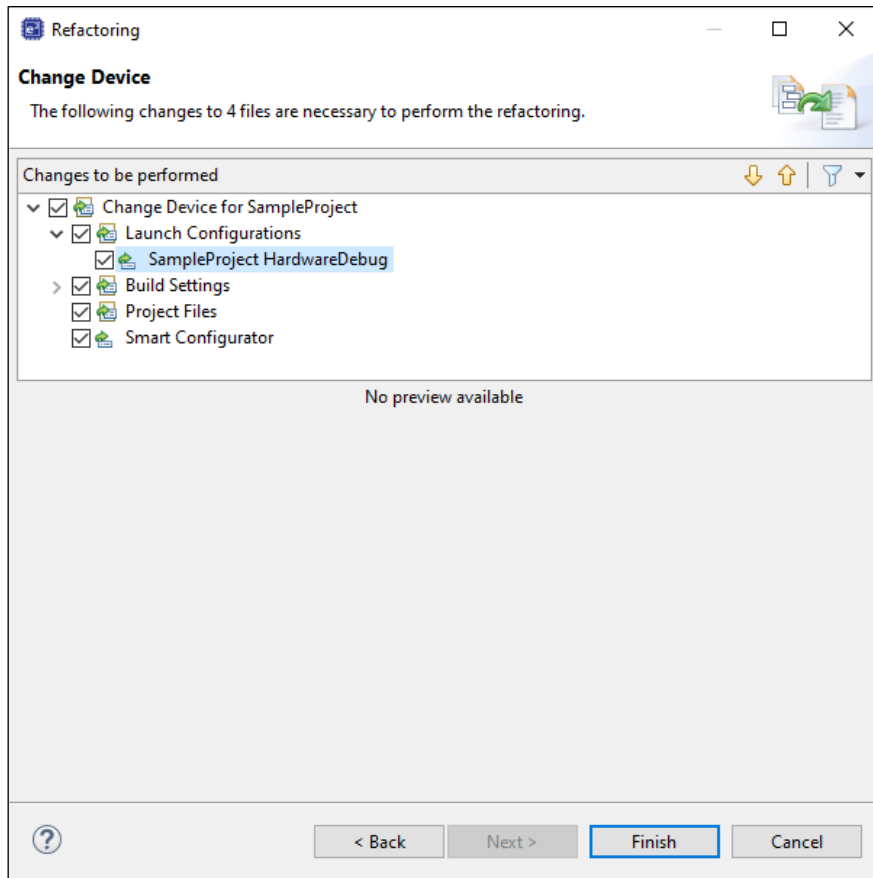
2. Select a desired board or device in the "Change Device" window and press the [Next] button.



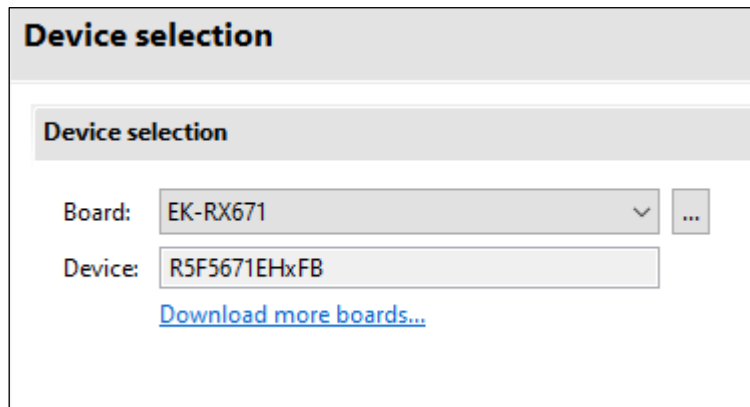
3. If a warning message appears, read it and check if there is a problem in proceeding with the procedure. Press [Next] to move to the next step.



- The changes you have made in the settings will be displayed. Press the [Finish] button to apply the changes to the project.



- Select the "Board" tabbed page to check that the board and device have been changed correctly.





(2) Clocks

Set up the clocks in the "Clocks" tabbed page.

When "Custom User Board (Any Device)" is selected for "Board", set up the clocks according to the specifications of the target board to be used.

When a Renesas board is selected for "Board", the clocks are automatically set up.

The screenshot displays the 'Clocks configuration' window with the following settings and components:

- VCC:** 3.3 (V) (Actual value: 3.3)
- Main clock:**
  - Oscillation source: Resonator
  - Frequency: 24 (MHz)
  - Oscillation wait time: 9980 (µs) (Actual value: 10000)
- Sub-clock:**
  - Frequency: 32.768 (kHz)
  - Oscillator drive capacity: Standard CL
  - Oscillation wait time: 2000 (ms) (Actual value: 2047.939)
- HOCO clock:**
  - Enable HOCO oscillation after reset:
  - Frequency: 16 (MHz)
  - Enable PLL function:
- LOCO clock:**
  - Frequency: 240 (kHz)
- IWDT-dedicated clock:**
  - Frequency: 120 (kHz)
- PLL circuit:**
  - Frequency Division: x1
  - Frequency Multiplication: x10.0
- Other Clocks and Dividers:**
  - SCKCR (FCK[3:0]): x1/4
  - SCKCR (ICK[3:0]): x1/2
  - SCKCR (PCKA[3:0]): x1/2
  - SCKCR (PCKB[3:0]): x1/4
  - SCKCR (PCKC[3:0]): x1/4
  - SCKCR (PCKD[3:0]): x1/4
  - SCKCR (BCK[3:0]): x1/4
  - SCKCR2 (LCK[3:0]): x1/5
  - CKOCR (CKODIV[2:0]): x1/8
- Output Clocks:**
  - FlashF clock (FCLK): 60 (MHz)
  - System clock (ICLK): 120 (MHz)
  - Peripheral module clock (PCLKA): 120 (MHz)
  - Peripheral module clock (PCLKB): 60 (MHz)
  - Peripheral module clock (PCLKC): 60 (MHz)
  - Peripheral module clock (PCLKD): 60 (MHz)
  - External bus clock (BCLK): 60 (MHz)
  - External bus clock pin (BCLK pin): - (MHz)
  - SDRAM clock (SDCLK): - (MHz)
  - USB clock (UCLK): 48 (MHz)
  - CLKOUT pin: - (MHz)
  - CANMCLK/CACMCLK: 24 (MHz)
  - CACLCLK: - (kHz)
  - CACHCLK: - (MHz)
  - IWDTCLK/CACILCLK: - (kHz)
  - CACSCCLK: 32.768 (kHz)
  - REMSCLK: 32.768 (kHz)
  - VBATCLK: 32.768 (kHz)
  - RTCSCLK: 32.768 (kHz)

The interface includes a block diagram showing the interconnections between these clock sources and dividers. At the bottom, there is a navigation bar with tabs: Overview, Board, Clocks, System, Components, Pins, Interrupts.

### (3) Components

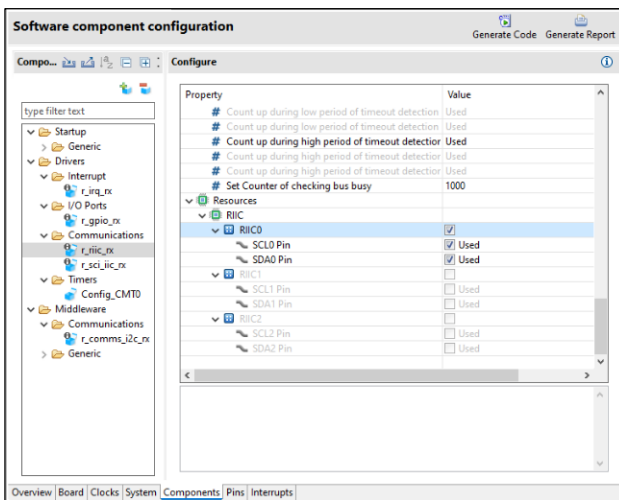
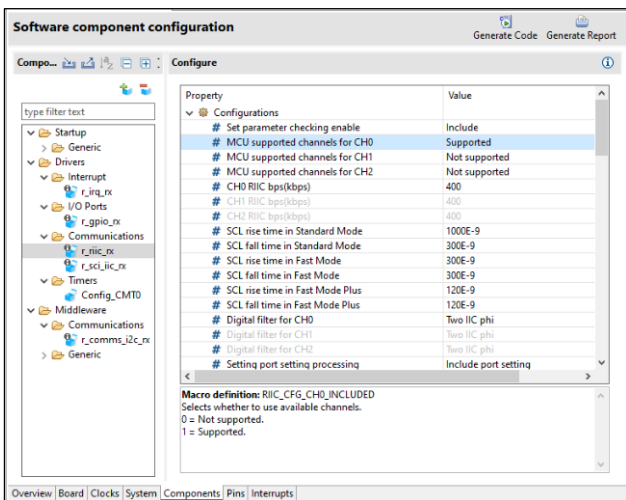
Modify the settings of individual components in the "Components" tabbed page according to the specifications of the target board.

#### (a) Changing I2C Driver Settings

RIIC0 is assigned to Pmod1 and SCI11 to Pmod2 on the EK-RX671 board.

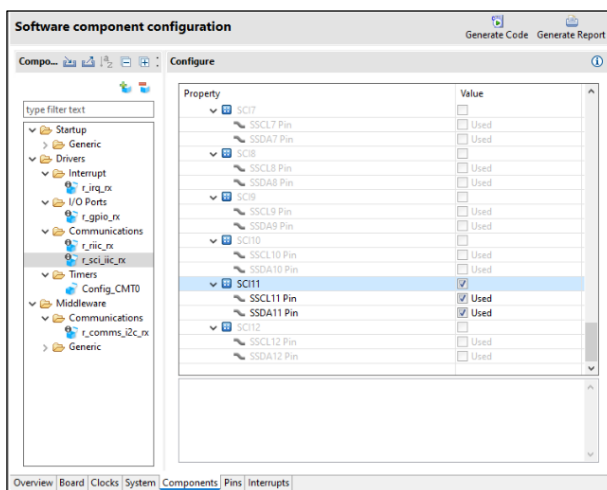
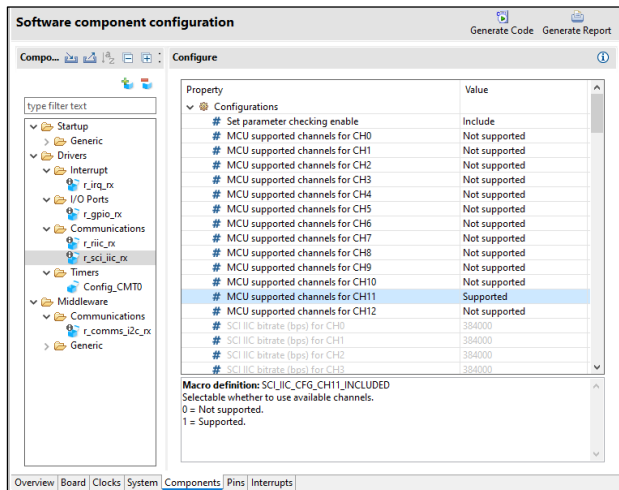
When using Pmod1 (Option Type 6A), set it as follows:

- For r\_riic\_rx, set "MCU supported channels for CH0" to "Supported".
- For r\_riic\_rx, add the check settings of "RIIC0", "SCL0 Pin" and "SDA0 Pin" under "Resources".



When using Pmod2 (Option Type 6A), set it as follows:

- For r\_sci\_iic\_rx, set "MCU supported channels for CH5" to "Not supported", and set "MCU supported channels for CH11" to "Supported".
- For r\_sci\_iic\_rx, add the check settings of "SCI11", "SSCL11 Pin" and "SSDA11 Pin" under "Resources".

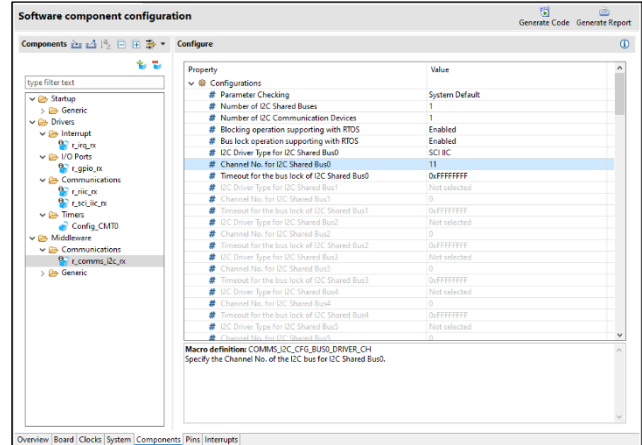
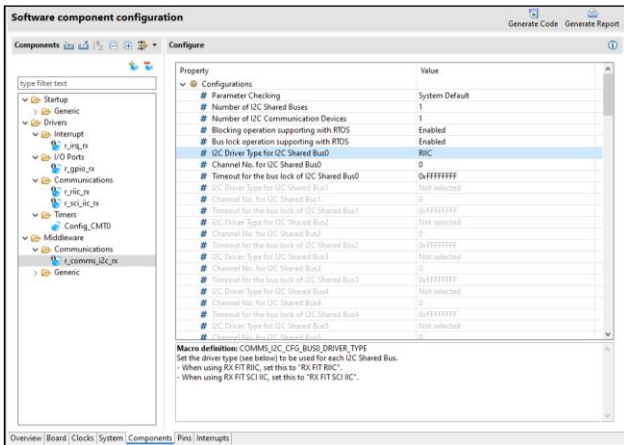


**(b) Changing COMMS\_I2C Settings**

If you have changed the I2C driver or channel, you will need to change these settings.

Set “I2C Driver Type for I2C Shared BusX” (X: Bus No.) and “Channel No. for I2C Shared BusX” (X: Bus No.) in `r_comms_i2c_rx` as follows:

- When using Pmod1 (Option Type 6A), set “I2C Driver Type for I2C Shared Bus0” to “RIIC”, set “Channel No. for I2C Shared Bus0” to “0”.
- When using Pmod2 (Option Type 6A), set “I2C Driver Type for I2C Shared Bus0” to “SCI\_IIC”, set “Channel No. for I2C Shared Bus0” to “11”.



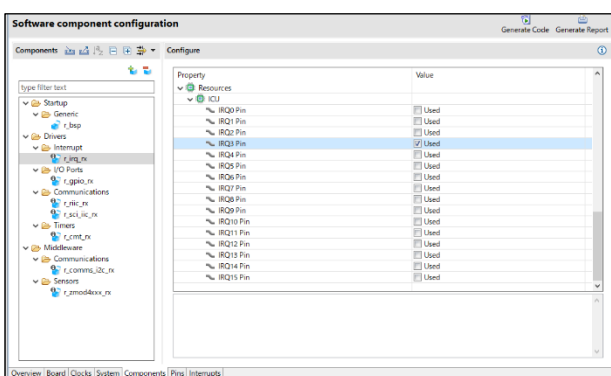
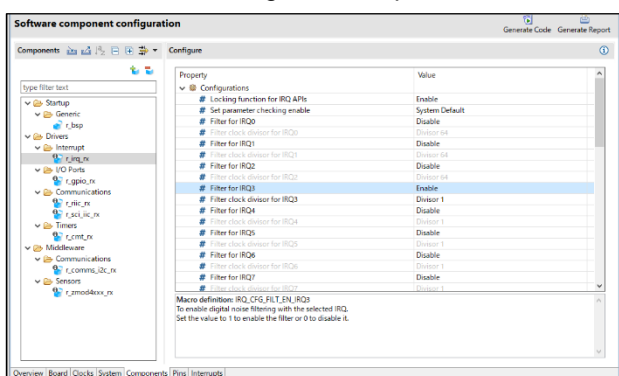
**(c) Changing IRQ Driver Settings**

For information on matching the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

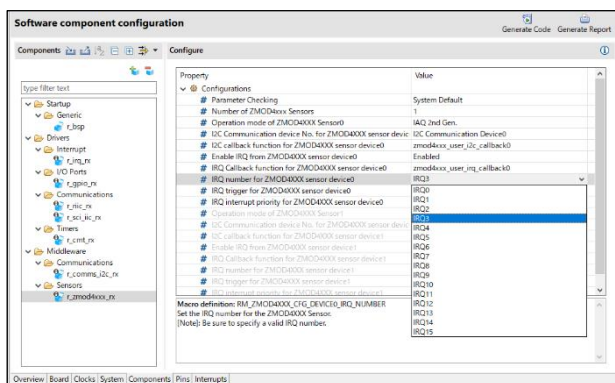
P83 IRQ3 is assigned to Pmod1 #1 and P74 IRQ12 to Pmod2 #1 on the EK-RX671 board.

Therefore, set IRQ channel in “r\_irq\_rx” as follows and disable any unnecessary settings, also:

- When using Pmod1 (Option Type 6A), set “Filter of IRQ4” to “Disable”, set “Filter of IRQ3” to “Enable”, remove the check setting of “IRQ4 pin” under “Resources”, add the check setting of “IRQ3 pin” under “Resources”.
- When using Pmod2 (Option Type 6A), set “Filter of IRQ4” to “Disable”, set “Filter of IRQ12” to “Enable”, remove the check setting of “IRQ4 pin” under “Resources”, add the check setting of “IRQ3 pin” under “Resources”.



Change the settings of “IRQ number for ZMOD4XXX sensor device0” to “IRQ3” or “IRQ12” in r\_zmod4xxx\_rx.



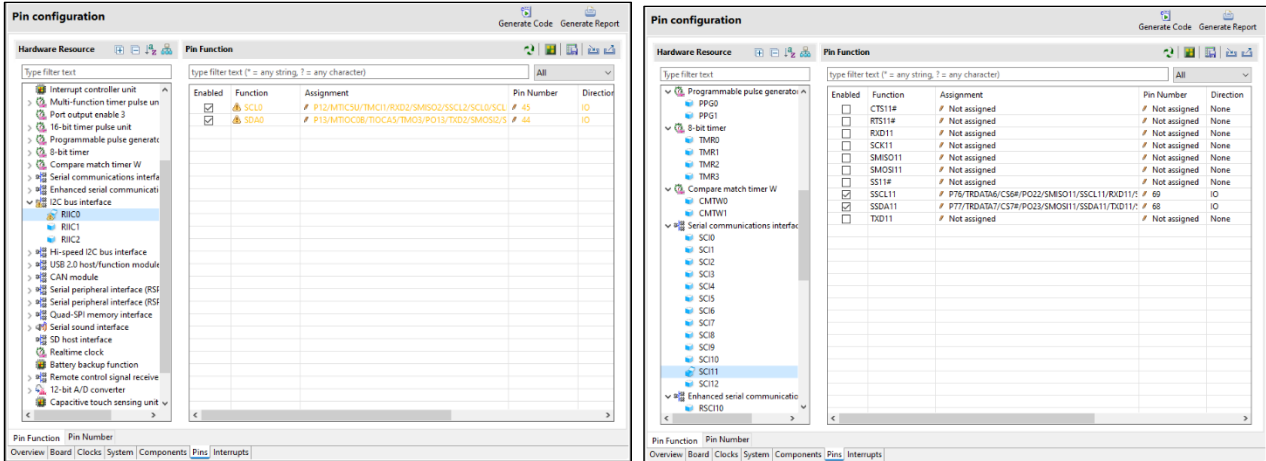
(4) Pins

(a) Changing I2C I/F Pins

RIIC0 is assigned to Pmod1 and SCI11 to Pmod2 on the EK-RX671 board.

Therefore, set the pins used for I2C communication in “Pin Function” on “Pin” tabbed page as follows:

- When using Pmod1 (Option Type 6A): Enable RIIC0, P12 SCL0 and P13 SDA0.
- When using Pmod2 (Option Type 6A): Enable SCI11, P76 SSCL11 and PB7 SDA11.



As the use of Pmod1 at “High-Speed I2C Bus Interface (RIICHS)” is specified in the EK-RX671 board information, a warning message will appear when RIIC is used, but this does not produce any problems.

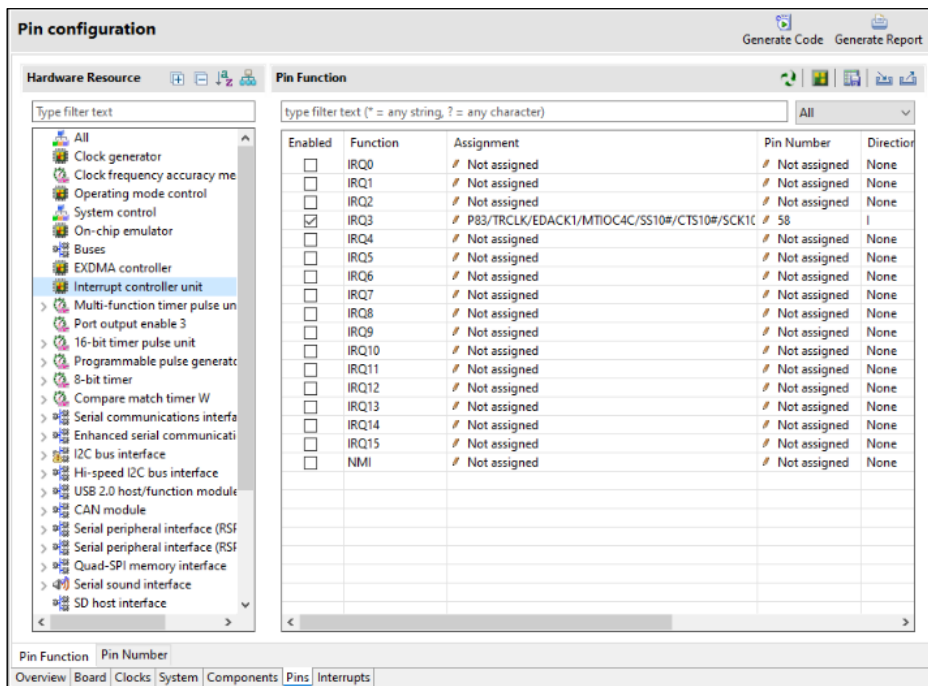
(b) Changing IRQ Pin

For information on configuring the pin of the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

P83 IRQ3 is assigned to Pmod1 #1 and P74 IRQ12 to Pmod2 #1 on the EK-RX671 board.

Therefore, select “Interrupt controller unit” in “Pin Function” on “Pin” tabbed page and set pins as follows:

- When using Pmod1 (Option Type 6A): Enable IRQ3 and set the IRQ3 pin to P83.
- When using Pmod2 (Option Type 6A): Enable IRQ12 and set the IRQ12 pin to P74.



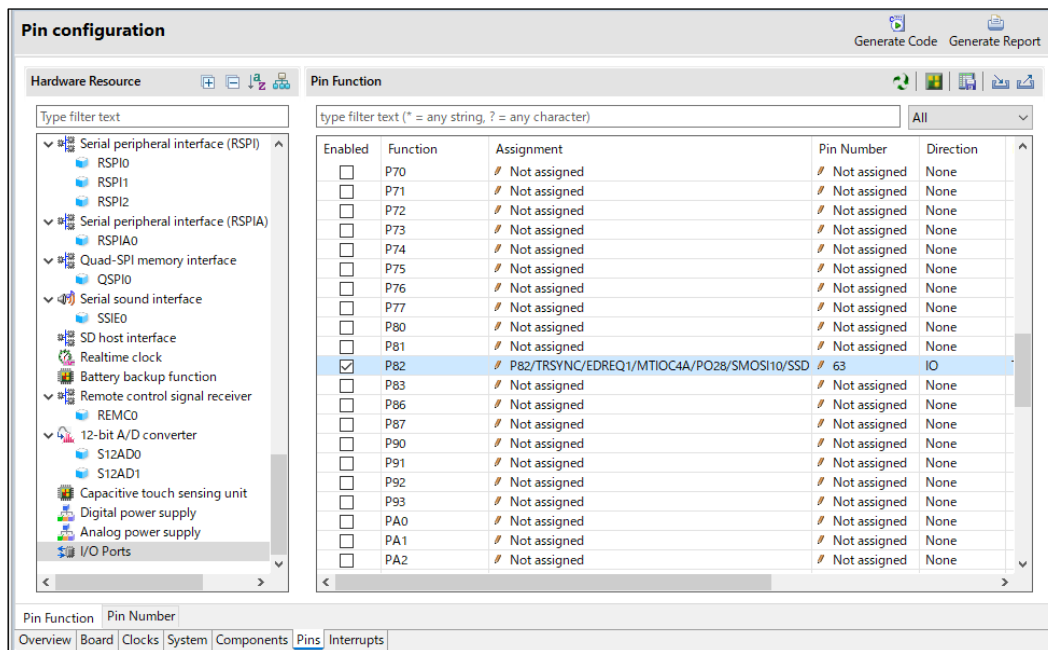
**(c) Changing General Purpose I/O Port Pin: RESET**

For information on configuring the pin of the RESET signal circuit, refer to “8.6 Notes for RESET Signal Circuits”.

The RESET pin is assigned to P82 Port for Pmod1 #2 and P77 Port for Pmod2 #2 on the EK-RX671 board.

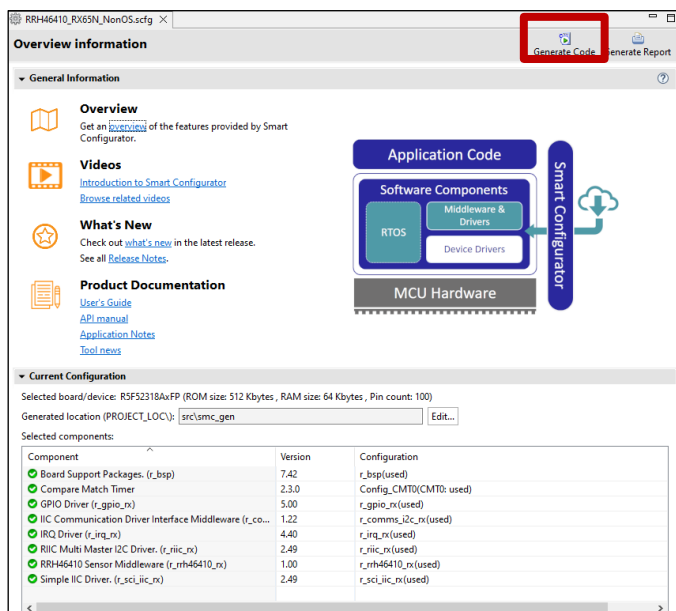
Therefore, select “I/O Ports” in “Pin Function” on “Pin” tabbed page and set pins as follows:

- When using Pmod1 (Option Type 6A): Enable P82.
- When using Pmod2 (Option Type 6A): Enable P77.



**(5) Code Generation and Build**

Press the [Generate Code] icon to generate code.



Build the project after implementing “8.3.2 Changing Sample Code”.

Select [Debug Configurations] from the menu and modify the debugger settings according to the specifications of the emulator to be connected to the target board.

### 8.3.2 Changing Sample Code

#### (1) RESET Signal Control

For pin configuration, refer to “8.3.1(3)(c) Changing General Purpose I/O Port Pin: RESET”.

Open main.c and change the reset operation when resetting the sensor. Also, change the reset control logic to an appropriate one.

Modify RESET pin designation according to the specifications of the target board.

With the above settings, the EK-RX671 board is configured as follows:

- When using Pmod1 (Option Type 6A), RESET pin is assigned to P82.
- When using Pmod2 (Option Type 6A), RESET pin is assigned to P77.

```

/* Reset ZMOD sensor (active low). Please change to the IO port
connected to the RES_N pin of the ZMOD sensor on the customer board. */
R_GPIO_PinWrite(GPIO_PORT_8_PIN_2, GPIO_LEVEL_HIGH);
R_GPIO_PinDirectionSet(GPIO_PORT_8_PIN_2, GPIO_DIRECTION_OUTPUT);
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);
R_GPIO_PinWrite(GPIO_PORT_8_PIN_2, GPIO_LEVEL_LOW);
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);
R_GPIO_PinWrite(GPIO_PORT_8_PIN_2, GPIO_LEVEL_HIGH);
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);

```

### 8.3.3 Changing Toolchain Setting

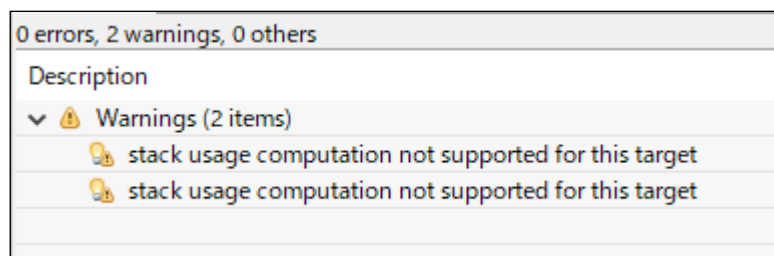
If you want to use a toolchain other than the CC-RX toolchain, copy main.c and RX\_ZMOD4XXX.c (Non-OS), or main.c and zmod4xxx\_sensor\_thread\_entry.c (FreeRTOS) from this project to create a new project.

### 8.3.4 Notes for Build on GCC

The following Warning occurs when building an GCC project.

These Warnings are occurring because specifying a stack size limit in the compiler options, and the target portion (the inline assembler processing portion) is not included in the calculation of stack usage.

Therefore, there is no problem in operation even if warnings occur.



### 8.3.5 When using IAR Integrated Development Environment "IAR Embedded Workbench"

You can use the RX Smart Configurator to import source files into IAR Embedded Workbench.

For instructions, see below.

[RX Smart Configurator User's Guide: IAREW](#)



### 8.4 RL78 Sample Project

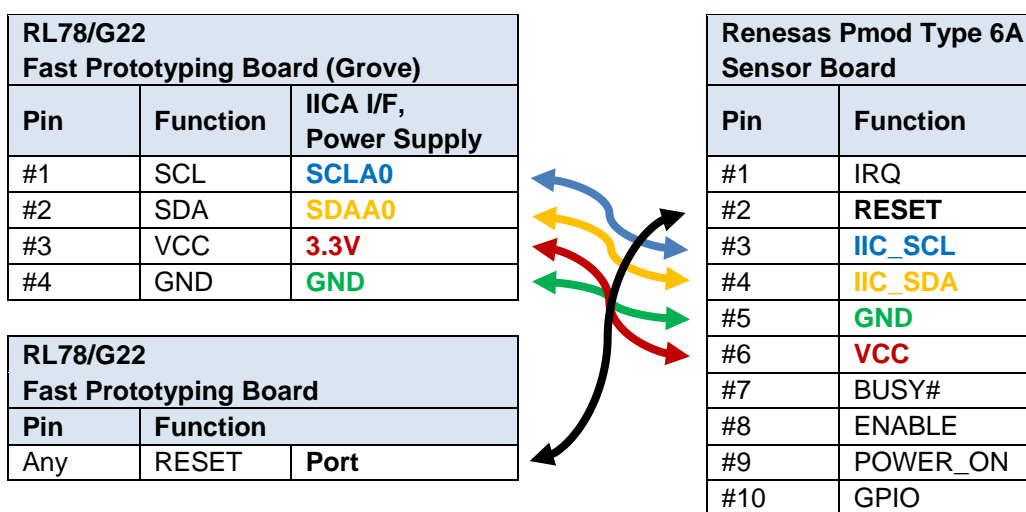
After importing the sample projects, follow the steps below. Please refer to “8.1 Importing Sample Project” for importing instructions.

The following explains the change procedure for the following board change example.

- Sample project "ZMOD4XXX\_RL78G23\_NonOS":
  - Pmod2 (Type 6A: IICA1)
    - Grove (IICA0) of RL78/G22 Fast Prototyping Board
    - With interrupt control
      - Without interrupt control
    - RESET control pin
      - Change to any Port pin

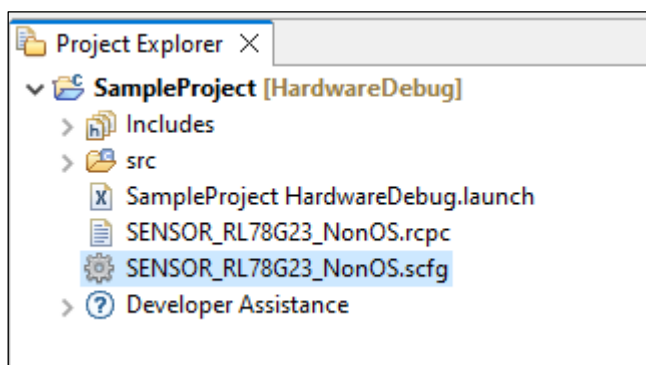
Set J17 to 2-3 to change Grove's VDD to 3.3V.

Also, connect with jumper wires as shown below.



#### 8.4.1 Modifying Settings of Smart Configurator

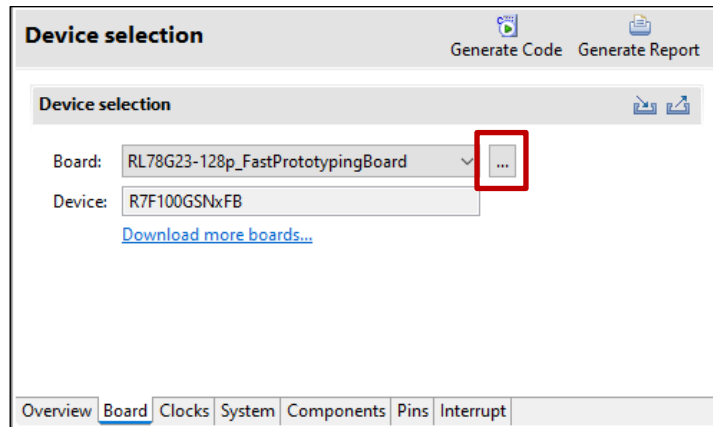
On the project tree, double-click on the .scfg file of the imported project in the Smart Configurator window will open.



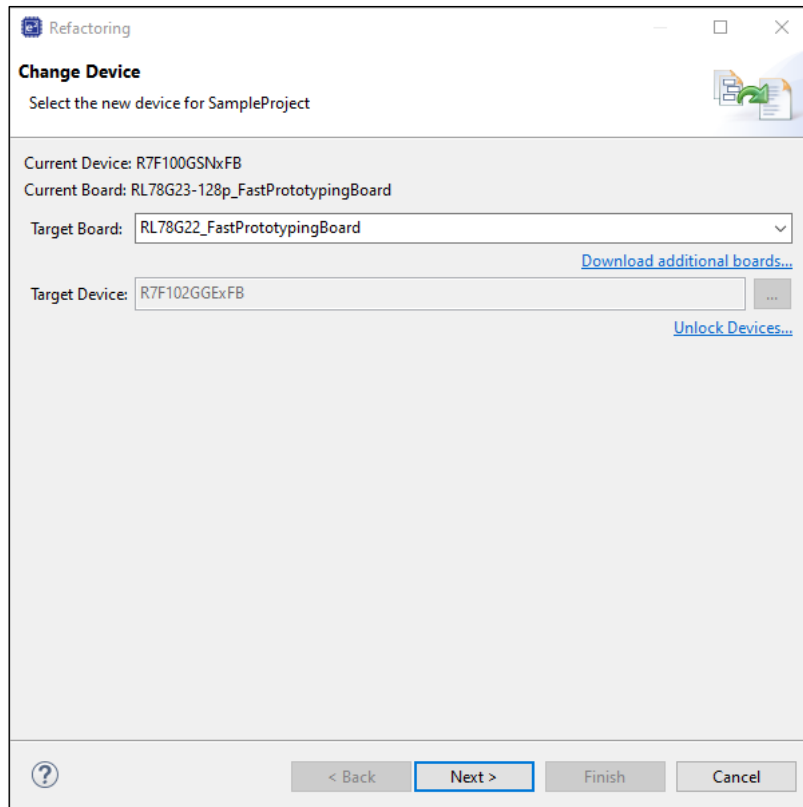


(1) Board

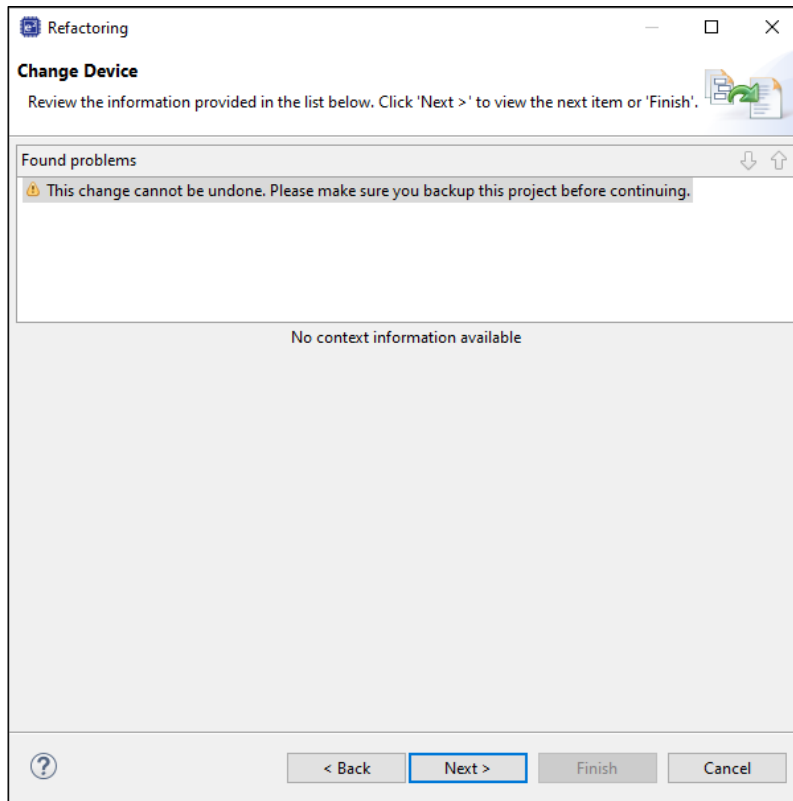
1. On the Board tab, click the [...] button.



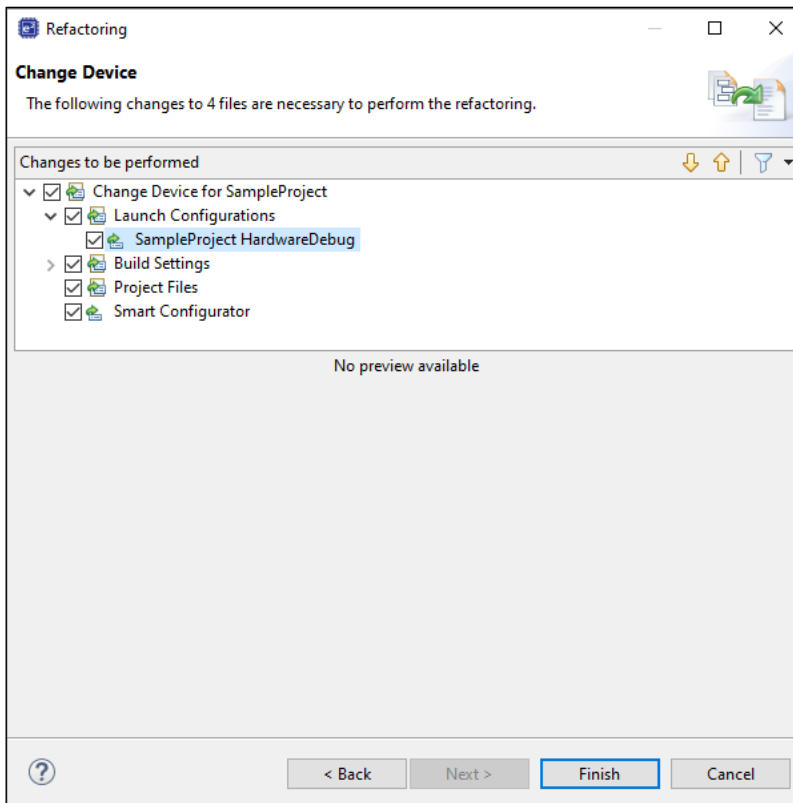
2. Select a desired board or device in the "Change Device" window and press the [Next] button.



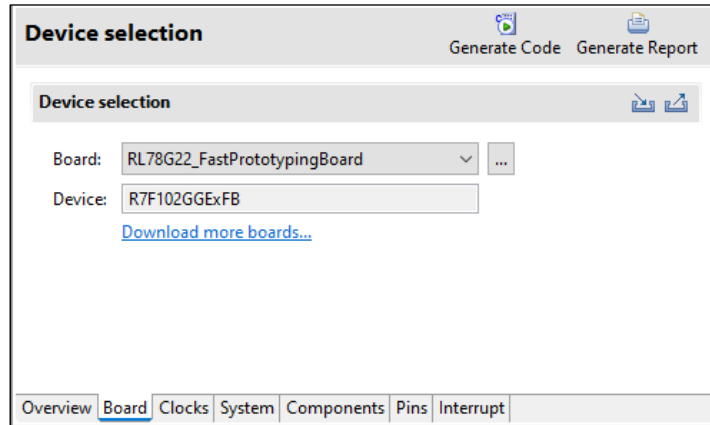
- If a warning message appears, read it and check if there is a problem in proceeding with the procedure. Press [Next] to move to the next step.



- The changes you have made in the settings will be displayed. Press the [Finish] button to apply the changes to the project.



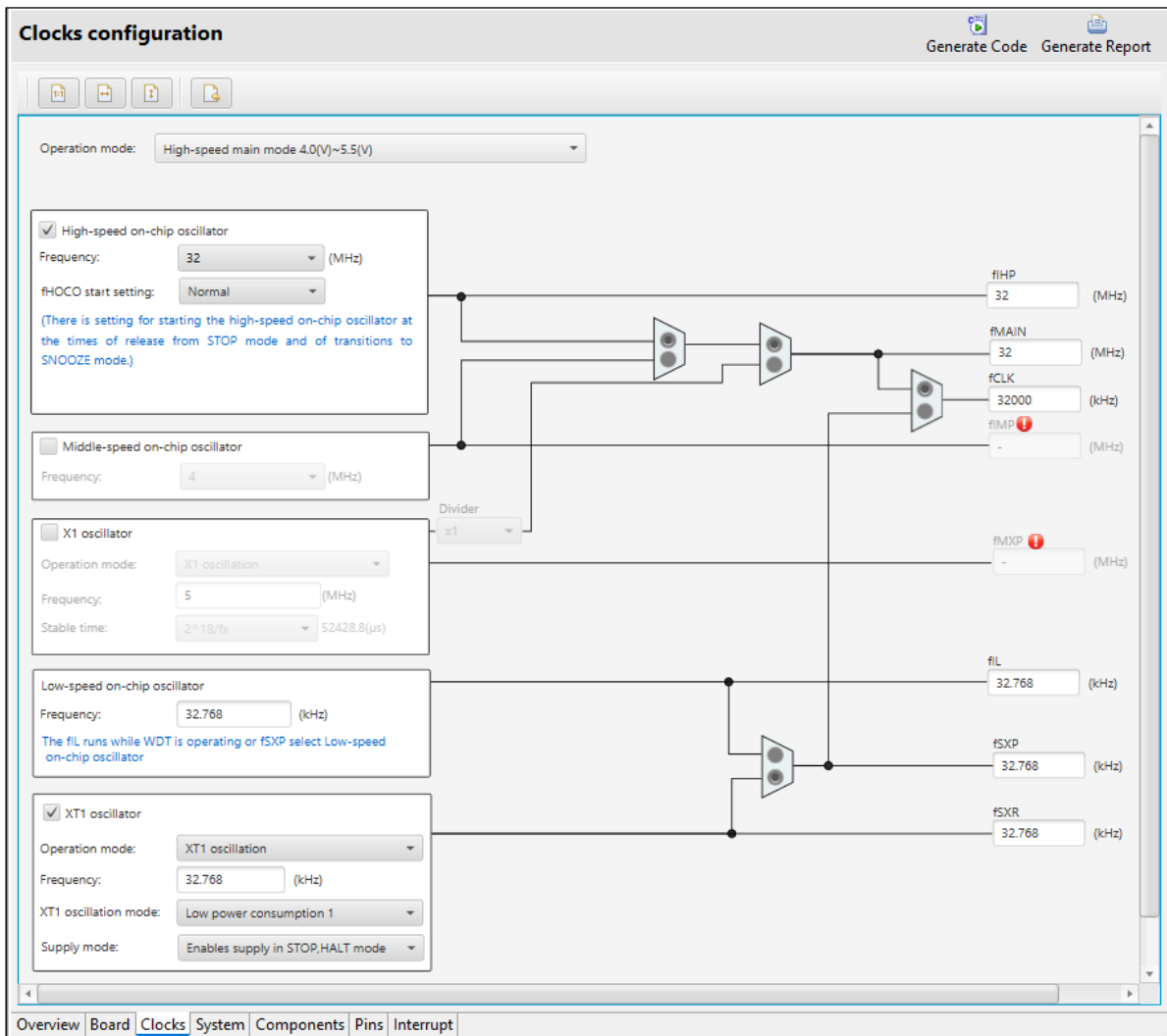
5. Select the "Board" tabbed page to check that the "Board" and "Device" have been changed correctly.



**(2) Clocks**

When "Custom User Board (Any Device)" is selected for "Board", set up the clocks according to the specifications of the target board to be used.

When a Renesas board is selected for "Board", the clocks are automatically set up.



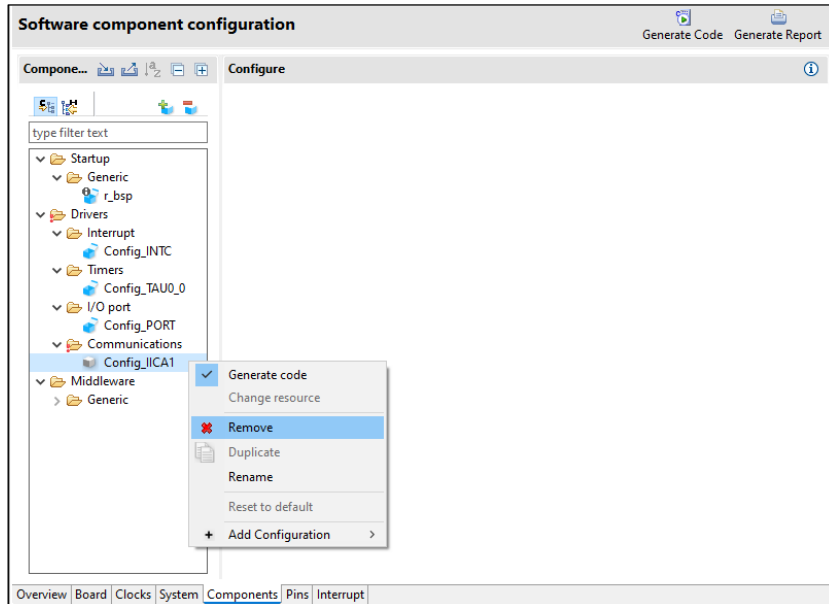
**(3) Components**

Modify the settings of individual components in the "Components" tabbed page according to the specifications of the target board.

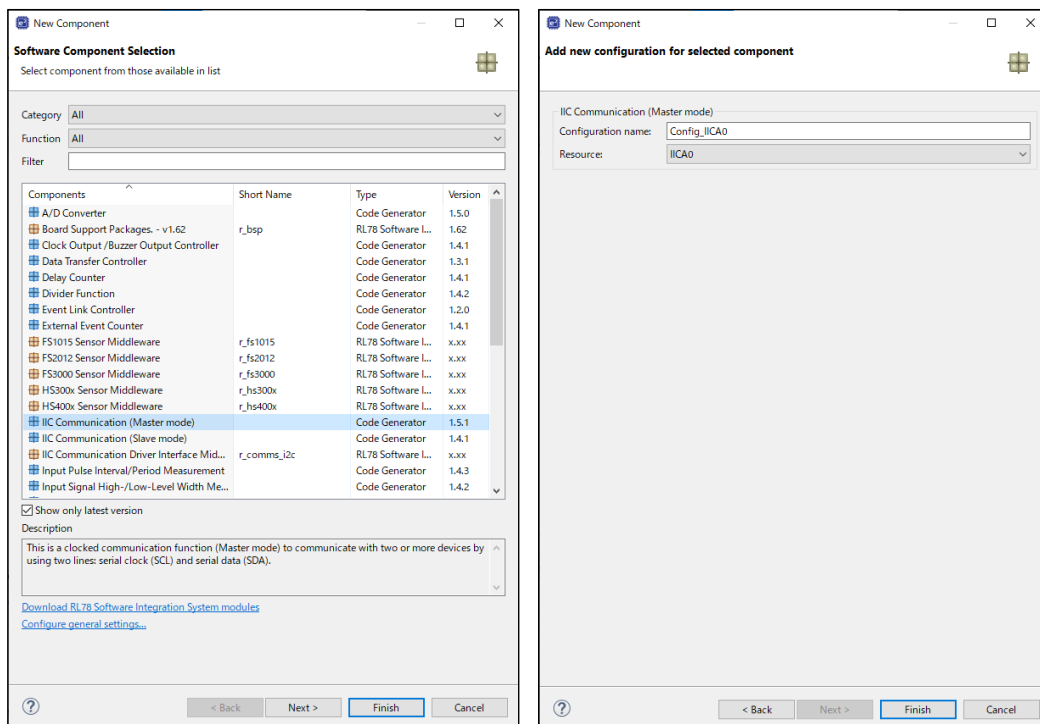
**(a) Changing I2C Driver Settings**

To change I2C driver setting, follow the steps below.

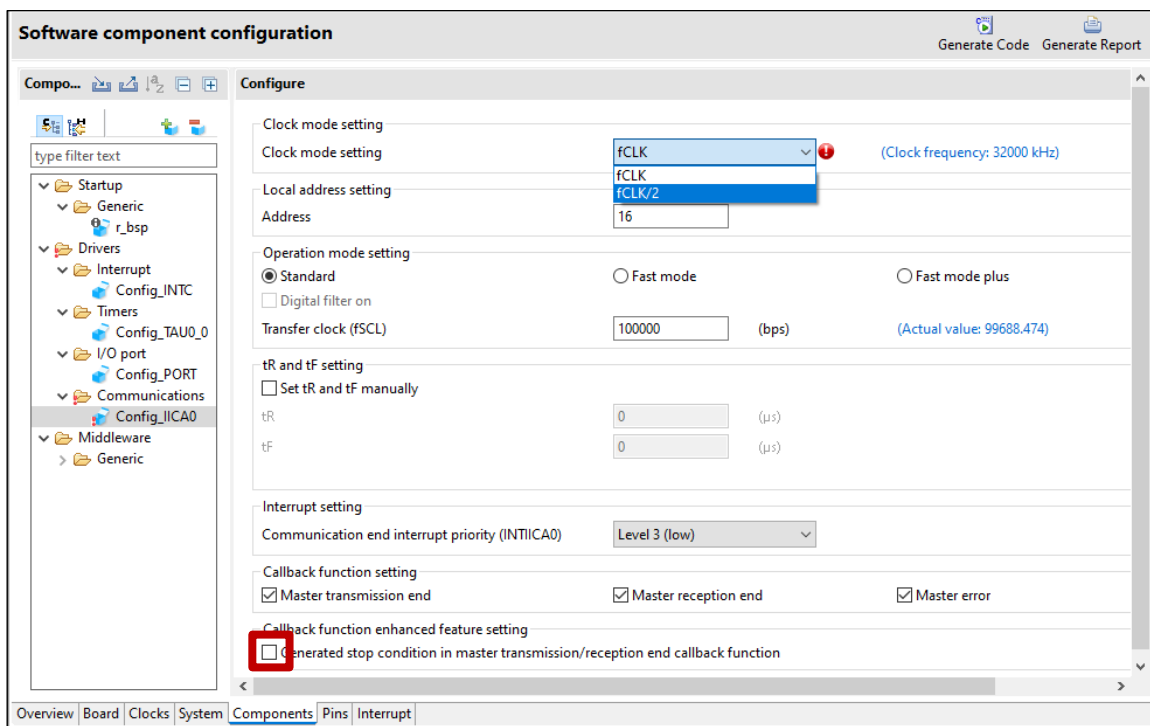
1. In RL78/G22, the only resource that can be used as IICA is IICA0, so delete Config\_IICA1.



2. In "Software Component Selection", select "IIC Communication (Master Mode)" and specify "IICA0" as the resource.



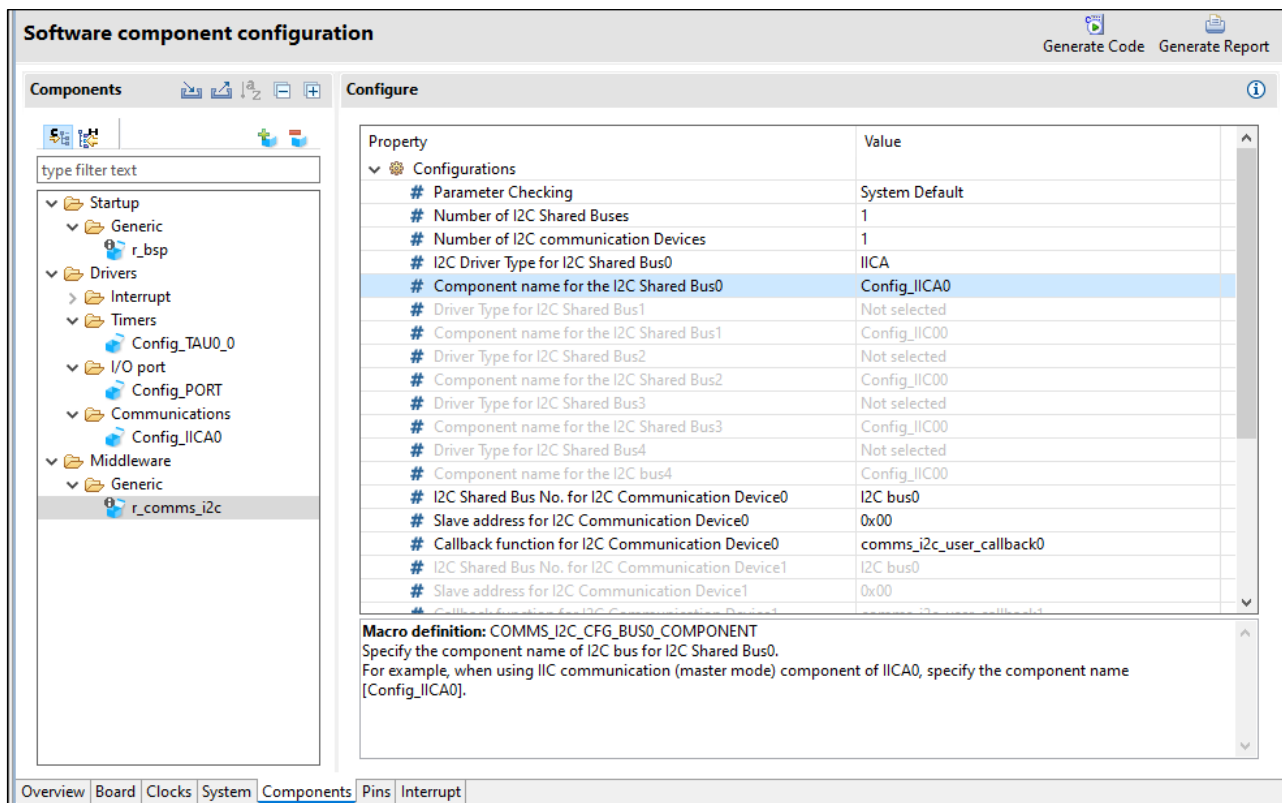
- Change the setting of "Clock mode setting" to "fCLK/2" and uncheck "Generated stop condition in master transmission/reception end callback function".



**(b) Changing COMMS\_I2C Settings**

Review the settings to make sure they are appropriate. If you have changed the I2C driver, you will need to review them.

Change the setting of "Component name for the I2C Shared Bus0" to "Config\_IICA0" in r\_comms\_i2c.

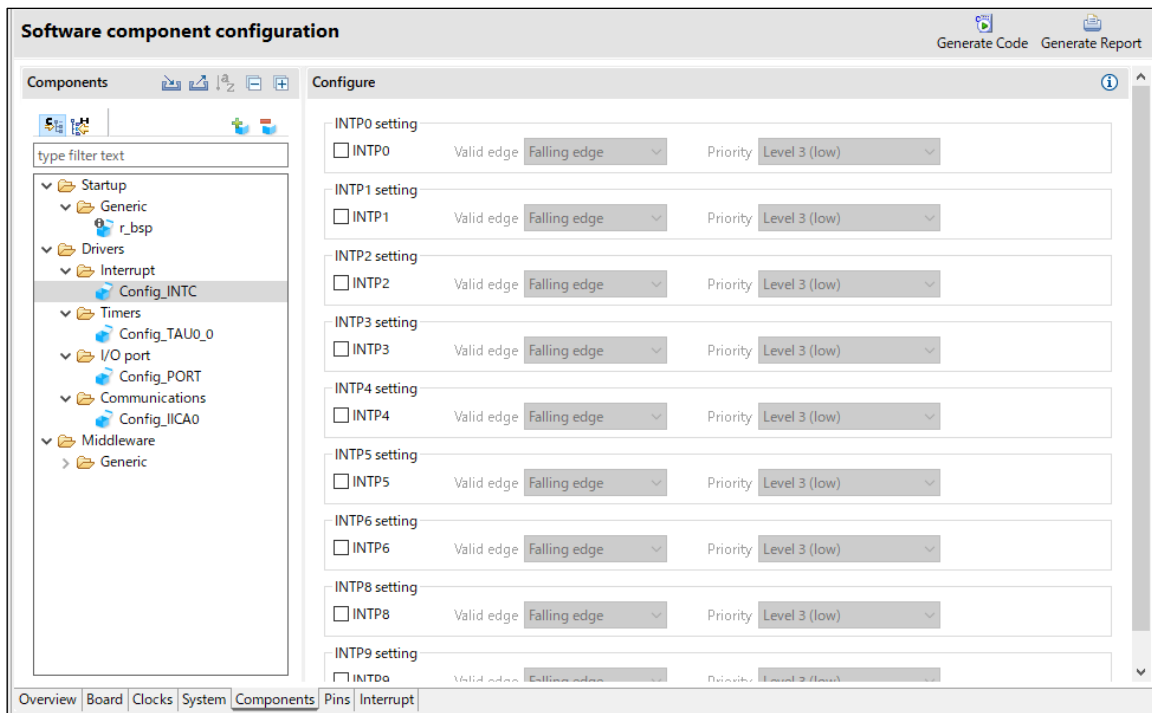


**(c) Changing INTC Driver Settings**

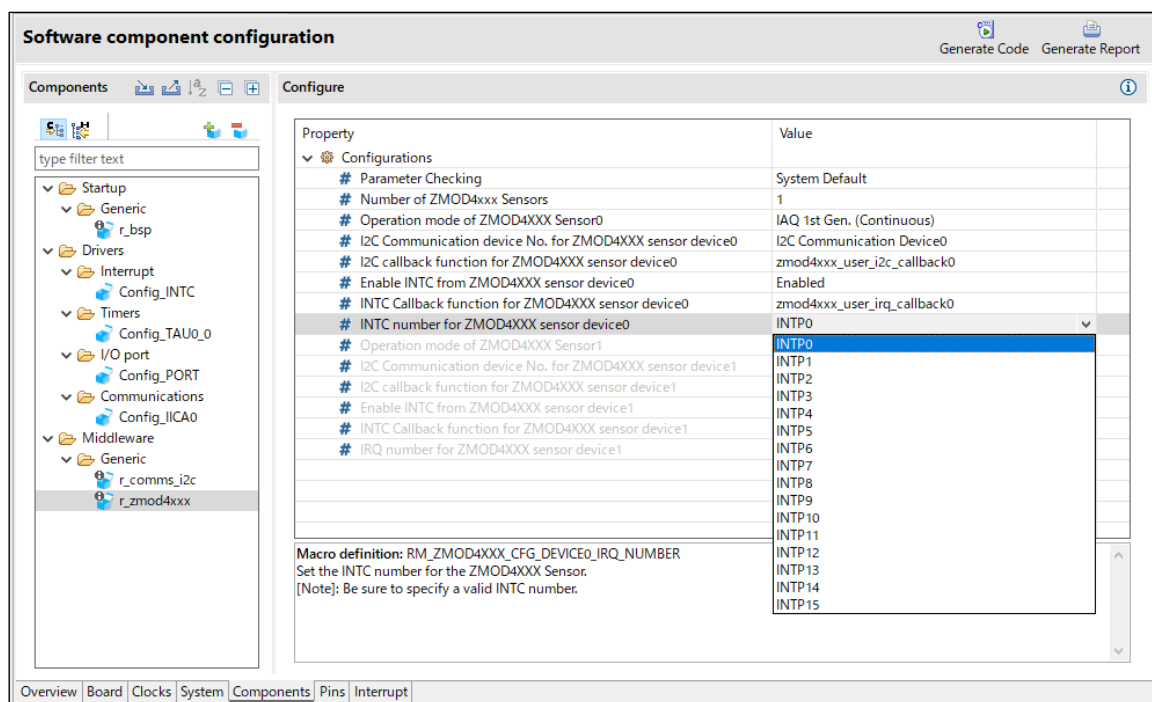
For information on matching the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

There is no interrupt pin assigned to Grove on the RL78/G22 Fast Prototyping Board. Please use another pin if necessary.

1. Set INTP pin in Config\_INTC according to the specifications of the target board.



2. Change “INTC number for ZMOD4XXX sensor device0” in r\_zmod4xxx\_rl to the INTP pin set above.



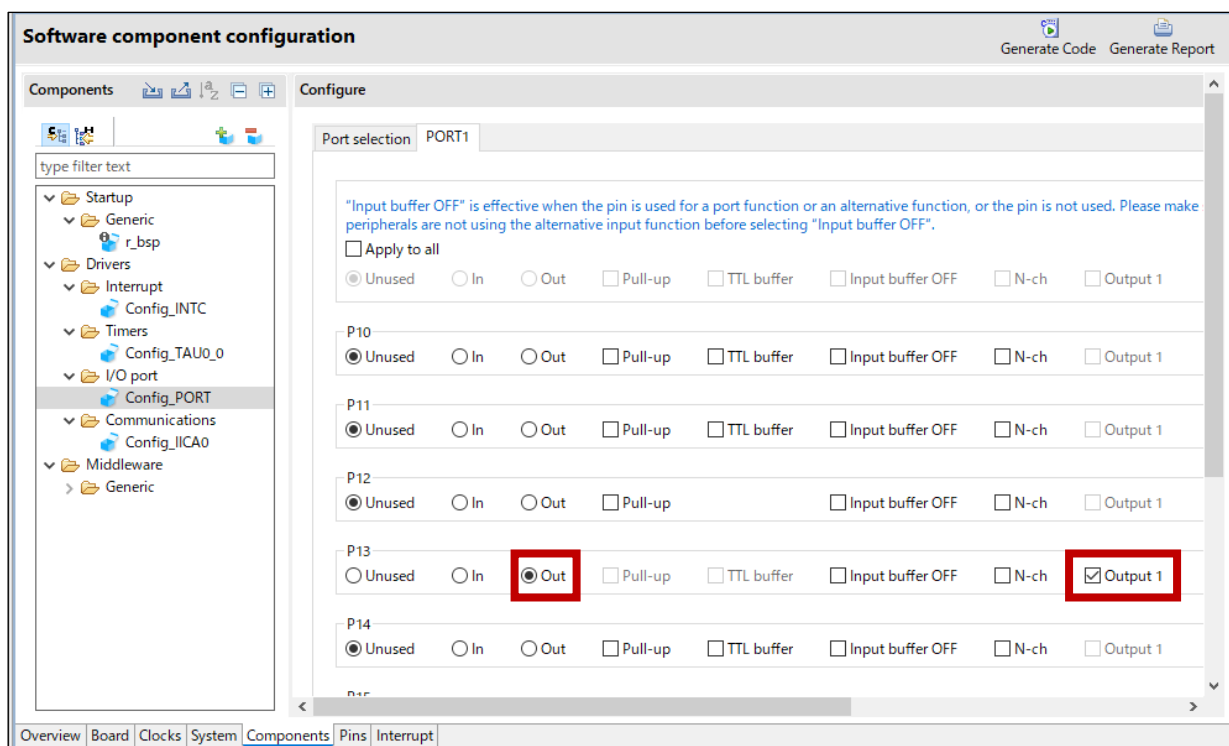
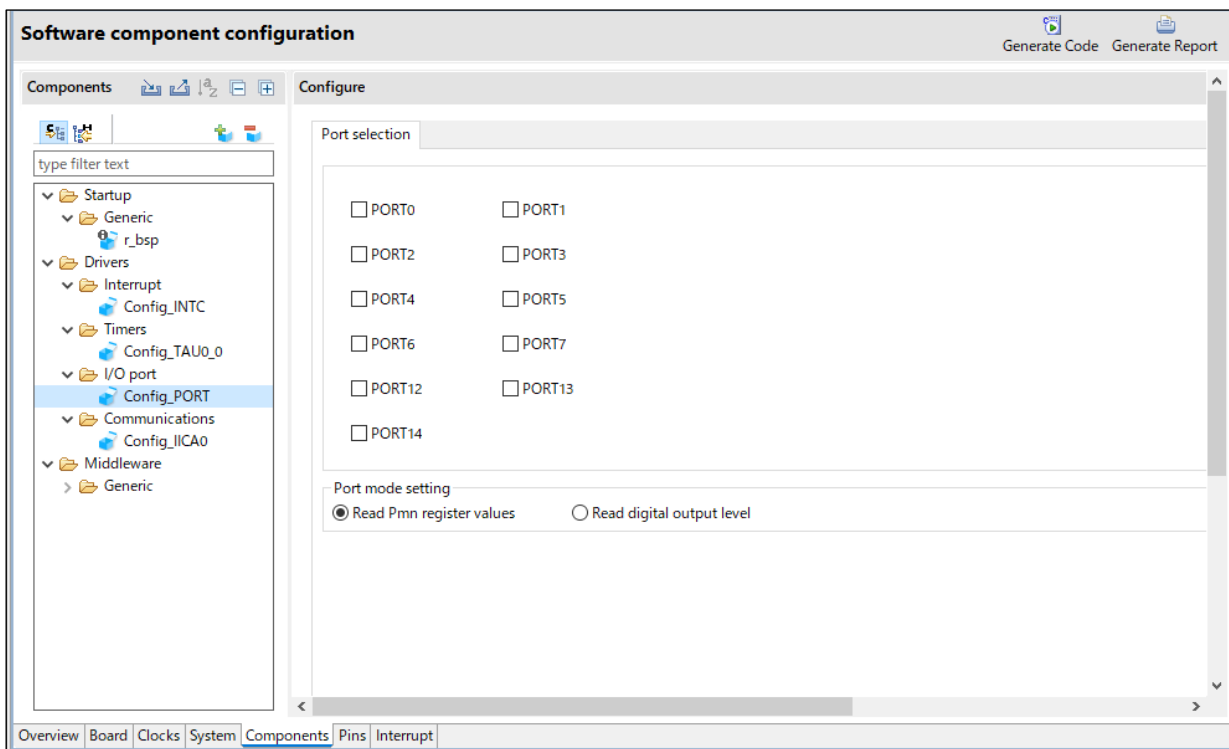
**(d) Changing General Purpose I/O Port Driver Settings: RESET**

For information on how to modify the source of the RESET signal control, refer to “8.4.3(1) RESET Signal Control”.

There is no RESET pin assigned to Grove on the RL78/G22 Fast Prototyping Board. Therefore, assign any port.

Set RESET pin on Pmod in Config\_PORT according to the specifications of the target board.

Set “Out” and enable “Output 1” in the “Port selection” tabbed page and the “PORT(x)” tabbed page.



(4) Pins

(a) Changing I2C I/F Pins

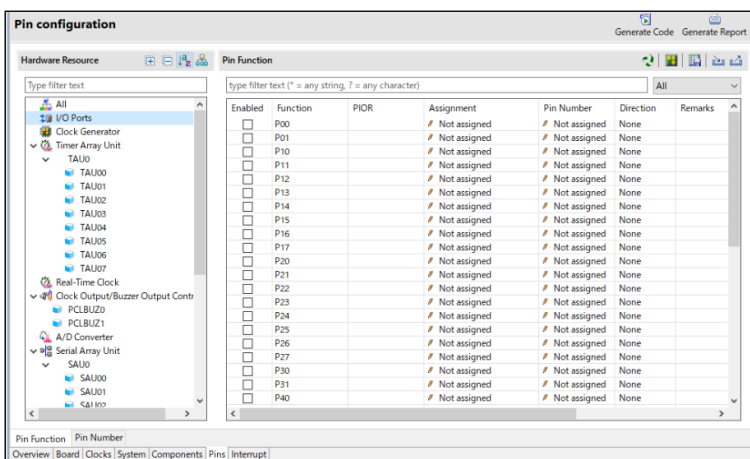
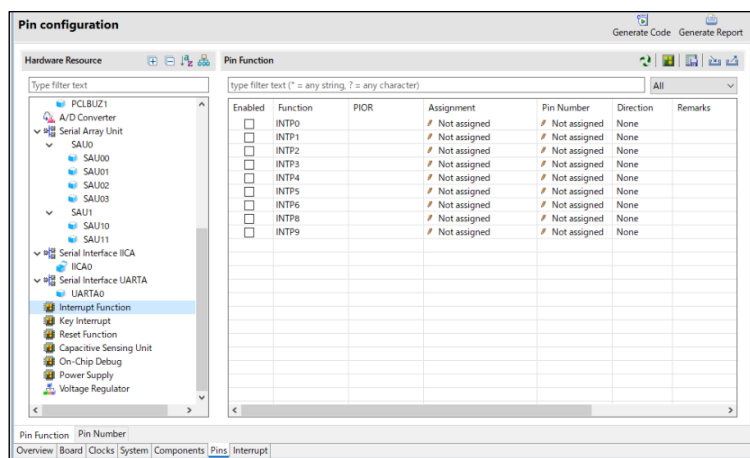
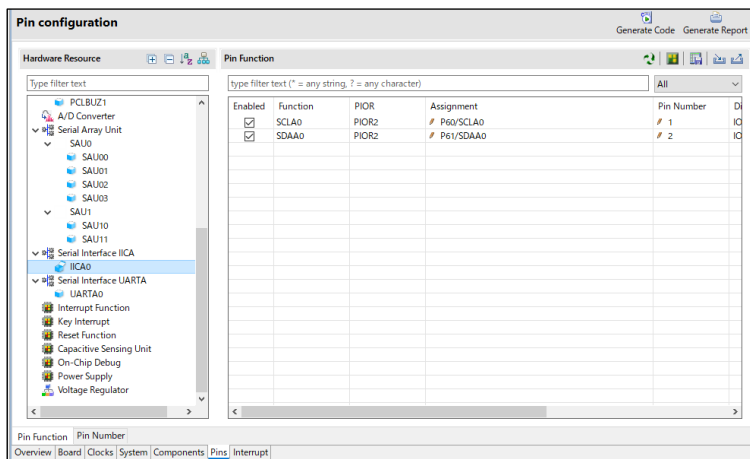
(b) Changing INTP Pins

(c) Changing General Purpose I/O Port Pin: RESET

For information on configuring the pin of the interrupt signal circuit, refer to “8.5 Notes for Interrupt Signal Circuits”.

For information on configuring the pin of the RESET signal circuit, refer to “8.6 Notes for RESET Signal Circuits”.

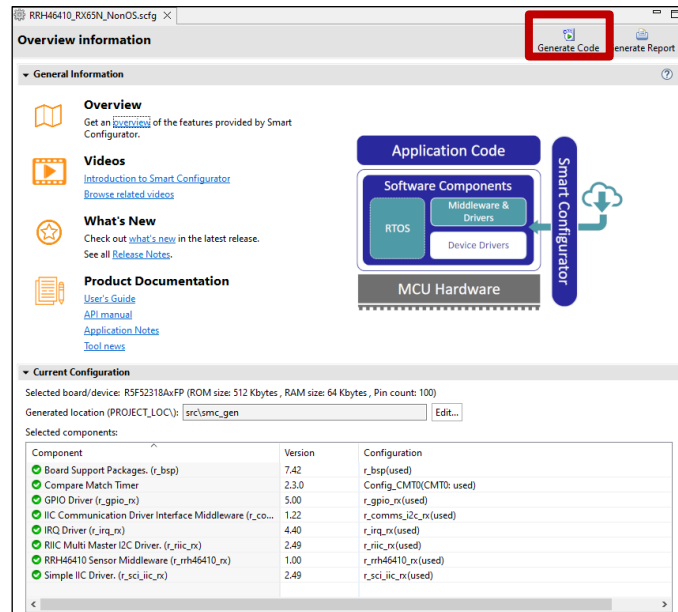
Select “IICA0”, “Interrupt Function” and “I/O Ports” in "Pins" tabbed page and check that functions are assigned to the IICA pins, INTP pin, and RESET pin in the "Pin function" panel.





(5) Code Generation and Build

Press the [Generate Code] icon to generate code.



Build the project after implementing “8.4.2 Modifying Generated Code“ and “8.4.3 Changing Sample Code”

Select [Debug Configurations] from the menu and modify the debugger settings according to the specifications of the emulator to be connected to the target board.

## 8.4.2 Modifying Generated Code

Open Config\_IICA0\_user.c and add the following code.

Definition for including r\_comms\_i2c\_if.h:

```

/*****
Includes
*****/
#include "r_cg_macrodriver.h"
#include "r_cg_userdefine.h"
#include "Config_IICA0.h"
/* Start user code for include. Do not edit comment generated here */
#include "r_comms_i2c_if.h"
/* End user code. Do not edit comment generated here */

```

Addition of the rm\_comms\_i2c\_bus0\_callback() function to the callback function:

Specify the "false" parameter for the transmission and reception end callback functions and the "true" parameter for the error callback function.

```

/*****
* Function Name: r_Config_IICA0_callback_master_sendend
* Description  : This function is a callback function when IICA0 finishes master
transmission.
* Arguments    : None
* Return Value : None
*****/
static void r_Config_IICA0_callback_master_sendend(void)
{
/* Start user code for r_Config_IICA0_callback_master_sendend. Do not edit comment
generated here */
    rm_comms_i2c_bus0_callback(false);
/* End user code. Do not edit comment generated here */
}

/*****
* Function Name: r_Config_IICA0_callback_master_receiveend
* Description  : This function is a callback function when IICA0 finishes master
reception.
* Arguments    : None
* Return Value : None
*****/
static void r_Config_IICA0_callback_master_receiveend(void)
{
/* Start user code for r_Config_IICA0_callback_master_receiveend. Do not edit comment
generated here */
    rm_comms_i2c_bus0_callback(false);
/* End user code. Do not edit comment generated here */
}

/*****
* Function Name: r_Config_IICA0_callback_master_error
* Description  : This function is a callback function when IICA0 master error occurs.
* Arguments    : flag -
                status flag
* Return Value : None
*****/
static void r_Config_IICA0_callback_master_error(MD_STATUS flag)
{
    /* Start user code for r_Config_IICA0_callback_master_error. Do not edit comment
generated here */
    rm_comms_i2c_bus0_callback(true);
    /* End user code. Do not edit comment generated here */
}

```

### 8.4.3 Changing Sample Code

#### (1) RESET Signal Control

For pin configuration, refer to “8.4.1(4)(c) Changing General Purpose I/O Port Pin: RESET”.

There is no RESET pin assigned to Grove on the RL78/G22 Fast Prototyping Board. Therefore, assign any port.

Open ZMOD4XXX\_RL78G23\_NonOS.c and change the reset operation when resetting the sensor.

Modify RESET pin designation according to the specifications of the target board. Also, change the reset control logic to an appropriate one.

The following is an example of using P13 as RESET pin.

```
/* Reset ZMOD sensor (active low). Please change to the IO port
connected to the RES_N pin of the ZMOD sensor on the customer board. */
P1 = _00_Pn7_OUTPUT_0 | _00_Pn6_OUTPUT_0 | _00_Pn5_OUTPUT_0 |
_00_Pn4_OUTPUT_0 | _08_Pn3_OUTPUT_1 |
_00_Pn2_OUTPUT_0 | _00_Pn1_OUTPUT_0 | _00_Pn0_OUTPUT_0;
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);
P1 = _00_Pn7_OUTPUT_0 | _00_Pn6_OUTPUT_0 | _00_Pn5_OUTPUT_0 |
_00_Pn4_OUTPUT_0 | _00_Pn3_OUTPUT_0 |
_00_Pn2_OUTPUT_0 | _00_Pn1_OUTPUT_0 | _00_Pn0_OUTPUT_0;
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);
P1 = _00_Pn7_OUTPUT_0 | _00_Pn6_OUTPUT_0 | _00_Pn5_OUTPUT_0 |
_00_Pn4_OUTPUT_0 | _08_Pn3_OUTPUT_1 |
_00_Pn2_OUTPUT_0 | _00_Pn1_OUTPUT_0 | _00_Pn0_OUTPUT_0;
R_BSP_SoftwareDelay(10, BSP_DELAY_MILLISECS);
```

### 8.4.4 Changing Toolchain Setting

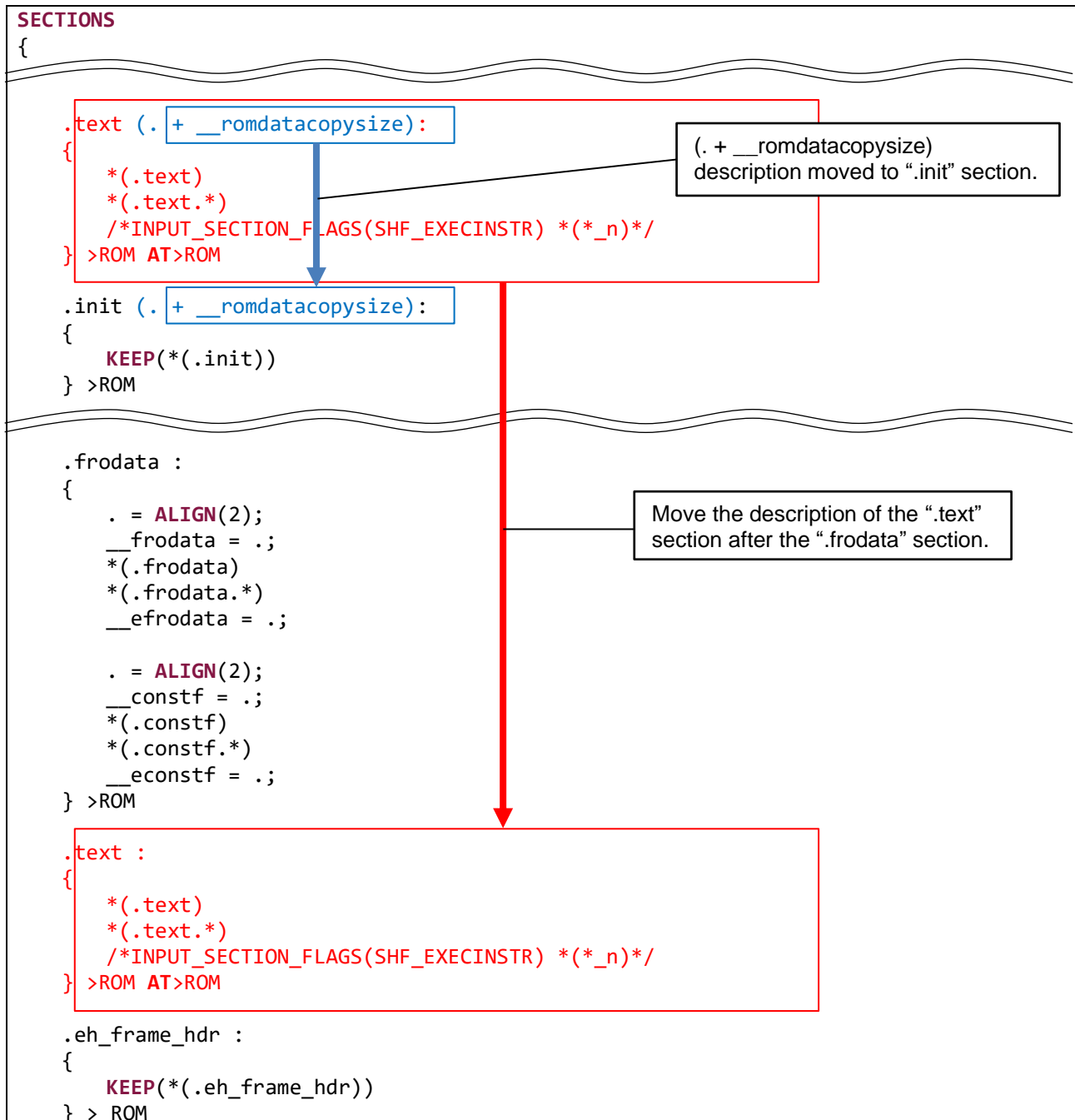
If you want to use a toolchain the LLVM toolchain, use "ZMOD4XXX\_RL78G23\_NonOS\_LLVM".

If you want to use a toolchain other than the CC-RL toolchain or LLVM toolchain, copy ZMOD4XXX\_RL78G23\_NonOS.c and RL78\_ZMOD4XXX.c from this project to create a new project.

Also, when using the LLVM toolchain, build errors may occur due to section placement. In this case, the linker script must be modified.

The following describes an example of modifying linker\_script.ld in the sample project "ZMOD4XXX\_RL78G23\_NonOS\_LLVM".

1. Place the ".text" section after the ".frodata" section.



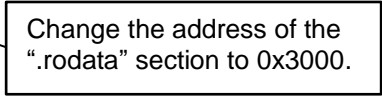
2. Fix the address of the “.rodata” section to the top address of the mirror area.

```

.fini :
{
    KEEP(*(.fini))
} >ROM

PROVIDE(__rodata_limit = CONSTANT(MIRRORAREASTART)+ 0x3000 + LENGTH(MIRROR));

/* The rodata section is placed in MIRROR area in order to access as near
addressing. */
.rodata MAX(., (CONSTANT(MIRRORAREASTART)+ 0x3000)):
.rodata 0x3000 : AT(0x3000)
{
    . = ALIGN(2);
    __rodata = .;
    *(.rodata)
    *(.rodata.*)
    . = ALIGN(2);
}
    
```



3. Change the “.data” section to the address after the “.ocd\_traceram” section.

```

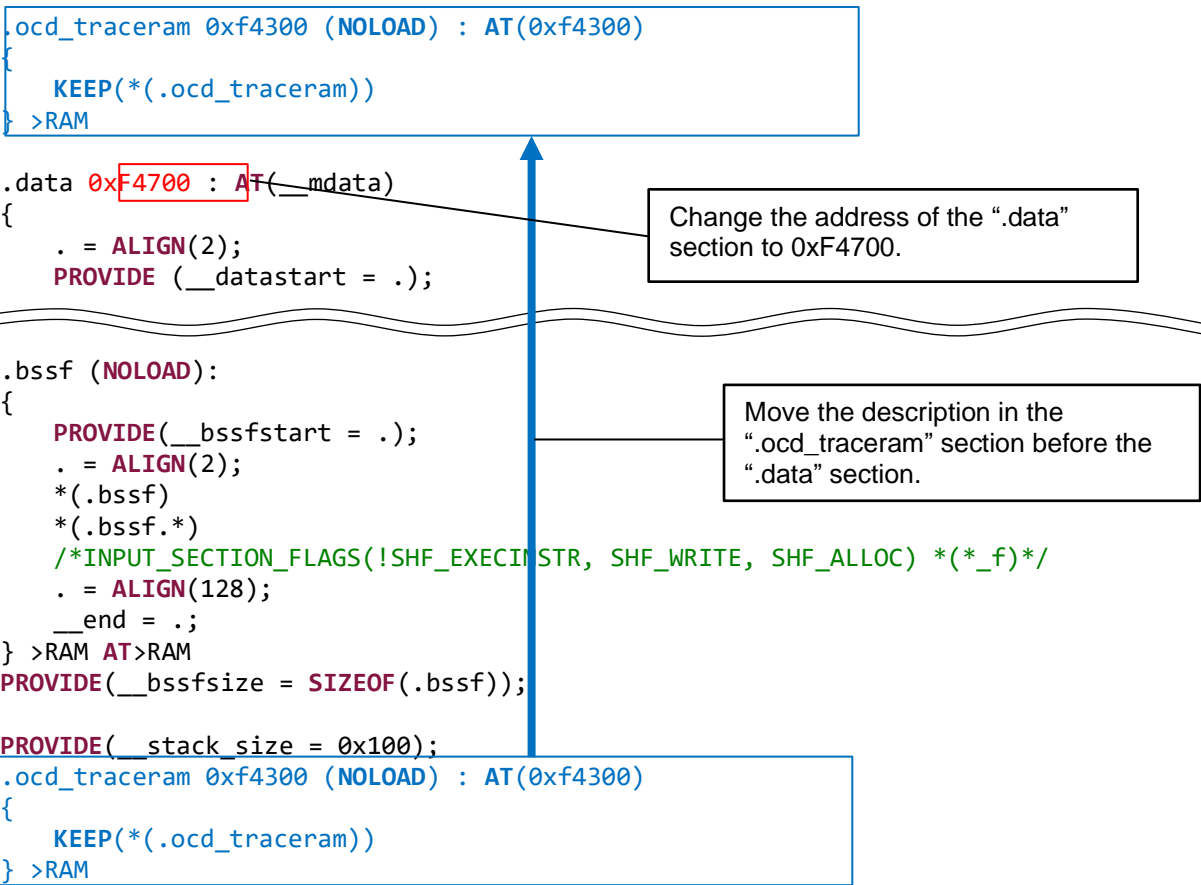
.eh_frame :
{
    KEEP(*(.eh_frame))
} > ROM

.ocd_traceram 0xf4300 (NOLOAD) : AT(0xf4300)
{
    KEEP(*(.ocd_traceram))
} >RAM

.data 0xF4700 : AT(__mdata)
{
    . = ALIGN(2);
    PROVIDE (__datastart = .);
}

.bssf (NOLOAD):
{
    PROVIDE(__bssfstart = .);
    . = ALIGN(2);
    *(.bssf)
    *(.bssf.*)
    /*INPUT_SECTION_FLAGS(!SHF_EXECINSTR, SHF_WRITE, SHF_ALLOC) *(*_f)*/
    . = ALIGN(128);
    __end = .;
} >RAM AT>RAM
PROVIDE(__bssfsize = SIZEOF(.bssf));

PROVIDE(__stack_size = 0x100);
.ocd_traceram 0xf4300 (NOLOAD) : AT(0xf4300)
{
    KEEP(*(.ocd_traceram))
} >RAM
.stack 0xFFE20 (NOLOAD) : AT(0xFFE20)
{
    PROVIDE(__stack = .);
}
    
```



Note: Changing the section address reduces the available ROM / RAM area.

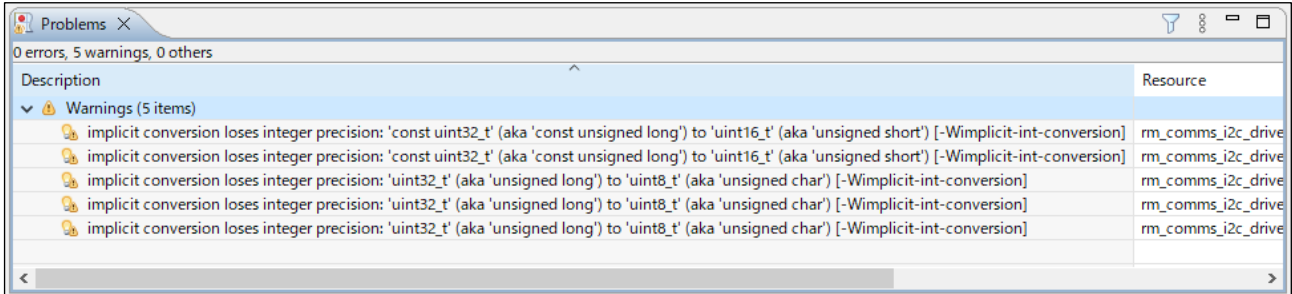
### 8.4.5 Notes for Build on LLVM

The following Warning occurs when building an LLVM project.

These Warnings are occurring because “slave\_address” and “bytes” used in COMMS\_I2C are handled as 32-bit type.

Since “slave\_address” is 7-bit data and “bytes” is 16-bit data, no loss occurs due to conversion.

Therefore, warning messages will appear, but this does not affect the operation.



## 8.5 Notes for Interrupt Signal Circuits

It is necessary to configure settings according to the circuit to be embedded.

When using the sensor interrupt request signal (SensorINT), set the trigger level correctly.

### (1) Circuit Configuration of ZMOD4XXX Sensor Pmod Board

The circuit configuration of the interrupt signal of the ZMOD4XXX Sensor Pmod Board is shown below. For details, please refer to the board datasheet.

**Table 8-2 Circuit Configuration of Interrupt Signal on ZMOD4XXX Sensor Pmod Board**

Sensor Pmod Board	Circuit Configuration (Pull-up resistor can be set to on or off)
#1: IRQ# H Output: Data Available	inverted output of SensorINT (Open-Drain output) and fixedly connected to IRQ#. (Output of the inverted SensorINT signal cannot be disabled.)
#7: BUSY#	- (No SensorINT signal output circuit)

### (2) How to Control Interrupt Trigger Control when Multiple Sensor Pmod Boards are Daisy-chained

If the IRQ# of each board is an open-drain output, daisy chain connection is possible. (Note)

With this connection, if an interrupt request is asserted, it will be difficult to identify the I2C device that generated the interrupt request.

**If multiple interrupt output signals are connected to IRQ# of Pmod #1 due to the circuit configuration, operate all I2C devices without interrupts. (Note)**

Note: Do not connect I2C devices with outputs other than Open-Drain (Push-Pull output of CMOS or TTL).  
If connected, the I2C device and the peripheral circuit ICs connected to the interrupt signal may be damaged.

### (3) How to Control Interrupt Triggers when using a Sensor Pmod Board that Allows Changing an Interrupt Request Output Destination

Some sensor Pmod boards can output an interrupt request signal to Pmod #7 (BUSY#) by changing a circuit on the board.

By daisy-chaining a board that switches this interrupt request signal to BUSY# output and a board with a standard IRQ# output, it becomes possible to trigger interrupts from two devices at the same time.

Note that an operating condition is that Pmod #1 and #7 on the MCU side have to input interrupt signals.

**Table 8-3 Circuit Configuration of Interrupt Signal on Sensor Pmod Board (Board Dependent)**

Sensor Pmod Board	Circuit Configuration (Pull-up resistor can be set to on or off)
#1: IRQ# #7: BUSY#	The Sensor Pmod Board allows the following circuit modifications (Board dependent): <ul style="list-style-type: none"> <li>• SensorINT non-inverted or inverted output (open-drain output or push-pull output) is possible for either #1 or #7. (Note 1)</li> <li>• SensorINT Output is disabled and open for both #1 and #7.</li> </ul>

Note 1: Non-inverted or inverted output, Open-Drain or Push-Pull output are board-dependent.

## 8.6 Notes for RESET Signal Circuits

It is necessary to configure settings according to the circuit to be embedded.

### (1) Circuit Configuration of ZMOD4XXX Sensor Pmod Board

The circuit configuration of the RESET signal of the ZMOD4XXX Sensor Pmod Board is shown below. For details, please refer to the board datasheet.

**Table 8-4 Circuit Configuration of RESET Signal on ZMOD4XXX Sensor Pmod Board**

Sensor Pmod Board	Circuit Configuration (Pull-up resistor can be set to on or off)
#2: RESET# L Input: Device Reset	Directly connected to the RESET signal of the sensor device

### (2) When You Want to Daisy-chain Multiple I2C Devices and Operate them using Interrupt Triggers

A single RESET output signal from the MCU can control the reset of multiple I2C devices.

In that case, it is necessary to configure a RESET signal circuit that matches the reset control logic of the multiple I2C devices and to implement a reset sequence that meets the requirements of the multiple I2C devices.

In addition, when connecting multiple I2C devices, the pin load capacitance will be large, so be sure to generate a sufficient reset pulse period.

## 8.7 Pull-up Resistor Circuit Configuration when Daisy Chain Connections of Renesas Sensor Pmod Boards

The recommended method for connecting the pull-up resistors in a daisy chain is shown below. Also, disable the pull-ups on other Renesas Sensor boards.

If the pull-up resistors of many Renesas Sensor boards are enabled at the same time, the sensor boards may not function properly.

**Table 8-5 Target Board that Enable Pull-up Resistors when Daisy-chaining**

Pmod Sensor Board Type 6A Singal Name	Recommended Circuit Configuration of Pull-up Resistors
#1: IRQ# (Note 1)	Enable only the board closest to the MCU board for boards with pull-up resistor circuits.
#2: RESET# (Note 1)	Enable only the board closest to the MCU board for boards with pull-up resistor circuits.
#3: SCL	Enable only the board closest to the MCU board for boards.
#4: SDA	Enable only the board closest to the MCU board for boards.
#7: BUSY# (Note 1, 2)	Enable only the board closest to the MCU board for boards with pull-up resistor circuits.

Note 1 There are the boards without the pull-up resistors.

Note 2 Configure when using as an Interrupt signal.



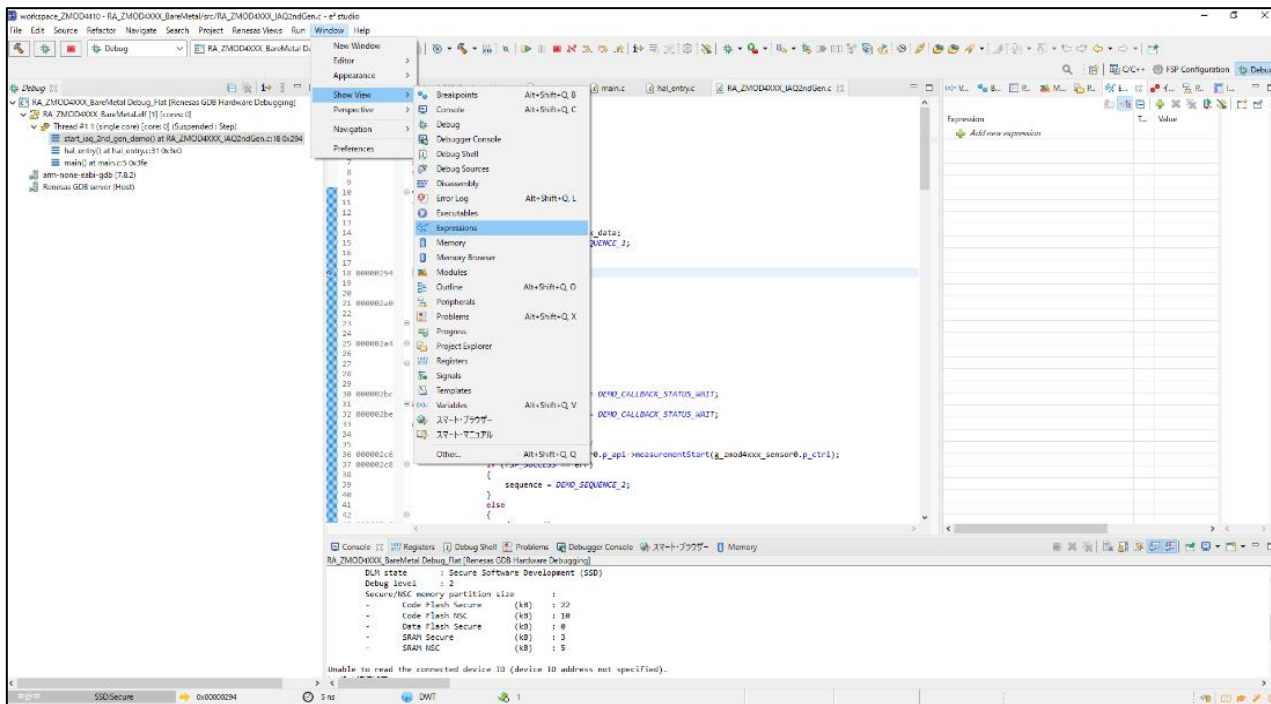
## 9. Viewing Gas Data

To check the real-time gas data, follow the steps below.

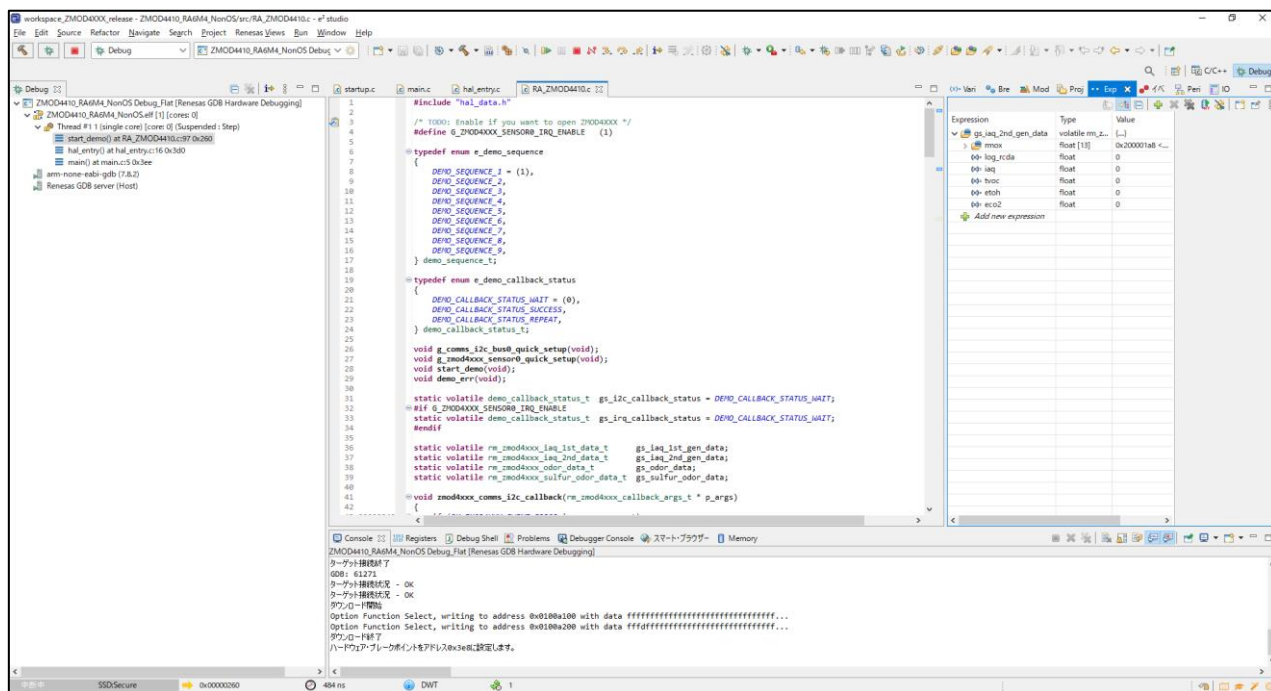
Take the “IAQ 2<sup>nd</sup> Generation” as an example.

1. After running the Debug, open the “Expressions” window.

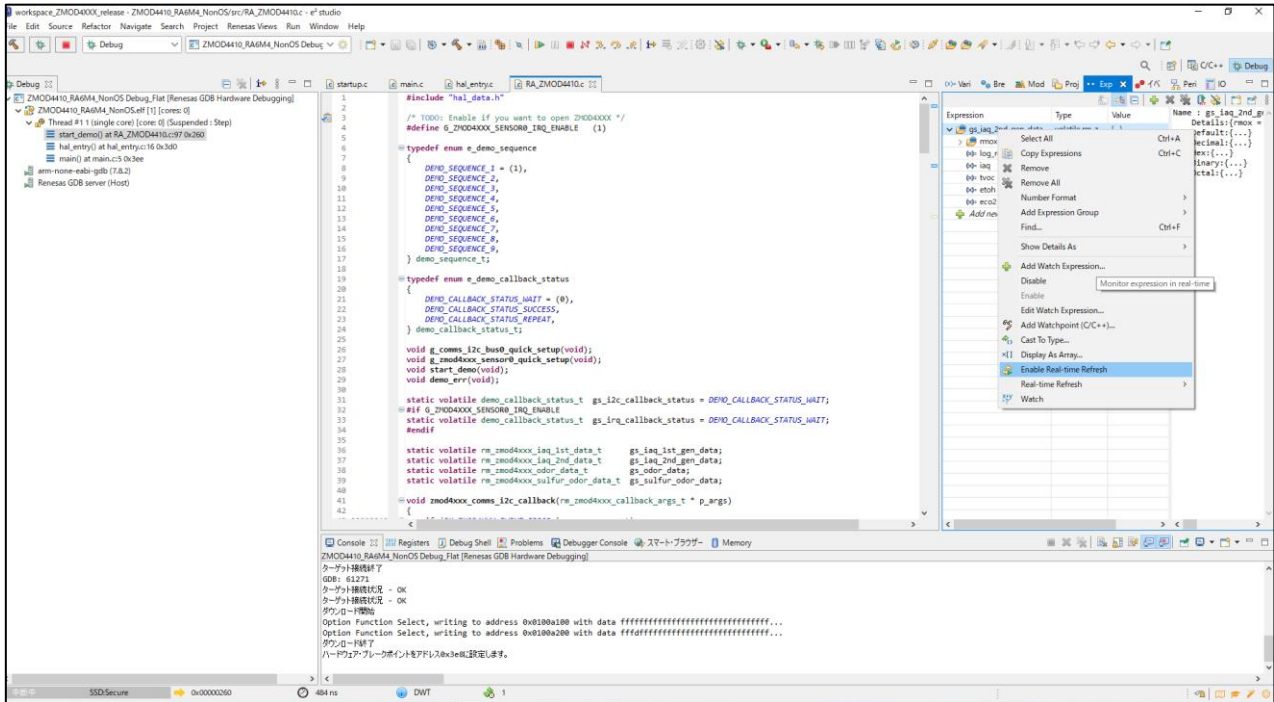
“Expressions” window is available from [Window]→[Show View]→[Expressions].



2. Click “Add new expression” in the “Expressions” and add “gs\_iaq\_2nd\_gen\_data”.

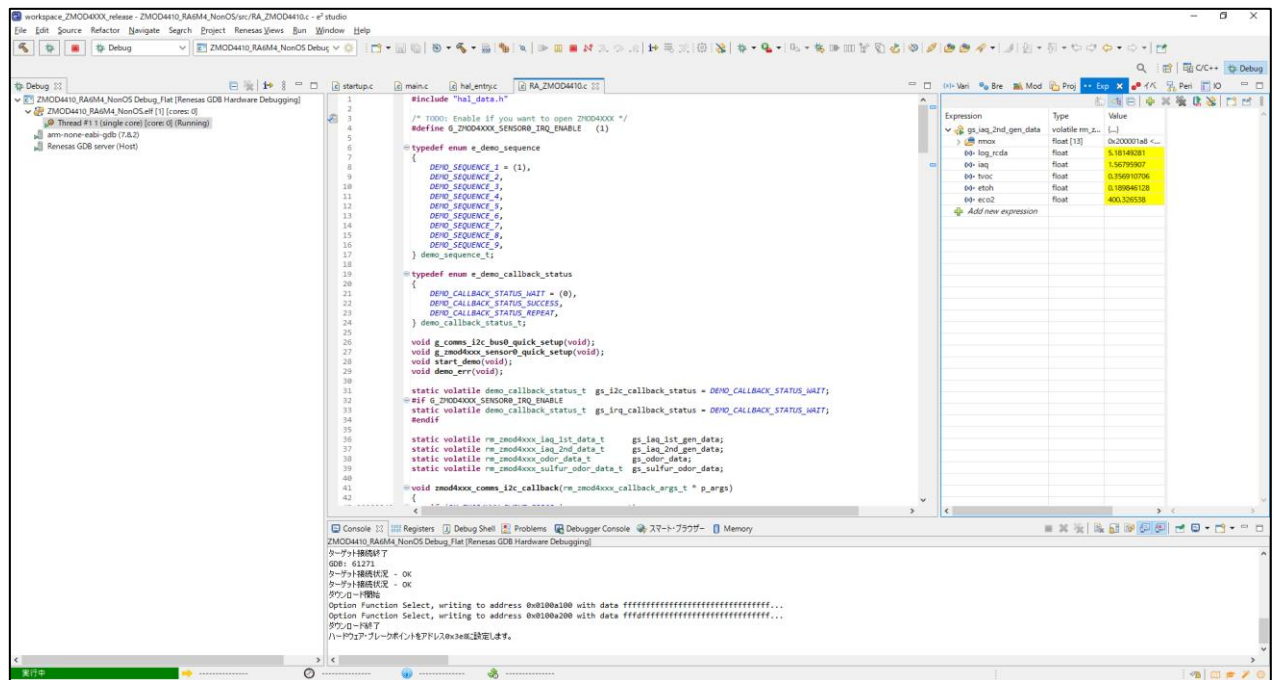


3. Right-click on the added variable and select the “Enable Real-time Refresh”.



4. Start the Debug.

It is possible to check the real-time values.



## Revision History

Rev.	Date	Description	
		Page	Summary
Rev 1.00	June 30, 2021	-	First Release
Rev 1.10	September 30, 2021	-	Add RX Family, RL78 Family and RE01 256KB Group supporting
Rev.1.20	December 20, 2021	-	Add: Support multiple ZMOD sensors usage Add: Support RX Azure Other minor changes
Rev.1.30	March 1, 2022	-	Update chapters 2, 5 and 6 to add IAQ 2nd Gen ULP mode
Rev.1.31	March 11, 2022	-	Fix RA sample projects
		P9	Changed Figure 2-6 Hardware Connections for RE01 1500KB
Rev.1.40	June 30, 2022	P8, P9	Changed Environment for Confirming Operation on an RE01 Fixed: RE01 sample project codes
	August 31, 2022	-	Added: ZMOD4450 new. Fixed: Sample project codes
Rev.1.50	November 30, 2022	-	Added: RZ items Other minor changes
Rev.1.51	February 20, 2023	-	Bug fixes Updated: Environments for RL78
Rev.1.52	March 29, 2023	-	Updated: Environments for RA, RX, RL78, RZ Updated: Main Processing Flow of Sample Software Updated: Guide for Changing the Target Device
Rev.1.53	June 28, 2023	-	Updated: Environments for RA Updated: ZMOD4410 Sensor specifications Updated: Specification of Sample Software Updated: MOD4xxx Settings for RX, RL78
Rev.1.54	September 7, 2023	-	Updated: Guide for Changing the Target Device Deleted: RE01 items
Rev.1.55	May 17,2024	-	Added: Environments for RA0E1 Updated: I2C Driver Settings of RA Family Updated: Guide for Changing the Target Device of RA Sample Project Fix misprint
Rev.1.60	Dec.23.24	-	Deleted: Azure items Deleted: RZ items Added: Terms/Abbreviations Updated: Environments for RA, RX, RL78 Updated: ZMOD4510 Sensor specifications Updated: Specification of Sample Software Updated: Configuration Settings Updated: Guide for Changing the Target Device

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

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

### 1. Precaution against Electrostatic Discharge (ESD)

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

### 2. Processing at power-on

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

### 3. Input of signal during power-off state

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

### 4. Handling of unused pins

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

### 5. Clock signals

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

### 6. Voltage application waveform at input pin

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

### 7. Prohibition of access to reserved addresses

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

### 8. Differences between products

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

## Notice

- 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.
- 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.
- No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
- 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](http://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/](http://www.renesas.com/contact/).