RA Family, RX Family, RL78 Family

RRH62000 Sample Software Manual

Introduction

This Application Note describes sample software for RRH62000 All-In-One Air Quality sensor that operates on RA Family, RX Family and RL78 Family.

Target Device

RA2E1 Group

RA0E1 Group

RX140 Group

RL78/G23 Group

Target Sensor Board

All-In-One Air Quality Pmod Evaluation Board (QCIOT-RRH62000POCZ)

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 “6.5 Notes for Interrupt Signal Circuits”.
— RESET Signal Circuit: Refer to “6.6 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

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 RRH62000 All-In-One Air Quality 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 RRH62000 through the I2C in the MCU to measure air quality, acquire Measurement data, converts and calculates the acquired results.

1.1 Terms/Abbreviations

The terms and their abbreviations are listed below.

Table 1-1 List of Terms/Abbreviations

Terms	Abbreviation
RRH62000 Sensor Control Module	Sensor Control Module When MCU is RA Family, “rm_rrh62000” When MCU is RX Family, “r_rrh62000_rx” When MCU is RL78 Family, “r_rrh62000”
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” When MCU is RX Family, “r_riic_rx”, “r_sci_iic_rx” When MCU is RL78 Family, “r_iica_master”
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”
Pin No.1 (#1) of Renesas Pmod Type 6A Sensor Board	“IRQ#” (L output when an interrupt occurs)

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^2 studio 2024-10
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	v5.7.0
RTOS	FreeRTOS v10.6.1
Emulator	On board (J-LINK)
Interposer	Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ)
Sensor board	All-In-One Air Quality Pmod Evaluation Board (QCIOT-RRH62000POCZ)

Table 2-2 Memory Size Used in RA2E1

Area	Size (Non-OS) [Bytes]	Size (FreeRTOS) [Bytes]
ROM	2,461	4,290 (Note 1)
RAM	377	588

Note Memory size is calculated for sample code, RRH62000 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.

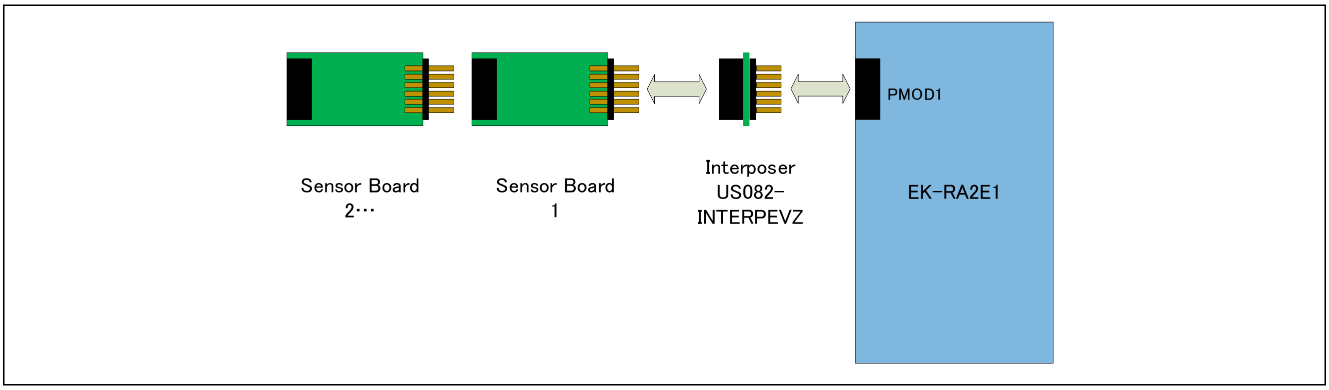


Figure 2-1 Hardware Connections for EK-RA2E1

(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^2 studio 2024-10
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	v5.7.0
Emulator	On board (J-LINK)
Interposer	Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ)
Sensor board	All-In-One Air Quality Pmod Evaluation Board (QCIOT-RRH62000POCZ)

Table 2-4 Memory Size Used in RA0E1

Area	Size (Non-OS) [Bytes]
ROM	1,933
RAM	365

Note Memory size is calculated for sample code, RRH62000 sensor control module, and COMMS_I2C.

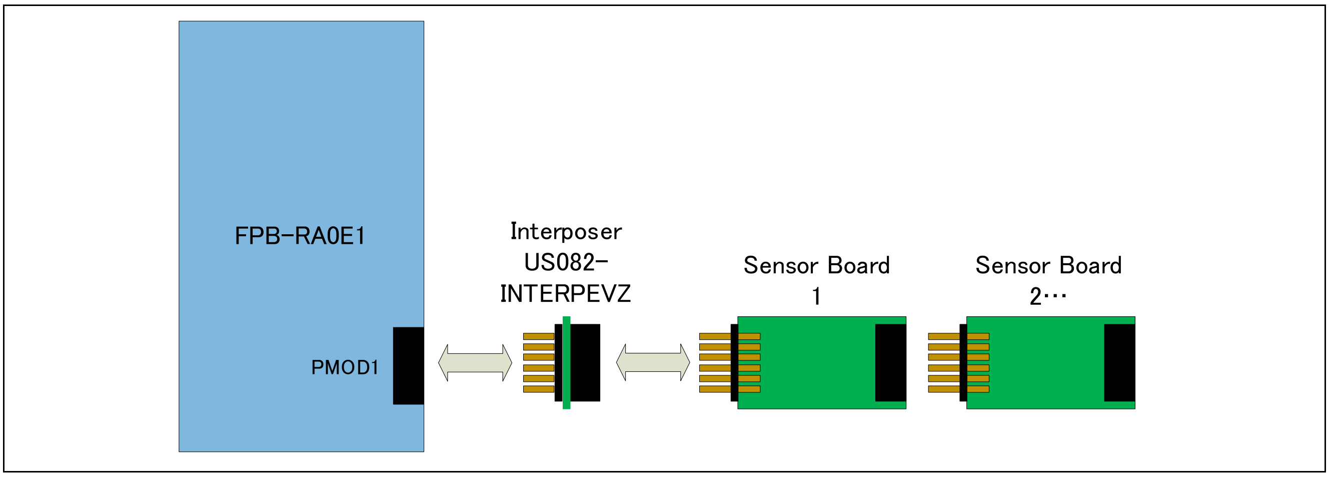


Figure 2-2 Hardware Connections for FPB-RA0E1

(3) Use of Interposer Board

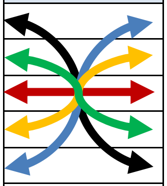
The Interposer Board is an I/F conversion board for connecting Pmod Type 6A sensors by switching the Pmod Type 2A/Type 3A connector of the SCI I/F to the Simple IIC function.

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 “6.5 Notes for Interrupt Signal Circuits”.

Application example: FPB-RA0E1 Pmod1 is applicable.

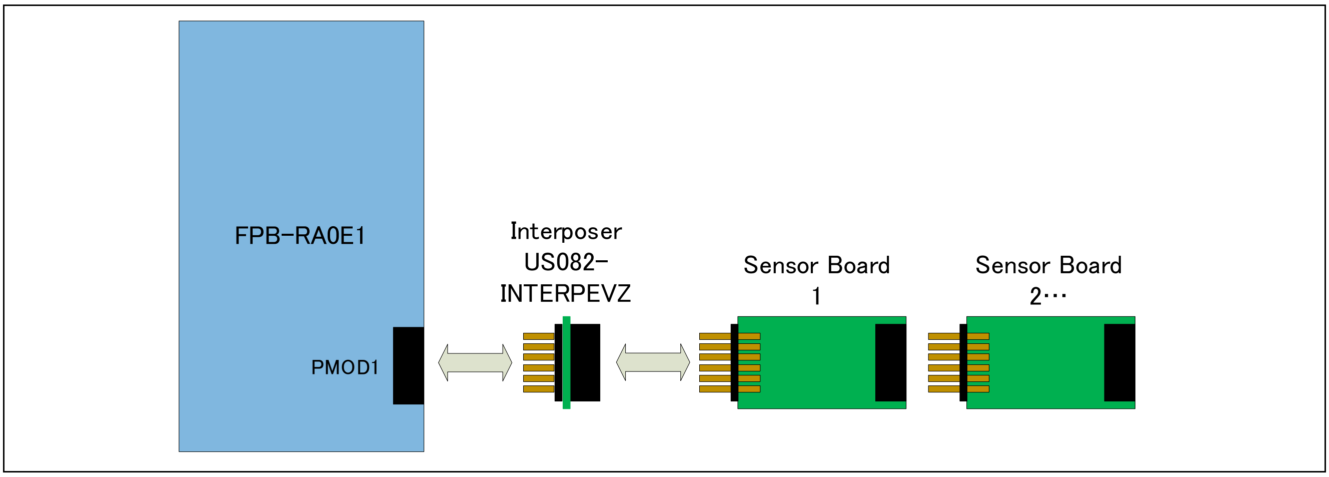


Figure 2-3 Hardware Connections for using IICA at Pmod Type 2A, Type 3A on FPB-RA0E1

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^2 studio 2024-10
C compiler	Renesas Electronics CC-RX V.3.06.00 GCC for Renesas RX 8.3.0.202411
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 RRH62000 Sensor Control Module (r_rrh62000_rx) v1.00 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 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	All-In-One Air Quality Pmod Evaluation Board (QCIOT-RRH62000POCZ)

Table 2-7 Memory Size Used in RX140

Area	Size (Non-OS) [Bytes] (CC-RX)	Size (FreeRTOS) [Bytes] (CC-RX)
ROM	2,689	2,752
RAM	384	371

Note Memory size is calculated for sample code, RRH62000 sensor control module, and COMMS_I2C. In RTOS, memory size does not include memory size of the thread.

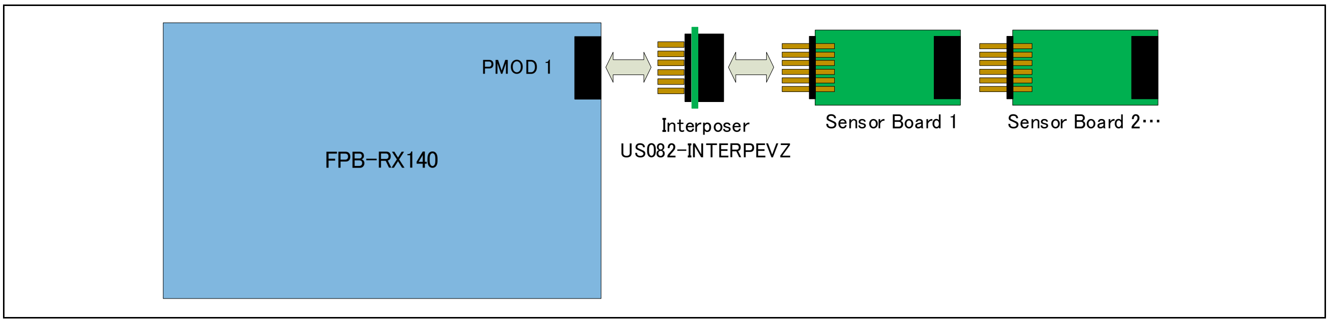


Figure 2-4 Hardware Connections for FPB-RX140

(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 e2 studio 2024-10
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 RRH62000 Sensor Middleware (r_rrh62000) v1.00 IIC Communication Driver Interface Middleware (r_comms_i2c) v1.11 IIC Communication (Master mode) v1.6.0 Interval Timer v1.5.0
Emulator	On board (COM Port)
Sensor board	All-In-One Air Quality Pmod Evaluation Board (QCIOT-RRH62000POCZ)

Table 2-9 Memory Size Used in RL78/G23

Area	Size (Non-OS) [Bytes] (CC-RL)
ROM	2,887
RAM	297

Note Memory size is calculated for sample code, RRH62000 sensor control module, and COMMS_I2C.

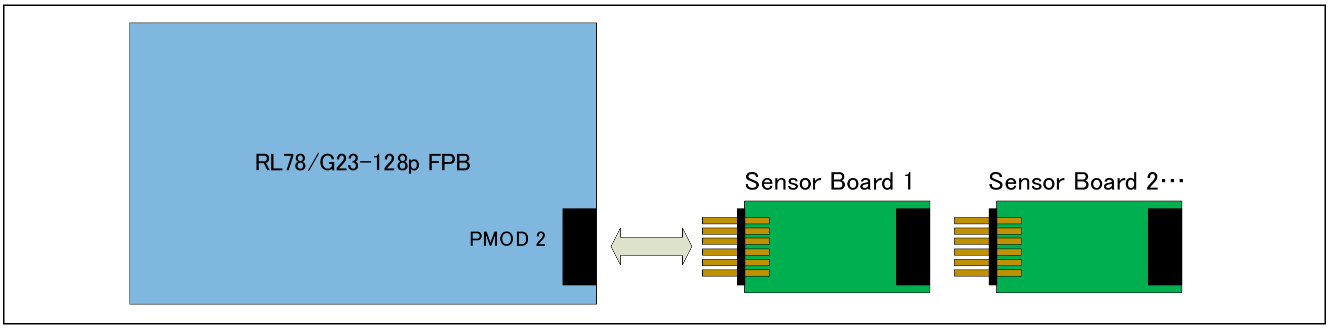


Figure 2-5 Hardware Connections for RL78/G23-128 FPB

(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. RRH62000 Sensor Specifications

3.1 Sensor Specifications Overview

The RRH62000 All-In-One Air Quality sensor is an integrated sensor module for the measurement of critical air quality parameters. Sensors for particulate matter (PM), total volatile organic compounds (TVOC), Indoor Air Quality Index (IAQ), estimation of carbon dioxide (eCO_2), humidity (RH), and temperature (T) are combined in a single package. Please refer to the RRH62000 datasheet for more information about the sensor module, including parameters that describe the module's characteristics.

3.2 Sensor Function and Methods

3.2.1 Conversion of Output Data – Firmware / API / Algorithms

To operate the RRH62000, a firmware provided by Renesas containing an API 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 RRH62000 webpage.

4. Specification of Sample Software

This sample software package contains a total of 8 projects:

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

For the FreeRTOS settings for RX family, refer to the FAQ.

4.1 Sample Software Structure

Figure 4-1 Layer diagram of Sample Software shows structure of sample software layer.

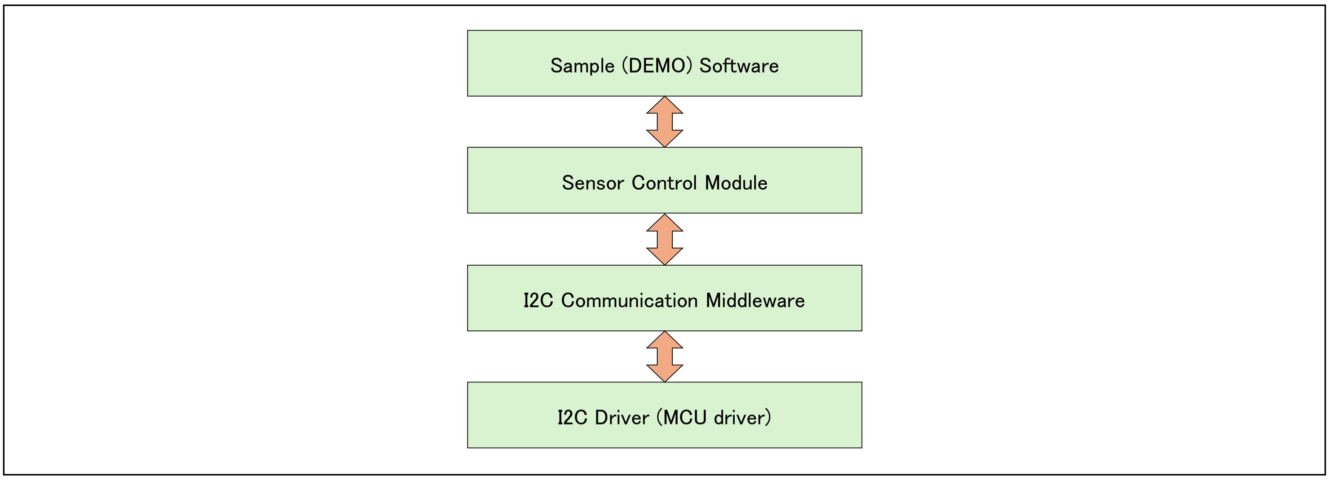


Figure 4-1 Layer diagram of Sample Software

4.2 Specification of Sensor Control Module API Functions

4.2.1 List of Sensor Control Module API Functions

The Sensor Control Module API includes the following functions.

For details on the Function API, see below.

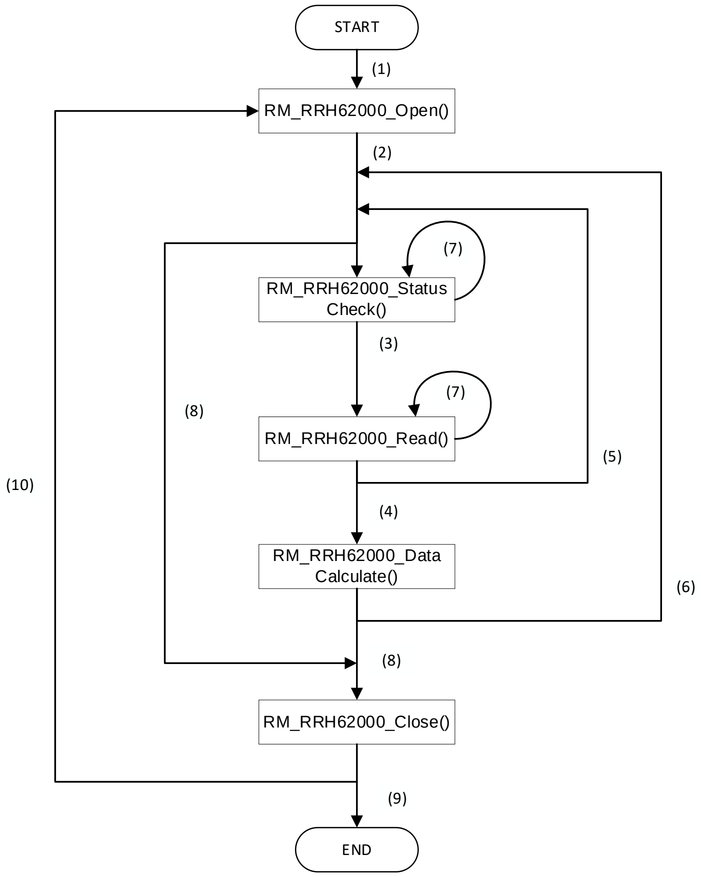
RA Flexible Software Package Documentation
Renesas Sensor Control Modules Firmware Integration Technology (R01AN5892)
Renesas Sensor Control Modules Software Integration System (R01AN6192)

Table 4-1 List of Sensor Control Module API Functions

Function	Feature
RM_RRH62000_Open()	Open the sensor
RM_RRH62000_Close()	Close the sensor
RM_RRH62000_Read()	Read measurement data
RM_RRH62000_StatusCheck()	Read status of the sensor
RM_RRH62000_DataCalculate()	Calculate Air environment data. values from measurement data
RM_RRH62000_FirmwareVersionGet()	Read firmware version of sensor
RM_RRH62000_AlgorithmVersionGet()	Read algorithm version of sensor

4.2.2 API Usage Guide

Figure 4-2 Transition of API Functions shows the transition diagram of functions calling order as the usage condition of API functions.



Flow No.	Function	Transition Case
(1)	Start RRH62000	not open RRH62000
(2)	Check status	open success
(3)	Measurement data Read	get status success
(4)	calculate	Measurement data Read success (callback state success)
(5)	Retry Check status	Measurement not finished
(6)	next measurement data	Data Calculate return success and next Measurement data Get
(7)	Retry Process	Waiting for response (callback none)
(8)	Close RRH62000	Measurement completed
(9)	Stop RRH62000	Close success
(10)	Restart RRH62000	Close success and user restart

Figure 4-2 Transition of API Functions

The calling conditions for each function are as follows:

- RM_RRH62000_Open(): (1) When starting RRH62000
  (10)Re-stating after RM_RRH62000_Close()
- RM_RRH62000_Close(): (8) When measurement is completed
- RM_RRH62000_Read(): (3) When acquiring measurement data
  (7) Retry due to waiting for data acquisition response
- RM_RRH62000_StatusCheck(): (2) Status check by polling
  (5) Retry due to measurement not finished
  (6) Get next measurement data
  (7) Retry due to waiting for data acquisition response
- RM_RRH62000_DataCalculate(): (4) Calculate data after RM_RRH62000_Read()

Note:

Since RM_RRH62000_Open() checks the state of the I2C driver, the I2C driver must be opened before the RM_RRH62000_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 a RTOS system, bus controlling by using semaphore by user is required if controlling the sensors at the same time in multiple threads/tasks.

4.3 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.

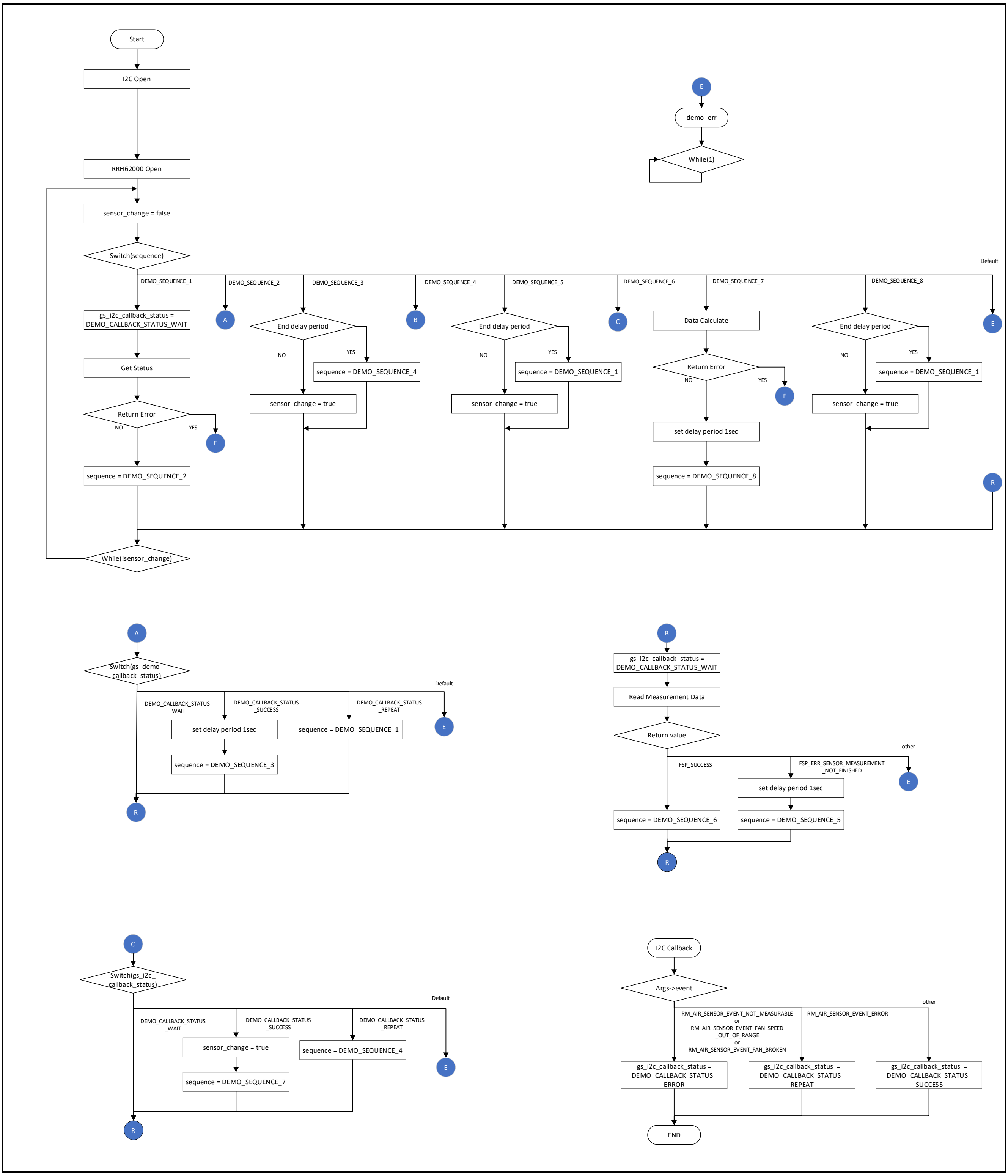

Start
I2C Open
RRH62000 Open
sensor_change = false
Switch(sequence)
Default
DEMO_SEQUENCE_1
DEMO_SEQUENCE_2
DEMO_SEQUENCE_3
DEMO_SEQUENCE_4
DEMO_SEQUENCE_5
DEMO_SEQUENCE_6
DEMO_SEQUENCE_7
DEMO_SEQUENCE_8
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_WAIT
Get Status
Return Error
NO
YES
sequence = DEMO_SEQUENCE_2
A
B
C
E
End delay period
sequence = DEMO_SEQUENCE_4
sensor_change = true
sequence = DEMO_SEQUENCE_1
Data Calculate
set delay period 1sec
sequence = DEMO_SEQUENCE_8
R
While(!sensor_change)
demo_err
While(1)
Switch(gs_demo_ callback_status)
DEMO_CALLBACK_STATUS _WAIT
DEMO_CALLBACK_STATUS _SUCCESS
DEMO_CALLBACK_STATUS _REPEAT
set delay period 1sec
sequence = DEMO_SEQUENCE_1
sequence = DEMO_SEQUENCE_3
Read Measurement Data
Return value
other
FSP_SUCCESS
FSP_ERR_SENSOR_MEASUREMENT _NOT_FINISHED
sequence = DEMO_SEQUENCE_6
sequence = DEMO_SEQUENCE_5
Switch(gs_i2c_ callback_status)
sensor_change = true
sequence = DEMO_SEQUENCE_7
sequence = DEMO_SEQUENCE_4
I2C Callback
Args->event
RM_AIR_SENSOR_EVENT_NOT_MEASURABLE or RM_AIR_SENSOR_EVENT_FAN_SPEED _OUT_OF_RANGE or RM_AIR_SENSOR_EVENT_FAN_BROKEN
RM_AIR_SENSOR_EVENT_ERROR
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ ERROR
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ REPEAT
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ SUCCESS
END

Figure 4-3 Flowchart of Main Processing in Non-OS Version of RRH62000 Sample Software

4.4 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.

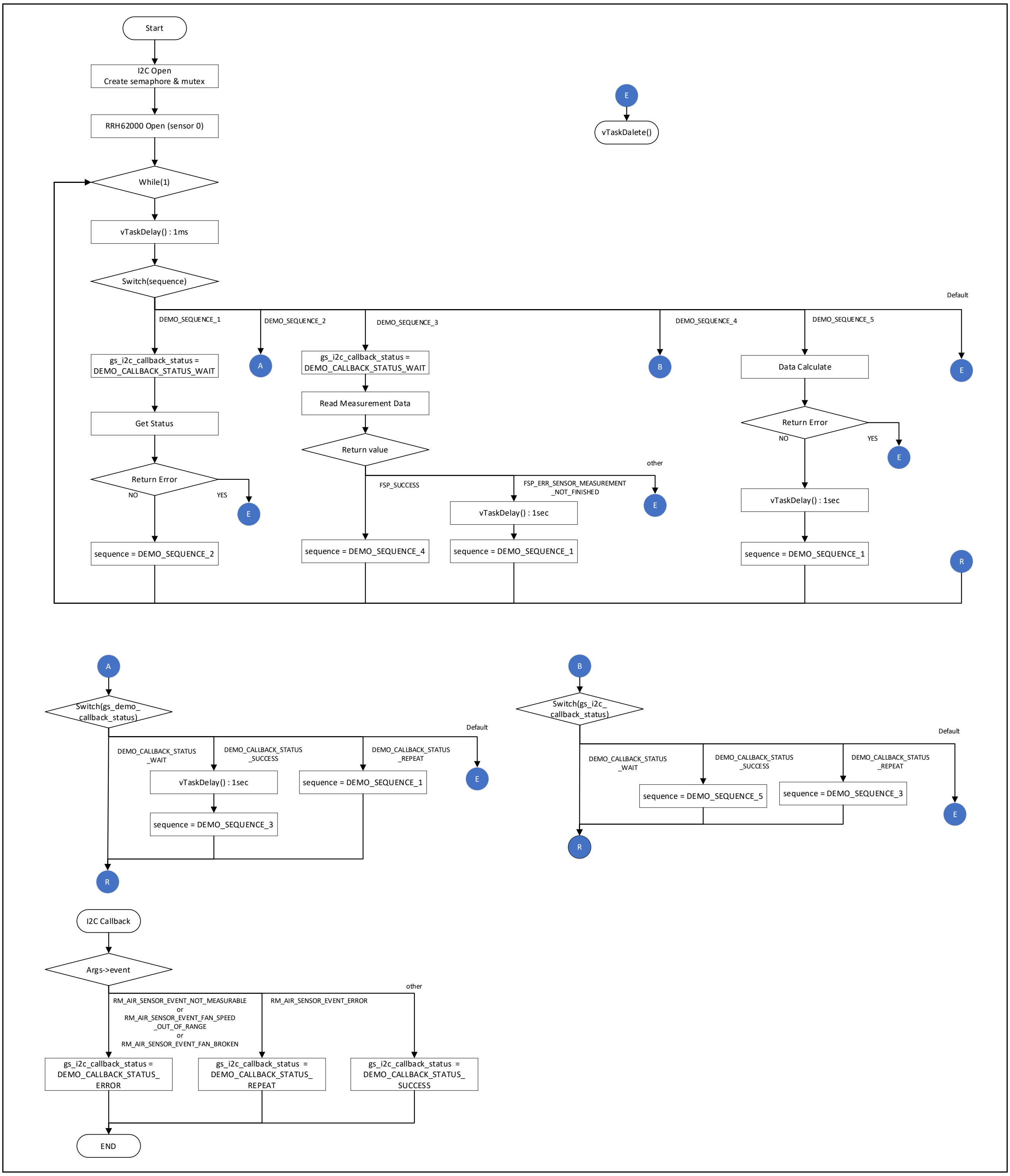

Start
I2C Open
Create semaphore & mutex
RRH62000 Open (sensor 0)
While(1)
vTaskDelay() : 1ms
Switch(sequence)
Default
DEMO_SEQUENCE_1
DEMO_SEQUENCE_2
DEMO_SEQUENCE_3
DEMO_SEQUENCE_4
DEMO_SEQUENCE_5
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_WAIT
A
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_WAIT
B
Data Calculate
E
Get Status
Read Measurement Data
Return Error
NO
YES
E
Return value
other
Return Error
NO
YES
E
FSP_SUCCESS
FSP_ERR_SENSOR_MEASUREMENT _NOT_FINISHED
vTaskDelay() : 1sec
E
vTaskDelay() : 1sec
sequence = DEMO_SEQUENCE_2
sequence = DEMO_SEQUENCE_4
sequence = DEMO_SEQUENCE_1
sequence = DEMO_SEQUENCE_1
R
E
vTaskDalete()
A
Switch(gs_demo_ callback_status)
Default
DEMO_CALLBACK_STATUS _WAIT
DEMO_CALLBACK_STATUS _SUCCESS
DEMO_CALLBACK_STATUS _REPEAT
vTaskDelay() : 1sec
sequence = DEMO_SEQUENCE_1
E
sequence = DEMO_SEQUENCE_3
R
B
Switch(gs_i2c_ callback_status)
Default
DEMO_CALLBACK_STATUS _WAIT
DEMO_CALLBACK_STATUS _SUCCESS
DEMO_CALLBACK_STATUS _REPEAT
sequence = DEMO_SEQUENCE_5
sequence = DEMO_SEQUENCE_3
E
R
I2C Callback
Args->event
other
RM_AIR_SENSOR_EVENT_NOT_MEASURABLE or RM_AIR_SENSOR_EVENT_FAN_SPEED _OUT_OF_RANGE or RM_AIR_SENSOR_EVENT_FAN_BROKEN
RM_AIR_SENSOR_EVENT_ERROR
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ ERROR
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ REPEAT
gs_i2c_callback_status = DEMO_CALLBACK_STATUS_ SUCCESS
END

Figure 4-4 Flowchart of Main Processing in OS Version of RRH62000 Sample Software

5. 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.

5.1 RRH62000 Sensor Control Module Settings

5.1.1 RA Family

Select the **"rm_rrh62000"** stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

Table 5-1 RRH62000 Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP) Enabled Disabled	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
Module g_rrh62000_sensor0 RRH62000 All-in-one Air Quality Module (rm_rrh62000)		
Name	g_rrh62000_sensor0	Specify the name of the module.
Number of moving average [times]	10	Specify the number of moving average. Values range from 1 to 60.
Fan speed control [%]	86	Specify the fan speed control. Values range from 60 to 100.
Comms I2C Callback	rrh62000_comms_i2c_callback	Specify the name of the user callback function. When "NULL" is specified, no callback function is used.

5.1.2 RX Family

Select the **"r_rrh62000_rx"** component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the "Configure" panel.

Table 5-2 RRH62000 Settings for RX Family

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default Enabled Disabled	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
Number of RRH62000 Sensors	1	Specify the number of RRH62000 sensor
Moving Average number of RRH62000 Sensor{x} (x = 0)	10	Specify the number of moving average. Values range from 1 to 60.
Fan speed control of RRH62000 Sensor{x} (x = 0)	86	Specify the fan speed control. Values range from 60 to 100.
I2C Communication device No. for RRH62000 sensor device{x} (x = 0)	I2C Communication Device{x} (x = 0 - 15)	Specify the communications device number to be used by the sensor.
I2C callback function for RRH62000 sensor device{x} (x = 0)	rrh62000_user_i2c_callback0	Specify the name of I2C callback function. When "NULL" is specified, no callback function is used.

5.1.3 RL78 Family

Select the **“r_rrh62000”** component on the “Component” tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

Table 5-3 RRH62000 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 RRH62000 Sensors	1	Specify the number of RRH62000 sensor
Moving Average number of RRH62000 Sensor{x} (x = 0)	10	Specify the number of moving average. Values range from 1 to 60.
Fan speed control of RRH62000 Sensor{x} (x = 0)	86	Specify the fan speed control. Values range from 60 to 100.
I2C Communication device No. for RRH62000 sensor device{x} (x = 0)	I2C Communication Device{x} (x = 0 - 4)	Specify the communications device number to be used by the sensor.
I2C callback function for RRH62000 sensor device{x} (x = 0)	rrh62000_user_i2c_callback0	Specify the name of I2C callback function. When "NULL" is specified, no callback function is used.

5.2 I2C Communication Middleware (COMMS_I2C) Settings

5.2.1 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 5-4 COMMS_I2C Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP) Enabled Disabled	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
Module g_comms_i2c_device0 I2C Communication Device (rm_comms_i2c)		
Name	g_comms_i2c_device0	Specify the name of the module.
Semaphore Timeout	0xFFFFFFFF	For an RTOS project, specify the time of semaphore timeout.
Slave Address	0x69	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	rm_rrh62000_comms_i2c_callback	Specify the name of the user callback function. No setting is required as this will be overwritten by the Sensor Control module.
Module g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)		
Name	g_comms_i2c_bus0	Specify the name of the I2C module.
Bus Timeout	0xFFFFFFFF	Specify the time of I2C bus timeout.
Semaphore for blocking	Unuse Use	For an RTOS project, enable or disable the blocking processing.
Recursive Mutex for Bus	Unuse Use	For an RTOS project, enable or disable the recursive operation when blocking processing is enabled.
Channel	0	Specify the channel number to be used. This setting is valid only when the I2C driver is “r_iic_master”. When using other I2C drivers, this setting is invalid.
Rate	Standard Fast-mode Fast-mode plus	Specify the bit rate. When using RRH62000, 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 “r_iica_master”, specify “Standard”due to the electrical characteristics of IICAx. When using “r_sci_i2c”, this setting is invalid.

5.2.2 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 5-5 COMMS_I2C Settings for RX Family

Configurable Item	Value	Description
Configurations		
Parameter Checking	System Default Enabled Disabled	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
Number of I2C Shared Buses	Unused 1 2 - 16	Specify the number of communications bus lines that can be connected.
Number of I2C Communication Devices	Unused 1 2 - 16	Specify the number of I2C device that can be connected.
Blocking operation supporting with RTOS	Disabled Enabled	For an RTOS project, enable or disable the blocking operation.
Bus lock operation supporting with RTOS	Disabled Enabled	For an RTOS project, enable or disable the bus lock operation.
I2C Driver Type for I2C Shared bus{x} (x = 0 - 15)	RIIC SCI IIC Not selected	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.
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 RRH62000, specify 0x69.
Address mode for I2C Communication device{x} (x = 0 - 15)	7 bit address mode	Specify the slave address mode. When using RRH62000, 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_ rrh62000_rx, specify rm_ rrh62000_callback{y} (y = 0).
Timeout for the blocking bus of I2C Communication device{x} (x = 0 - 15)	0xFFFFFFFF	Specify the time of I2C bus blocking timeout.

5.2.3 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 5-6 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 RRH62000, 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 RRH62000, specify 0x69.
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_rrh62000, specify rm_rrh62000_callback{y} (y = 0).

5.3 I2C Driver Settings

5.3.1 RA Family

Select the **“r_iic_master”**, **“r_sci_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.

Simplified I2C using Serial Array Unit (SAU) cannot be used because the clock stretch function is not supported.

(1) r_iic_master

Table 5-7 r_iic_master Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP)	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
	Enabled	
	Disabled	
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. No setting is required as this will be overwritten by COMMS_I2C.
	Fast-mode	
	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 salve 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. Use the "Pins" tabbed page to modify the pin configuration.
SCL	Pxxx	

(2) r_sci_i2c

Table 5-8 r_sci_i2c Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP) Enabled Disabled	Specify the include parameter check processing in code. When “Disabled” is specified, excluding in the code. When “Enabled” is specified, including in the code.
DTC on Transmission and Reception	Enabled Disabled	Specify whether to use the DTC for transmission and reception.
10-bit slave addressing	Enabled Disabled	Specify whether to support 10-bit addressing for the slave address. No setting is required as this will be overwritten by COMMS_I2C.
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 10-bit	Specify the salve address mode for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
Rate	Standard Fast-mode	Specify the bit rate. When using RRH62000, 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.
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 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	Specify the noise filter to be used for input signals.
Bit Rate Modulation	Enable Disable	Enable or disable the bit rate modulation function.
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) Priority 1 Priority 2 Priority 3	Specify the interrupt priority level of the I2C bus driver.
RX Interrupt Priority Level [Only used when DTC is enabled]	Priority 0 (highest) Priority 1 Priority 2 Priority 3 Disabled	When using DTC, specify the priority level of the reception interrupt.
Pins		
SDA	Pxxx	The pin numbers to be used by the driver are displayed. Use the "Pins" tabbed page to modify the pin configuration.
SCL	Pxxx	

(3) r_iica_master

Table 5-9 r_iica_master Settings for RA Family

Configurable Item	Value	Description
Common		
Parameter Checking	Default (BSP) Enabled Disabled	Specify the include parameter check processing in code. When "Disabled" is specified, excluding in the code. When "Enabled" is specified, including in the code.
10-bit slave addressing	Enabled Disabled	Specify whether to support 10-bit addressing for the slave address. No setting is required as this will be overwritten by COMMS_I2C.
Enable Single Channel	Enabled Disabled	Enable single channel to reduce code size if only one channel is to be configured for IICA.
Module g_iica_master0 IICA Master (r_iica_master)		
Name	g_iica_master0	Specify the name of the module.
Channel	0	Specify the IICA channel.
Rate	Standard Fast-mode Fast-mode plus	Specify the bit rate. No setting is required as this will be overwritten by COMMS_I2C.
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 Disabled	Specify whether to use the digital filter.
Address Mode	7-Bit 10-Bit	Specify the salve address mode for the device to be connected. No setting is required as this will be overwritten by COMMS_I2C.
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 Disabled	Specify whether to use the communication reservation.
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) Priority 1 Priority 2 Priority 3	Specify the interrupt priority level of the I2C bus driver.
Pins		
SCLAx	Pxxx	The pin numbers to be used by the driver are displayed. Use the "Pins" tabbed page to modify the pin configuration.
SDAAx	Pxxx	

5.3.2 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 5-10 r_riic_rx Settings for RX Family

Configurable Item	Value	Description
Configurations		
Set parameter checking enable	System Default Not Include	Specify the include parameter check processing in code. When "Not" is specified, excluding in the code. When "Include" is specified, including in the code.
MCU supported channels for CH{x} (x = 0 - 2) - When x = 0, the default value = "Supported" - When x = a number other than 0, the default value = "Not supported"	Not supported 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.
CH{x} RIIC bps(kbps) (x = 0 - 2)	400	Specify the bit rate. When using RRH62000, 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.
SCL rise time in Standard Mode	1000E-9	Specify the SCL rise time (s) in Standard Mode (up to 100 kbps).
SCL fall time in Standard Mode	300E-9	Specify the SCL fall time (s) in Standard Mode (up to 100 kbps).
SCL rise time in Fast Mode	300E-9	Specify the SCL rise time (s) in Fast Mode (up to 400 kbps).
SCL fall time in Fast Mode	300E-9	Specify the SCL fall time (s) in Fast Mode (up to 400 kbps).
SCL rise time in Fast Mode Plus	120E-9	Specify the SCL rise time (s) in Fast Mode Plus (up to 1 Mbps).
SCL fall time in Fast Mode Plus	120E-9	Specifies the SCL fall time (s) in Fast Mode Plus (up to 1 Mbps).
Digital filter for CH{x} (x = 0 - 2)	Not One IIC phi Two IIC phi Three IIC phi Four IIC phi	Specify the digital filter for input signals. When "Not" is specified, disable the digital filter.
Setting port setting processing	Not include port setting 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.
Master arbitration lost detection function for CH{x} (x = 0 - 2)	Unused Used	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.
Address {y} format for CH{x} (x = 0 - 2, y = 0 - 2)	Not 7 bit address format 10 bit address format	Specify whether to support 7-bit addressing or 10-bit addressing for the slave address. When using RRH62000, select "7 bit address format". Do not connect devices with different address formats on the same bus.

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.
Resources		
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 5-11 r_sci_iic_rx Settings for RX Family

Configurable Item	Value	Description
Configurations		
Set parameter checking enable	System Default Not Include	Specify the include parameter check processing in code. When “Not” is specified, excluding in the code. When “Include” is specified, including in the code.
MCU supported channels for CH{x} (x = 0 - 12)	Not supported Supported	Specify whether to support the operation of channel.
SCI IIC bitrate (bps) for CH{x} (x = 0 - 12)	384000	Specify the bit rate. When using RRH62000, 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 Level 2 … Level 14 Level 15 (highest)	Specify the interrupt priority level.
Digital noise filter (NFEN bit) for CH{x} (x = 0 - 12)	Disable Enable	Specify whether to use the digital noise filter.
Noise Filter Setting Register (NFCS bit) for CH{x} (x = 0 - 12)	The clock divided by 1 The clock divided by 2 The clock divided by 4 The clock divided by 8	Specify the sample clock of the digital noise filter.
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 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
Resources		
SSCLx Pins	Checked Unchecked	Specify the pins to be used. Set the pin to “Checked”.
SSDAx Pins	Checked Unchecked	

5.3.3 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.

Simplified I2C using Serial Array Unit (SAU) cannot be used because the clock stretch function is not supported.

(1) IICAx

Table 5-12 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. When using RRH62000, 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	
	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	

6. 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.

The RRH62000 does not have an interrupt signal output pin or a RESET signal input pin, so there is no need to control these signals.

However, on the RRH62000 Pmod Board, there is a jumper pin for a pull-up resistor on the IRQ# signal line. Please leave the jumper pin open. See also the following.

— Interrupt Signal Circuit: Refer to “6.5 Notes for Interrupt Signal Circuits”.

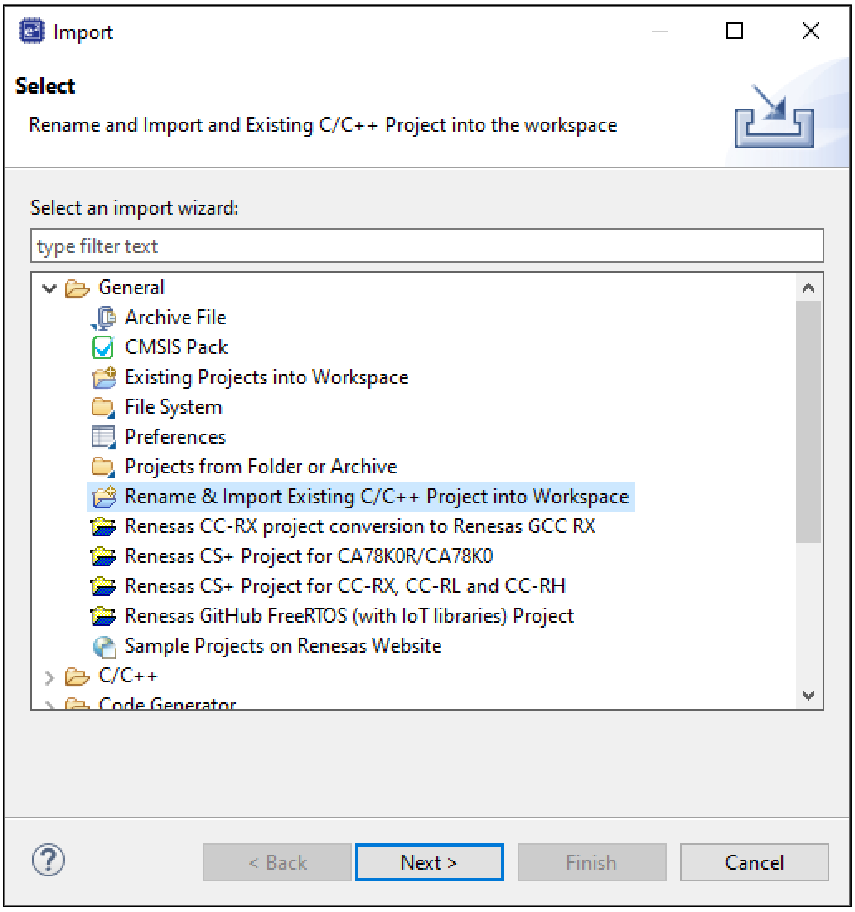
6.1 Importing Sample Project

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.



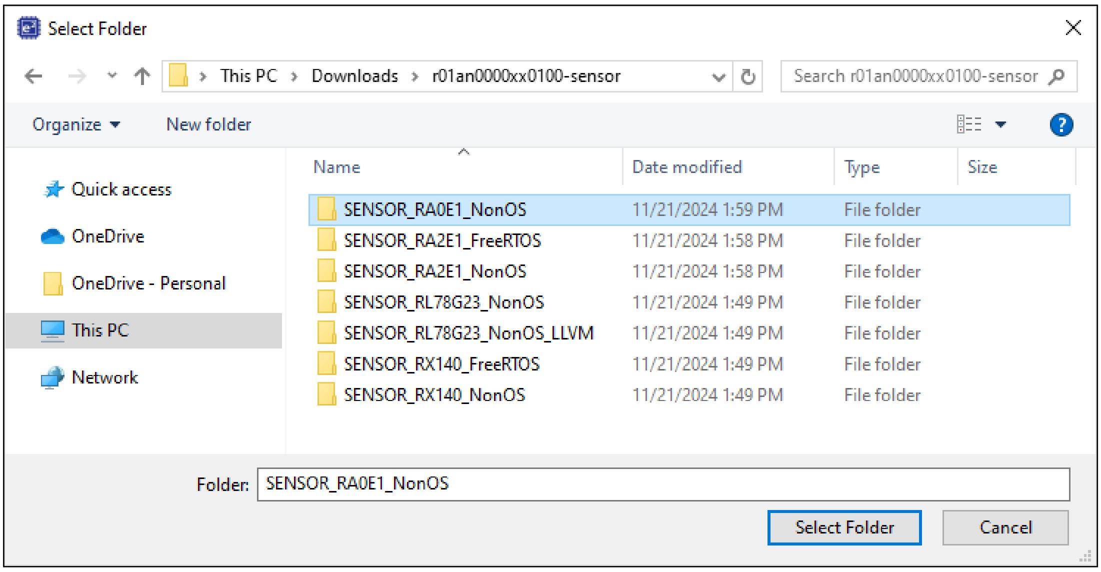
Select

Rename and Import and Existing C/C++ Project into the workspace

Import
Select an import wizard:
type filter text
General
Archive File
CMSIS Pack
Existing Projects into Workspace
File System
Preferences
Projects from Folder or Archive
Rename & Import Existing C/C++ Project into Workspace
Renesas CC-RX project conversion to Renesas GCC RX
Renesas CS+ Project for CA78K0R/CA78K0
Renesas CS+ Project for CC-RX, CC-RL and CC-RH
Renesas GitHub FreeRTOS (with IoT libraries) Project
Sample Projects on Renesas Website
C/C++
Code Generator
< Back
Next >
Finish
Cancel

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

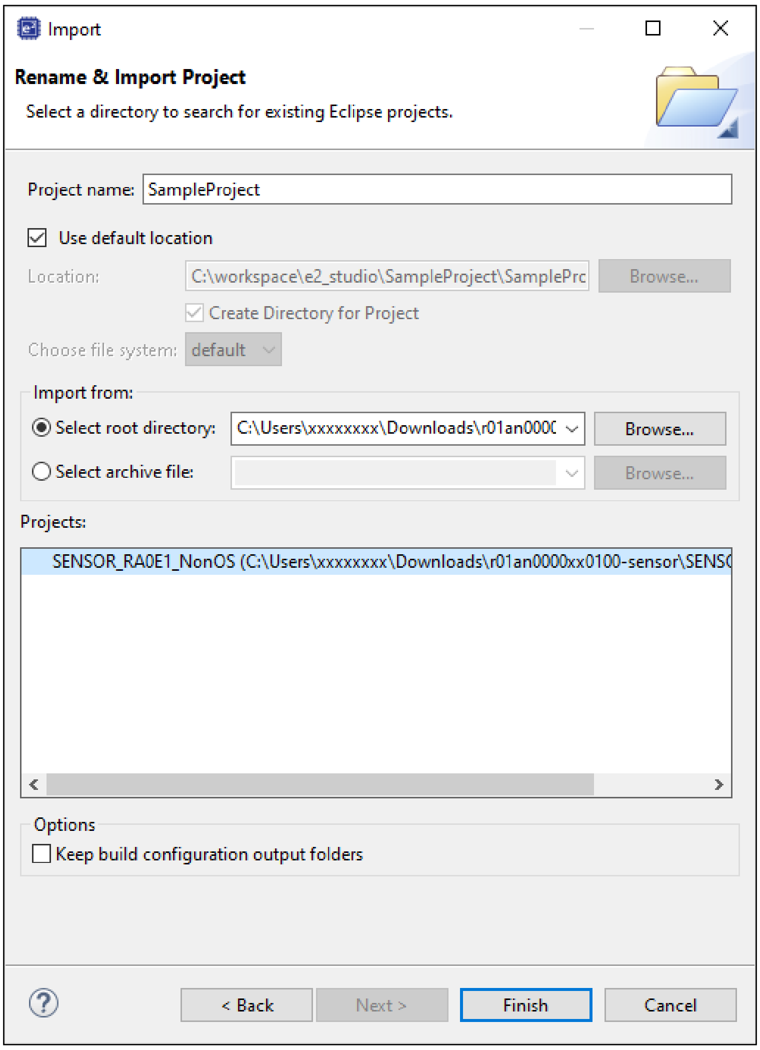
Select the folder of the original project for the current device from a list of imported sample projects and press the [Select Folder] button.



Name	Date modified	Type	Size
SENSOR_RA0E1_NonOS	11/21/2024 1:59 PM	File folder	
SENSOR_RA2E1_FreeRTOS	11/21/2024 1:58 PM	File folder	
SENSOR_RA2E1_NonOS	11/21/2024 1:58 PM	File folder	
SENSOR_RL78G23_NonOS	11/21/2024 1:49 PM	File folder	
SENSOR_RL78G23_NonOS_LLVM	11/21/2024 1:49 PM	File folder	
SENSOR_RX140_FreeRTOS	11/21/2024 1:49 PM	File folder	
SENSOR_RX140_NonOS	11/21/2024 1:49 PM	File folder	

Folder: SENSOR_RA0E1_NonOS

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



Rename & Import Project

Select a directory to search for existing Eclipse projects.

Project name: SampleProject

☑ Use default location

Location: C:\workspace\e2_studio\SampleProject\SamplePrc

☑ Create Directory for Project

Choose file system: default

Import from:

● Select root directory: C:\Users\xxxxxxxx\Downloads\r01an000(

○ Select archive file:

Projects:

SENSOR_RA0E1_NonOS (C:\Users\xxxxxxxx\Downloads\r01an0000xx0100-sensor\SENS(

Options

☐ Keep build configuration output folders

6.2 RA Sample Project

After importing the sample projects, follow the steps below. Please refer to “6.1 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 "RRH62000_RA2E1_NonOS":
  Pmod1 (Type 2A/3A: SCI0)
  → Pmod1 (Option Type 6A: IIC1) or Pmod2 (Type 2A: SCI0) of the EK-RA6M4 board

6.2.1 Modifying Settings of FSP Configurator

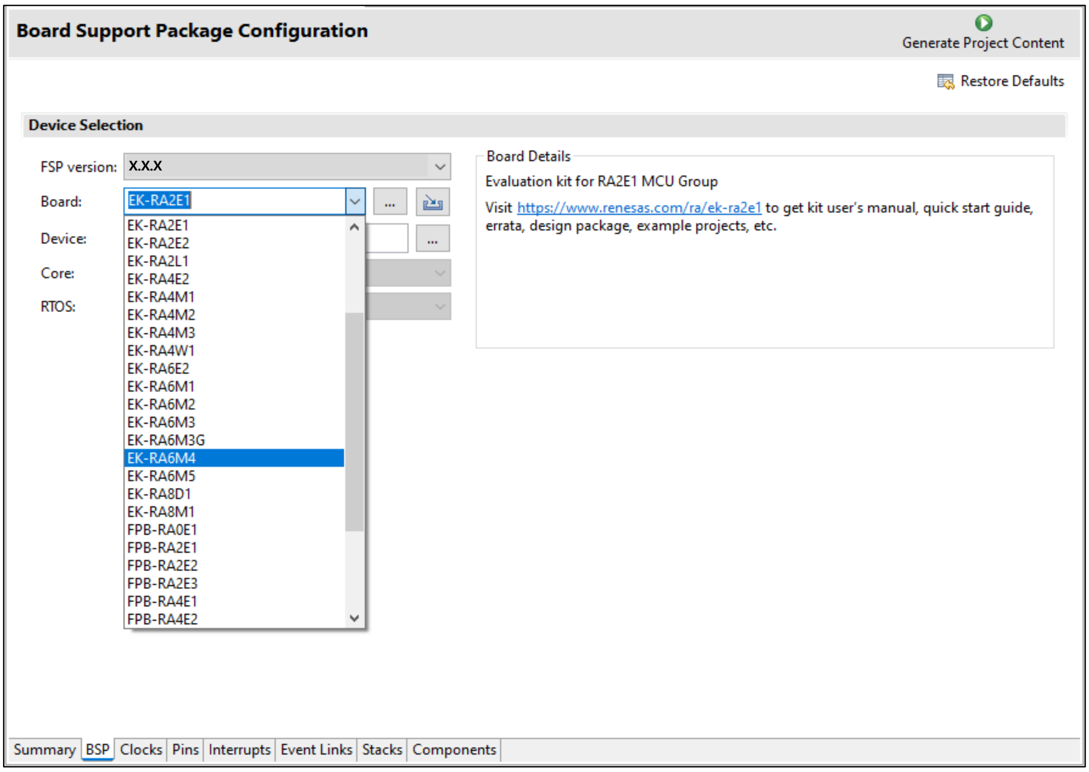
Double-click on "Configuration.xml" in the project tree to open the FSP Configurator.

(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.



Board Support Package Configuration

Device Selection

FSP version: X.X.X

Board: EK-RA2E1

Device:

Core:

RTOS:

Board Details

Evaluation kit for RA2E1 MCU Group

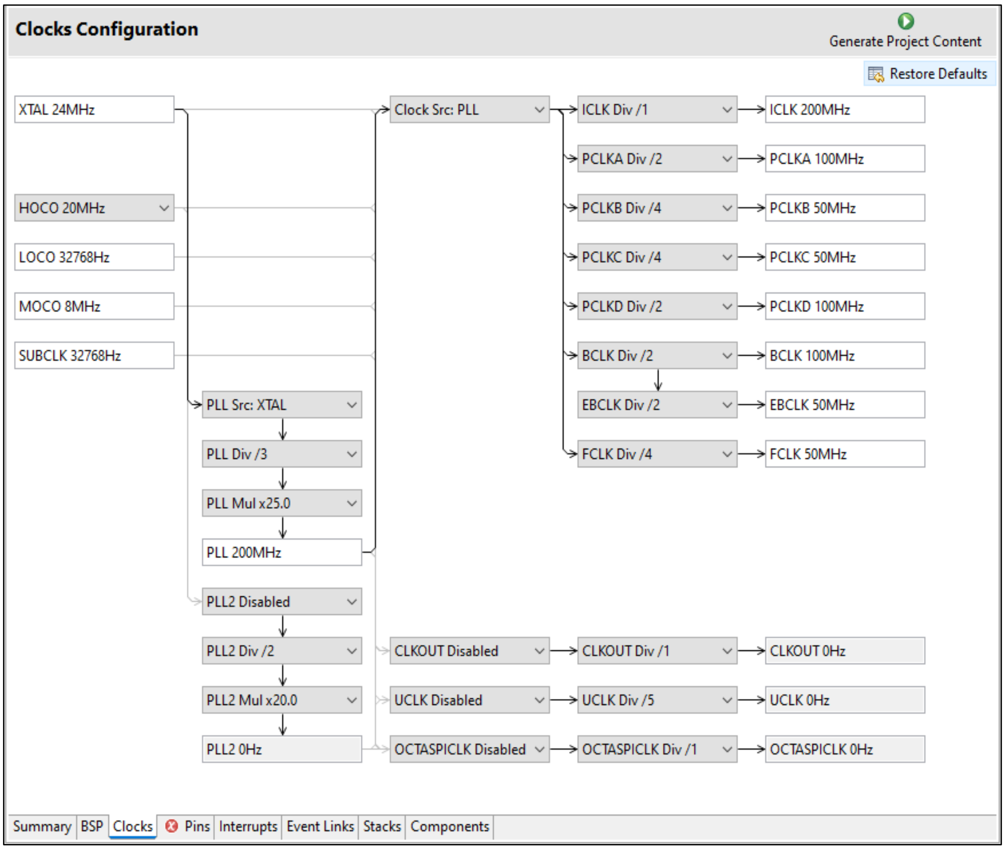
Visit https://www.renesas.com/ra/ek-ra2e1 to get kit user's manual, quick start guide, errata, design package, example projects, etc.

(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.



Clocks Configuration

XTAL 24MHz

HOCO 20MHz

LOCO 32768Hz

MOCO 8MHz

SUBCLK 32768Hz

PLL Src: XTAL

PLL Div /3

PLL Mul x25.0

PLL 200MHz

PLL2 Disabled

PLL2 Div /2

PLL2 Mul x20.0

PLL2 0Hz

Clock Src: PLL

ICLK Div /1 → ICLK 200MHz

PCLKA Div /2 → PCLKA 100MHz

PCLKB Div /4 → PCLKB 50MHz

PCLKC Div /4 → PCLKC 50MHz

PCLKD Div /2 → PCLKD 100MHz

BCLK Div /2 → BCLK 100MHz

EBCLK Div /2 → EBCLK 50MHz

FCLK Div /4 → FCLK 50MHz

CLKOUT Disabled → CLKOUT Div /1 → CLKOUT 0Hz

UCLK Disabled → UCLK Div /5 → UCLK 0Hz

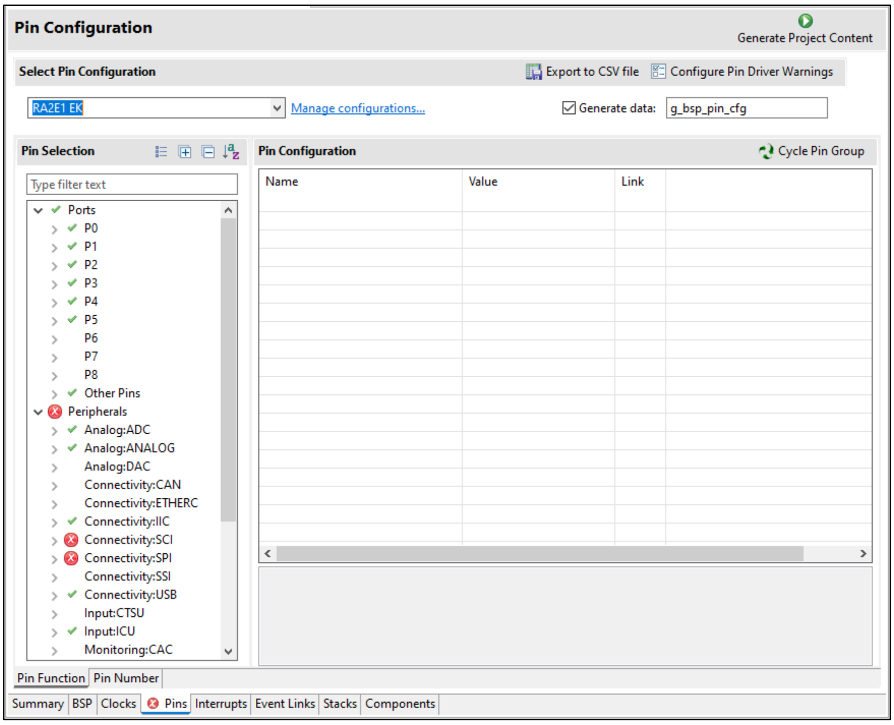
OCTASPICLK Disabled → OCTASPICLK Div /1 → OCTASPICLK 0Hz

(3) Pins

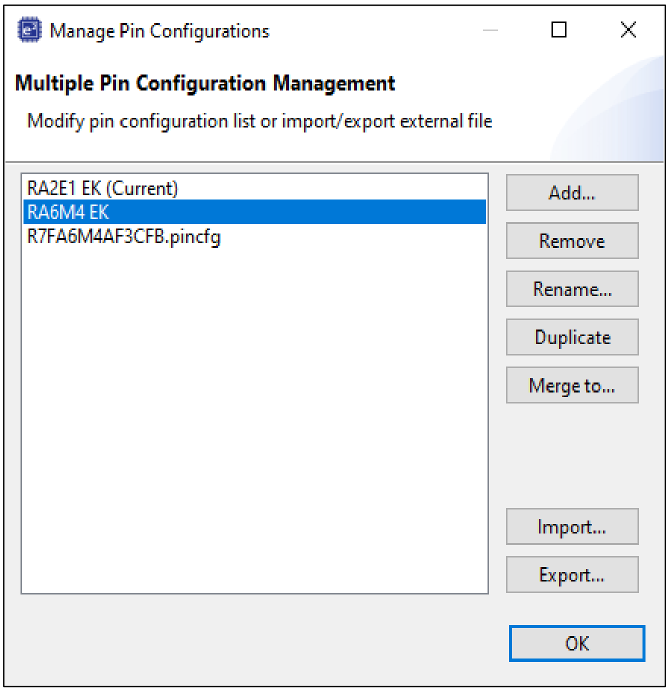
(a) Changing Board

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.


Manage Pin Configurations
Multiple Pin Configuration Management
Modify pin configuration list or import/export external file
RA2E1 EK (Current)
RA6M4 EK
R7FA6M4AF3CFB.pincfg
Add...
Remove
Rename...
Duplicate
Merge to...
Import...
Export...
OK

(b) Changing I2C I/F Pins

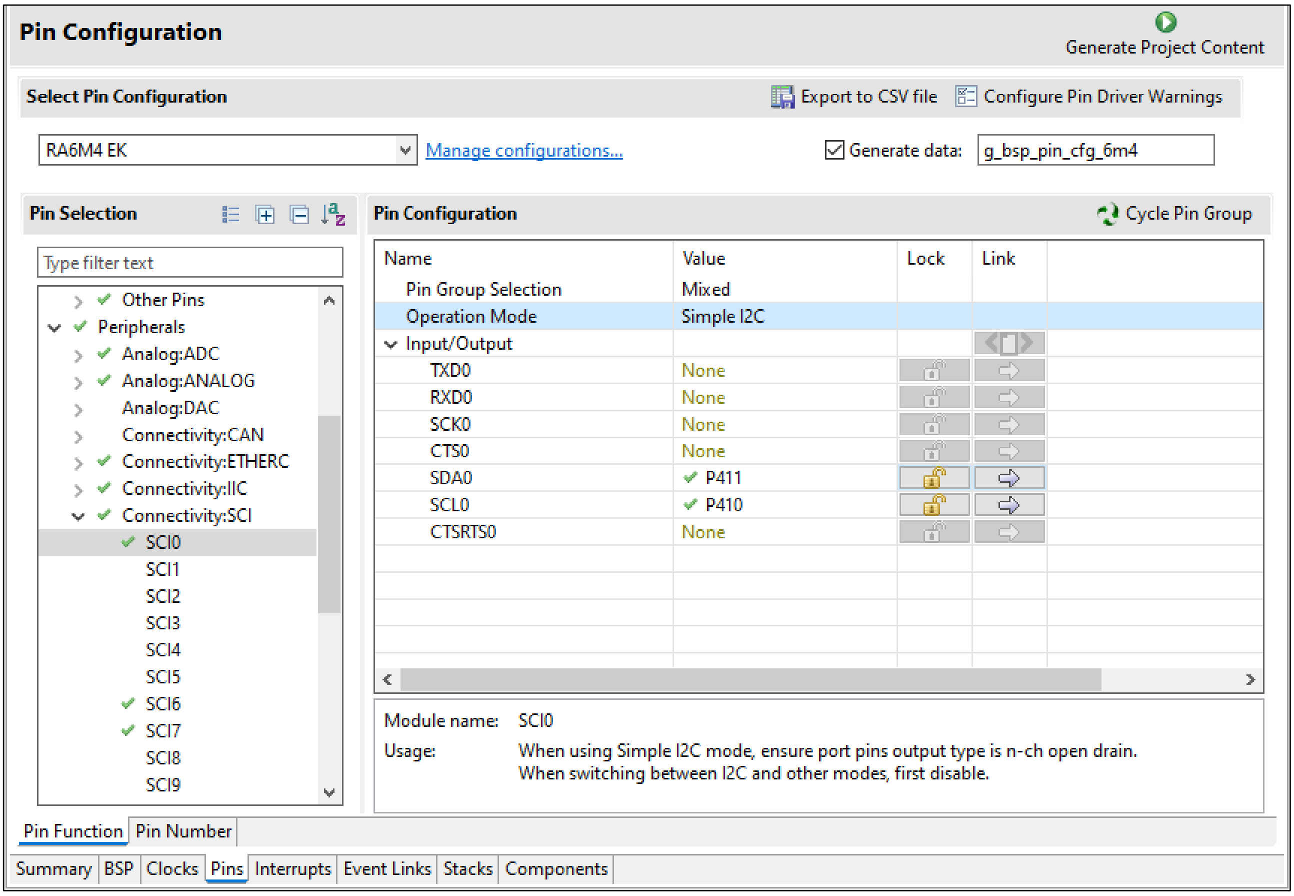
However, the assignment on the above “(a)Changing Board” will apply the SPI communication pin settings that support Pmod Type 2A on the EK-RA6M4 board.

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.



RA6M4 EK

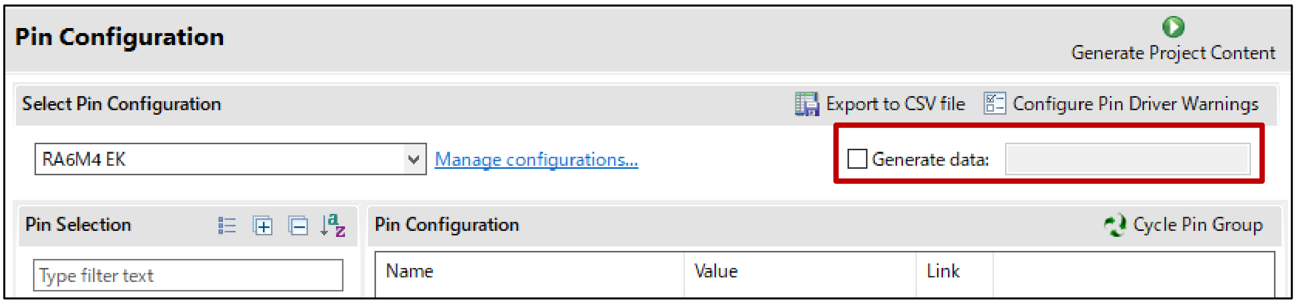
☑ Generate data: g_bsp_pin_cfg_6m4

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.

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



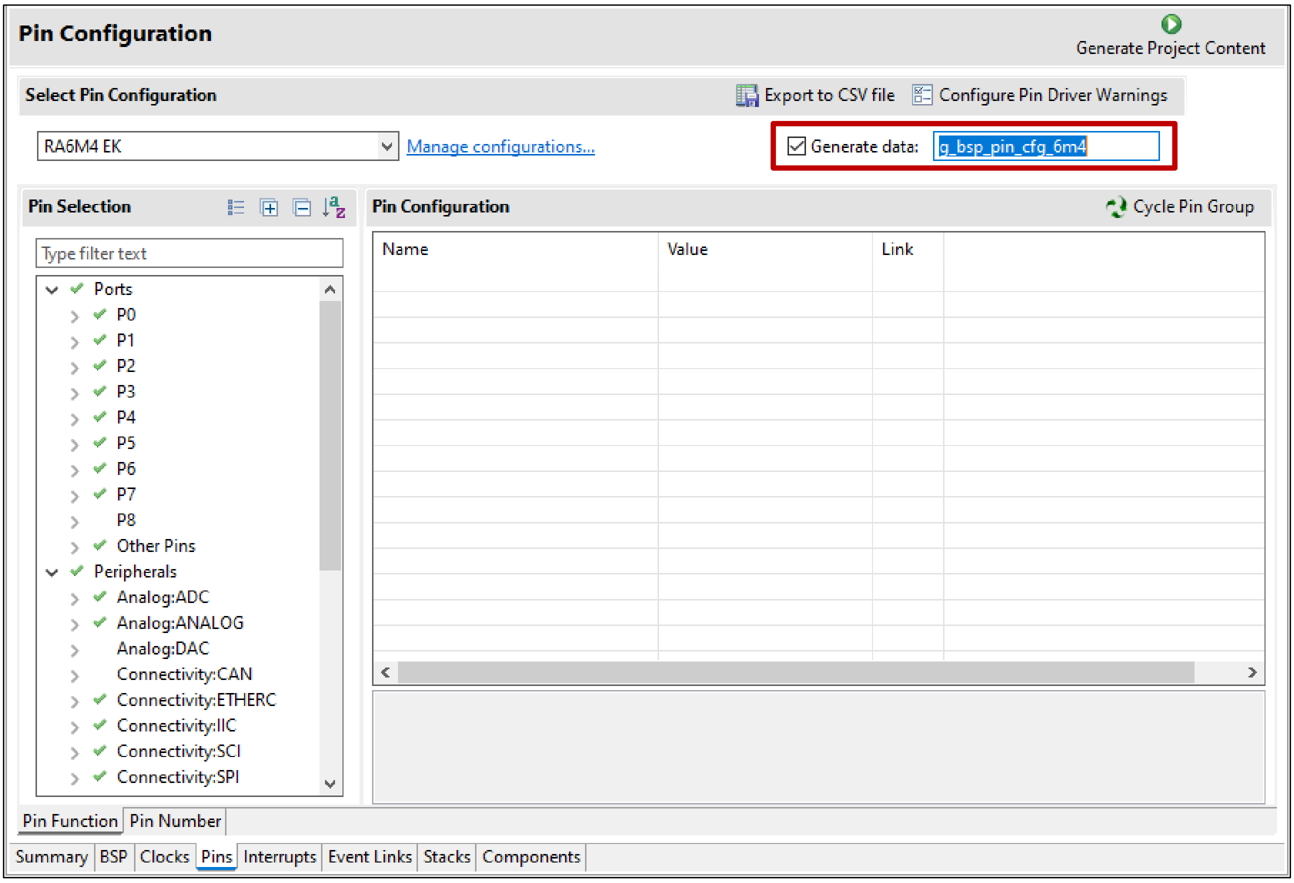
RA6M4 EK

□ Generate data:

To enable generation of pin settings, check [Generate data] check-box and enter a desired name in the text field.

The entered name is linked to the pin configuration, therefore must use a unique name that does not duplicate with other pin configurations.

The following is an example named “g_bsp_pin_cfg_6m4”.



Select Pin Configuration: RA6M4 EK

☑ Generate data: g_bsp_pin_cfg_6m4

(4) Stacks

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

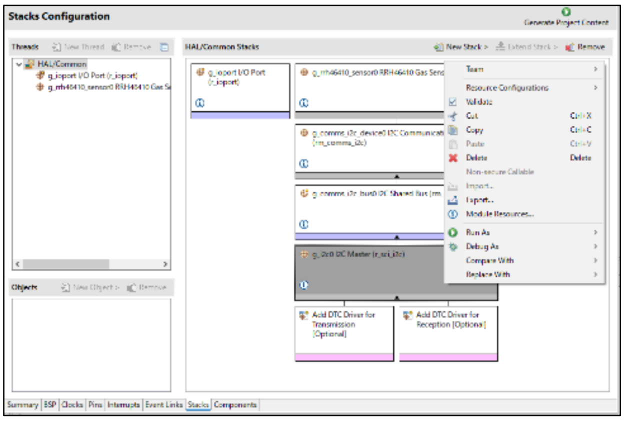
If an error is displayed, modify the specified item according to the displayed error.

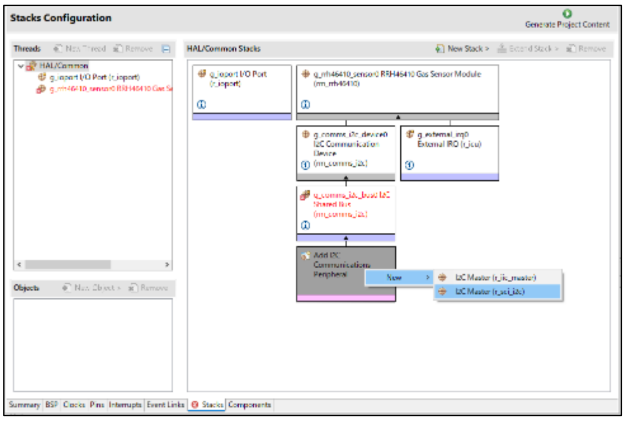
(a) Changing COMMS_I2C Settings and I2C Driver Settings

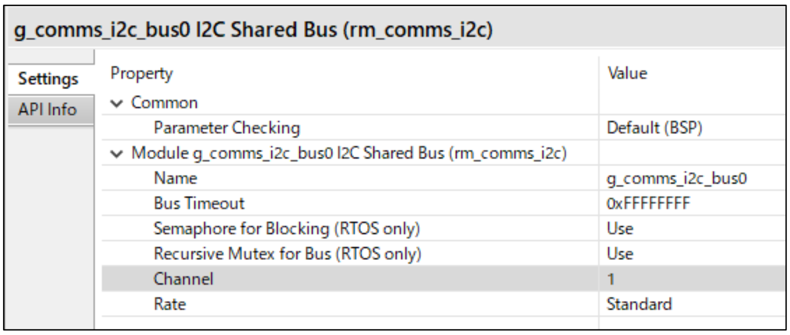
Modify the settings of COMMS_I2C and I2C driver according to the specifications of the target board. To use the pins of the I2C I/F, delete the unnecessary stack and add the new stack to use.

Table 6-1 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



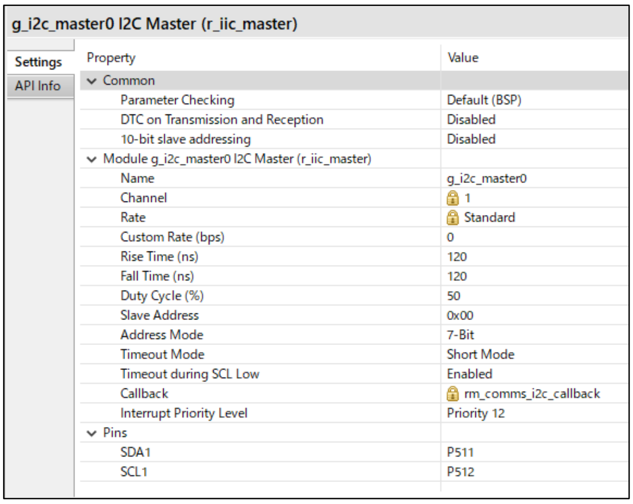

Stacks Configuration
Generate Project Content
Threads
New Thread
Remove
HAL/Common
g_ioport I/O Port (r_ioport)
g_rrh46410_sensor0 RRH46410 Gas Se
Objects
New Object >
Remove
HAL/Common Stacks
New Stack >
Extend Stack >
Remove
g_ioport I/O Port (r_ioport)
g_rrh46410_sensor0 RRH46410 Gas Sensor Module (rm_rrh46410)
g_comms_i2c_device0 I2C Communication Device (rm_comms_i2c)
g_external_irq0 External IRQ (r_icu)
g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)
Add I2C Communications Peripheral
New
I2C Master (r_iic_master)
I2C Master (r_sci_i2c)
Summary
BSP
Clocks
Pins
Interrupts
Event Links
Stacks
Components



g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)

Settings | API Info

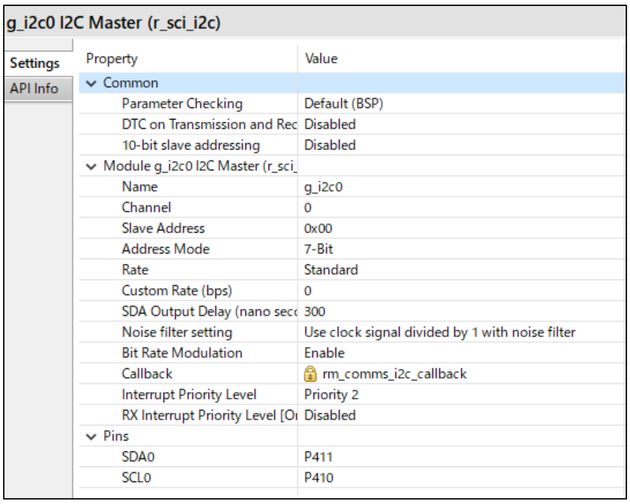
Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)	
Name	g_comms_i2c_bus0
Bus Timeout	0xFFFFFFFF
Semaphore for Blocking (RTOS only)	Use
Recursive Mutex for Bus (RTOS only)	Use
Channel	1
Rate	Standard



g_i2c_master0 I2C Master (r_iic_master)

Settings | API Info

Property	Value
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)

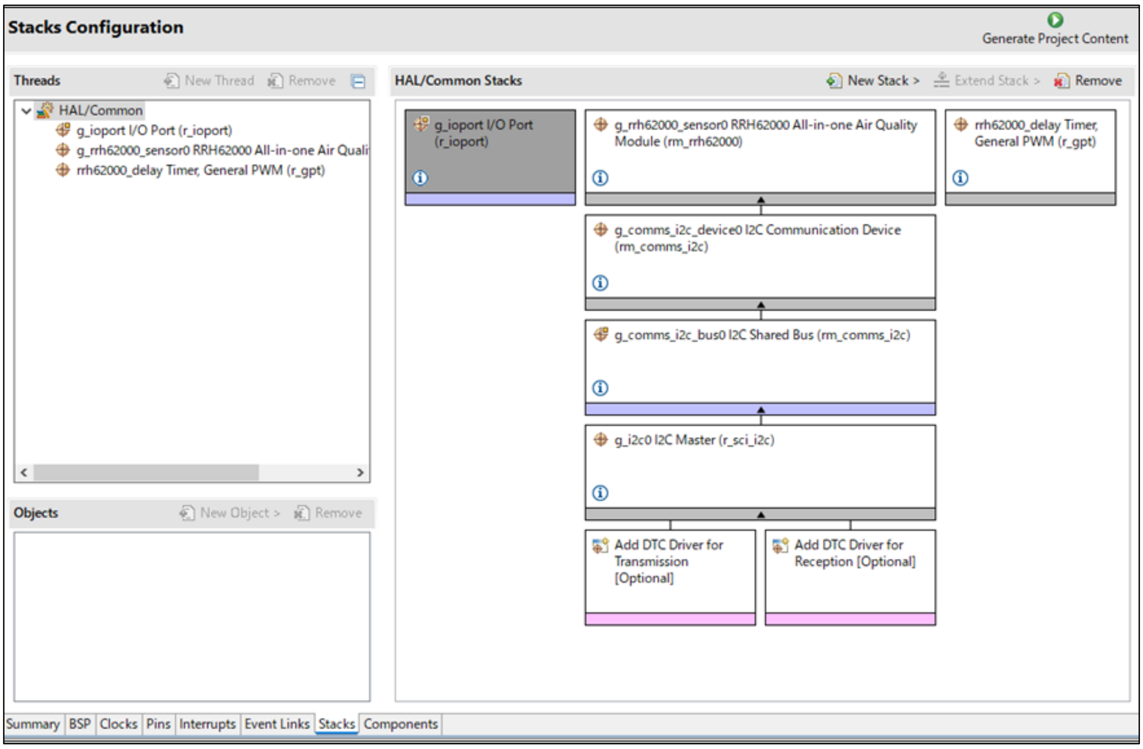
Settings | API Info

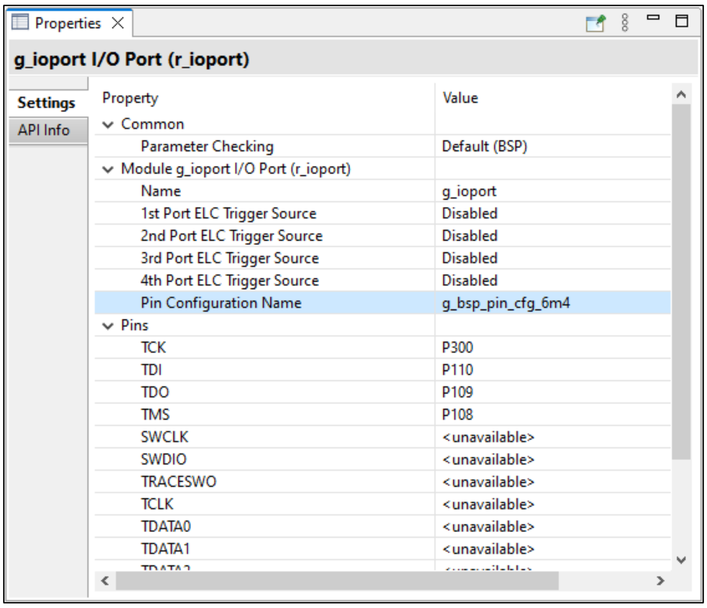
Property	Value
Common	
Parameter Checking	Default (BSP)
DTC on Transmission and Rec	Disabled
10-bit slave addressing	Disabled
Module g_i2c0 I2C Master (r_sci_	
Name	g_i2c0
Channel	0
Slave Address	0x00
Address Mode	7-Bit
Rate	Standard
Custom Rate (bps)	0
SDA Output Delay (nano seco	300
Noise filter setting	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

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”.


Stacks Configuration
Generate Project Content
Threads
New Thread
Remove
HAL/Common
g_ioport I/O Port (r_ioport)
g_rrh62000_sensor0 RRH62000 All-in-one Air Quali
rrh62000_delay Timer, General PWM (r_gpt)
Objects
New Object >
Remove
HAL/Common Stacks
New Stack >
Extend Stack >
Remove
g_ioport I/O Port (r_ioport)
g_rrh62000_sensor0 RRH62000 All-in-one Air Quality Module (rm_rrh62000)
rrh62000_delay Timer, General PWM (r_gpt)
g_comms_i2c_device0 I2C Communication Device (rm_comms_i2c)
g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)
g_i2c0 I2C Master (r_sci_i2c)
Add DTC Driver for Transmission [Optional]
Add DTC Driver for Reception [Optional]
Summary
BSP
Clocks
Pins
Interrupts
Event Links
Stacks
Components



g_ioport I/O Port (r_ioport)

Property	Value
Common	
Parameter Checking	Default (BSP)
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>

(5) Code Generation and Build

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

Build the project.

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.

6.2.2 Changing Toolchain Setting

If you want to use a toolchain other than the GCC ARM Embedded toolchain, copy RA_RRH62000.c (Non-OS) or rrh62000_sensor_thread_entry.c, sensor_thread_common.c and sensor_thread_common.h (FreeRTOS) from this project to create a new project.

6.3 RX Sample Project

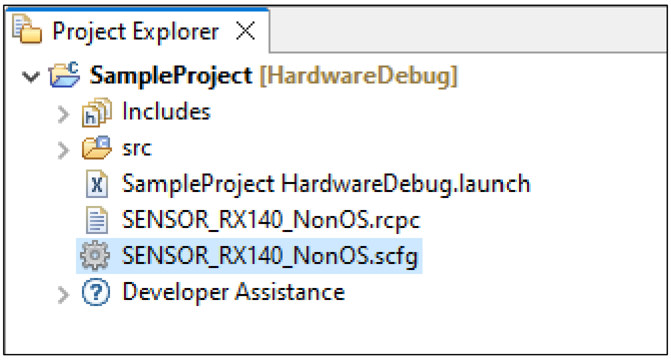
After importing the sample projects, follow the steps below. Please refer to “6.1 Importing Sample Project” for importing instructions.

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 "RRH62000_RX140_NonOS":
  Pmod1 (Type2A: SCI5)
  → Pmod1 (Option Type 6A: RIIC0) or Pmod2 (Option Type6A: SCI11) of the EK-RX671 board

6.3.1 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.

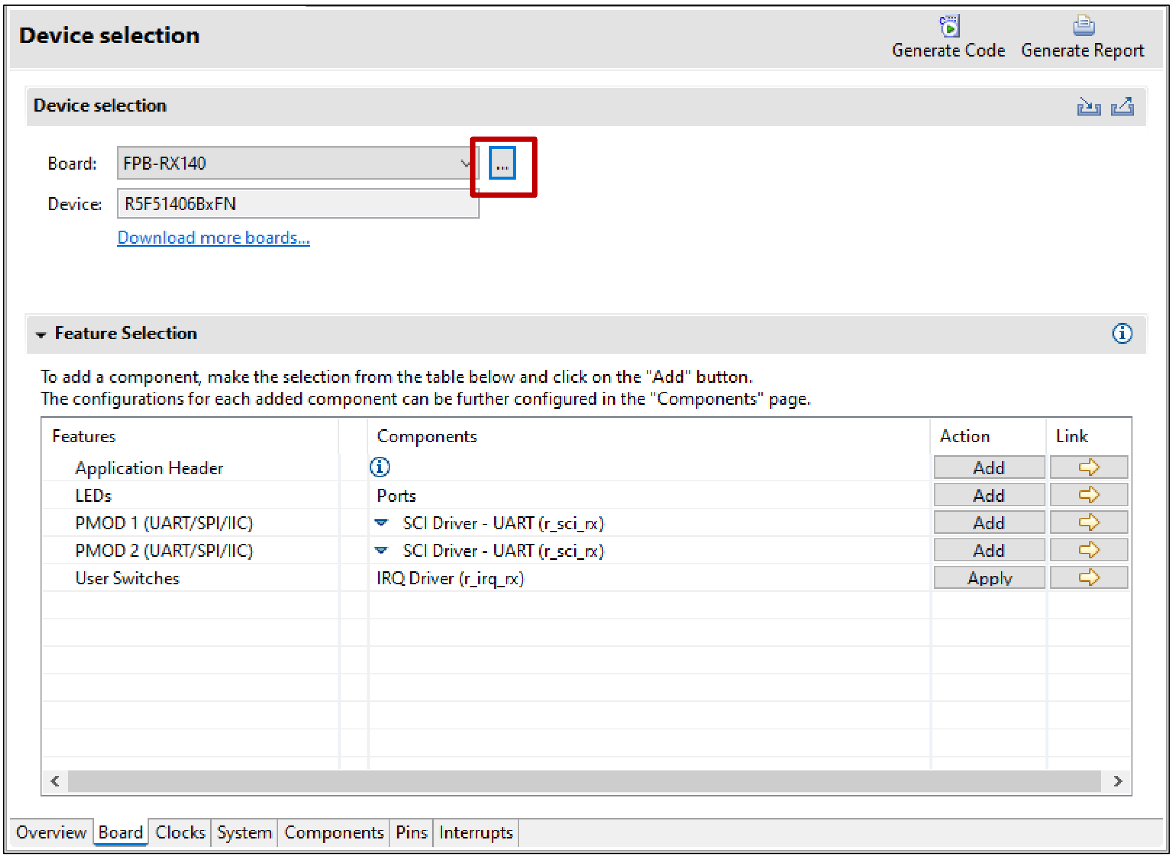


Project Explorer

SampleProject [HardwareDebug]
- Includes
- src
- SampleProject HardwareDebug.launch
- SENSOR_RX140_NonOS.rcpc
- SENSOR_RX140_NonOS.scfg
- Developer Assistance

(1) **Board**

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



Device selection

Device selection

Board: FPB-RX140 ...

Device: R5F51406BxFN

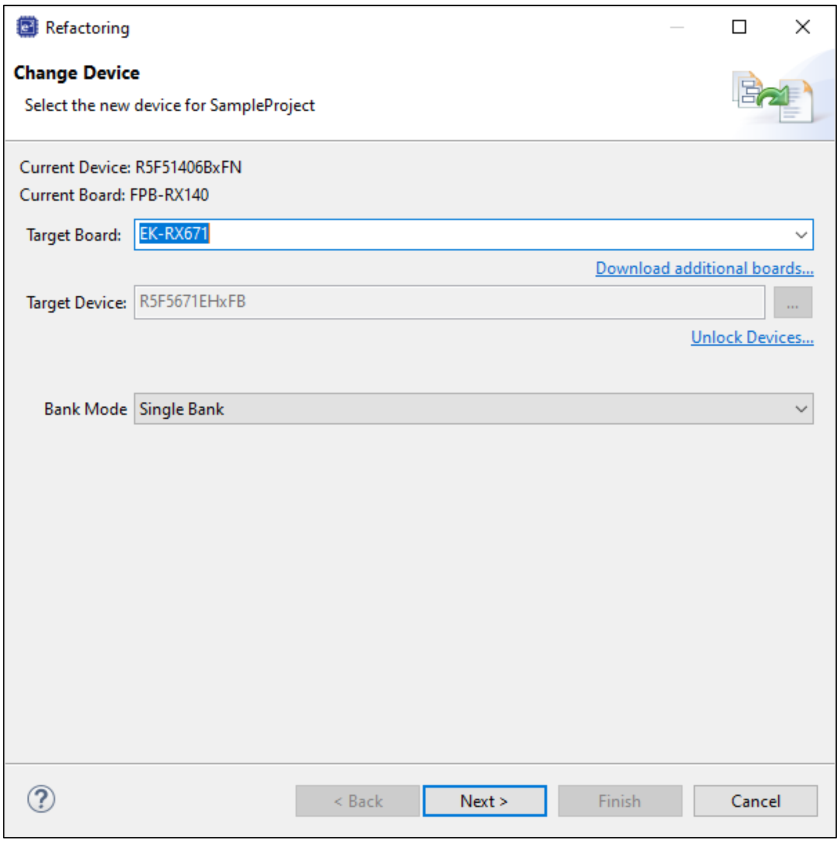
Download more boards...

Feature Selection

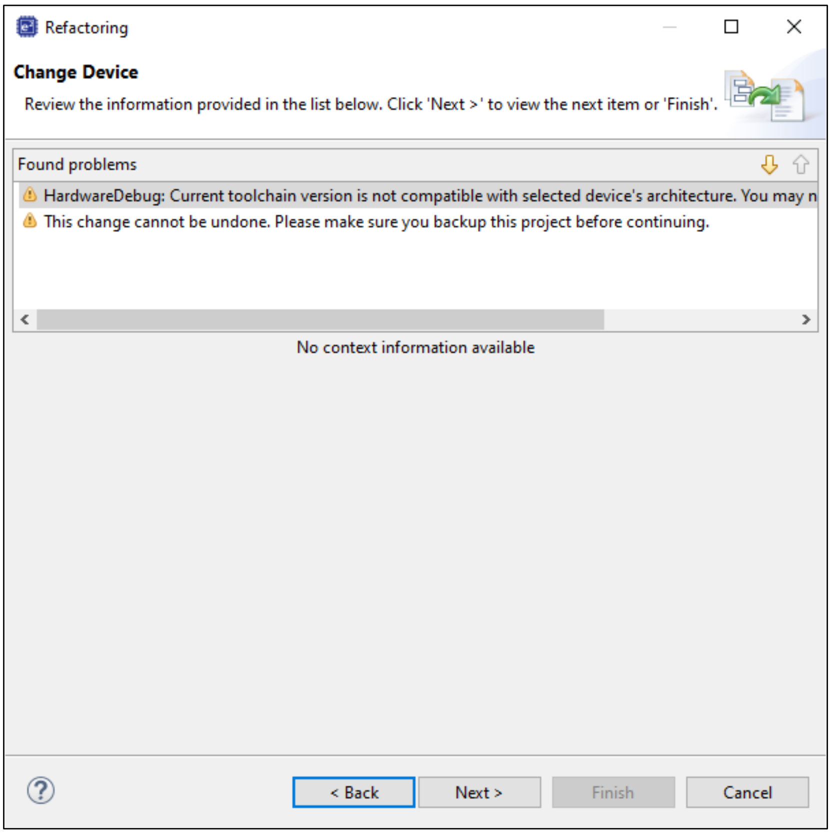
To add a component, make the selection from the table below and click on the "Add" button.
The configurations for each added component can be further configured in the "Components" page.

Features	Components	Action	Link
Application Header	ⓘ	Add	⇨
LEDs	Ports	Add	⇨
PMOD 1 (UART/SPI/IIC)	SCI Driver - UART (r_sci_rx)	Add	⇨
PMOD 2 (UART/SPI/IIC)	SCI Driver - UART (r_sci_rx)	Add	⇨
User Switches	IRQ Driver (r_irq_rx)	Apply	⇨

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.

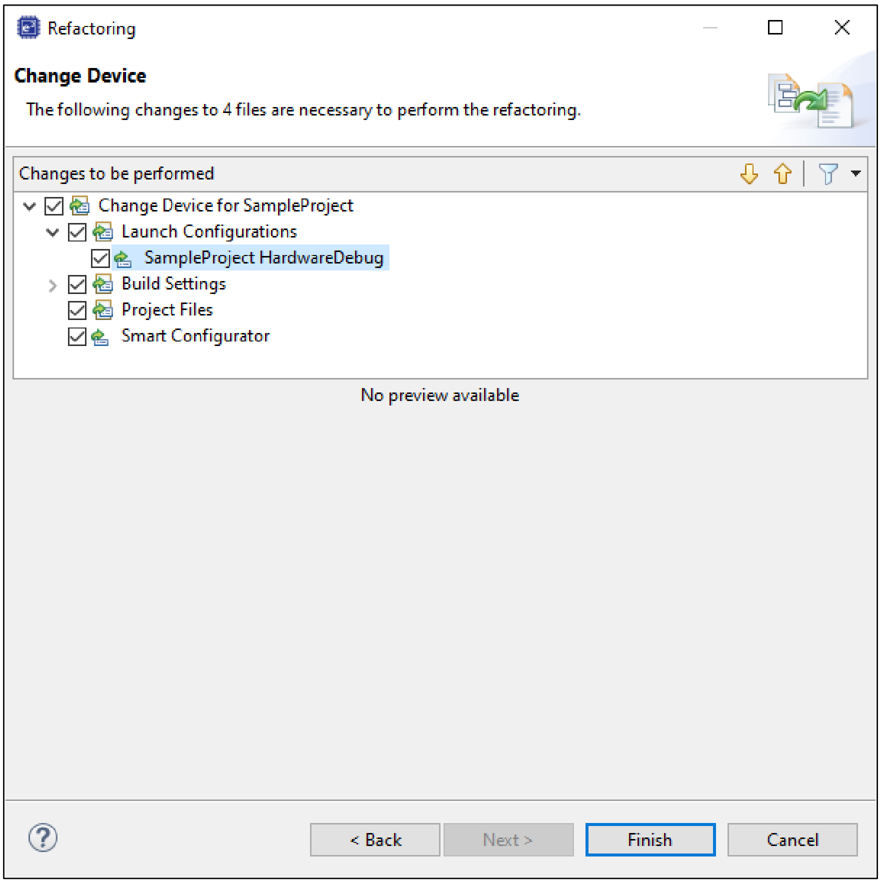


Refactoring
Change Device
Review the information provided in the list below. Click 'Next >' to view the next item or 'Finish'.
Found problems
No context information available
< Back
Next >
Finish
Cancel

⚠ HardwareDebug: Current toolchain version is not compatible with selected device's architecture. You may n

⚠ This change cannot be undone. Please make sure you backup this project before continuing.

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



Refactoring

Change Device

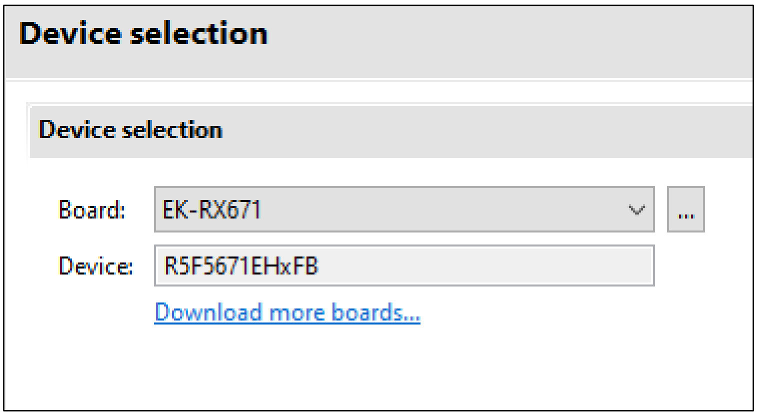
The following changes to 4 files are necessary to perform the refactoring.

Changes to be performed

- ☑ Change Device for SampleProject
  - ☑ Launch Configurations
    - ☑ SampleProject HardwareDebug
  - ☑ Build Settings
  - ☑ Project Files
  - ☑ Smart Configurator

No preview available

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



Device selection

Device selection

Board: EK-RX671

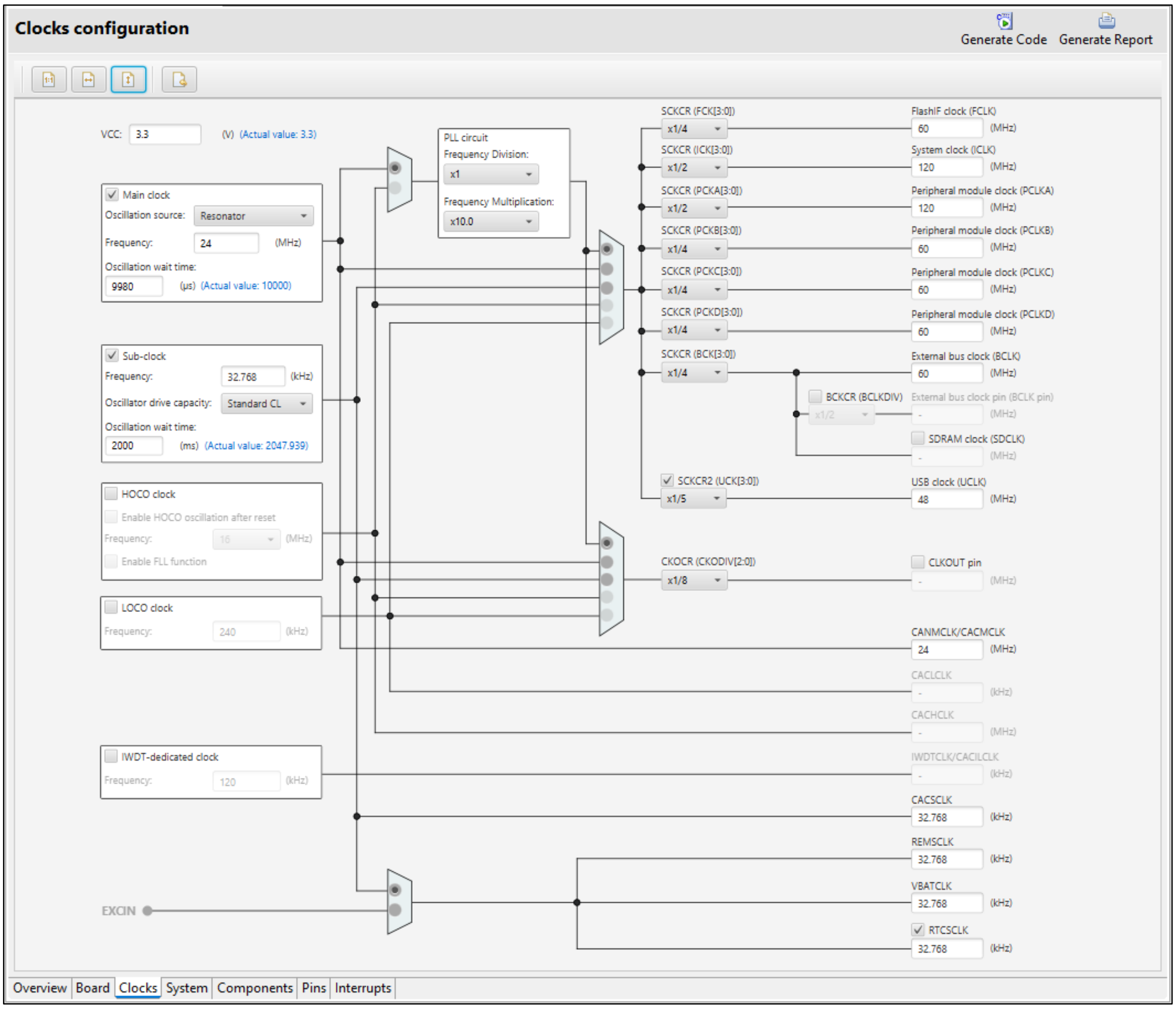
Device: R5F5671EHxFB

(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.



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 FLL function

☐ LOCO clock
Frequency: 240 (kHz)

☐ IWDT-dedicated clock
Frequency: 120 (kHz)

PLL circuit
Frequency Division: x1
Frequency Multiplication: x10.0

SCKCR (FCK[3:0]) x1/4 — FlashIF clock (FCLK) 60 (MHz)
SCKCR (ICK[3:0]) x1/2 — System clock (ICLK) 120 (MHz)
SCKCR (PCKA[3:0]) x1/2 — Peripheral module clock (PCLKA) 120 (MHz)
SCKCR (PCKB[3:0]) x1/4 — Peripheral module clock (PCLKB) 60 (MHz)
SCKCR (PCKC[3:0]) x1/4 — Peripheral module clock (PCLKC) 60 (MHz)
SCKCR (PCKD[3:0]) x1/4 — Peripheral module clock (PCLKD) 60 (MHz)
SCKCR (BCK[3:0]) x1/4 — External bus clock (BCLK) 60 (MHz)
☐ BCKCR (BCLKDIV) x1/2 — External bus clock pin (BCLK pin) - (MHz)
☐ SDRAM clock (SDCLK) - (MHz)
☑ SCKCR2 (UCK[3:0]) x1/5 — USB clock (UCLK) 48 (MHz)
CKOCR (CKODIV[2:0]) x1/8 — ☐ CLKOUT pin - (MHz)

CANMCLK/CACMCLK 24 (MHz)
CACLCLK - (kHz)
CACHCLK - (MHz)
IWDTCLK/CACILCLK - (kHz)
CACSCLK 32.768 (kHz)
REMSCLK 32.768 (kHz)
VBATCLK 32.768 (kHz)
☑ RTCSCLK 32.768 (kHz)

(3) Components

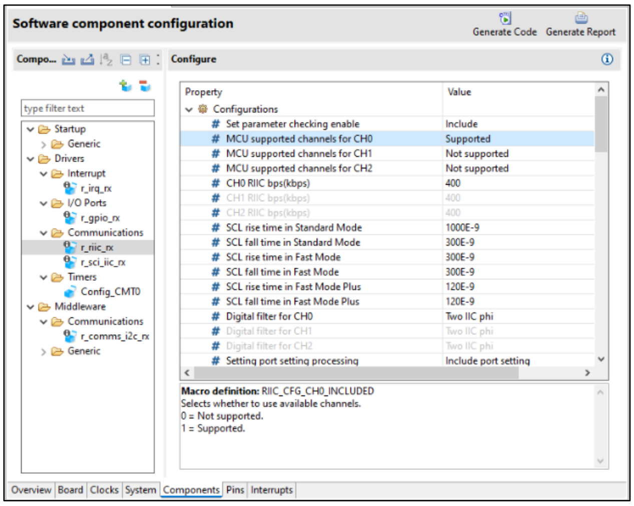
Modify the settings of individual components in the "Components" tabbed page according to the specifications of the target board.

(a) Changing I2C Driver Settings

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

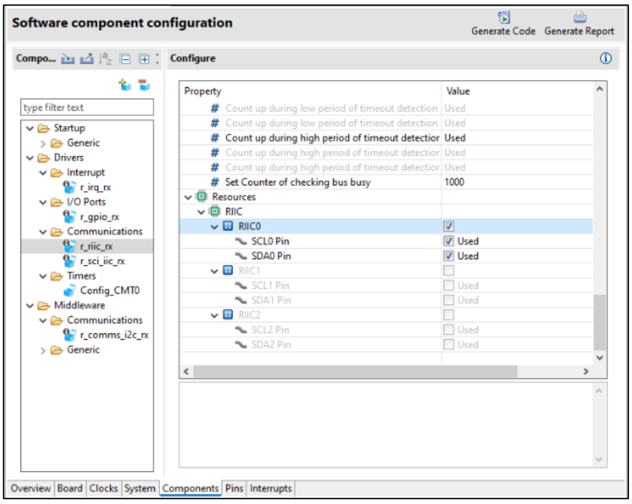
When using Pmod1 (Option Type 6A), set it as follows:

- 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".



Property	Value
Configurations	
Set parameter checking enable	Include
MCU supported channels for CH0	Supported
MCU supported channels for CH1	Not supported
MCU supported channels for CH2	Not supported
CH0 RIIC bps(kbps)	400
CH1 RIIC bps(kbps)	400
CH2 RIIC bps(kbps)	400
SCL rise time in Standard Mode	1000E-9
SCL fall time in Standard Mode	300E-9
SCL rise time in Fast Mode	300E-9
SCL fall time in Fast Mode	300E-9
SCL rise time in Fast Mode Plus	120E-9
SCL fall time in Fast Mode Plus	120E-9
Digital filter for CH0	Two IIC phi
Digital filter for CH1	Two IIC phi
Digital filter for CH2	Two IIC phi
Setting port setting processing	Include port setting

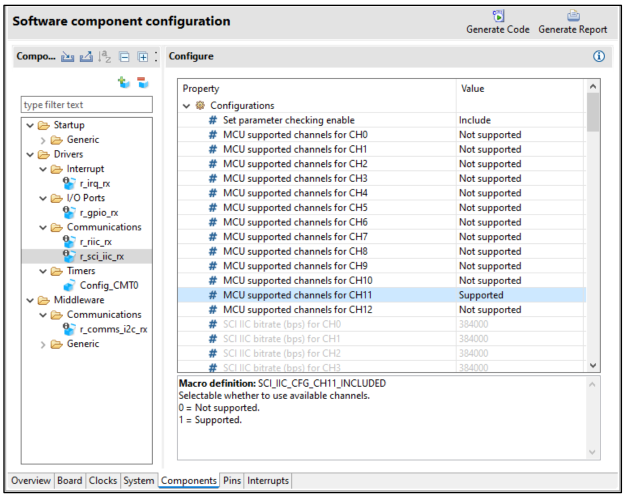
Macro definition: RIIC_CFG_CH0_INCLUDED
Selects whether to use available channels.
0 = Not supported.
1 = Supported.



Property	Value
Count up during low period of timeout detection	Used
Count up during low period of timeout detection	Used
Count up during high period of timeout detectior	Used
Count up during high period of timeout detectior	Used
Count up during high period of timeout detectior	Used
Set Counter of checking bus busy	1000
Resources	
RIIC	
RIIC0	☑
SCL0 Pin	☑ Used
SDA0 Pin	☑ Used
RIIC1	☐
SCL1 Pin	☐ Used
SDA1 Pin	☐ Used
RIIC2	☐
SCL2 Pin	☐ Used
SDA2 Pin	☐ Used

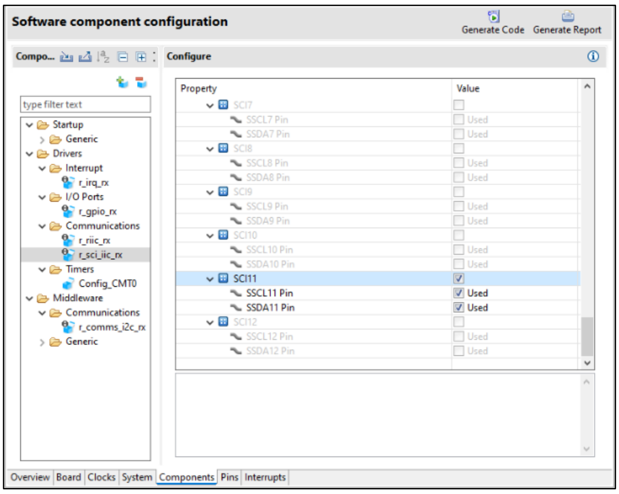
When using Pmod2 (Option Type 6A), set it as follows:

- 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".



Property	Value
Configurations	
Set parameter checking enable	Include
MCU supported channels for CH0	Not supported
MCU supported channels for CH1	Not supported
MCU supported channels for CH2	Not supported
MCU supported channels for CH3	Not supported
MCU supported channels for CH4	Not supported
MCU supported channels for CH5	Not supported
MCU supported channels for CH6	Not supported
MCU supported channels for CH7	Not supported
MCU supported channels for CH8	Not supported
MCU supported channels for CH9	Not supported
MCU supported channels for CH10	Not supported
MCU supported channels for CH11	Supported
MCU supported channels for CH12	Not supported
SCI IIC bitrate (bps) for CH0	384000
SCI IIC bitrate (bps) for CH1	384000
SCI IIC bitrate (bps) for CH2	384000
SCI IIC bitrate (bps) for CH3	384000

Macro definition: SCI_IIC_CFG_CH11_INCLUDED
Selectable whether to use available channels.
0 = Not supported.
1 = Supported.



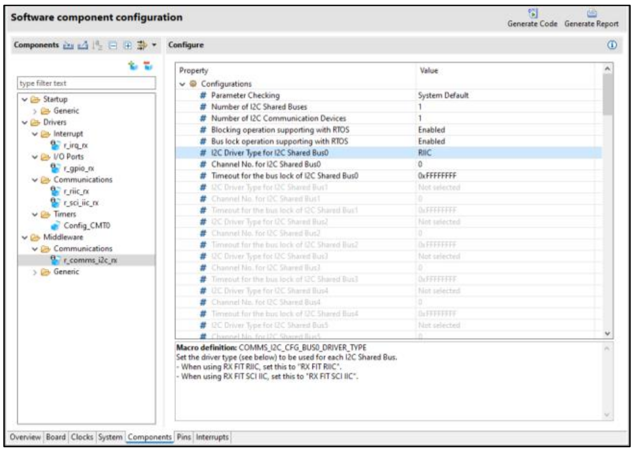
Property	Value
SCI7	☐
SSCL7 Pin	☐ Used
SSDA7 Pin	☐ Used
SCI8	☐
SSCL8 Pin	☐ Used
SSDA8 Pin	☐ Used
SCI9	☐
SSCL9 Pin	☐ Used
SSDA9 Pin	☐ Used
SCI10	☐
SSCL10 Pin	☐ Used
SSDA10 Pin	☐ Used
SCI11	☑
SSCL11 Pin	☑ Used
SSDA11 Pin	☑ Used
SCI12	☐
SSCL12 Pin	☐ Used
SSDA12 Pin	☐ Used

(b) Changing COMMS_I2C Settings

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

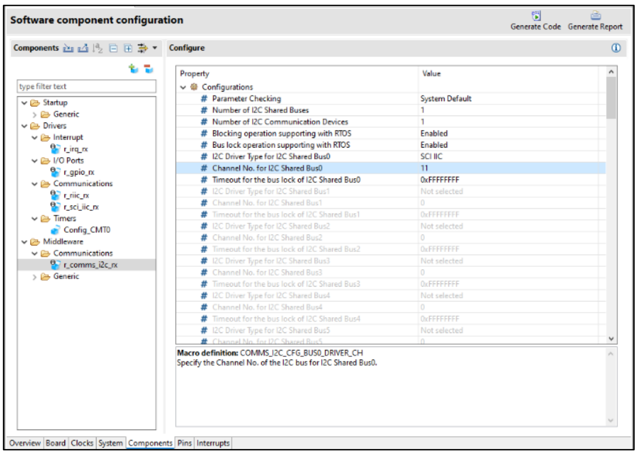
Set “I2C Driver Type for I2C Shared BusX” (X: Bus No.) and “Channel No. for I2C Shared BusX” (X: Bus No.) in **r_comms_i2c_rx** as follows:

- 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”.



Property	Value
Configurations	
Parameter Checking	System Default
Number of I2C Shared Buses	1
Number of I2C Communication Devices	1
Blocking operation supporting with RTOS	Enabled
Bus lock operation supporting with RTOS	Enabled
I2C Driver Type for I2C Shared Bus0	RIIC
Channel No. for I2C Shared Bus0	0
Timeout for the bus lock of I2C Shared Bus0	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus1	Not selected
Channel No. for I2C Shared Bus1	0
Timeout for the bus lock of I2C Shared Bus1	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus2	Not selected
Channel No. for I2C Shared Bus2	0
Timeout for the bus lock of I2C Shared Bus2	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus3	Not selected
Channel No. for I2C Shared Bus3	0
Timeout for the bus lock of I2C Shared Bus3	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus4	Not selected
Channel No. for I2C Shared Bus4	0
Timeout for the bus lock of I2C Shared Bus4	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus5	Not selected
Channel No. for I2C Shared Bus5	0

Macro definition: COMMS_I2C_CFG_BUS0_DRIVER_TYPE
Set the driver type (see below) to be used for each I2C Shared Bus.
- When using RX FIT RIIC, set this to "RX FIT RIIC".
- When using RX FIT SCI IIC, set this to "RX FIT SCI IIC".



Property	Value
Configurations	
Parameter Checking	System Default
Number of I2C Shared Buses	1
Number of I2C Communication Devices	1
Blocking operation supporting with RTOS	Enabled
Bus lock operation supporting with RTOS	Enabled
I2C Driver Type for I2C Shared Bus0	SCI IIC
Channel No. for I2C Shared Bus0	11
Timeout for the bus lock of I2C Shared Bus0	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus1	Not selected
Channel No. for I2C Shared Bus1	0
Timeout for the bus lock of I2C Shared Bus1	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus2	Not selected
Channel No. for I2C Shared Bus2	0
Timeout for the bus lock of I2C Shared Bus2	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus3	Not selected
Channel No. for I2C Shared Bus3	0
Timeout for the bus lock of I2C Shared Bus3	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus4	Not selected
Channel No. for I2C Shared Bus4	0
Timeout for the bus lock of I2C Shared Bus4	0xFFFFFFFF
I2C Driver Type for I2C Shared Bus5	Not selected
Channel No. for I2C Shared Bus5	0

Macro definition: COMMS_I2C_CFG_BUS0_DRIVER_CH
Specify the Channel No. of the I2C bus for I2C Shared Bus0.

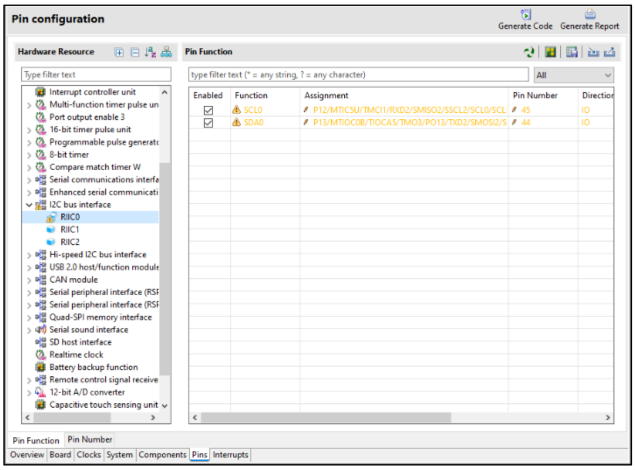
(4) Pins

(a) Changing I2C I/F Pins

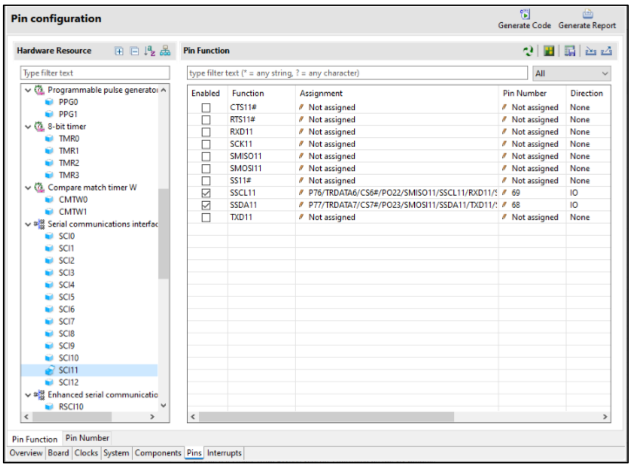
RIIC0 is assigned to Pmod1 and SCI11 to Pmod2 on the EK-RX671 board.

Therefore, set the pins used for I2C communication in “Pin Function” on “Pins” 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 SSDA11.



Enabled	Function	Assignment	Pin Number	Direction
☑	SCL0	P12/MTIC5U/TMCI1/RXD2/SMISO2/SSCL2/SCL0/SCL	45	IO
☑	SDA0	P13/MTIOC0B/TIOCA5/TMO3/PO13/TXD2/SMOSI2/S	44	IO

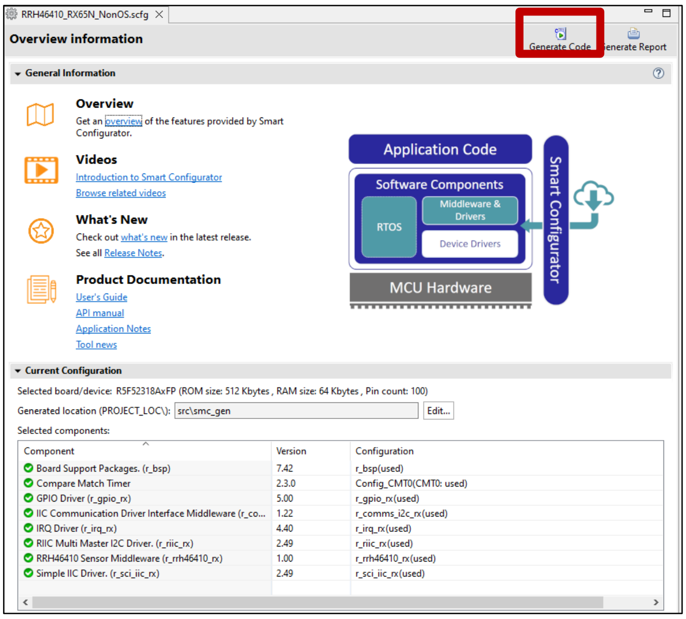


Enabled	Function	Assignment	Pin Number	Direction
□	CTS11#	Not assigned	Not assigned	None
□	RTS11#	Not assigned	Not assigned	None
□	RXD11	Not assigned	Not assigned	None
□	SCK11	Not assigned	Not assigned	None
□	SMISO11	Not assigned	Not assigned	None
□	SMOSI11	Not assigned	Not assigned	None
□	SS11#	Not assigned	Not assigned	None
☑	SSCL11	P76/TRDATA6/CS6#/PO22/SMISO11/SSCL11/RXD11/S	69	IO
☑	SSDA11	P77/TRDATA7/CS7#/PO23/SMOSI11/SSDA11/TXD11/S	68	IO
□	TXD11	Not assigned	Not assigned	None

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.

(5) Code Generation and Build

Press the [Generate Code] icon to generate code.



Overview
Get an overview of the features provided by Smart Configurator.

Videos
Introduction to Smart Configurator
Browse related videos

What's New
Check out what's new in the latest release.
See all Release Notes.

Product Documentation
User's Guide
API manual
Application Notes
Tool news

Selected board/device: R5F52318AxFP (ROM size: 512 Kbytes , RAM size: 64 Kbytes , Pin count: 100)

Generated location (PROJECT_LOC\): src\smc_gen

Selected components:

Component	Version	Configuration
Board Support Packages. (r_bsp)	7.42	r_bsp(used)
Compare Match Timer	2.3.0	Config_CMT0(CMT0: used)
GPIO Driver (r_gpio_rx)	5.00	r_gpio_rx(used)
IIC Communication Driver Interface Middleware (r_co...	1.22	r_comms_i2c_rx(used)
IRQ Driver (r_irq_rx)	4.40	r_irq_rx(used)
RIIC Multi Master I2C Driver. (r_riic_rx)	2.49	r_riic_rx(used)
RRH46410 Sensor Middleware (r_rrh46410_rx)	1.00	r_rrh46410_rx(used)
Simple IIC Driver. (r_sci_iic_rx)	2.49	r_sci_iic_rx(used)

Build the project.

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.

6.3.2 Changing Toolchain Setting

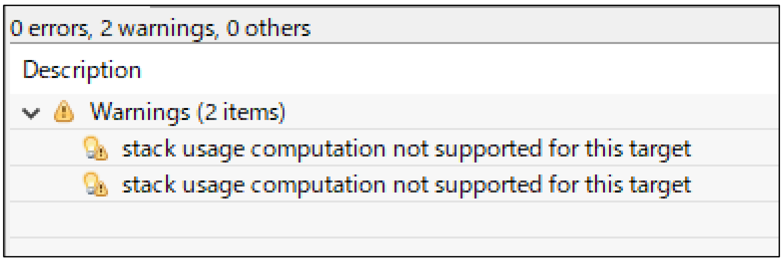
If you want to use a toolchain other than the CC-RX toolchain, copy main.c and RX_RRH62000.c (Non-OS), or main.c and rrh62000_sensor_thread_entry.c (FreeRTOS) from this project to create a new project.

6.3.3 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.



0 errors, 2 warnings, 0 others

Description
⚠ Warnings (2 items)
stack usage computation not supported for this target
stack usage computation not supported for this target

6.3.4 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

6.4 RL78 Sample Project

After importing the sample projects, follow the steps below. Please refer to “6.1 Importing Sample Project” for importing instructions.

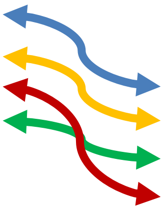
The following explains the change procedure for the following board change example.

- Sample project "RRH62000_RL78G23_NonOS":
  Pmod2 (Type 6A: IICA1)
  → Grove (IICA0) of RL78/G22 Fast Prototyping Board

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

Also, connect with jumper wires as shown below.

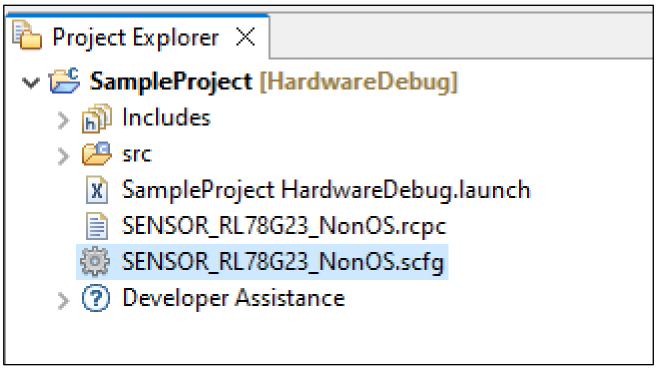
RL78/G22 Fast Prototyping Board (Grove)		
Pin	**Function**	**IICA I/F, Power Supply**
#1	SCL	**SCLA0**
#2	SDA	**SDAA0**
#3	VCC	**3.3V**
#4	GND	**GND**



Renesas Pmod Type 6A Sensor Board	
Pin	**Function**
#1	IRQ
#2	RESET
#3	**IIC_SCL**
#4	**IIC_SDA**
#5	**GND**
#6	**VCC**
#7	BUSY#
#8	ENABLE
#9	POWER_ON
#10	GPIO

6.4.1 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.



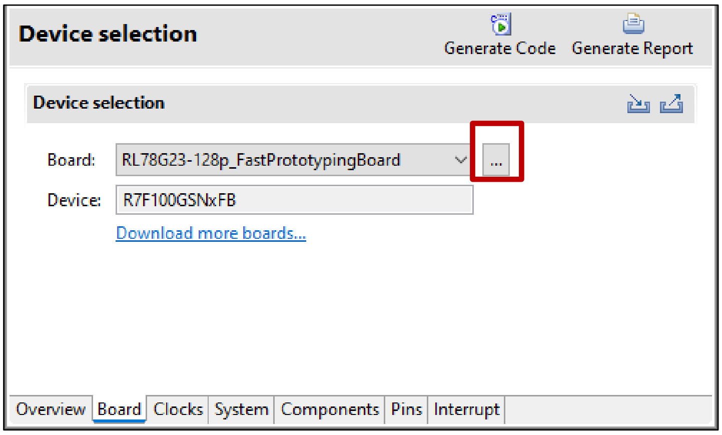
Project Explorer

SampleProject [HardwareDebug]

- Includes
- src
- SampleProject HardwareDebug.launch
- SENSOR_RL78G23_NonOS.rcpc
- SENSOR_RL78G23_NonOS.scfg
- Developer Assistance

(1) Board

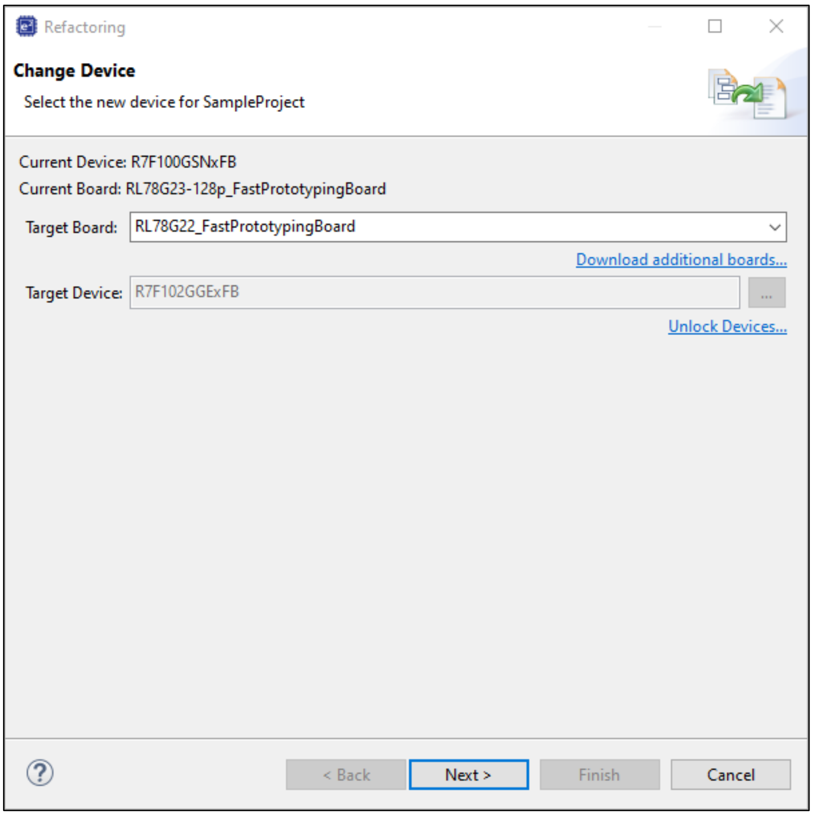
1. On the Board tab, click the [...] button.



Board: RL78G23-128p_FastPrototypingBoard

Device: R7F100GSNxFB

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



Select the new device for SampleProject

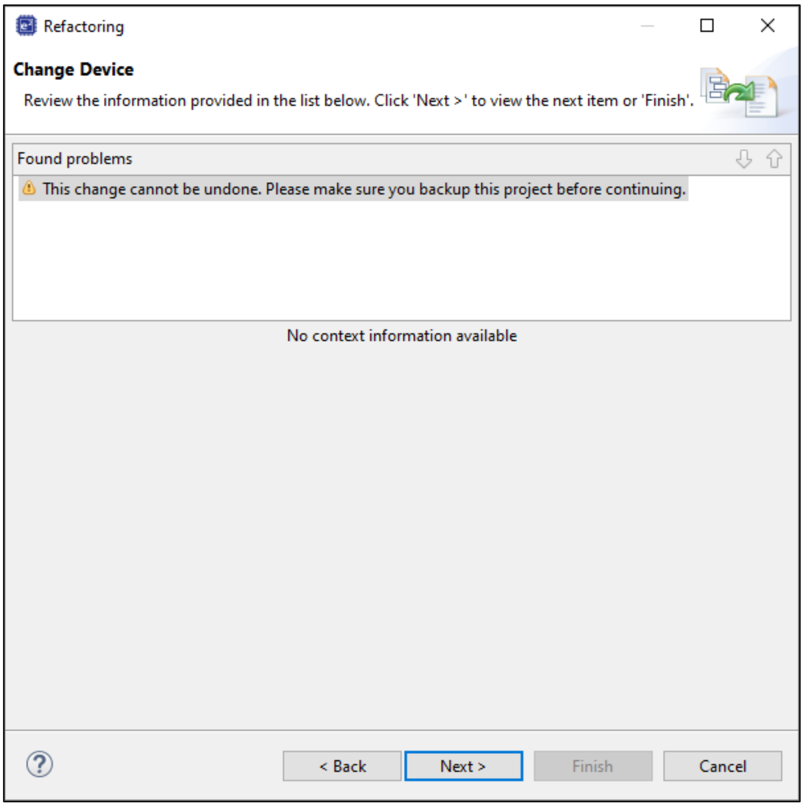
Current Device: R7F100GSNxFB

Current Board: RL78G23-128p_FastPrototypingBoard

Target Board: RL78G22_FastPrototypingBoard

Target Device: R7F102GGExFB

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.


Refactoring

Change Device

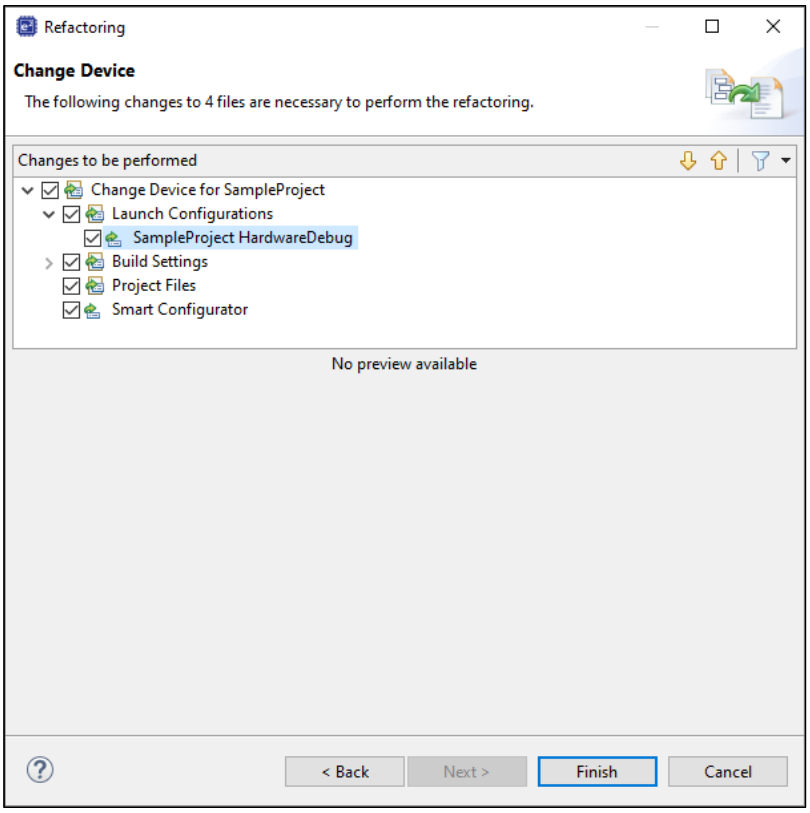
Review the information provided in the list below. Click 'Next >' to view the next item or 'Finish'.

Found problems

⚠ This change cannot be undone. Please make sure you backup this project before continuing.

No context information available

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


Refactoring

Change Device

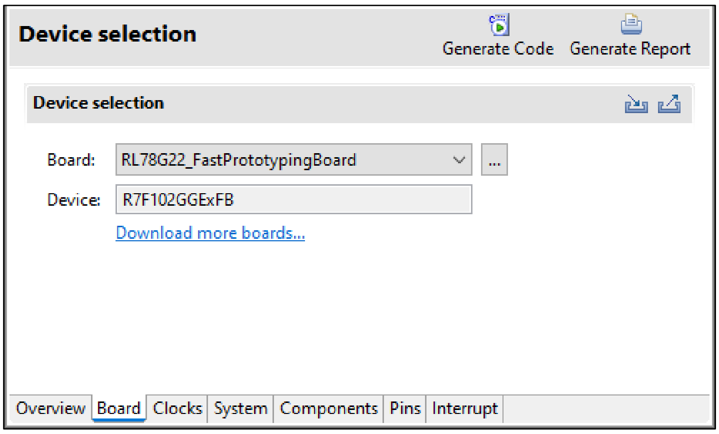
The following changes to 4 files are necessary to perform the refactoring.

Changes to be performed

- ☑ Change Device for SampleProject
  - ☑ Launch Configurations
    - ☑ SampleProject HardwareDebug
  - ☑ Build Settings
  - ☑ Project Files
  - ☑ Smart Configurator

No preview available

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



Device selection

Board: RL78G22_FastPrototypingBoard

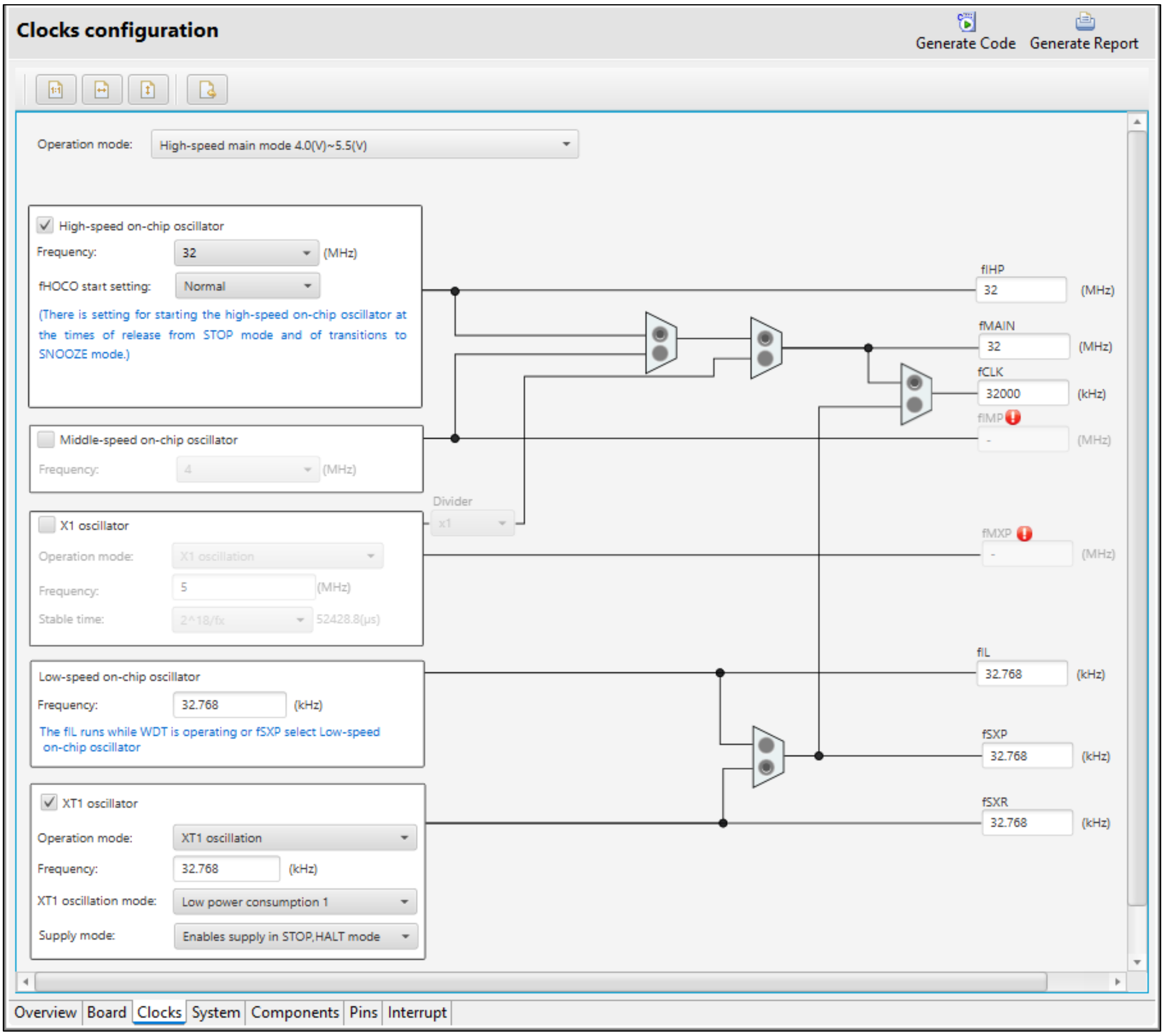
Device: R7F102GGExFB

Download more boards...

(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.



Operation mode: High-speed main mode 4.0(V)~5.5(V)

☑ High-speed on-chip oscillator
Frequency: 32 (MHz)
fHOCO start setting: Normal
(There is setting for starting the high-speed on-chip oscillator at the times of release from STOP mode and of transitions to SNOOZE mode.)

□ Middle-speed on-chip oscillator
Frequency: 4 (MHz)

□ X1 oscillator
Operation mode: X1 oscillation
Frequency: 5 (MHz)
Stable time: 2^18/fx 52428.8(μs)

Low-speed on-chip oscillator
Frequency: 32.768 (kHz)
The fIL runs while WDT is operating or fSXP select Low-speed on-chip oscillator

☑ XT1 oscillator
Operation mode: XT1 oscillation
Frequency: 32.768 (kHz)
XT1 oscillation mode: Low power consumption 1
Supply mode: Enables supply in STOP,HALT mode

fIHP 32 (MHz)
fMAIN 32 (MHz)
fCLK 32000 (kHz)
fIMP - (MHz)
fMXP - (MHz)
fIL 32.768 (kHz)
fSXP 32.768 (kHz)
fSXR 32.768 (kHz)

(3) Components

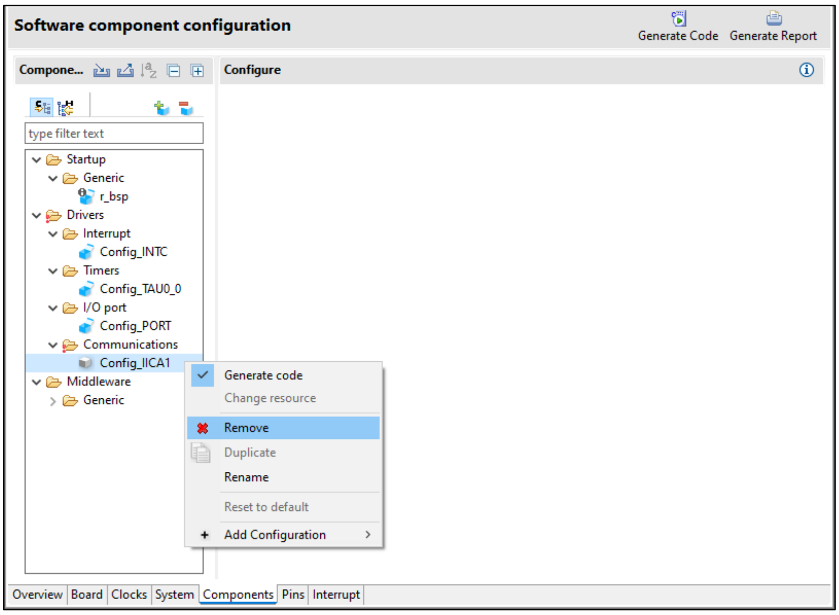
Modify the settings of individual components in the "Components" tabbed page according to the specifications of the target board.

(a) Changing I2C Driver Settings

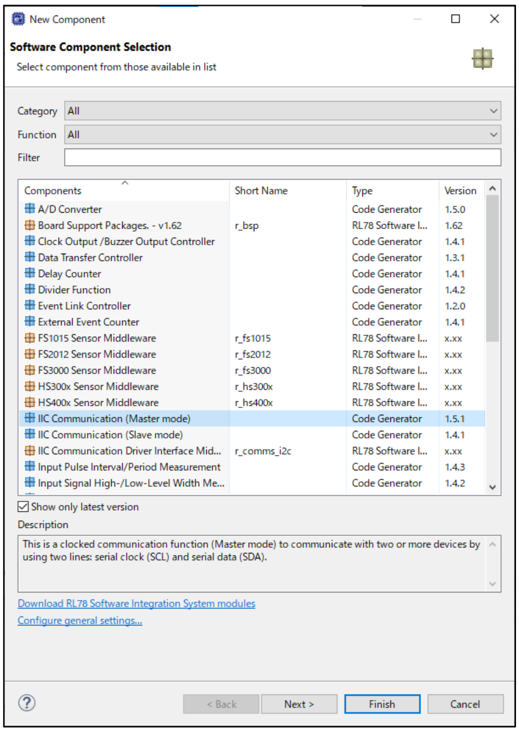
The SAU cannot be used because it does not support the clock stretching function required by RRH62000 sensor.

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.


New Component
Software Component Selection
Select component from those available in list

Category: All
Function: All
Filter:

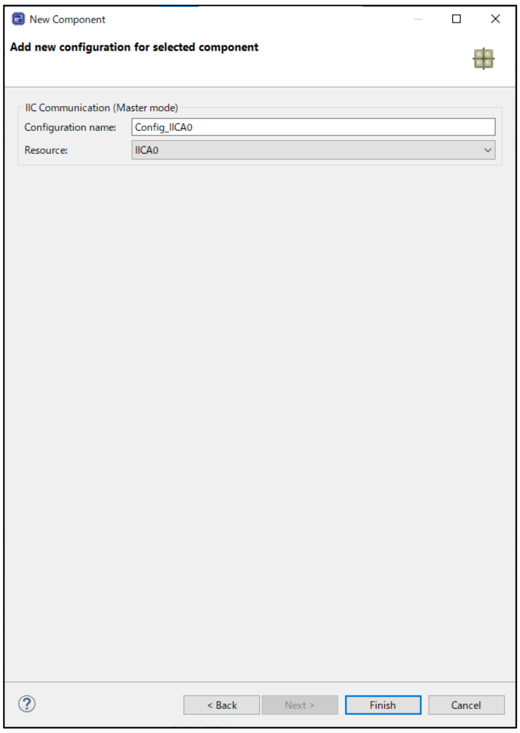
Components	Short Name	Type	Version
A/D Converter		Code Generator	1.5.0
Board Support Packages. - v1.62	r_bsp	RL78 Software I...	1.62
Clock Output /Buzzer Output Controller		Code Generator	1.4.1
Data Transfer Controller		Code Generator	1.3.1
Delay Counter		Code Generator	1.4.1
Divider Function		Code Generator	1.4.2
Event Link Controller		Code Generator	1.2.0
External Event Counter		Code Generator	1.4.1
FS1015 Sensor Middleware	r_fs1015	RL78 Software I...	x.xx
FS2012 Sensor Middleware	r_fs2012	RL78 Software I...	x.xx
FS3000 Sensor Middleware	r_fs3000	RL78 Software I...	x.xx
HS300x Sensor Middleware	r_hs300x	RL78 Software I...	x.xx
HS400x Sensor Middleware	r_hs400x	RL78 Software I...	x.xx
IIC Communication (Master mode)		Code Generator	1.5.1
IIC Communication (Slave mode)		Code Generator	1.4.1
IIC Communication Driver Interface Mid...	r_comms_i2c	RL78 Software I...	x.xx
Input Pulse Interval/Period Measurement		Code Generator	1.4.3
Input Signal High-/Low-Level Width Me...		Code Generator	1.4.2

☑ Show only latest version

Description

This is a clocked communication function (Master mode) to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA).

Download RL78 Software Integration System modules
Configure general settings...
< Back
Next >
Finish
Cancel


New Component
Add new configuration for selected component

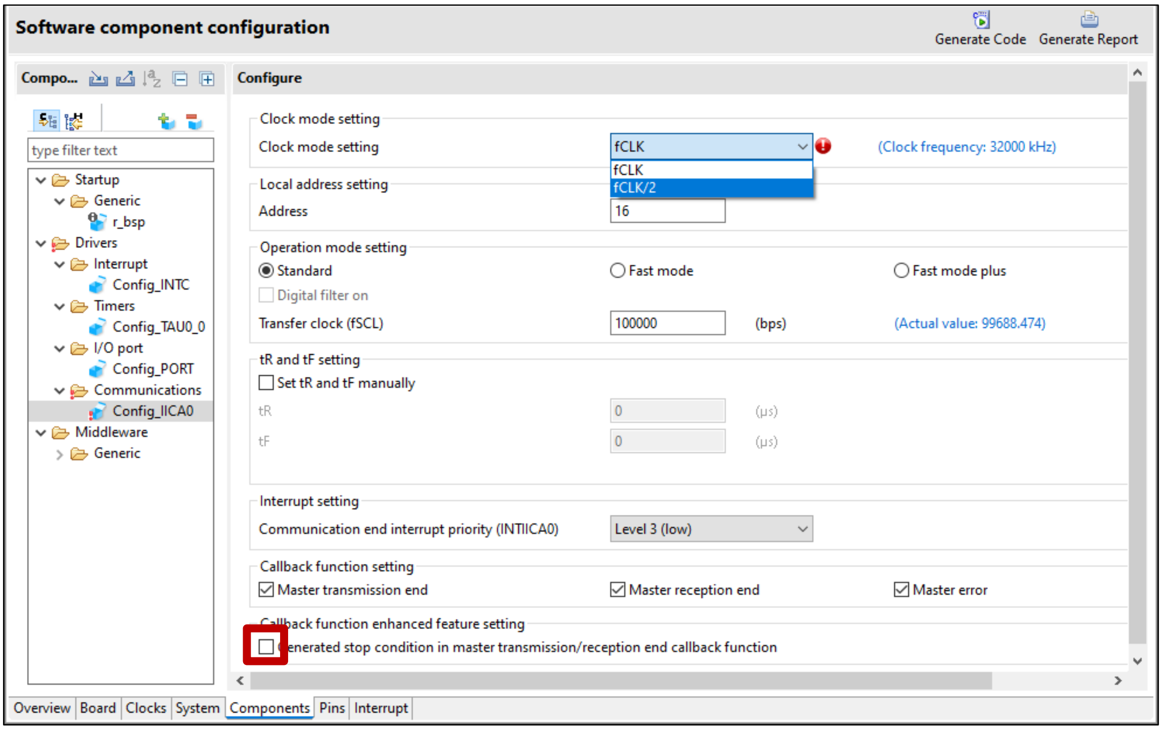
IIC Communication (Master mode)

Configuration name: Config_IICA0

Resource: IICA0

< Back
Next >
Finish
Cancel

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



Software component configuration

Configure

Clock mode setting
Clock mode setting: fCLK (options: fCLK, fCLK/2) (Clock frequency: 32000 kHz)

Local address setting
Address: 16

Operation mode setting
● Standard ○ Fast mode ○ Fast mode plus
☐ Digital filter on
Transfer clock (fSCL): 100000 (bps) (Actual value: 99688.474)

tR and tF setting
☐ Set tR and tF manually
tR: 0 (μs)
tF: 0 (μs)

Interrupt setting
Communication end interrupt priority (INTIICA0): Level 3 (low)

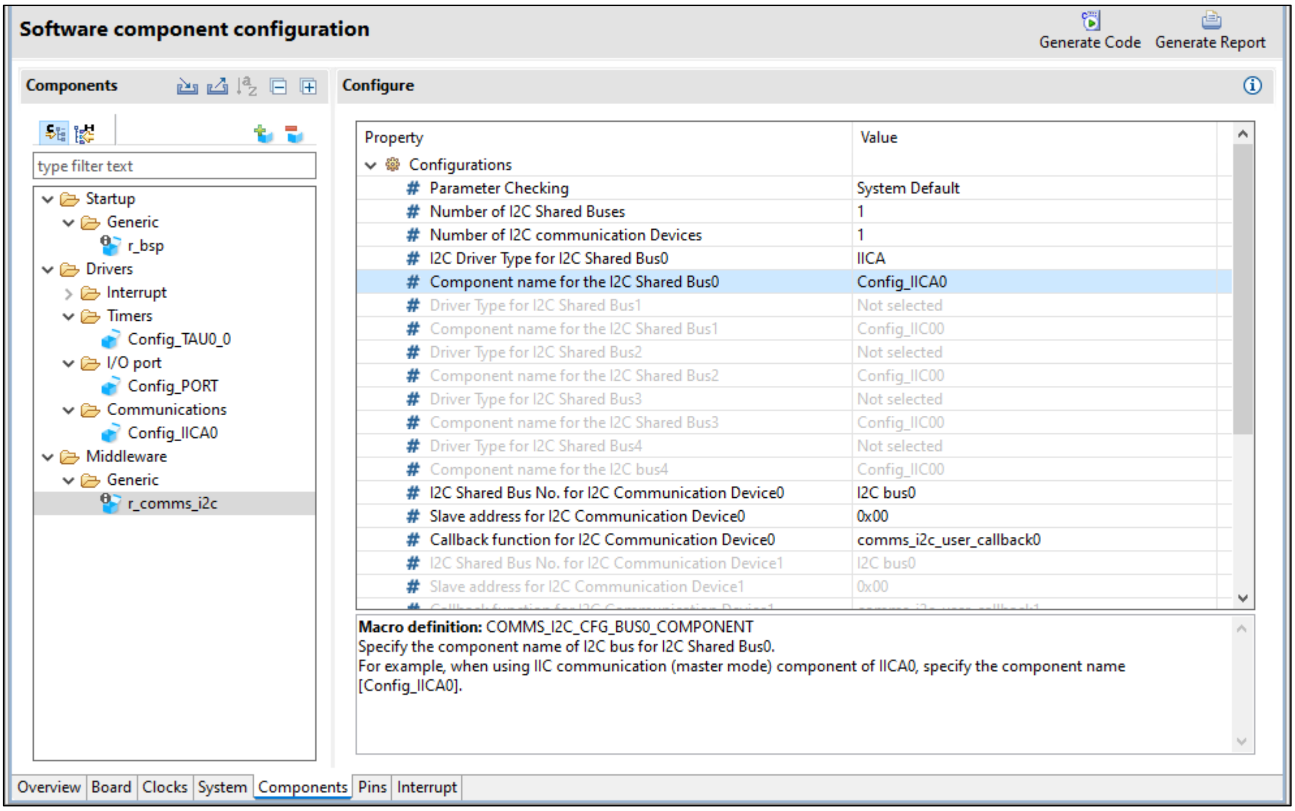
Callback function setting
☑ Master transmission end ☑ Master reception end ☑ Master error

Callback function enhanced feature setting
☐ 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.



Software component configuration

Configure

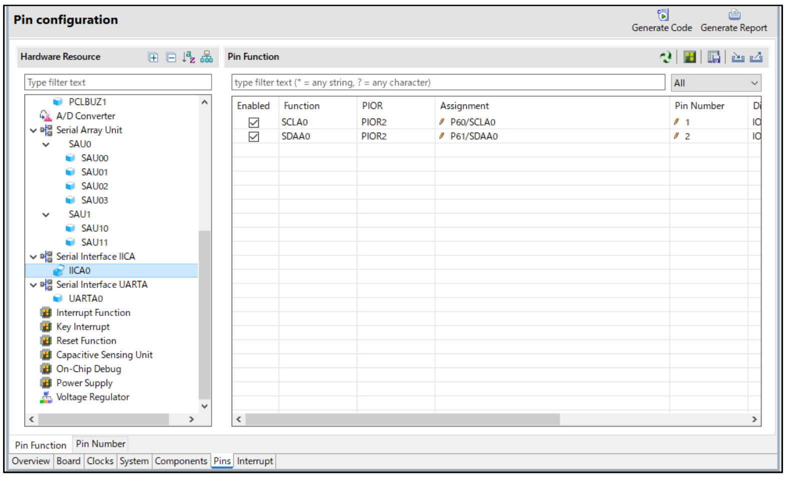
Property	Value
Configurations	
Parameter Checking	System Default
Number of I2C Shared Buses	1
Number of I2C communication Devices	1
I2C Driver Type for I2C Shared Bus0	IICA
Component name for the I2C Shared Bus0	Config_IICA0
Driver Type for I2C Shared Bus1	Not selected
Component name for the I2C Shared Bus1	Config_IIC00
Driver Type for I2C Shared Bus2	Not selected
Component name for the I2C Shared Bus2	Config_IIC00
Driver Type for I2C Shared Bus3	Not selected
Component name for the I2C Shared Bus3	Config_IIC00
Driver Type for I2C Shared Bus4	Not selected
Component name for the I2C bus4	Config_IIC00
I2C Shared Bus No. for I2C Communication Device0	I2C bus0
Slave address for I2C Communication Device0	0x00
Callback function for I2C Communication Device0	comms_i2c_user_callback0
I2C Shared Bus No. for I2C Communication Device1	I2C bus0
Slave address for I2C Communication Device1	0x00
Callback function for I2C Communication Device1	comms_i2c_user_callback1

Macro definition: COMMS_I2C_CFG_BUS0_COMPONENT
Specify the component name of I2C bus for I2C Shared Bus0.
For example, when using IIC communication (master mode) component of IICA0, specify the component name [Config_IICA0].

(4) Pins

(a) Changing I2C I/F Pins

Select “IICA0” in "Pins" tabbed page and check that functions are assigned to the IICA pins in the "Pin Function" panel.



Pin configuration

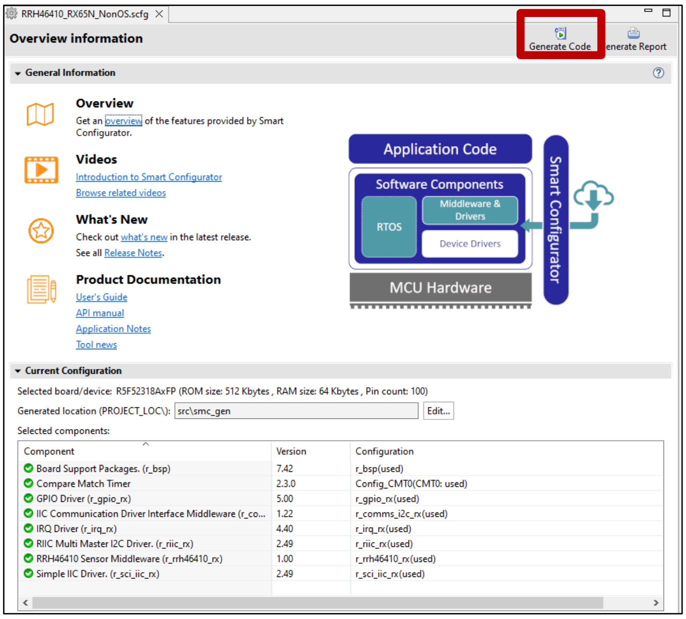
Hardware Resource: IICA0 selected under Serial Interface IICA

Pin Function

Enabled	Function	PIOR	Assignment	Pin Number	Di
☑	SCLA0	PIOR2	P60/SCLA0	1	IO
☑	SDAA0	PIOR2	P61/SDAA0	2	IO

(5) Code Generation and Build

Press the [Generate Code] icon to generate code.



Selected board/device: R5F52318AxFP (ROM size: 512 Kbytes , RAM size: 64 Kbytes , Pin count: 100)

Generated location (PROJECT_LOC\): src\smc_gen Edit...

Selected components:

Component	Version	Configuration
Board Support Packages. (r_bsp)	7.42	r_bsp(used)
Compare Match Timer	2.3.0	Config_CMT0(CMT0: used)
GPIO Driver (r_gpio_rx)	5.00	r_gpio_rx(used)
IIC Communication Driver Interface Middleware (r_co...	1.22	r_comms_i2c_rx(used)
IRQ Driver (r_irq_rx)	4.40	r_irq_rx(used)
RIIC Multi Master I2C Driver. (r_riic_rx)	2.49	r_riic_rx(used)
RRH46410 Sensor Middleware (r_rrh46410_rx)	1.00	r_rrh46410_rx(used)
Simple IIC Driver. (r_sci_iic_rx)	2.49	r_sci_iic_rx(used)

Build the project after implementing “6.4.2 Modifying Generated 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.

6.4.2 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 */
}
```

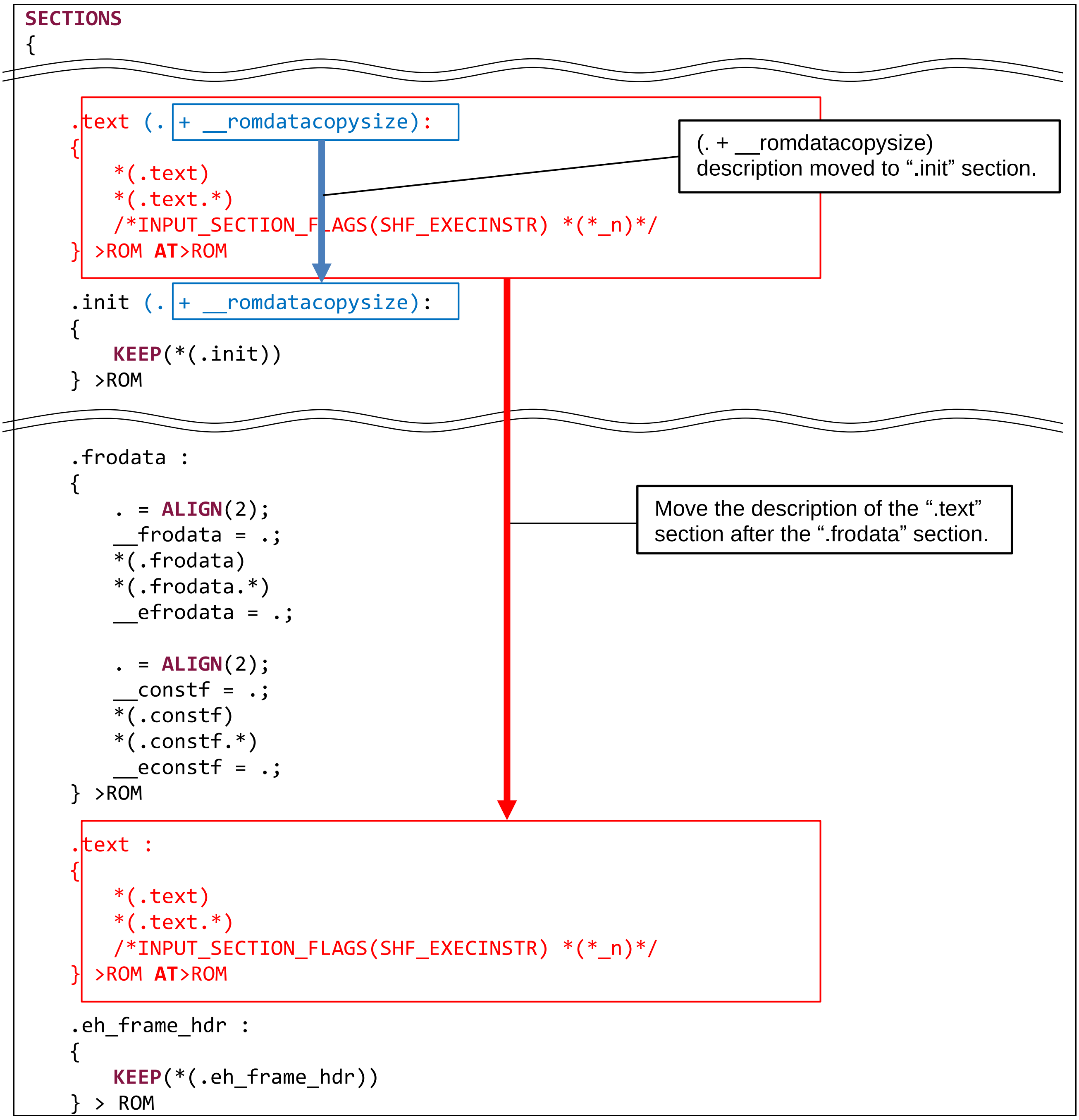
6.4.3 Changing Toolchain Setting

If you want to use a toolchain the LLVM toolchain, use ”RRH62000_RL78G23_NonOS_LLVM”.

If you want to use a toolchain other than the CC-RL toolchain or LLVM toolchain, copy RRH62000_RL78G23_NonOS.c and RL78_RRH62000.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 “RRH62000_RL78G23_NonOS_LLVM”.

1. Place the “.text” section after the “.frodata” section.

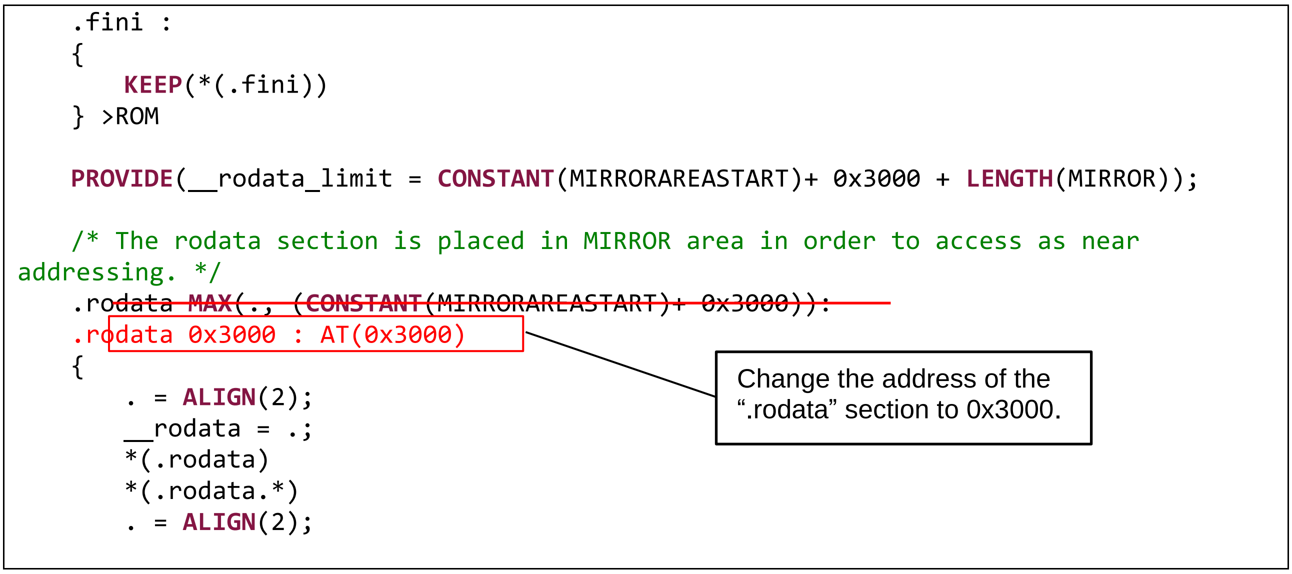


```
SECTIONS
{

    .text (. + __romdatacopysize):
    {
        *(.text)
        *(.text.*)
        /*INPUT_SECTION_FLAGS(SHF_EXECINSTR) *(*_n)*/
    } >ROM AT>ROM
    .init (. + __romdatacopysize):
    {
        KEEP(*(.init))
    } >ROM

    .frodata :
    {
        . = ALIGN(2);
        __frodata = .;
        *(.frodata)
        *(.frodata.*)
        __efrodata = .;

        . = ALIGN(2);
        __constf = .;
        *(.constf)
        *(.constf.*)
        __econstf = .;
    } >ROM
    .text :
    {
        *(.text)
        *(.text.*)
        /*INPUT_SECTION_FLAGS(SHF_EXECINSTR) *(*_n)*/
    } >ROM AT>ROM
    .eh_frame_hdr :
    {
        KEEP(*(.eh_frame_hdr))
    } > ROM
```

(. + __romdatacopysize) description moved to “.init” section.

Move the description of the “.text” section after the “.frodata” section.

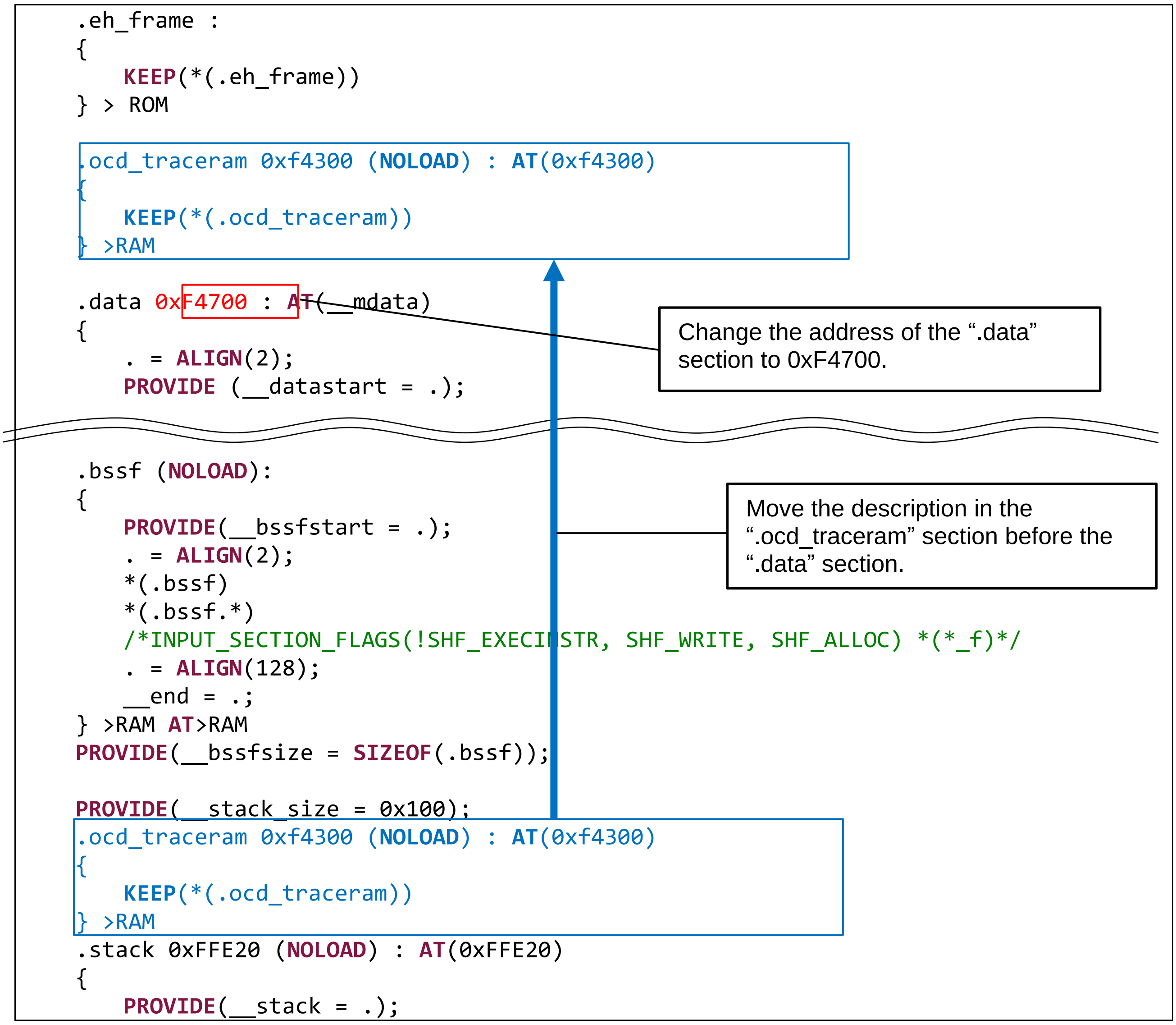
2. Fix the address of the ".rodata" section to the top address of the mirror area.



```
    .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);
```

Change the address of the ".rodata" section to 0x3000.

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 = .);
```

Change the address of the ".data" section to 0xF4700.

Move the description in the ".ocd_traceram" section before the ".data" section.

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

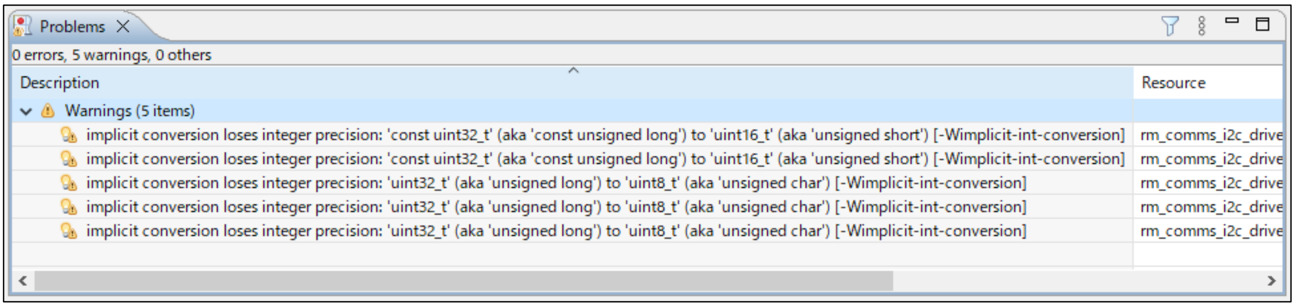
6.4.4 Notes for Build on LLVM

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.



0 errors, 5 warnings, 0 others

Description	Resource
Warnings (5 items)	
implicit conversion loses integer precision: 'const uint32_t' (aka 'const unsigned long') to 'uint16_t' (aka 'unsigned short') [-Wimplicit-int-conversion]	rm_comms_i2c_drive
implicit conversion loses integer precision: 'const uint32_t' (aka 'const unsigned long') to 'uint16_t' (aka 'unsigned short') [-Wimplicit-int-conversion]	rm_comms_i2c_drive
implicit conversion loses integer precision: 'uint32_t' (aka 'unsigned long') to 'uint8_t' (aka 'unsigned char') [-Wimplicit-int-conversion]	rm_comms_i2c_drive
implicit conversion loses integer precision: 'uint32_t' (aka 'unsigned long') to 'uint8_t' (aka 'unsigned char') [-Wimplicit-int-conversion]	rm_comms_i2c_drive
implicit conversion loses integer precision: 'uint32_t' (aka 'unsigned long') to 'uint8_t' (aka 'unsigned char') [-Wimplicit-int-conversion]	rm_comms_i2c_drive

6.5 Notes for Interrupt Signal Circuits

RRH62000 does not have an Interrupt request signal pin.

However, since the RRH62000 Sensor Pmod Board has a pull-up resistor circuit, please disable the pull-up resistor.

6.6 Notes for RESET Signal Circuits

RRH62000 does not have a RESET input signal pin.

Also, since the RRH62000 sensor Pmod board does not have the pull-up resistor circuit, there are no precautions to take when daisy-chaining Renesas sensor Pmod boards.

6.7 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 6-2 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.

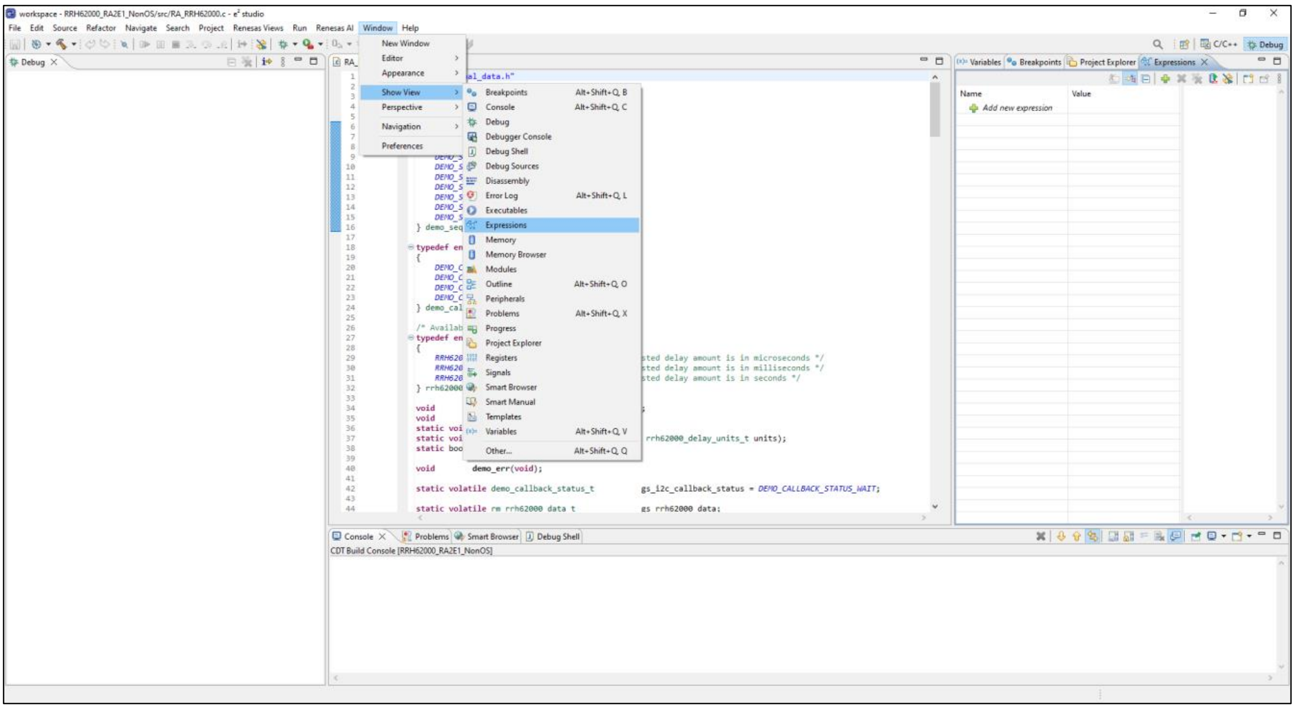
Note 2 Configure when using as an Interrupt signal.

7. Viewing Sensor Data

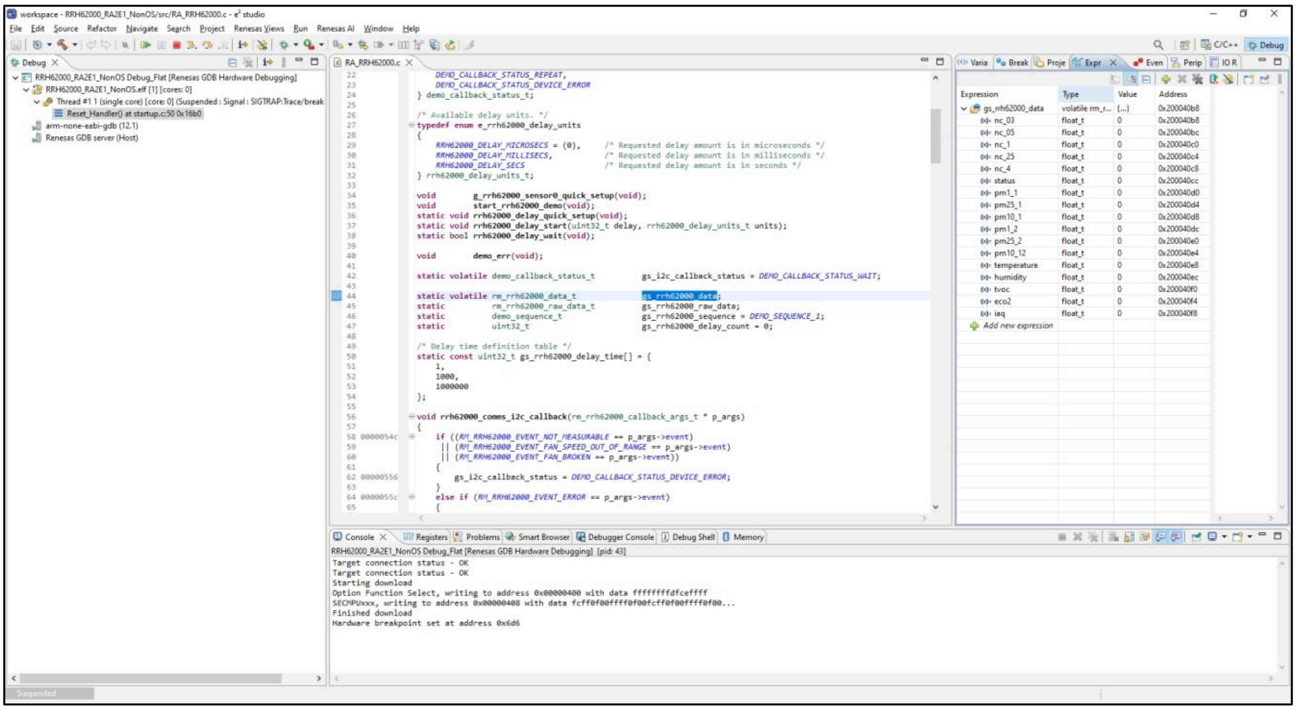
To check the real-time sensor data, follow the steps below.

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

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



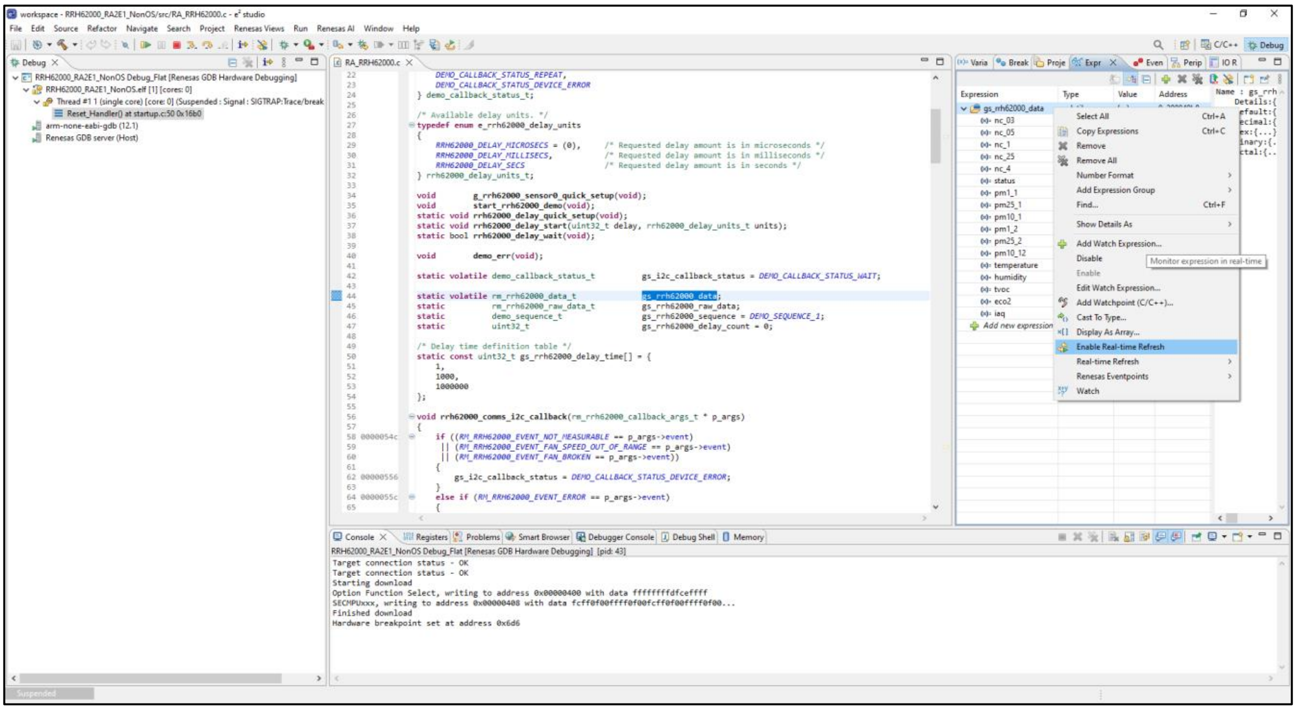
2. Click “Add new expression” in the “Expressions” and add “gs_rrh62000_data”.


workspace - RRH62000_RA2E1_NonOS/src/RA_RRH62000.c - e² studio
Debug
RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging]
RRH62000_RA2E1_NonOS.elf [1] [cores: 0]
Thread #1 1 (single core) [core: 0] (Suspended : Signal : SIGTRAP:Trace/break
Reset_Handler() at startup.c:50 0x16b0
arm-none-eabi-gdb (12.1)
Renesas GDB server (Host)
RA_RRH62000.c
Varia
Break
Proje
Expr
Even
Perip
IO R
Console
Registers
Problems
Smart Browser
Debugger Console
Debug Shell
Memory
Suspended

Expression	Type	Value	Address
gs_rrh62000_data	volatile rm_r...	{...}	0x200040b8
nc_03	float_t	0	0x200040b8
nc_05	float_t	0	0x200040bc
nc_1	float_t	0	0x200040c0
nc_25	float_t	0	0x200040c4
nc_4	float_t	0	0x200040c8
status	float_t	0	0x200040cc
pm1_1	float_t	0	0x200040d0
pm25_1	float_t	0	0x200040d4
pm10_1	float_t	0	0x200040d8
pm1_2	float_t	0	0x200040dc
pm25_2	float_t	0	0x200040e0
pm10_12	float_t	0	0x200040e4
temperature	float_t	0	0x200040e8
humidity	float_t	0	0x200040ec
tvoc	float_t	0	0x200040f0
eco2	float_t	0	0x200040f4
iaq	float_t	0	0x200040f8
Add new expression			

```
RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging]  [pid: 43]
Target connection status - OK
Target connection status - OK
Starting download
Option Function Select, writing to address 0x00000400 with data ffffffffdfceffff
SECMPUxxx, writing to address 0x00000408 with data fcff0f00ffff0f00fcff0f00ffff0f00...
Finished download
Hardware breakpoint set at address 0x6d6
```

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



workspace - RRH62000_RA2E1_NonOS/src/RA_RRH62000.c - e² studio

- RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging]
  - RRH62000_RA2E1_NonOS.elf [1] [cores: 0]
    - Thread #1 1 (single core) [core: 0] (Suspended : Signal : SIGTRAP:Trace/break
      - Reset_Handler() at startup.c:50 0x16b0
  - arm-none-eabi-gdb (12.1)
  - Renesas GDB server (Host)

RA_RRH62000.c

```c
    DEMO_CALLBACK_STATUS_REPEAT,
    DEMO_CALLBACK_STATUS_DEVICE_ERROR
} demo_callback_status_t;

/* Available delay units. */
typedef enum e_rrh62000_delay_units
{
    RRH62000_DELAY_MICROSECS = (0),     /* Requested delay amount is in microseconds */
    RRH62000_DELAY_MILLISECS,           /* Requested delay amount is in milliseconds */
    RRH62000_DELAY_SECS                 /* Requested delay amount is in seconds */
} rrh62000_delay_units_t;

void        g_rrh62000_sensor0_quick_setup(void);
void        start_rrh62000_demo(void);
static void rrh62000_delay_quick_setup(void);
static void rrh62000_delay_start(uint32_t delay, rrh62000_delay_units_t units);
static bool rrh62000_delay_wait(void);

void        demo_err(void);

static volatile demo_callback_status_t         gs_i2c_callback_status = DEMO_CALLBACK_STATUS_WAIT;

static volatile rm_rrh62000_data_t             gs_rrh62000_data;
static          rm_rrh62000_raw_data_t         gs_rrh62000_raw_data;
static          demo_sequence_t                gs_rrh62000_sequence = DEMO_SEQUENCE_1;
static          uint32_t                       gs_rrh62000_delay_count = 0;

/* Delay time definition table */
static const uint32_t gs_rrh62000_delay_time[] = {
    1,
    1000,
    1000000
};

void rrh62000_comms_i2c_callback(rm_rrh62000_callback_args_t * p_args)
{
    if ((RM_RRH62000_EVENT_NOT_MEASURABLE == p_args->event)
     || (RM_RRH62000_EVENT_FAN_SPEED_OUT_OF_RANGE == p_args->event)
     || (RM_RRH62000_EVENT_FAN_BROKEN == p_args->event))
    {
        gs_i2c_callback_status = DEMO_CALLBACK_STATUS_DEVICE_ERROR;
    }
    else if (RM_RRH62000_EVENT_ERROR == p_args->event)
    {
```

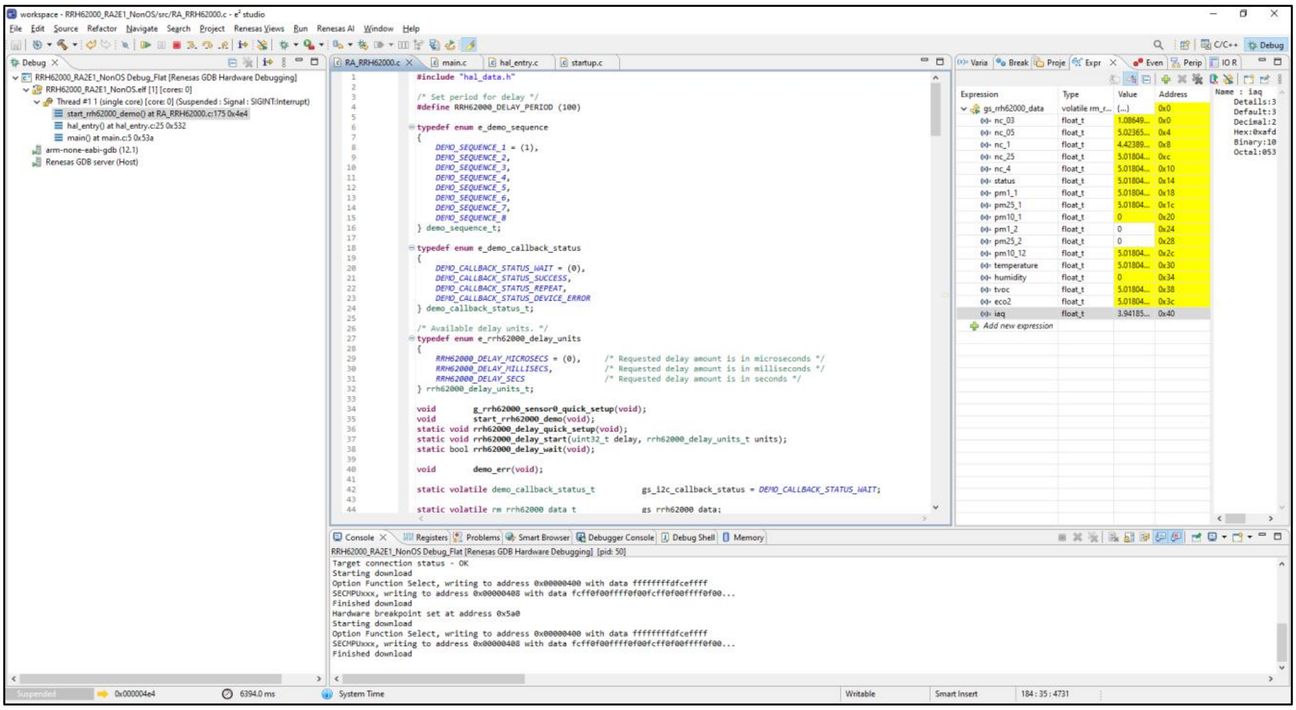
Expression: gs_rrh62000_data, nc_03, nc_05, nc_1, nc_25, nc_4, status, pm1_1, pm25_1, pm10_1, pm1_2, pm25_2, pm10_12, temperature, humidity, tvoc, eco2, iaq, Add new expression

RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging] [pid: 43]

```
Target connection status - OK
Target connection status - OK
Starting download
Option Function Select, writing to address 0x00000400 with data ffffffffdfceffff
SECMPUxxx, writing to address 0x00000408 with data fcff0f00ffff0f00fcff0f00ffff0f00...
Finished download
Hardware breakpoint set at address 0x6d6
```

4. Start the Debug.

It is possible to check the real-time values.



workspace - RRH62000_RA2E1_NonOS/src/RA_RRH62000.c - e² studio

- RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging]
  - RRH62000_RA2E1_NonOS.elf [1] [cores: 0]
    - Thread #1 1 (single core) [core: 0] (Suspended : Signal : SIGINT:Interrupt)
      - start_rrh62000_demo() at RA_RRH62000.c:175 0x4e4
      - hal_entry() at hal_entry.c:25 0x532
      - main() at main.c:5 0x53a
  - arm-none-eabi-gdb (12.1)
  - Renesas GDB server (Host)

RA_RRH62000.c

```c
#include "hal_data.h"

/* Set period for delay */
#define RRH62000_DELAY_PERIOD (100)

typedef enum e_demo_sequence
{
    DEMO_SEQUENCE_1 = (1),
    DEMO_SEQUENCE_2,
    DEMO_SEQUENCE_3,
    DEMO_SEQUENCE_4,
    DEMO_SEQUENCE_5,
    DEMO_SEQUENCE_6,
    DEMO_SEQUENCE_7,
    DEMO_SEQUENCE_8
} demo_sequence_t;

typedef enum e_demo_callback_status
{
    DEMO_CALLBACK_STATUS_WAIT = (0),
    DEMO_CALLBACK_STATUS_SUCCESS,
    DEMO_CALLBACK_STATUS_REPEAT,
    DEMO_CALLBACK_STATUS_DEVICE_ERROR
} demo_callback_status_t;

/* Available delay units. */
typedef enum e_rrh62000_delay_units
{
    RRH62000_DELAY_MICROSECS = (0),     /* Requested delay amount is in microseconds */
    RRH62000_DELAY_MILLISECS,           /* Requested delay amount is in milliseconds */
    RRH62000_DELAY_SECS                 /* Requested delay amount is in seconds */
} rrh62000_delay_units_t;

void        g_rrh62000_sensor0_quick_setup(void);
void        start_rrh62000_demo(void);
static void rrh62000_delay_quick_setup(void);
static void rrh62000_delay_start(uint32_t delay, rrh62000_delay_units_t units);
static bool rrh62000_delay_wait(void);

void        demo_err(void);

static volatile demo_callback_status_t         gs_i2c_callback_status = DEMO_CALLBACK_STATUS_WAIT;

static volatile rm_rrh62000_data_t             gs_rrh62000_data;
```

Expression	Type	Value	Address
gs_rrh62000_data	volatile rm_r...	{...}	0x0
nc_03	float_t	1.08649...	0x0
nc_05	float_t	5.02365...	0x4
nc_1	float_t	4.42389...	0x8
nc_25	float_t	5.01804...	0xc
nc_4	float_t	5.01804...	0x10
status	float_t	5.01804...	0x14
pm1_1	float_t	5.01804...	0x18
pm25_1	float_t	5.01804...	0x1c
pm10_1	float_t	0	0x20
pm1_2	float_t	0	0x24
pm25_2	float_t	0	0x28
pm10_12	float_t	5.01804...	0x2c
temperature	float_t	5.01804...	0x30
humidity	float_t	0	0x34
tvoc	float_t	5.01804...	0x38
eco2	float_t	5.01804...	0x3c
iaq	float_t	3.94185...	0x40
Add new expression			

```
Name : iaq
    Details:3
    Default:3
    Decimal:2
    Hex:0xafd
    Binary:10
    Octal:053
```

RRH62000_RA2E1_NonOS Debug_Flat [Renesas GDB Hardware Debugging] [pid: 50]

```
Target connection status - OK
Starting download
Option Function Select, writing to address 0x00000400 with data ffffffffdfceffff
SECMPUxxx, writing to address 0x00000408 with data fcff0f00ffff0f00fcff0f00ffff0f00...
Finished download
Hardware breakpoint set at address 0x5a0
Starting download
Option Function Select, writing to address 0x00000400 with data ffffffffdfceffff
SECMPUxxx, writing to address 0x00000408 with data fcff0f00ffff0f00fcff0f00ffff0f00...
Finished download
```

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Feb.21.25	-	First Release

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

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

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

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

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

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

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

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/.

Trademarks

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