# RA Family, RX Family, RL78 Family

## FS3000 Sample Software Manual

### Introduction

This application note describes the sample software that is for use with the FS3000 flow sensor and runs on certain MCUs of the RA family, and RX family, RL78 family.

### Target Devices

RA0E1 Group

RA2E1 Group

RX65N Group

RL78/G23 Group

### Target Sensor Board

Air Velocity Sensor Pmod$^{TM}$ Board (US082-FS3000EVZ)

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 “6.4 Notes for Interrupt Signal Circuits”
— RESET Signal Circuit: Refer to “6.5 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$^{TM}$ is a trademark of Amazon Web Services, Inc.

Pmod$^{TM}$ is a trademark of Digilent Inc.

# 1. Overview

This sample software acquires data from the FS3000-1005 air flow sensor and handles calculations on the data. In combination with the I2C driver of the FSP or FIT, the sample software controls the FS3000-1005 through the I2C in the MCU to acquire ADC data from the sensor and calculate the air velocity.

## 1.1 Terms/Abbreviations

The terms and their abbreviations are listed below.

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

| Terms | Abbreviation |
|---|---|
| FS3000 Sensor Control Module | Sensor Control Module<br>When MCU is RA Family, “rm_fs3000”<br>When MCU is RX Family, “r_fs3000_rx”<br>When MCU is RL78 Family, “r_fs3000” |
| I2C Communication Middleware | COMMS_I2C<br>When MCU is RA Family, “rm_comms_i2c”<br>When MCU is RX Family, “r_comms_i2c_rx”<br>When MCU is RL78 Family, “r_comms_i2c” |
| I2C Driver | When MCU is RA Family, “r_iic_master”, “r_sci_i2c”, “r_iica_master”<br>When MCU is RX Family, “r_riic_rx”, “r_sci_iic_rx”<br>When MCU is RL78 Family, “r_iica_master” |
| Serial Communications Interface | When MCU is RA Family, “SCI”, “SCI I/F”<br>When MCU is RX Family, “SCI”, “SCI I/F” |
| Serial Array Unit | When MCU is RA Family, “SAU”, “SAU I/F”<br>When MCU is RL78 Family, “SAU”, “SAU I/F” |
| I2C Bus Interface | When MCU is RA Family, “IIC”, “IIC I/F”<br>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<br>I2C Bus Interface,<br>I2C Bus Interface (IICA),<br>Serial Interface (IICA) | “I2C I/F” |
| General Term for Interrupt Controller | “ICU I/F” (Interrupt Controller Unit) |
| General purpose I/O Port | “GPIO”, “GPIO I/F” |
| Pin No.1 (#1) of Renesas Pmod Type 6A Sensor Board | “IRQ#” (L output when an interrupt occurs) |

# 2. Environment for Confirming Operation

## 2.1 Environment for Confirming Operation on RA Family MCU

The operation of this software has been confirmed on the MCU of the RA family 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 2024-07 |
| C compiler | GNU ARM Embedded 13.2.1.arm-13-7 |
| Configuration options | ISO C99 (-std=c99)<br>Optimization Level: Default settings (-O2) |
| FSP | v5.5.0 |
| RTOS | FreeRTOS v10.6.1 |
| Emulator | On board (J-LINK) |
| Interposer | Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ) |
| Sensor board | Air Velocity Sensor Pmod Board (US082-FS3000EVZ) |

**Table 2-2 Amount of Memory Used in RA2E1 Group**

| Area | Size (Non-OS) [Bytes] | Size (FreeRTOS) [Bytes] |
|---|---|---|
| ROM | 1,548 | 3,478 (Nore 1) |
| RAM | 136 | 372 |

Note Memory size is calculated for the sample code, FS3000 sensor control module, and COMMS_I2C. They do not include the sizes of FreeRTOS threads for the FreeRTOS versions.

Note 1 This includes an increase of 1,572 bytes due to the Relax function.

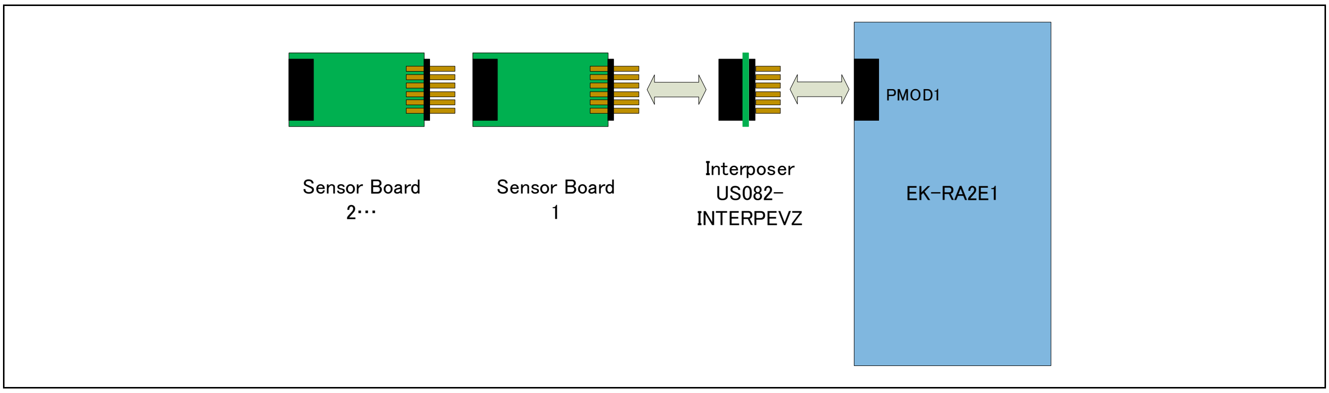


**Figure 2-1 Hardware Connections for EK-RA2E1**

### (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 2024-07 |
| C compiler | GNU ARM Embedded 13.2.1.arm-13-7 |
| Configuration options | ISO C99 (-std=c99)<br>Optimization Level: Default settings (-Oz) |
| FSP | v5.5.0 |
| Emulator | On board (J-LINK) |
| Interposer | Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ) |
| Sensor board | Air Velocity Sensor Pmod Board (US082-FS3000EVZ) |

Table 2-4 Amount of Memory Used in RA0E1 Group

| Area | Size [Bytes] |
|---|---|
| ROM | 1,306 |
| RAM | 124 |

Note Memory size is calculated for the sample code, FS3000 sensor control module, and COMMS_I2C.

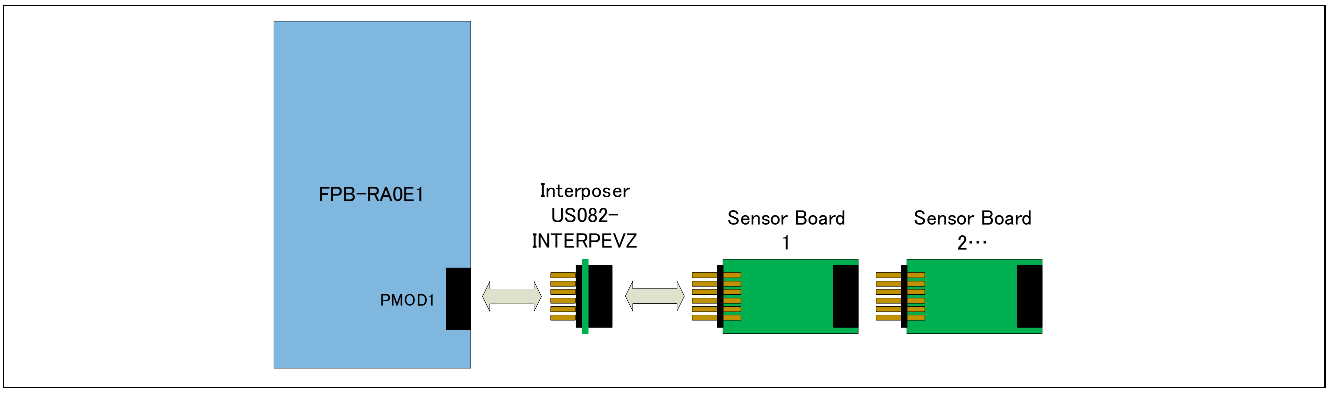


Figure 2-2 Hardware Connections for FPB-RA0E1

### (3) Use of Interposer Board

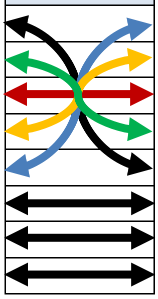
The Interposer Board is an I/F conversion board for connecting Pmod Type 6A sensors by switching the Pmod Type 2A/Type 3A connector of the SCI I/F to the Simple IIC function.

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,<br>Type 3A | SCI I/F,<br>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,<br>Type 6A | SCI I/F, IIC I/F,<br>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 is connected to Pmod #1 on the Pmod Sensor Board.

Note 3: For an interrupt signal circuit, refer to “6.4 Notes on Matching with Interrupt Signal Circuits”.

Application example: FPB-RA0E1 PMOD1 is applicable.

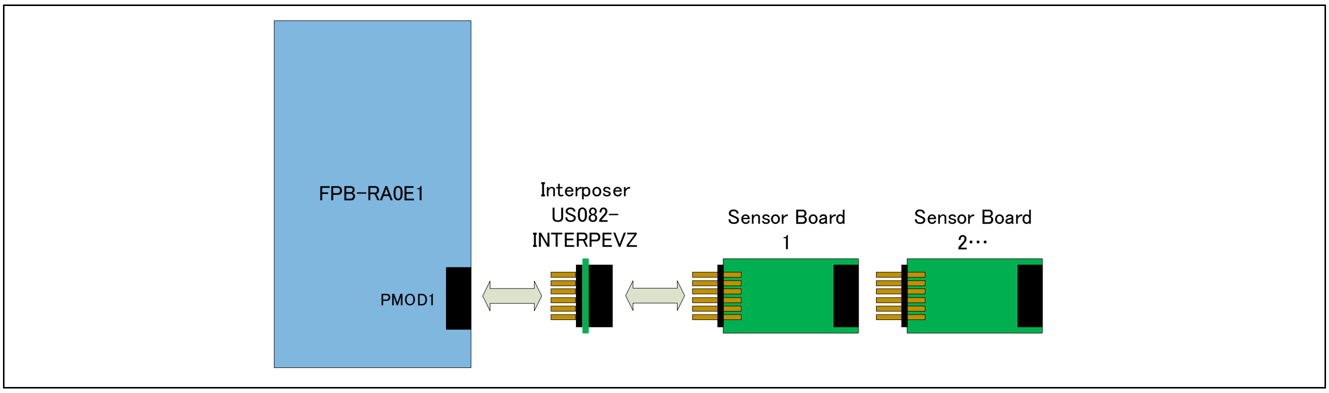


**Figure 2-3 Hardware Connections for using IICA at PMOD1 Type 2A, Type 3A on FPB-RA0E1**

## 2.2 Environment for Confirming Operation on RX Family MCU

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

### (1) RX65N Envision Kit

Table 2-6 Confirming Operating Environment for RX65N Envision Kit

| Item | Description |
|---|---|
| Demonstration board | RPBRX65N (RX65N Envision Kit) |
| Microcontroller | RX65N (R5F565NEDDFB: 144pin) |
| Operating frequency | 120MHz |
| Operating voltage | 5V |
| Integrated development environment | Renesas Electronics e$^2$ studio 2024-07 |
| C compiler | Renesas Electronics CC-RX V.3.02.00 |
| Configuration options | C99 (-lang = c99)<br>Optimization Level: Default settings (Level 2) |
| FIT | Board Support Packages (r_bsp) v7.20<br>FS3000 Sensor Middleware (r_fs3000_rx) v1.00<br>IIC Communication Driver Interface Middleware (r_comms_i2c_rx) v1.20<br>RIIC Multi Master I2C Driver (r_riic_rx) v2.49<br>Simple IIC Driver (r_sci_iic_rx) v2.49 |
| RTOS | FreeRTOS Kernal 10.4.3-rx-1.0.1、FreeRTOS Object 10.4.3-rx-1.0.1 |
| Emulator | On board (E2OB) |
| Interposer | Interposer Board for Pmod Type2/3 to 6A (US082-INTERPEVZ) |
| Sensor board | Air Velocity Sensor Pmod Board (US082-FS3000EVZ) |

Table 2-7 Amount of Memory Used in RX65N Group

| Area | Size (Non-OS) [Bytes] | Size (FreeRTOS) [Bytes] |
|---|---|---|
| ROM | 1,638 | 1,844 |
| RAM | 118 | 141 |

Note Memory size is calculated for the sample code, FS3000 sensor control module, and COMMS_I2C. They do not include the sizes of FreeRTOS threads for the FreeRTOS versions.

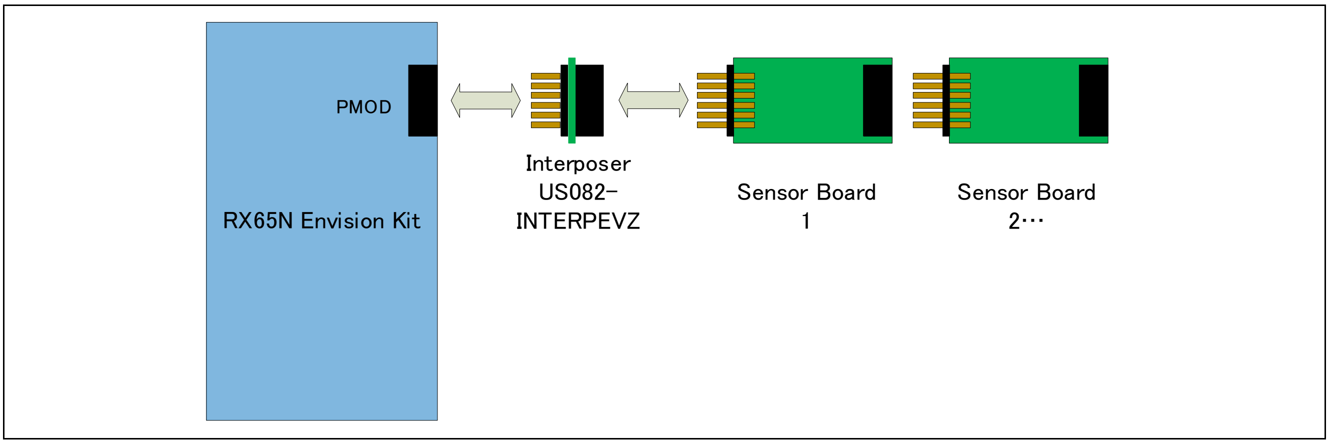


Figure 2-4 Hardware Connections for RX65N Envision Kit

### (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 Family MCU

The operation of this software has been confirmed on the MCU of the RL78/G23 group 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: 128 pins) |
| Operating frequency | 32MHz |
| Operating voltage | 3.3V |
| Integrated development environment | Renesas Electronics e$^2$ Studio 2024-07 |
| C compiler | Renesas Electronics CC-RL V1.14.00<br>LLVM for RL78 17.0.1.202409 |
| Configuration options | Add the following settings to the compiler default settings.<br>CC-RL: C99 (-lang = c99), Optimization Level: Default settings (-Odefault)<br>LLVM: GNU ISO C99 (-std = gnu99), Optimization Level: Default settings (-Og) |
| SIS / CG | Board Support Packages (r_bsp) v1.70<br>FS3000 Sensor Middleware (r_fs3000) v1.02<br>IIC Communication Driver Interface Middleware (r_comms_i2c) v1.11<br>IIC Communication (Master mode) v1.6.0 |
| Emulator | On board (COM Port) |
| Sensor board | Air Velocity Sensor Pmod Board (US082-FS3000EVZ) |

Table 2-9 Amount of Memory Used in RL78/G23 Group

| Area | Size [Bytes] (CC-RL) |
|---|---|
| ROM | 1,905 |
| RAM | 85 |

Note Memory size is calculated for the sample code, FS3000 sensor control module, and COMMS_I2C.

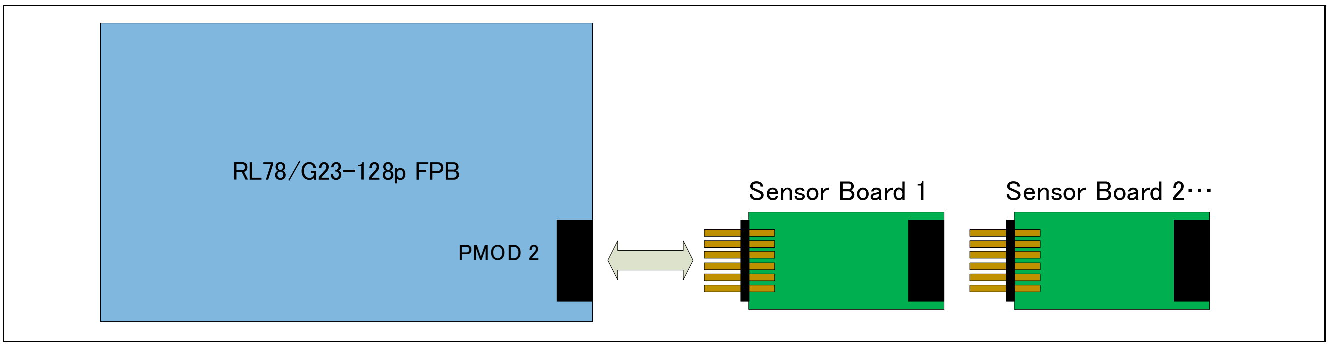


Figure 2-5 Hardware Connections for RL78/G23-128p FPB

### (2) Use of Interposer Board

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

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

# 3. Sensor Specifications

## 3.1 Overview of Sensor Specifications

The FS3000 is a surface-mount type air velocity module utilizing a MEMS thermopile-based sensor. The FS3000 features a digital output with 12-bit resolution. Please refer to the FS3000 datasheet for more information about the sensor module, including parameters that describe the module's characteristics.

## 3.2 Sensor Functions

The FS3000 sample software supports the FS3000-1005 air flow sensor; it does not support the FS3000-1015 sensor.

The sensor begins measurement as soon as the power supply is turned on.

To obtain data from the sensor, send the Flow Data Read command and then read the five bytes of data. For commands, please refer to the FS3000 datasheet.

The air velocity is represented by 12 bits; the four lower-order bits of the second byte (Byte 2) are valid.

# 4. Sample Software Specifications

This sample software package contains a total of 7 projects: non-OS and OS (FreeRTOS) versions for the RA2E1 group, non-OS version for the RA0E1 group, non-OS and OS (FreeRTOS) versions for the RX65N group, and a non-OS version (CC-RL / LLVM) for the RL78/G23 group. This section describes these projects.

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

## 4.1 Configuration of Sample Software

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

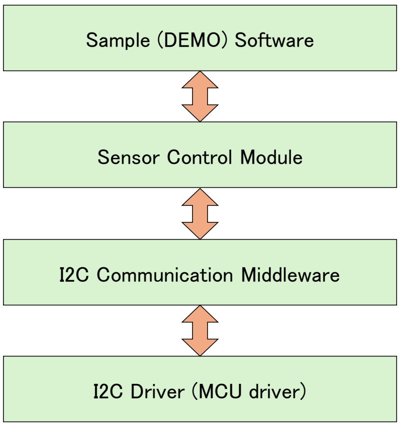


**Figure 4-1 Layer diagram of Sample Software**

## 4.2 Specifications of Sensor API Functions

### 4.2.1 List of Sensor API functions

The Sensor Control Module API includes the following functions.

For details on the Function API, see below.

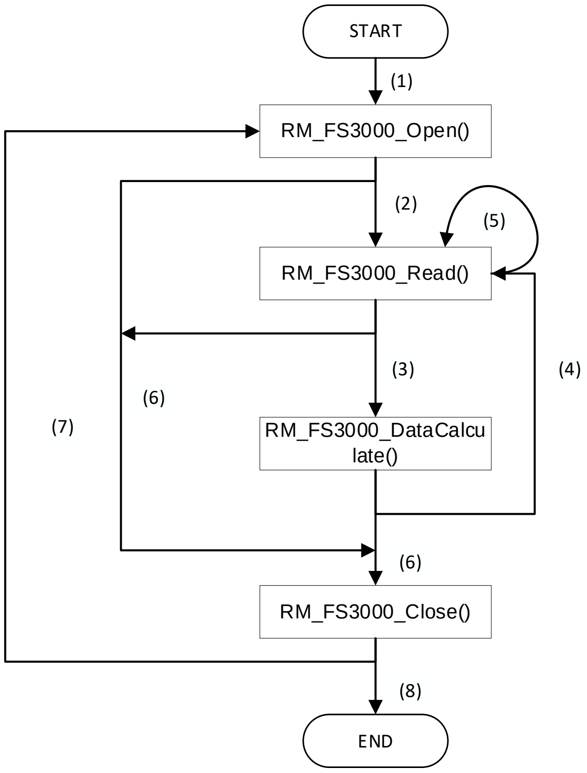
RA Flexible Software Package Documentation
Renesas Sensor Control Modules Firmware Integration Technology
Renesas Sensor Control Modules Software Integration System

**Table 4-1 List of Sensor API Functions**

| Function | Description |
|---|---|
| RM_FS3000_Open() | Starts control of the sensor. |
| RM_FS3000_Close() | Terminates control of the sensor. |
| RM_FS3000_Read() | Acquires data from the sensor. |
| RM_FS3000_DataCalculate() | Calculates values from the data acquired from the sensor. |

### 4.2.2 Guide to Using API Functions

The following shows the transition diagram of functions calling order as the usage condition of API functions.



| Flow No. | Function | Transition Case |
|---|---|---|
| (1) | Start FS3000 | not open FS3000 |
| (2) | ADC Read | open success |
| (3) | calculate | ADC read success (callback state success) |
| (4) | next read ADC | DataCalculate return success and next ADC data Get |
| (5) | get retry ADC data | user get retry ADC data or DataCalculate return invalid data |
| (6) | Close FS3000 | Error state or user stop |
| (7) | Restart FS3000 | Close success and user restart |
| (8) | Stop FS3000 | Close success or Error |

**Figure 4-2 Diagram of Transitions between API Function Calls**

The conditions for calling the individual functions are shown below.

- RM_FS3000_Open() (1) Activation of FS3000 or
  (7) Restart after a call of RM_FS3000_Close()
- RM_FS3000_Close() (6) Successful completion or abnormal end of individual processing
- RM_FS3000_Read() (2) Acquisition of measured data after the start of measurement or
  (5) Retry after waiting for the response to the data acquisition request
- RM_FS3000_DataCalculate() (3) Calculation of data after a call of RM_FS3000_Read()

Note: When using an OS and controlling the sensor with multiple threads or tasks simultaneously in use, the user will need to use a semaphore to control the bus. For the timing of the semaphore being raised and the control of blocking, refer to section “4.4 Flowchart of OS Version of Sample Software”.

## 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.

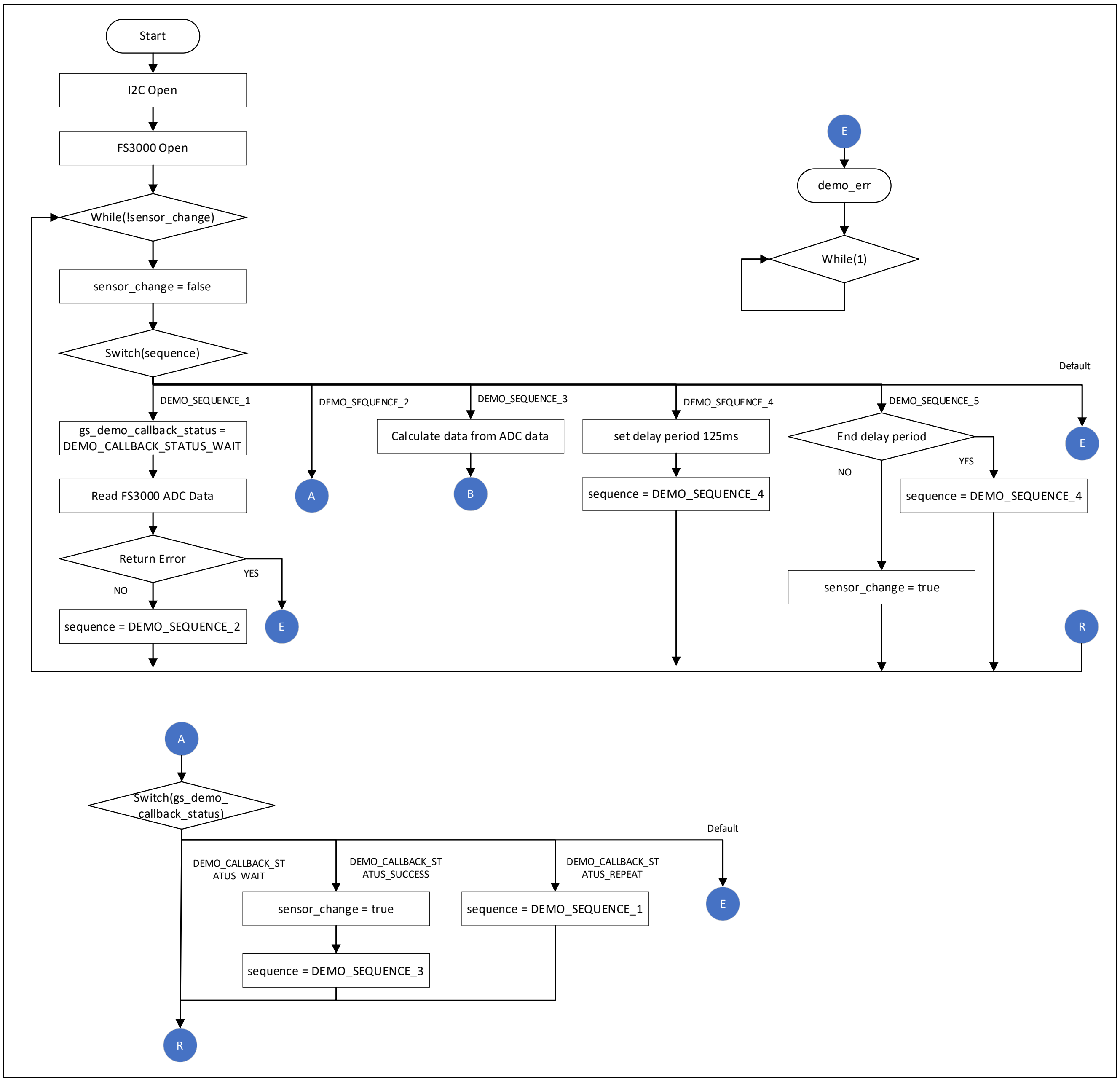

Start
I2C Open
FS3000 Open
While(!sensor_change)
sensor_change = false
Switch(sequence)
DEMO_SEQUENCE_1
gs_demo_callback_status = DEMO_CALLBACK_STATUS_WAIT
Read FS3000 ADC Data
Return Error
YES
NO
sequence = DEMO_SEQUENCE_2
E
DEMO_SEQUENCE_2
A
DEMO_SEQUENCE_3
Calculate data from ADC data
B
DEMO_SEQUENCE_4
set delay period 125ms
sequence = DEMO_SEQUENCE_4
DEMO_SEQUENCE_5
End delay period
YES
NO
sequence = DEMO_SEQUENCE_4
sensor_change = true
Default
E
R
E
demo_err
While(1)
A
Switch(gs_demo_callback_status)
DEMO_CALLBACK_STATUS_WAIT
DEMO_CALLBACK_STATUS_SUCCESS
sensor_change = true
sequence = DEMO_SEQUENCE_3
DEMO_CALLBACK_STATUS_REPEAT
sequence = DEMO_SEQUENCE_1
Default
E
R

**Figure 4-3 Flowchart of Main Processing in Non-OS Version of FS3000 Sample Software (1)**

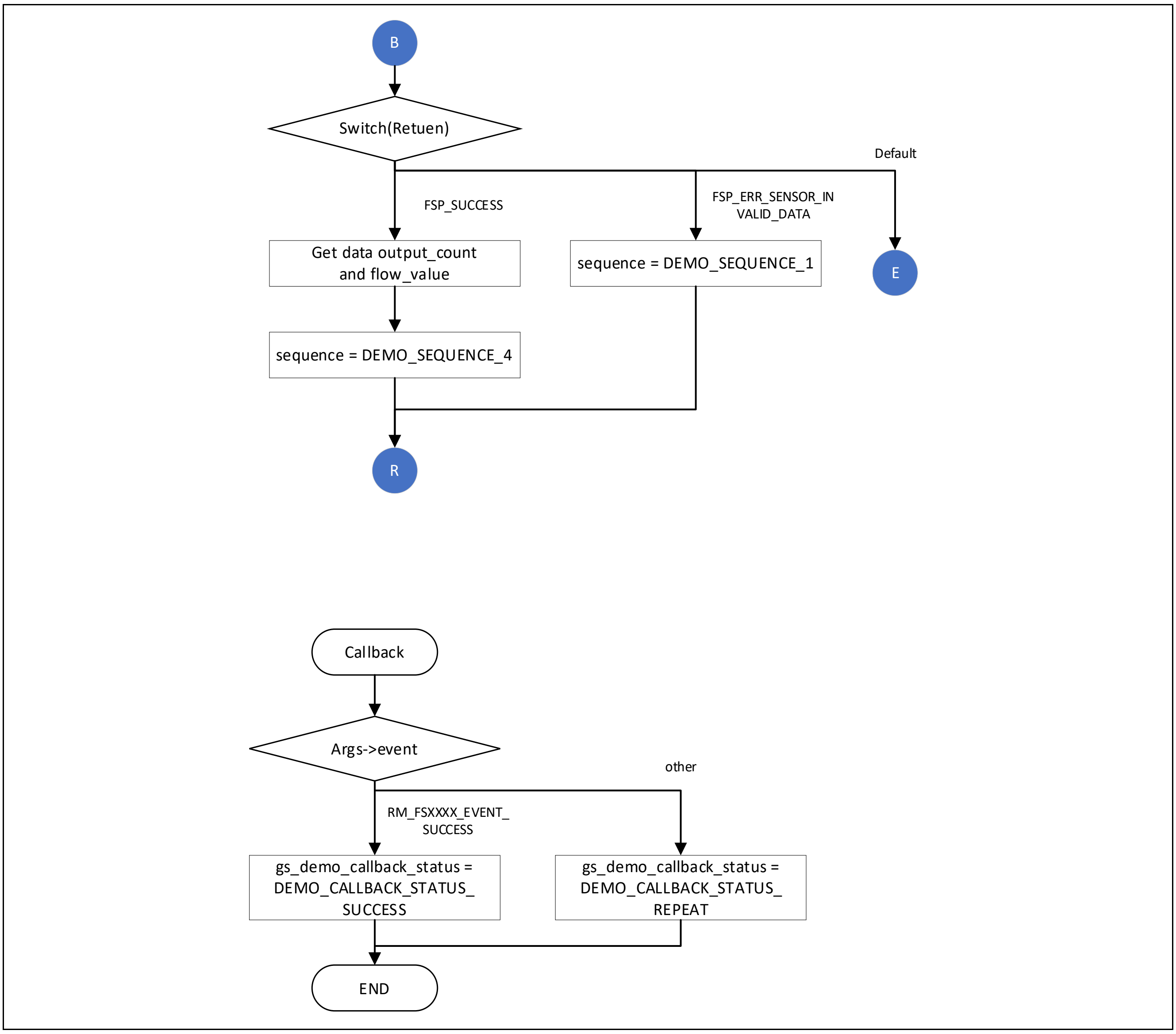

B
Switch(Retuen)
Default
FSP_SUCCESS
FSP_ERR_SENSOR_IN VALID_DATA
Get data output_count and flow_value
sequence = DEMO_SEQUENCE_1
E
sequence = DEMO_SEQUENCE_4
R
Callback
Args->event
other
RM_FSXXXX_EVENT_ SUCCESS
gs_demo_callback_status = DEMO_CALLBACK_STATUS_ SUCCESS
gs_demo_callback_status = DEMO_CALLBACK_STATUS_ REPEAT
END

**Figure 4-4 Flowchart of Main Processing in Non-OS Version of FS3000 Sample Software (2)**

## 4.4 Flowchart of OS Version of Sample Software

The OS version uses a semaphore in control of the sensor and operates two threads for controlling the sensor in parallel.

The sensor control in each thread first starts the driver and then repeats the processing for acquiring data from the sensor and calculating values from the results of measurement.

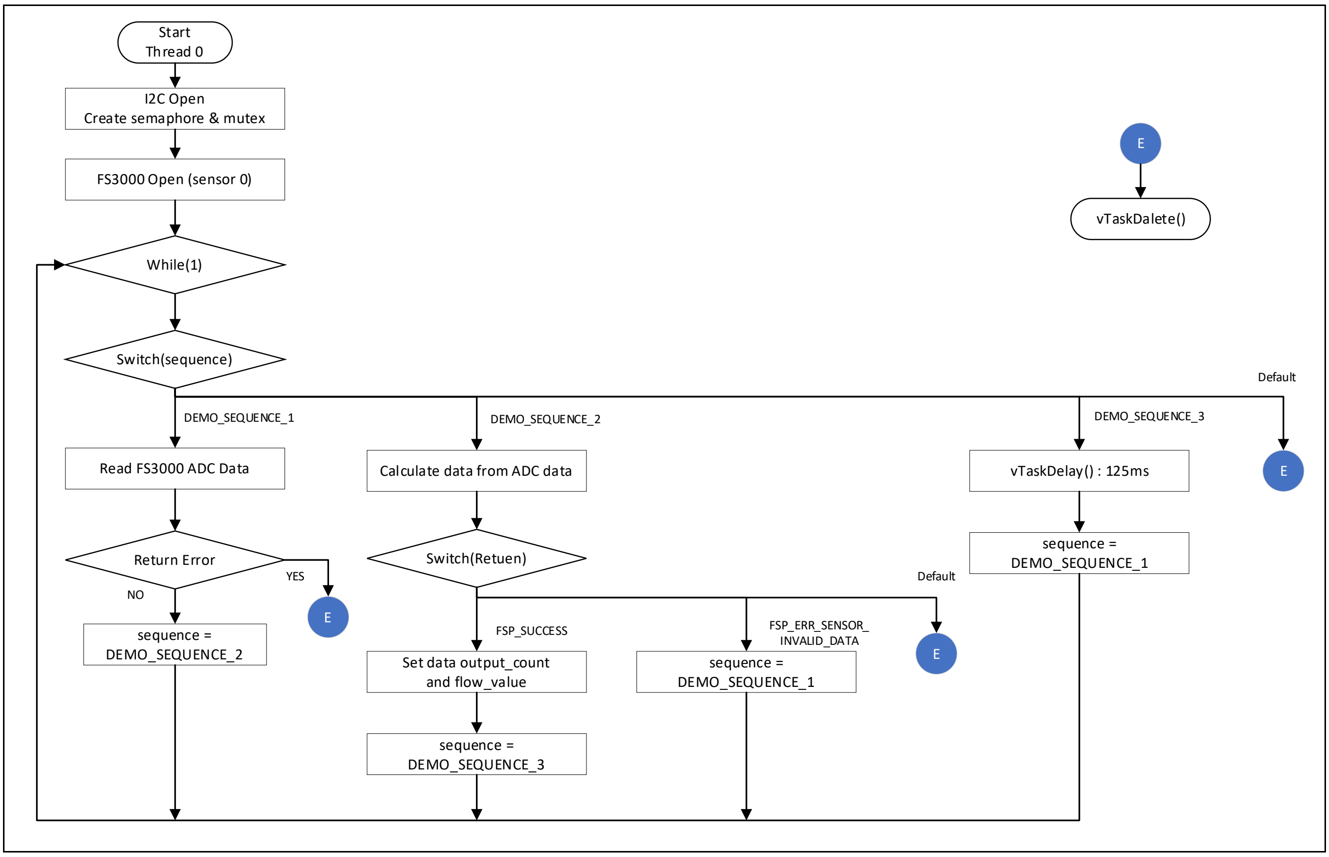


**Figure 4-5 Flowchart of Main Processing in OS Version of FS3000 Sample Software (1)**

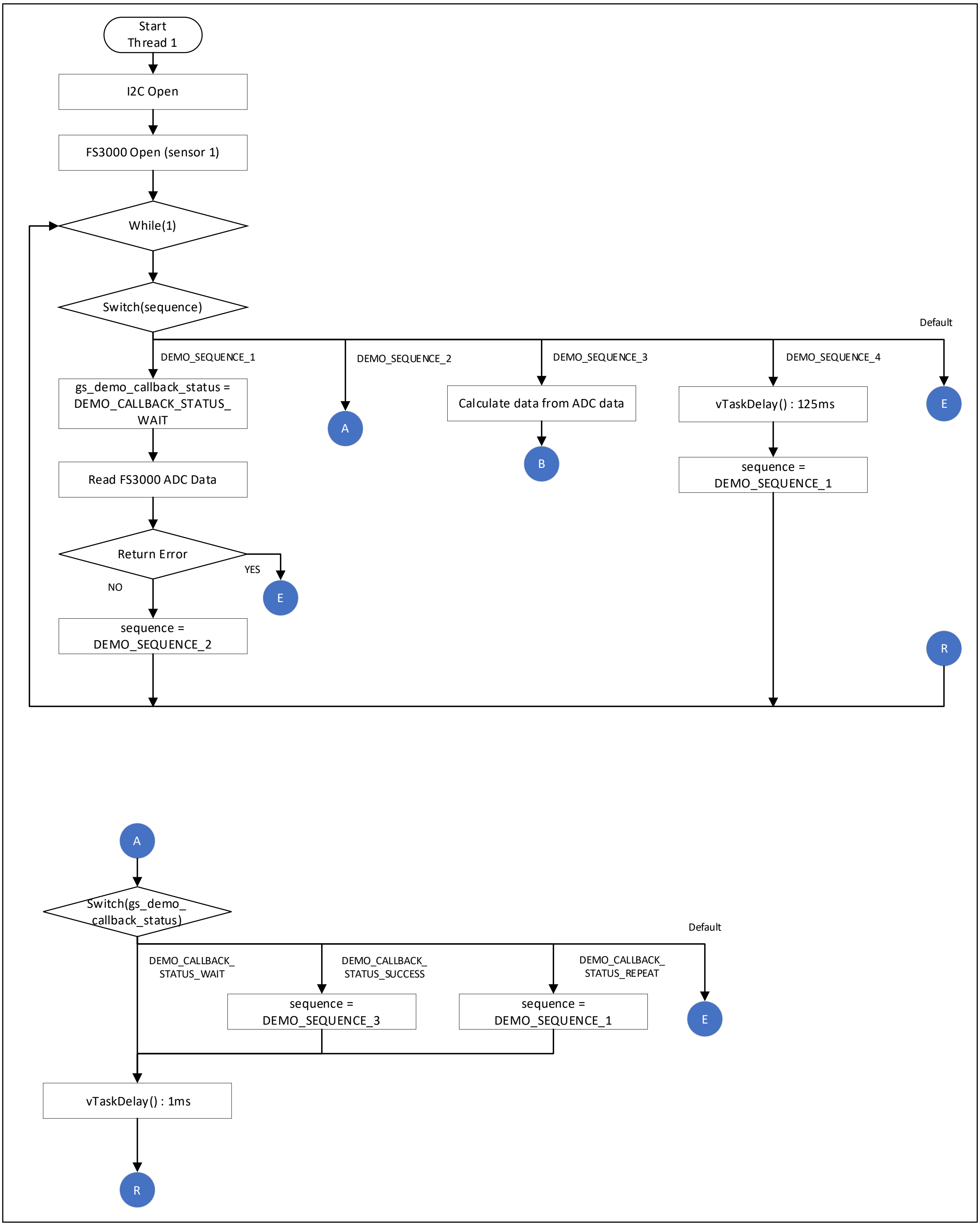

Start
Thread 1
I2C Open
FS3000 Open (sensor 1)
While(1)
Switch(sequence)
Default
DEMO_SEQUENCE_1
DEMO_SEQUENCE_2
DEMO_SEQUENCE_3
DEMO_SEQUENCE_4
gs_demo_callback_status = DEMO_CALLBACK_STATUS_WAIT
A
Calculate data from ADC data
B
vTaskDelay() : 125ms
sequence = DEMO_SEQUENCE_1
E
Read FS3000 ADC Data
Return Error
YES
NO
E
sequence = DEMO_SEQUENCE_2
R
A
Switch(gs_demo_callback_status)
Default
DEMO_CALLBACK_STATUS_WAIT
DEMO_CALLBACK_STATUS_SUCCESS
DEMO_CALLBACK_STATUS_REPEAT
sequence = DEMO_SEQUENCE_3
sequence = DEMO_SEQUENCE_1
E
vTaskDelay() : 1ms
R

**Figure 4-6 Flowchart of Main Processing in OS Version of FS3000 Sample Software (2)**

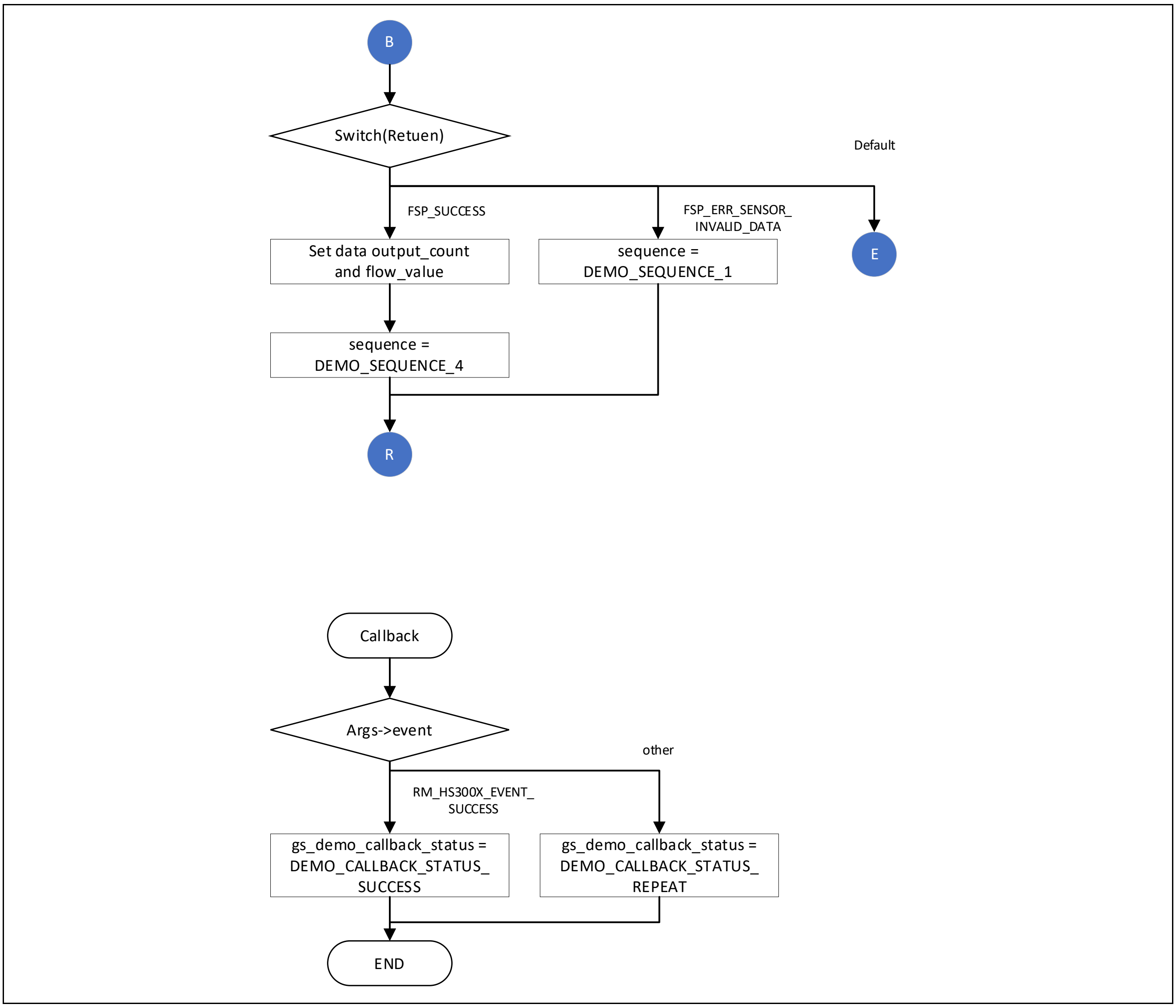


Figure 4-7 Flowchart of Main Processing in OS Version of FS3000 Sample Software (3)

# 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 latest version, the settings items and values shown below may differ.

## 5.1 FS3000 Air Velocity Sensor Settings

### 5.1.1 RA Family

Select the **"rm_fs3000"** stack in the "Stack" tabbed page of the FSP Configurator, and the configurable items are shown in the "Properties" tabbed page.

**Table 5-1 FS3000 Settings for RA Family**

| Configurable Item | Value | Description |
|---|---|---|
| Common | | |
| Parameter Checking | Default (BSP) | Specify the include parameter check processing in code.<br>When "Disabled" is specified, excluding in the code.<br>When "Enabled" is specified, including in the code. |
| | Enabled | |
| | Disabled | |
| Device type | FS3000-1005 | Specify the type of device to be acquired from the sensor.<br>"FS3000-1005" only can be selected. |
| Module g_fs3000_sensor FS3000 on rm_fs3000 | | |
| Name | g_fs3000_sensor0 | Specify the name of the module. |
| Callback | fs3000_callback | Specify the name of the user callback function.<br>When "NULL" is specified, no callback function is used. |

### 5.1.2 RX Family

Select the **"r_fs3000_rx"** component in the "Component" tabbed page of the Smart Configurator, and the configurable items are shown in the "Configure" panel.

**Table 5-2 FS3000 Settings for RX Family**

| Configurable Item | Value | Description |
|---|---|---|
| Configurations | | |
| Parameter Checking | System Default | Specify the include parameter check processing in code.<br>When "Disabled" is specified, excluding in the code.<br>When "Enabled" is specified, including in the code. |
| | Enabled | |
| | Disabled | |
| Number of FS3000 sensors | 1 | Specify the number of FS3000 sensors. |
| | 2 | |
| Device type of FS3000 Sensors | FS3000-1005 | Specify the type of sensor.<br>"FS3000-1005" only can be selected. |
| I2C Communication device No. for FS3000 sensor device{x}<br>(x = 0 or 1) | I2C Communication Device{y}<br>(y = 0 - 15) | Specify the communications device number to be used by the sensor. |
| Callback function for FS3000 sensor device{x}<br>(x = 0 or 1) | fs3000_user_callback{x}<br>(x = 0 or 1) | Specify the name of the user callback function.<br>When "NULL" is specified, no callback function is used. |

### 5.1.3 RL78 Family

Select the **“r_fs3000”** component on the "Component" tabbed page of the Smart Configurator, and the configurable items will be shown in the “Configure” panel.

**Table 5-3 FS3000 Settings for RL78 Family**

| Configurable Item | Value | Description |
|---|---|---|
| Configurations | | |
| Parameter Checking | 0 | Specify the include parameter check processing in code.<br>When “Disabled” is specified, excluding in the code.<br>When “Enabled” is specified, including in the code. |
| | 1 | |
| Number of FS3000 sensors | 1 | Specify the number of FS3000 sensors. |
| | 2 | |
| Device type of FS3000 Sensors | FS3000-1005 | Specify the type of sensor.<br>"FS3000-1005" only can be selected. |
| | FS3000-1015 | |
| Using communication line number for FS3000 sensor device{x}<br>(x = 0 or 1) | Comms{y}<br>(y = 0 - 4) | Specify the device number for COMMS_I2C. |
| Callback function for FS3000 sensor device{x}<br>(x = 0 or 1) | fs3000_user_callback{x}<br>(x = 0 or 1) | Specify the name of the user callback function.<br>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)<br>Enabled<br>Disabled | Specify the include parameter check processing in code.<br>When "Disabled" is specified, excluding in the code.<br>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 | 0x28 | Specify the slave address.<br>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.<br>No setting is required as this will be overwritten by the Sensor Control module. |
| Callback | rm_fs3000_comms_i2c_callback | Specify the name of the user callback function.<br>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<br>Use | For an RTOS project, enable or disable the blocking processing. |
| Recursive Mutex for Bus | Unuse<br>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.<br>This setting is valid only when the I2C driver is "r_iic_master".<br>When using other I2C drivers, this setting is invalid. |
| Rate | Standard<br>Fast-mode<br>Fast-mode plus | Specify the bit rate.<br>This setting is valid only when the I2C driver is "r_iic_master".<br>When using FS3000, 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.<br>When using other I2C drivers, this setting is invalid. |

### 5.2.2 RX Family

Select the **"r_comms_i2c_rx"** component in 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<br>Enabled<br>Disabled | Specify the include parameter check processing in code.<br>When "Disabled" is specified, excluding in the code.<br>When "Enabled" is specified, including in the code. |
| Number of I2C Shared Buses | Unused<br>1<br>2 - 16 | Specify the number of communications bus lines that can be connected. |
| Number of I2C Communication Devices | Unused<br>1<br>2 - 16 | Specify the number of I2C device that can be connected. |
| Blocking operation supporting with RTOS | Disabled<br>Enabled | For an RTOS project, enable or disable the blocking operation. |
| Bus lock operation supporting with RTOS | Disabled<br>Enabled | For an RTOS project, enable or disable the bus lock operation. |
| I2C Driver Type for I2C Shared bus{x}<br>(x = 0 - 15) | RIIC<br>SCI IIC<br>Not selected | Specify the I2C bus type to be used for the communication bus.<br>When using the "RIIC", r_riic_rx is necessary. When using the "SCI IIC", "r_sci_iic_rx is necessary.<br>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}<br>(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}<br>(x = 0 - 15) | 0xFFFFFFFF | Specify the time of I2C bus lock timeout. |
| I2C Shared Bus No. for I2C Communication Device{x}<br>(x = 0 - 15) | I2C Shared Bus{x}<br>(x = 0 - 15) | Specify the configuration of used communication bus. |
| Slave address for I2C Communication device{x}<br>(x = 0 - 15) | 0x00 | Specify the slave address of the device to be connected to the communications bus.<br>When using FS3000, specify 0x38. |
| Address mode for I2C Communication device{x}<br>(x = 0 - 15) | 7 bit address mode | Specify the slave address mode.<br>When using FS3000, specify the 7-bit address mode. |
| Callback function for I2C Communication device{x}<br>(x = 0 - 15) | comms_i2c_user_callback{x}<br>(x = 0 - 15) | Specify the name of the user callback function.<br>When using r_fs3000_rx, specify rm_ fs3000_callback{y} (y = 0 or 1). |
| Timeout for the blocking bus of I2C Communication device{x}<br>(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.<br>When “Disabled” is specified, excluding in the code.<br>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}<br>(x = 0 - 4) | IICA | Specify the I2C type to be used for the communication bus.<br>When using FS3000, specify “IICA”. |
| | SAU IIC | |
| | Not selected | |
| Component name for the I2C bus{x} | Config_IIC00 | Specify the I2C bus component name to be used for the communication bus. |
| I2C Shared Bus No. for I2C Communication Device{x}<br>(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}<br>(x = 0 - 4) | 0x00 | Specify the slave address of the device to be connected to the communications bus.<br>When using FS3000, specify 0x28. |
| Callback function for I2C Communication device{x}<br>(x = 0 - 4) | comms_i2c_user_callback{x}<br>(x = 0 - 4) | Specify the name of the user callback function.<br>When using r_fs3000_rx, specify rm_ fs3000_callback{y} (y = 0 or 1). |

## 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.<br>When “Disabled” is specified, excluding in the code.<br>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.<br>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.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| Rate | Standard | Specify the bit rate.<br>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.<br>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.<br>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.<br>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.<br>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.<br>Use the "Pins" tabbed page to modify the pin configuration. |
| SCL | Pxxx | |

### (2) r_sci_i2c

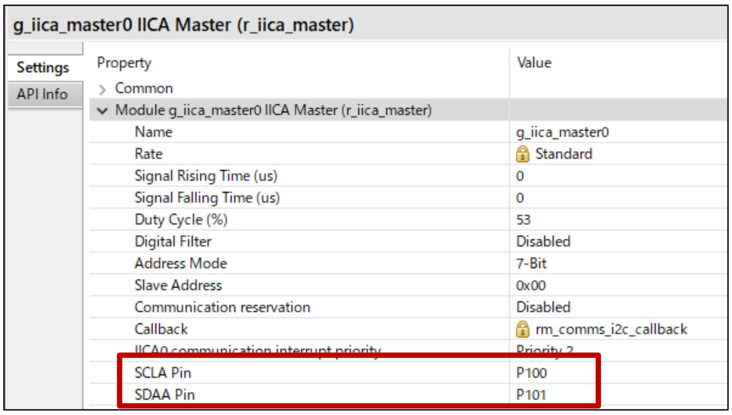
Table 5-8 r_sci_i2c Settings for RA Family

| Configurable Item | Value | Description |
|---|---|---|
| Common | | |
| Parameter Checking | Default (BSP) | Specify the include parameter check processing in code.<br>When “Disabled” is specified, excluding in the code.<br>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.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| | Disabled | |
| Module g_i2c0 I2C Master (r_sci_i2c) | | |
| Name | g_i2c0 | Specify the name of the module. |
| Channel | 0 | Specify the channel number to be used. |
| Slave Address | 0x00 | Specify the slave address for the device to be connected.<br>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.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| | 10-bit | |
| Rate | Standard | Specify the bit rate.<br>When using FS3000, Standard or Fast-mode can be set. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices. |
| | Fast-mode | |
| Custom Rate (bps) | 0 | Specify the custom bit rate.<br>This setting is valid when the value is other than 0. Use this setting when you want to set the low bitrate within the “Rate” setting range. |
| SDA Output Delay (nano seconds) | 300 | Specify the SDA output delay time. |
| Noise filter setting | Use clock signal divided by 1 with noise filter | Specify the noise filter to be used for input signals. |
| | Use clock signal divided by 2 with noise filter | |
| | Use clock signal divided by 4 with noise filter | |
| | Use clock signal divided by 8 with noise filter | |
| Bit Rate Modulation | Enable | Enable or disable the bit rate modulation function. |
| | Disable | |
| Callback | rm_comms_i2c_callback | Set the user callback function name.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| Interrupt Priority Level | Priority 0 (highest) | Specify the interrupt priority level of the I2C bus driver. |
| | Priority 1 | |
| | Priority 2 | |
| | Priority 3 | |
| RX Interrupt Priority Level [Only used when DTC is enabled] | Priority 0 (highest) | When using the DTC, specify the priority level of the reception interrupt. |
| | Priority 1 | |
| | Priority 2 | |
| | Priority 3 | |
| | Disabled | |
| Pins | | |
| SDA | Pxxx | The pin numbers to be used by the driver are displayed.<br>Use the "Pins" tabbed page to modify the pin configuration. |
| SCL | Pxxx | |

### (3) r_iica_master

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

Table 5-9 r_iica_master Settings for RA Family

| Configurable Item | Value | Description |
|---|---|---|
| Common | | |
| Parameter Checking | Default (BSP)<br>Enabled<br>Disabled | Specify the include parameter check processing in code.<br>When “Disabled” is specified, excluding in the code.<br>When “Enabled” is specified, including in the code. |
| 10-bit slave addressing | Enabled<br>Disabled | Specify whether to support 10-bit addressing for the slave address.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| Module g_iica_master0 IICA Master (r_iica_master) | | |
| Name | g_iica_master0 | Specify the name of the module. |
| Rate | Standard<br>Fast-mode<br>Fast-mode plus | Specify the bit rate.<br>Specify “Standard” due to the electrical characteristics of IICAx.<br>(For FS3000, it is possible to set to “Standard” or “Fast-mode”.) |
| Custom Rate (bps) | 0 | Specify the custom bit rate.<br>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<br>Disabled | Specify whether to use the digital filter. |
| Address Mode | 7-Bit<br>10-Bit | Specify the salve address mode for the device to be connected.<br>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.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| Communication reservation | Enabled<br>Disabled | Specify whether to use the communication reservation. |
| Callback | rm_comms_i2c_callback | Set the user callback function name.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| IICA0 communication interrupt priority | Priority 0 (highest)<br>Priority 1<br>Priority 2<br>Priority 3 | Specify the interrupt priority level of the I2C bus driver. |
| SCLA Pin | Pxxx | Specify the pin numbers to be used.<br>No setting is required in “Pins” tabbed page.<br> |
| SDAA Pin | Pxxx | |

g_iica_master0 IICA Master (r_iica_master)

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

### 5.3.2 RX Family

Select the **“r_riic_rx”** or **“r_sci_iic_rx”** component in the "Component" tabbed page of the Smart Configurator, and the configurable items are 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<br>Not<br>Include | Specify the include parameter check processing in code.<br>When “Not” is specified, excluding in the code.<br>When “Include” is specified, including in the code. |
| MCU supported channels for CH{x}<br>(x = 0 - 2) | Not supported<br>Supported | Specify whether to support the operation of channel. |
| CH{x} RIIC bps(kbps)<br>(x = 0 - 2) | 400 | Specify the bit rate.<br>When using FS3000, set it to 400kbps or less. If other devices are connected on the same bus, set the transfer rate taking into consideration the transfer rate that can be set for those devices. |
| Digital filter for CH{x}<br>(x = 0 - 2) | Not<br>One IIC phi<br>Two IIC phi<br>Three IIC phi<br>Four IIC phi | Specify the digital filter for input signals. |
| Setting port setting processing | Not include port setting<br>Include port setting | Specify whether to include the pin function settings in the code to be generated. |
| Master arbitration lost detection function for CH{x}<br>(x = 0 - 2) | Unused<br>Used | Specify whether to use the master arbitration lost detection function. |
| Address {y} format for CH{x}<br>(x = 0 - 2, y = 0 - 2) | Not<br>7 bit address format<br>10 bit address format | Specify whether to support 7-bit addressing or 10-bit addressing for the slave address.<br>When using FS3000, select "7 bit address format".<br>Do not connect devices with different address formats on the same bus. |
| Slave Address {y} for CH{x}<br>(x = 0 - 2, y = 0 - 2) | 0x0025 | Specify the slave address of the designated device.<br>No setting is required as this will be overwritten by COMMS_I2C. |
| General call address for CH{x} | Unused<br>Used | Specify whether to use the general call function. |
| CH{x} RXI INT Priority Level<br>(x = 0 - 2) | Level 1<br>Level 2<br>…<br>Level 14<br>Level 15 (highest) | Specify the priority level of the reception interrupt. |
| CH{x} TXI INT Priority Level<br>(x = 0 - 2) | Level 1<br>Level 2<br>…<br>Level 14<br>Level 15 (highest) | Specify the priority level of the transmission interrupt. |

| | | |
|---|---|---|
| CH{x} EEI INT Priority Level<br>(x = 0 - 2) | Level 1<br>Level 2<br>…<br>Level 14<br>Level 15 (highest) | Specify the priority level of the error interrupt. |
| CH{x} TEI INT Priority Level<br>(x = 0 - 2) | Level 1<br>Level 2<br>…<br>Level 14<br>Level 15 (highest) | Specify the priority level of the transmission end interrupt. |
| Timeout function for CH{x}<br>(x = 0 - 2) | Unused<br>Used | Specify whether to use the timeout function. |
| Timeout detection time for CH{x}<br>(x = 0 - 2) | Long mode<br>Short mode | Specify the time for timeout detection. |
| Count up during low period of timeout detection for CH{x}<br>(x = 0 - 2) | Unused<br>Used | Specify whether to increment the counter for detecting a timeout while SCL is at the low level. |
| Count up during high period of timeout detection for CH{x}<br>(x = 0 - 2) | Unused<br>Used | Specify whether to increment the counter for detecting a timeout while SCL is at the high level. |
| Set Counter of checking bus busy | 1000 | Specify the counter value to be judged to represent the bus busy state. |
| Resources | | |
| SCLx Pins | Checked<br>Unchecked | Specify the pins to be used.<br>Select the checkboxes for the desired pins. |
| SDAx Pins | Checked<br>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<br>Not<br>Include | Specify the include parameter check processing in code.<br>When “Not” is specified, excluding in the code.<br>When “Include” is specified, including in the code. |
| MCU supported channels for CH{x}<br>(x = 0 - 12) | Not supported<br>Supported | Specify whether to support the operation of channel. |
| SCI IIC bitrate (bps) for CH{x}<br>(x = 0 - 12) | 384000 | Specify the bit rate.<br>When using FS3000, 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}<br>(x = 0 - 12) | Level 1<br>Level 2<br>…<br>Level 14<br>Level 15 (highest) | Specify the interrupt priority level. |
| Digital noise filter (NFEN bit) for CH{x}<br>(x = 0 - 12) | Disable<br>Enable | Specify whether to use the digital noise filter. |
| Noise Filter Setting Register (NFCS bit) for CH{x}<br>(x = 0 - 12) | The clock divided by 1<br>The clock divided by 2<br>The clock divided by 4<br>The clock divided by 8 | Specify the function of the digital noise filter. |
| I2C Mode Register 1 (IICDL bit) for CH{x}<br>(x = 0 - 12) | 18 | Specify the number of SDA output delay cycles.<br>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<br>Include port setting | Specify whether to include the pin function settings in the code to be generated. |
| Resources | | |
| SSCLx Pins | Checked<br>Unchecked | Specify the pins to be used.<br>Select the checkboxes for the desired pins. |
| SSDAx Pins | Checked<br>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 Serial Settings for RL78 Family**

| Configurable Item | Value | Description |
|---|---|---|
| Configurations | | |
| Clock mode setting | fCLK<br>[✓ fCLK/2] | Specify the clock to drive counting. |
| Address | 16 | Specify the local address. |
| Operation mode setting | [✓ Standard]<br>Fast mode<br>Fast mode plus | Specify the operating mode.<br>When using FS3000, 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. |
| Digital filter on | Checked<br>[✓ Unchecked] | Specify whether to use the digital filtering. |
| Transfer clock (fSCL) | 100000 | Specify the bit rate.<br>Due to the electrical characteristics of IICAx, specify 100000bps or less. |
| Set tR and tF manually | Checked<br>[✓ Unchecked] | Manually set the SDAAn and SCLAn signal rising / falling times. |
| 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)<br>Level1<br>Level2<br>[✓ Level3 (low)] | Specify the priority level of the communication end interrupt. |
| Master transmission end | [✓ Checked]<br>Unchecked | Specify whether to use the callback function when master transmission ends. |
| Master reception end | [✓ Checked]<br>Unchecked | Specify whether to use the callback function when master reception ends. |
| Master error | [✓ Checked]<br>Unchecked | Specify whether to use the callback function when a communication error occurs. |
| Generated stop condition in master transmission / reception end callback function | [✓ Checked]<br>Unchecked | Specify whether to generate a stop condition in the callback function.<br>Set to “Unchecked”. |

# 6. Guide for Changing a Target Device

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

Before switching to a new device, import the original sample project for the current device to the workspace.

## 6.1 RA Sample Project

Use the following procedures to modify a sample project.

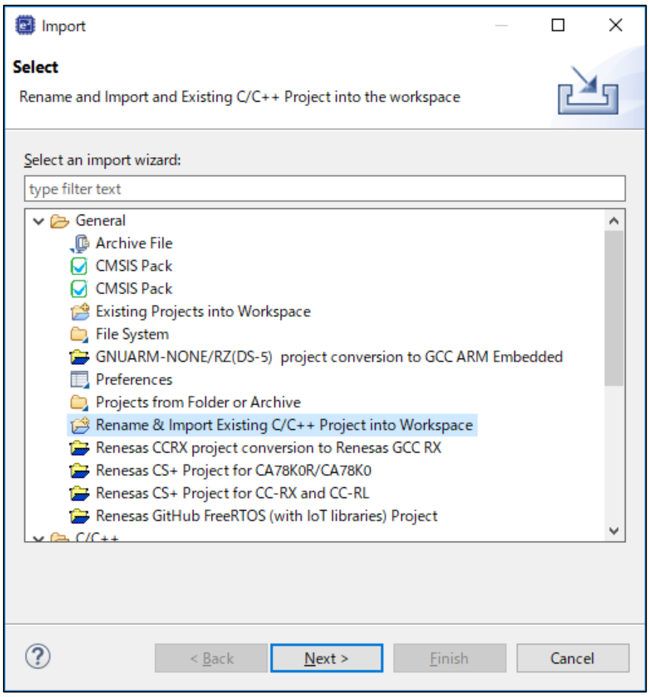
The following explains the change procedure for the following board change example. In addition, an Interposer Board is required when using a Pmod Type 2A/3A connector.

- Sample Project "FS3000_RA2E1_NonOS":
  PMOD1 (Type 2A/3A: SCI0)
  → PMOD1 (Option Type 6A: IIC1) or PMOD2 (Type 2A: SCI0) of the EK-RA6M4 board

### 6.1.1 Importing a Sample Project

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.



Import

**Select**

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

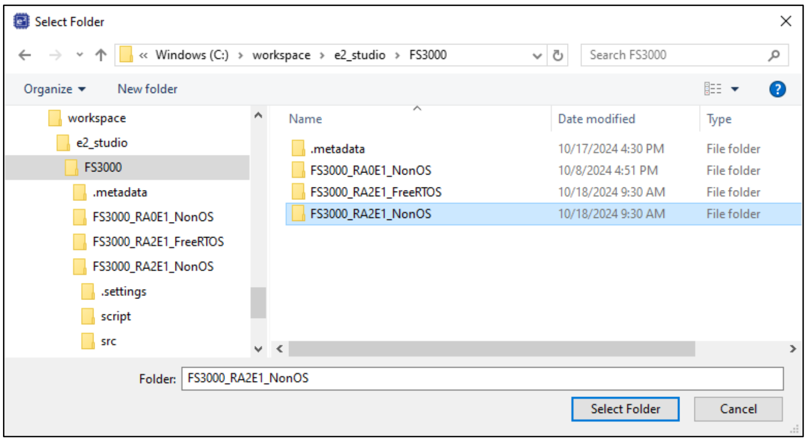
Select an import wizard:

type filter text

- General
  - Archive File
  - CMSIS Pack
  - CMSIS Pack
  - Existing Projects into Workspace
  - File System
  - GNUARM-NONE/RZ(DS-5) project conversion to GCC ARM Embedded
  - Preferences
  - Projects from Folder or Archive
  - Rename & Import Existing C/C++ Project into Workspace
  - Renesas CCRX project conversion to Renesas GCC RX
  - Renesas CS+ Project for CA78K0R/CA78K0
  - Renesas CS+ Project for CC-RX and CC-RL
  - Renesas GitHub FreeRTOS (with IoT libraries) Project
- C/C++

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

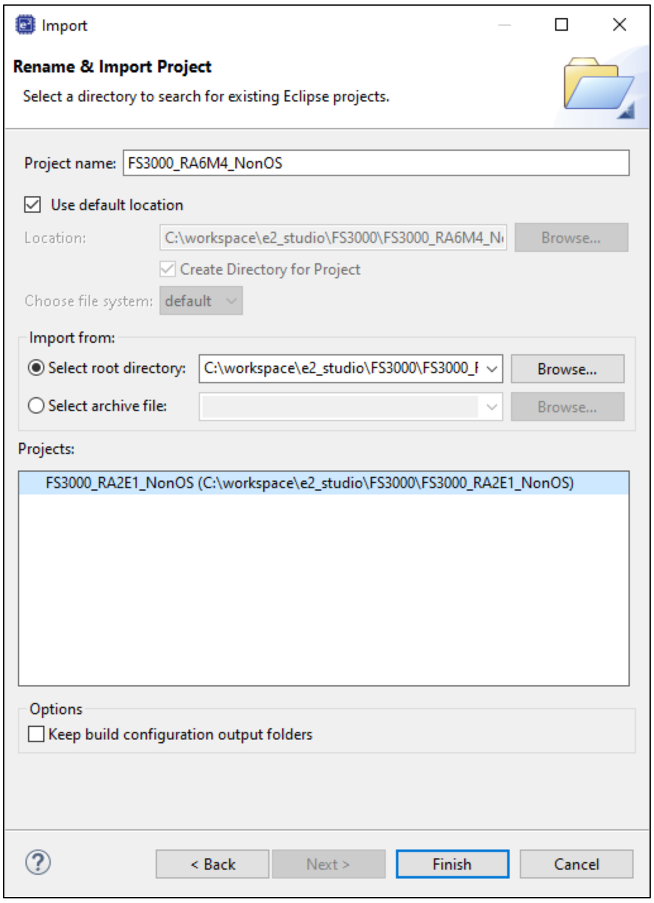
Select the folder of the original project for the current device from a list of imported sample projects and press the [Select Folder] button.



| Name | Date modified | Type |
|---|---|---|
| .metadata | 10/17/2024 4:30 PM | File folder |
| FS3000_RA0E1_NonOS | 10/8/2024 4:51 PM | File folder |
| FS3000_RA2E1_FreeRTOS | 10/18/2024 9:30 AM | File folder |
| FS3000_RA2E1_NonOS | 10/18/2024 9:30 AM | File folder |

Folder: FS3000_RA2E1_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: FS3000_RA6M4_NonOS

☑ Use default location

Location: C:\workspace\e2_studio\FS3000\FS3000_RA6M4_N

☑ Create Directory for Project

Choose file system: default

Import from:

● Select root directory: C:\workspace\e2_studio\FS3000\FS3000_I

○ Select archive file:

Projects:

FS3000_RA2E1_NonOS (C:\workspace\e2_studio\FS3000\FS3000_RA2E1_NonOS)

Options

☐ Keep build configuration output folders

### 6.1.2 Modifying Settings of FSP Configurator

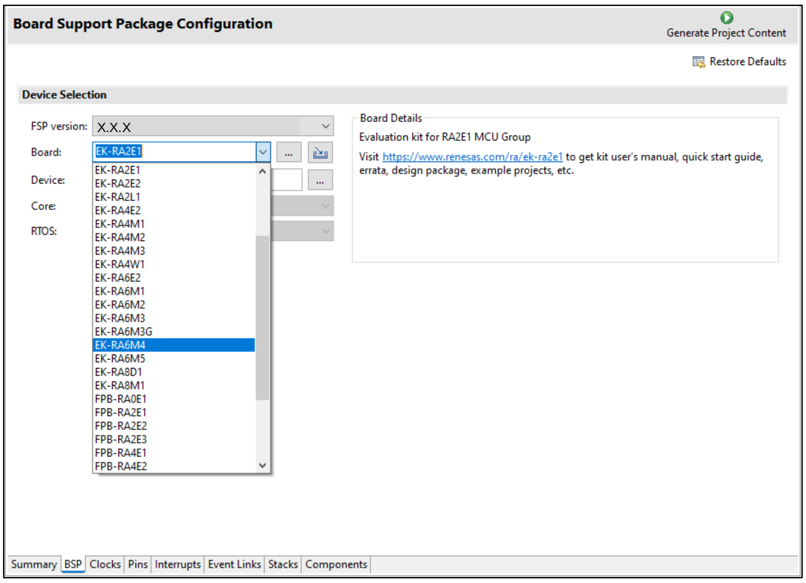
Double-click on "Configuratorn.xml" in the project tree to open the FSP Configurator.

#### (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

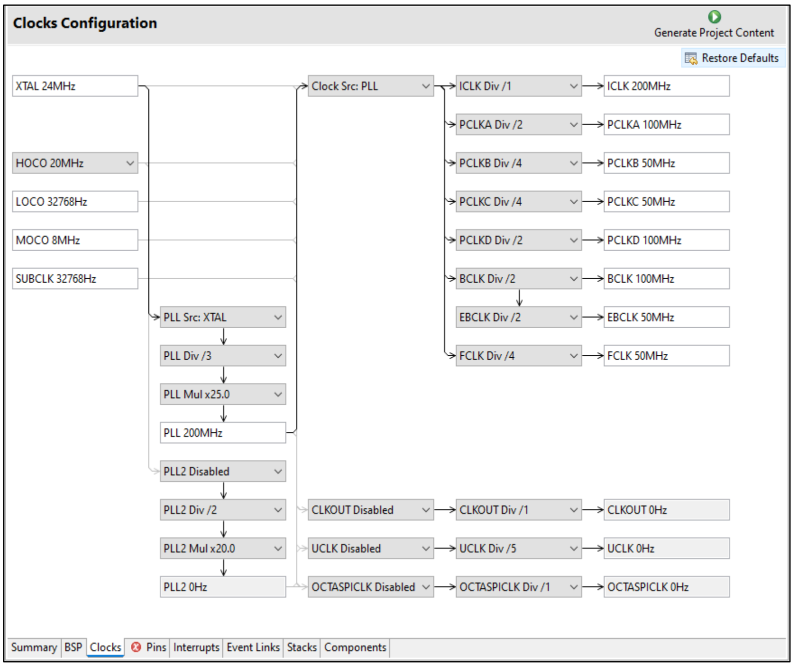
Visit https://www.renesas.com/ra/ek-ra2e1 to get kit user's manual, quick start guide, errata, design package, example projects, etc.

#### (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

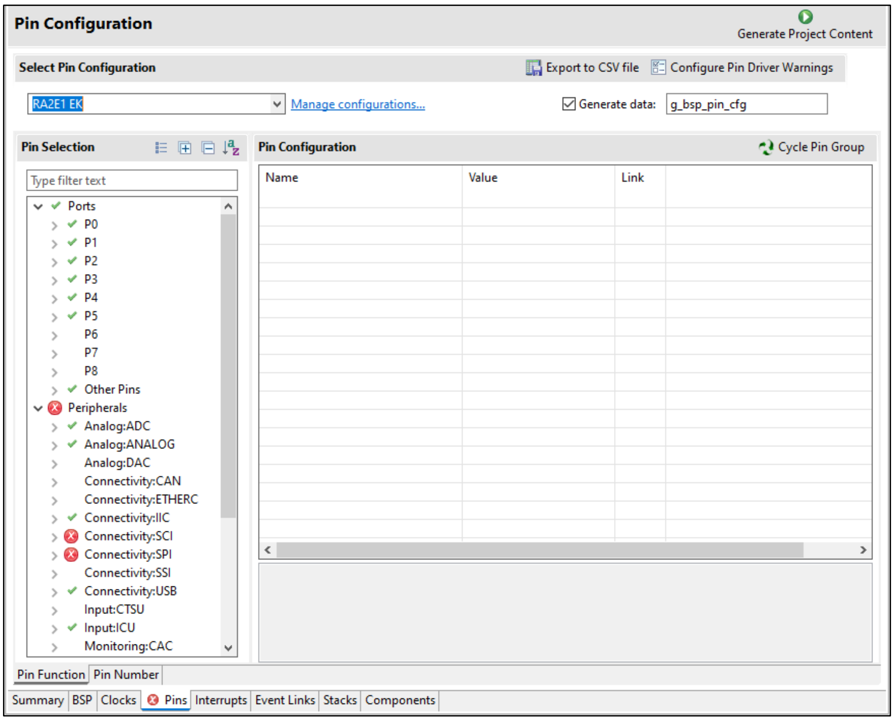
OCTASPICLK Disabled → OCTASPICLK Div /1 → OCTASPICLK 0Hz

### (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.

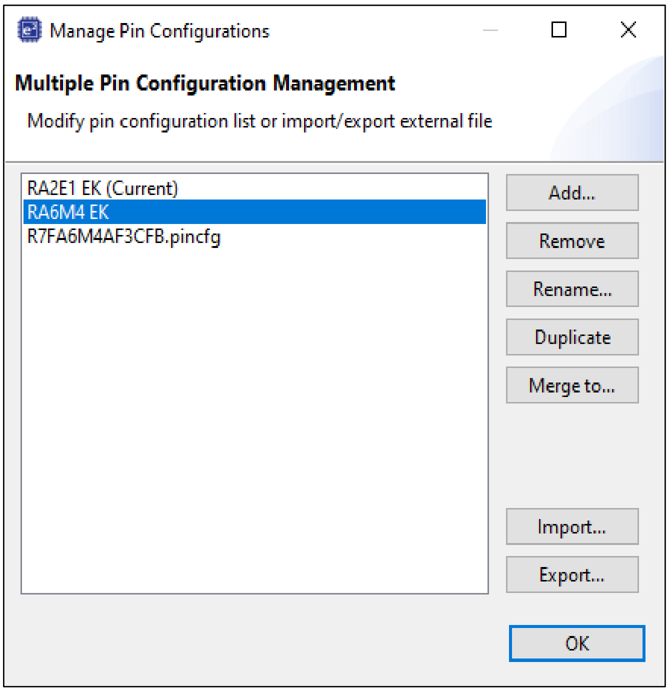
When using a Renesas board, change the selection for "Select Pin Configuration" from ”RA2E1 EK” to the target board; appropriate pins are automatically assigned.



Select Pin Configuration: RA2E1 EK

☑ Generate data: g_bsp_pin_cfg

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 Configuration" window and select the desired board in the window.



RA2E1 EK (Current)
RA6M4 EK
R7FA6M4AF3CFB.pincfg

Manage Pin Configurations
Multiple Pin Configuration Management
Modify pin configuration list or import/export external file
Add...
Remove
Rename...
Duplicate
Merge to...
Import...
Export...
OK

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

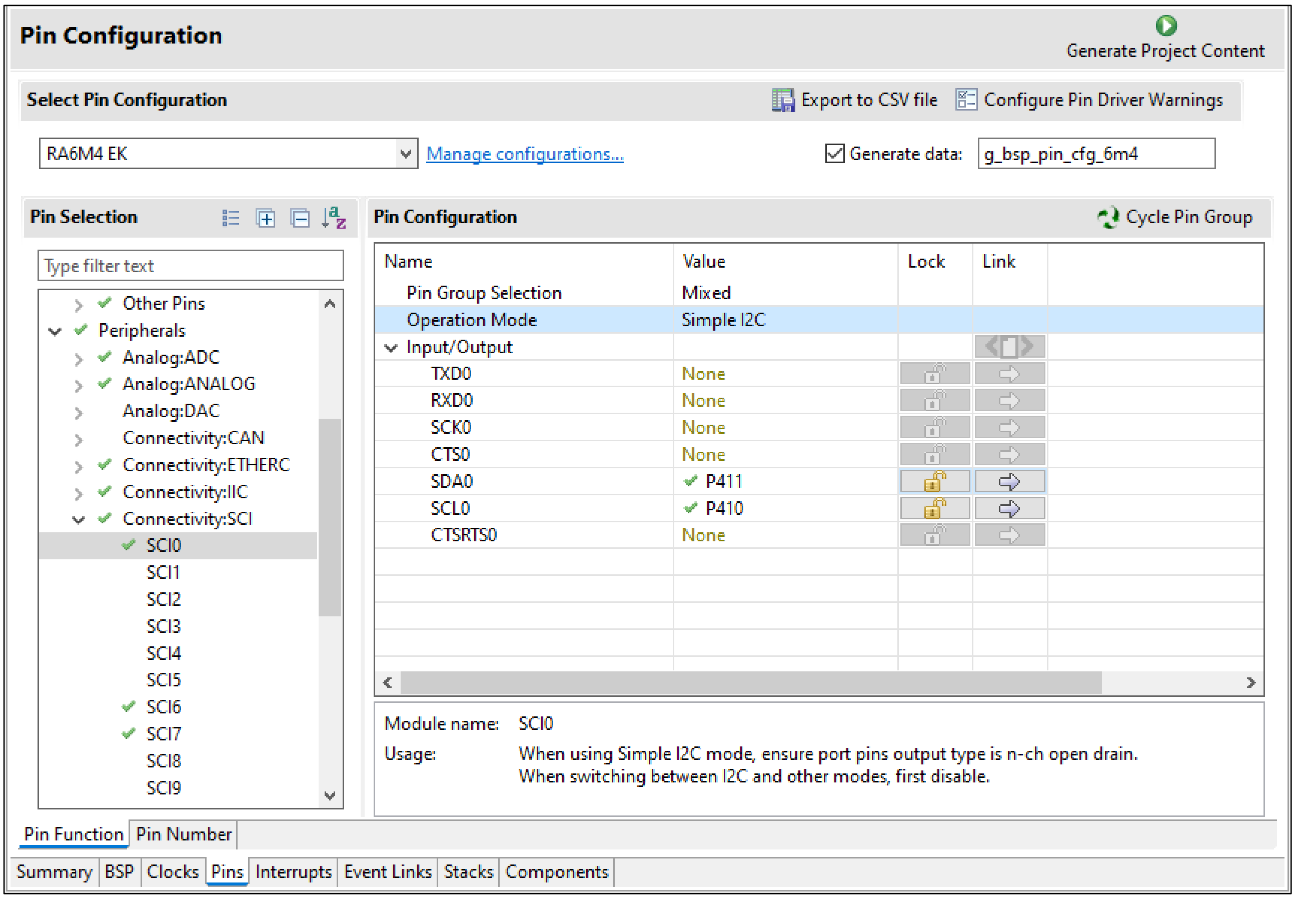
However, the "Select Pin Configuration" assignment will apply the SPI communication pin settings that support Pmod Type 2A on the EK-RA6M4 board.

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

IIC1 is assigned to PMOD1 and SCI0 to PMOD2 on the EK-RA6M4 board.

I2C communication is assigned to P511 and P512 on PMOD1(Option Type 6A), and it is assigned to P410 and P411 on PMOD2.

After automatic assignment of "Select Pin Configuration", reconfigure in "Pin Configuration".



Pin Configuration

Generate Project Content

Select Pin Configuration

Export to CSV file | Configure Pin Driver Warnings

RA6M4 EK | Manage configurations... | ☑ Generate data: g_bsp_pin_cfg_6m4

Pin Selection

Type filter text

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

Pin Configuration

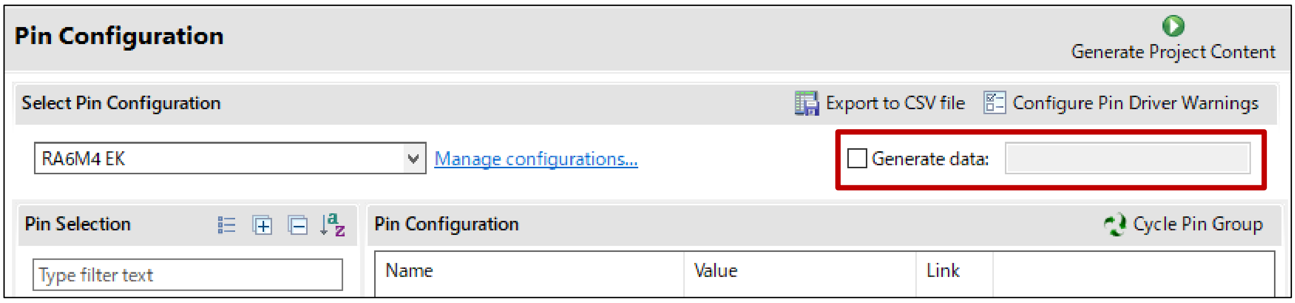
Cycle Pin Group

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

Module name: SCI0

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

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



Pin Configuration

Generate Project Content

Select Pin Configuration

Export to CSV file | Configure Pin Driver Warnings

RA6M4 EK | Manage configurations... | ☐ Generate data:

Pin Selection

Type filter text

Pin Configuration

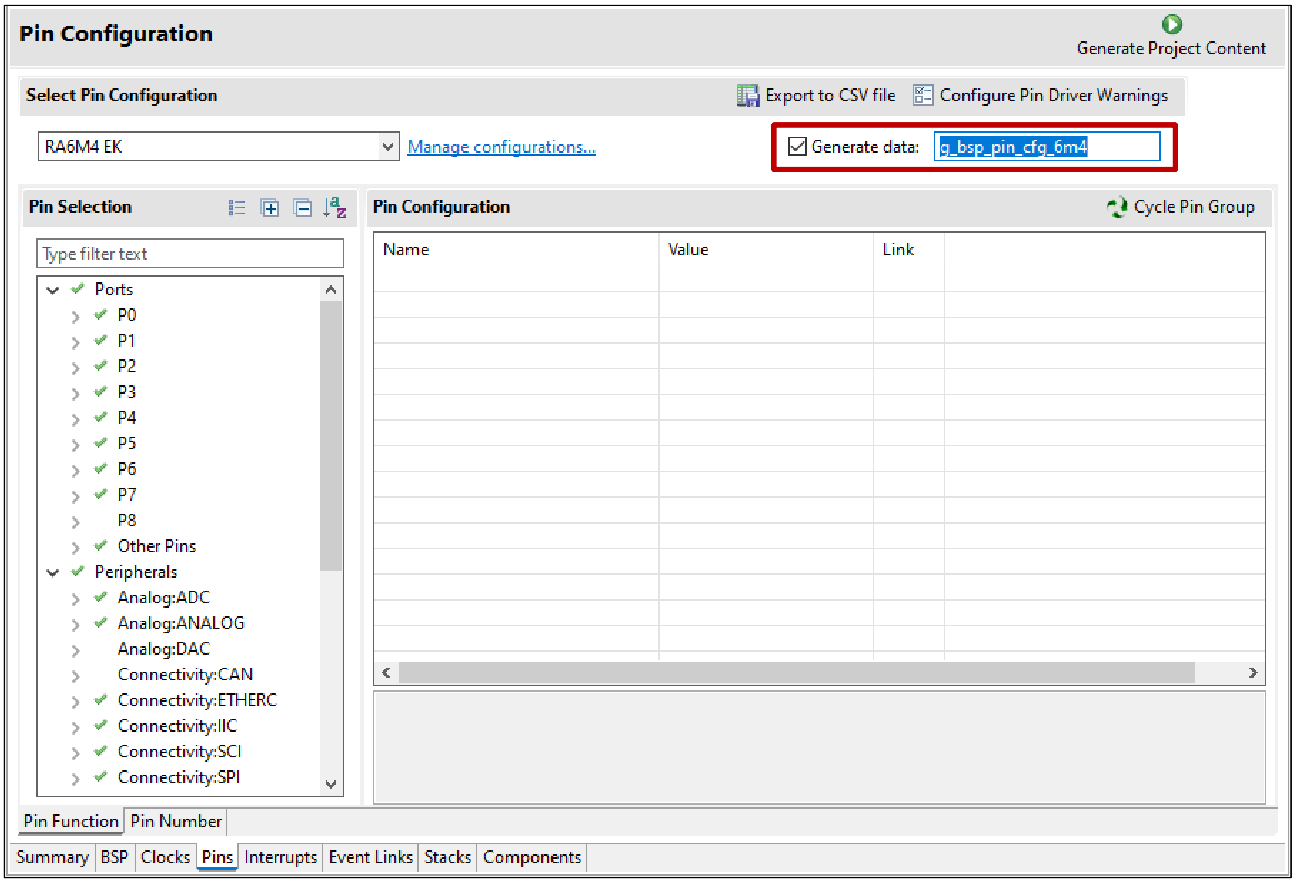
Cycle Pin Group

| Name | Value | Link |
|---|---|---|

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.

In our example, it is “g_bsp_pin_cfg_6m4”.



RA6M4 EK

☑ Generate data: g_bsp_pin_cfg_6m4

### (4) Stacks

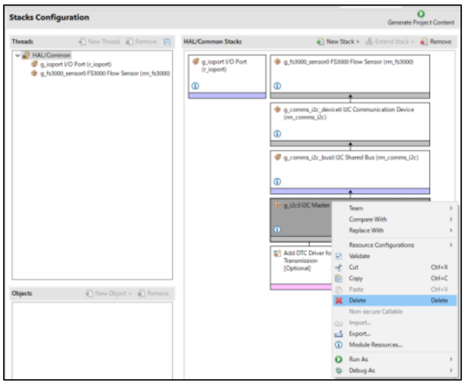
Modify the configuration of individual components in the "Stacks" tabbed page.

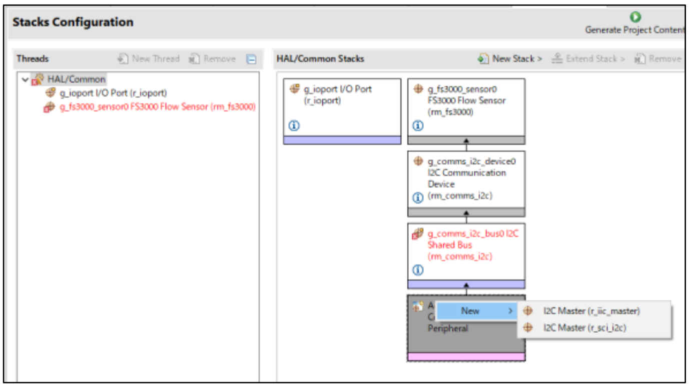
### (a) Changing COMMS_I2C Setting and I2C Driver Settings

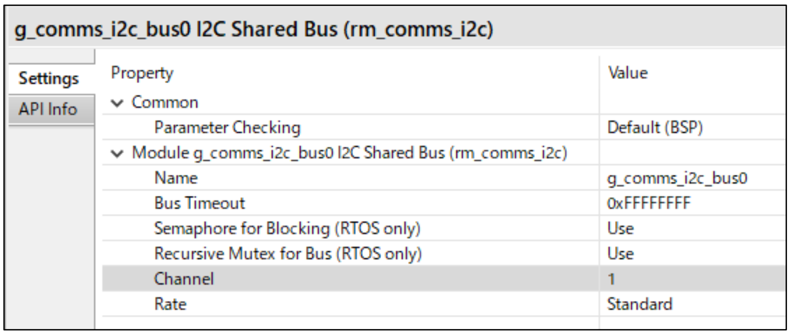
Modify the settings of COMMS_I2C and I2C driver according to the specifications of the target board. To use the pins of the I2C I/F, delete the unnecessary stack and add the new stack to use.

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 |



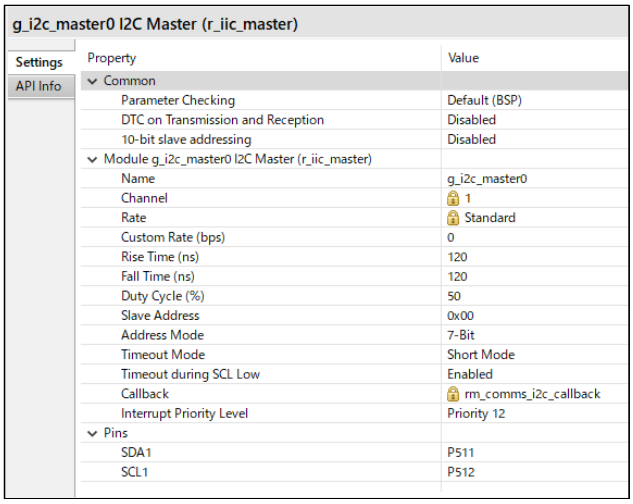

Stacks Configuration
Generate Project Content
Threads
New Thread
Remove
HAL/Common Stacks
New Stack >
Extend Stack >
Remove
HAL/Common
g_ioport I/O Port (r_ioport)
g_fs3000_sensor0 FS3000 Flow Sensor (rm_fs3000)
g_ioport I/O Port (r_ioport)
g_fs3000_sensor0 FS3000 Flow Sensor (rm_fs3000)
g_comms_i2c_device0 I2C Communication Device (rm_comms_i2c)
g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)
Peripheral
New
I2C Master (r_iic_master)
I2C Master (r_sci_i2c)



g_comms_i2c_bus0 I2C Shared Bus (rm_comms_i2c)

| 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 |

When setting IIC1



g_i2c_master0 I2C Master (r_iic_master)

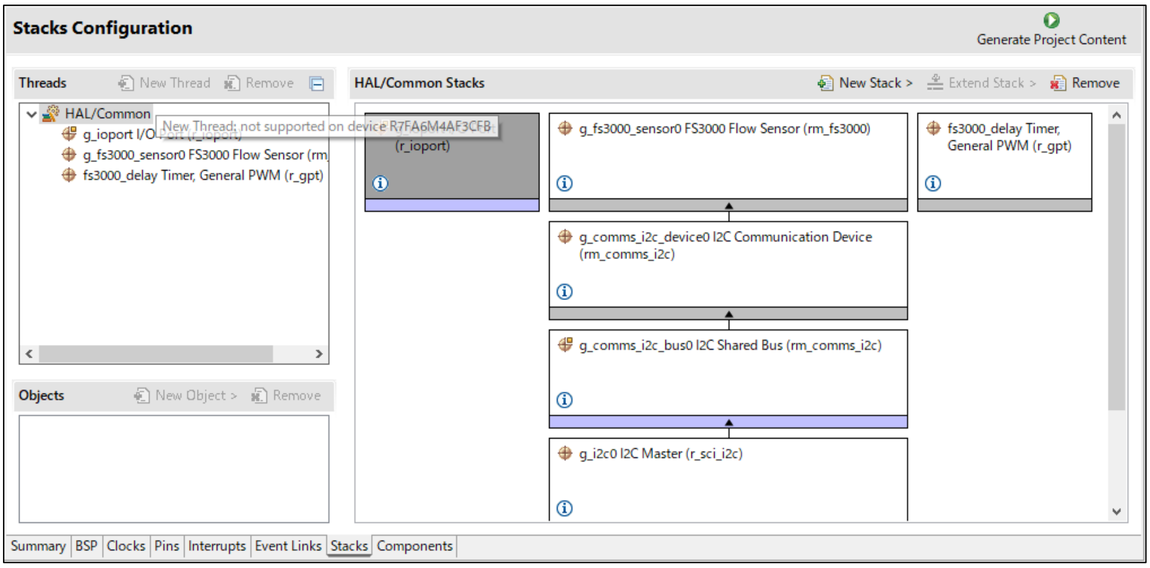
| Property | Value |
|---|---|
| Common | |
| Parameter Checking | Default (BSP) |
| DTC on Transmission and Reception | Disabled |
| 10-bit slave addressing | Disabled |
| Module g_i2c_master0 I2C Master (r_iic_master) | |
| Name | g_i2c_master0 |
| Channel | 1 |
| Rate | Standard |
| Custom Rate (bps) | 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 |

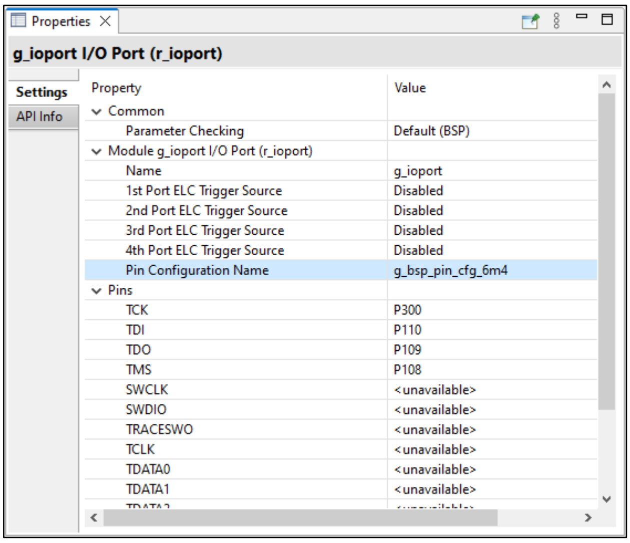
When setting IIC1

### (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".

In our example, it is ”g_bsp_pin_cfg_6m4”.


Stacks Configuration
Generate Project Content
Threads
New Thread
Remove
HAL/Common Stacks
New Stack >
Extend Stack >
Remove
HAL/Common
g_ioport I/O Port (r_ioport)
g_fs3000_sensor0 FS3000 Flow Sensor (rm
fs3000_delay Timer, General PWM (r_gpt)
New Thread: not supported on device R7FA6M4AF3CFB
(r_ioport)
g_fs3000_sensor0 FS3000 Flow Sensor (rm_fs3000)
fs3000_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)
Objects
New Object >
Remove
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> |
| TDATA2 | <unavailable> |

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

### (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.

## 6.1.3 Changing Toolchain Setting

If you want to use a toolchain other than the GCC ARM Embedded toolchain, copy RA_FS3000.c (Non-OS) or fs3000_sensor_thread_entry.c, sensor_thread_common.c, and sensor_thread_common.c (FreeRTOS) from this project to create a new project.

## 6.2 RX Sample Project

Use the following procedures to modify a sample project.

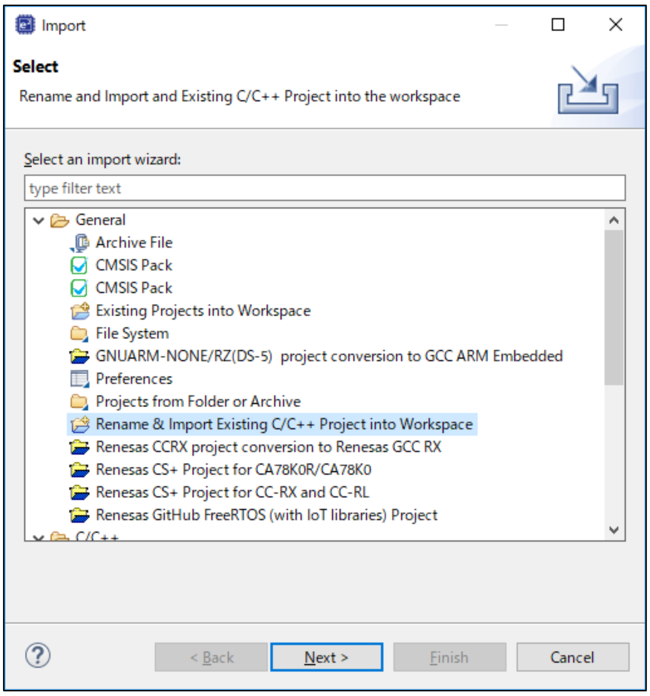
The following explains the change procedure for the following board change example. In addition, an Interposer Board is required when using a Pmod Type 2A/3A connector.

- Sample project "FS3000_RX65N_NonOS":
  PMOD1 (Type2A: SCI2)
  → PMOD1 (Type 2A: SCI8) of the RSK-RX231 board

### 6.2.1 Importing a Sample Project

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.



Import

Select

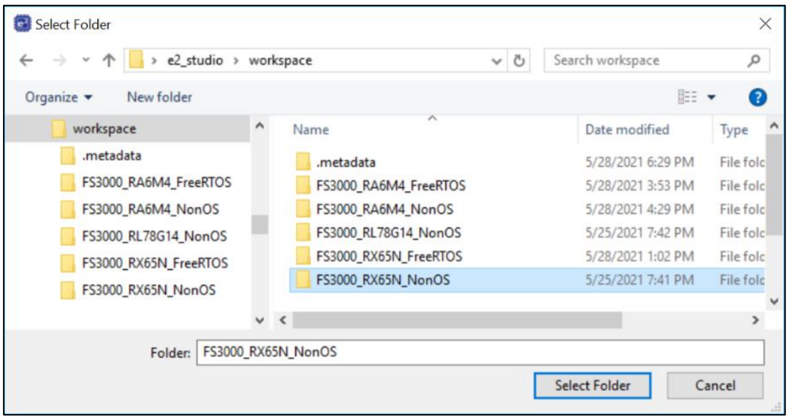
Rename and Import and Existing C/C++ Project into the workspace

Select an import wizard:

type filter text
General
Archive File
CMSIS Pack
CMSIS Pack
Existing Projects into Workspace
File System
GNUARM-NONE/RZ(DS-5) project conversion to GCC ARM Embedded
Preferences
Projects from Folder or Archive
Rename & Import Existing C/C++ Project into Workspace
Renesas CCRX project conversion to Renesas GCC RX
Renesas CS+ Project for CA78K0R/CA78K0
Renesas CS+ Project for CC-RX and CC-RL
Renesas GitHub FreeRTOS (with IoT libraries) Project
C/C++
< 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.

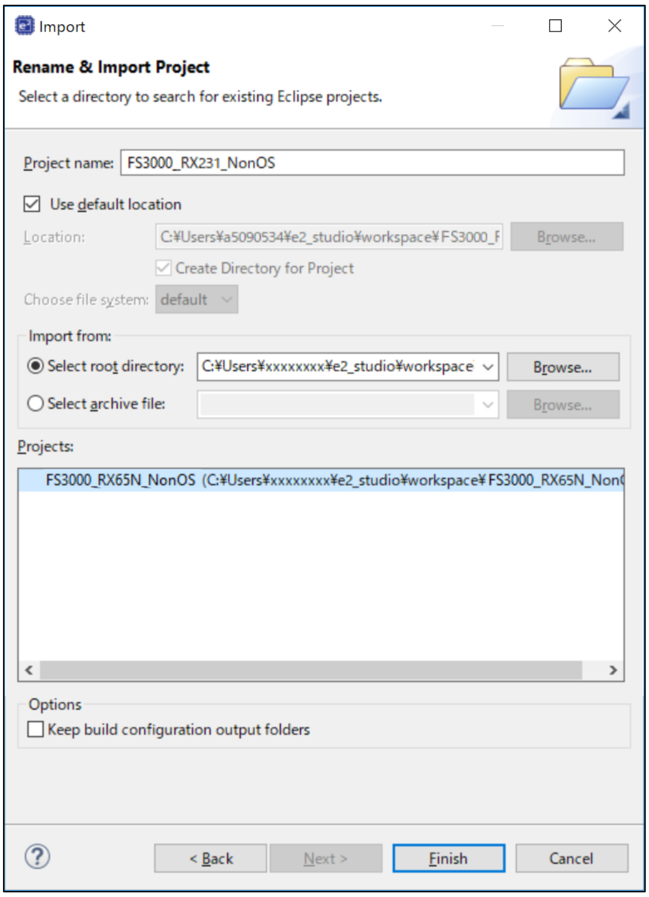


Select Folder

| Name | Date modified | Type |
|---|---|---|
| .metadata | 5/28/2021 6:29 PM | File folc |
| FS3000_RA6M4_FreeRTOS | 5/28/2021 3:53 PM | File folc |
| FS3000_RA6M4_NonOS | 5/28/2021 4:29 PM | File folc |
| FS3000_RL78G14_NonOS | 5/25/2021 7:42 PM | File folc |
| FS3000_RX65N_FreeRTOS | 5/28/2021 1:02 PM | File folc |
| FS3000_RX65N_NonOS | 5/25/2021 7:41 PM | File folc |

Folder: FS3000_RX65N_NonOS

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



Import

**Rename & Import Project**

Select a directory to search for existing Eclipse projects.

Project name: FS3000_RX231_NonOS

☑ Use default location

Location: C:¥Users¥a5090534¥e2_studio¥workspace¥FS3000_F

☑ Create Directory for Project

Choose file system: default

Import from:

● Select root directory: C:¥Users¥xxxxxxxx¥e2_studio¥workspace

○ Select archive file:

Projects:

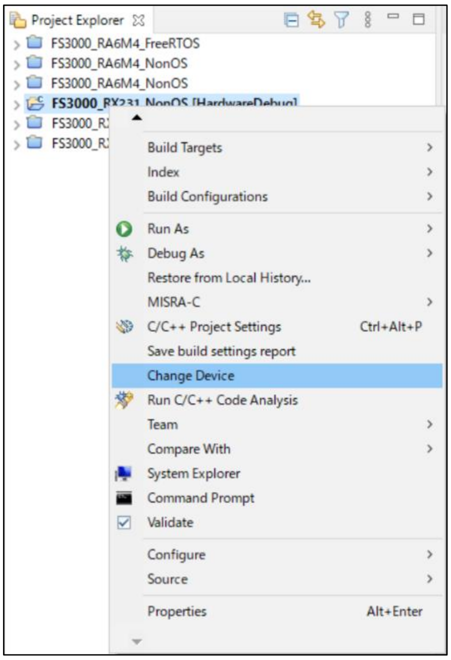
FS3000_RX65N_NonOS (C:¥Users¥xxxxxxxx¥e2_studio¥workspace¥FS3000_RX65N_Non(

Options

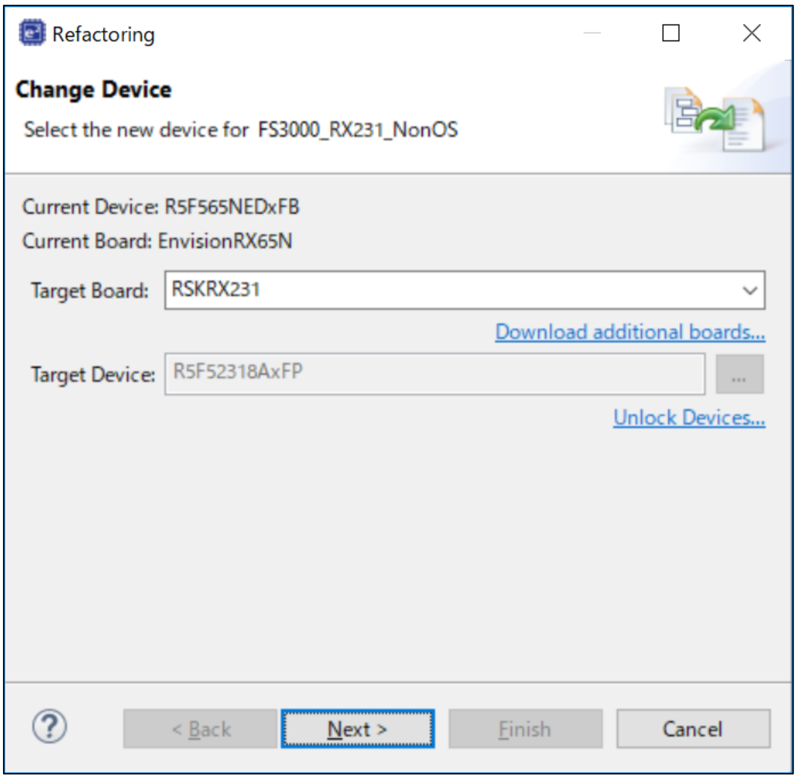
□ Keep build configuration output folders

### 6.2.2 Changing a Device

1. Select the imported project from the project tree and right-click on int to open the context menu. Select "Change Device" from the menu.



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


Refactoring

**Change Device**

Select the new device for FS3000_RX231_NonOS

Current Device: R5F565NEDxFB

Current Board: EnvisionRX65N

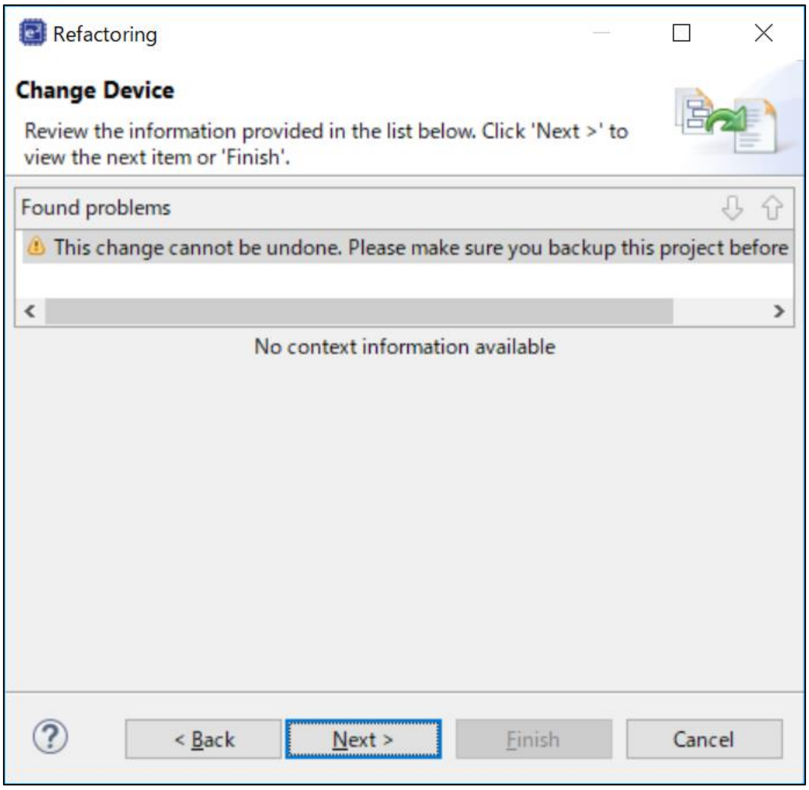
Target Board: RSKRX231

Download additional boards...

Target Device: R5F52318AxFP

Unlock Devices...

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

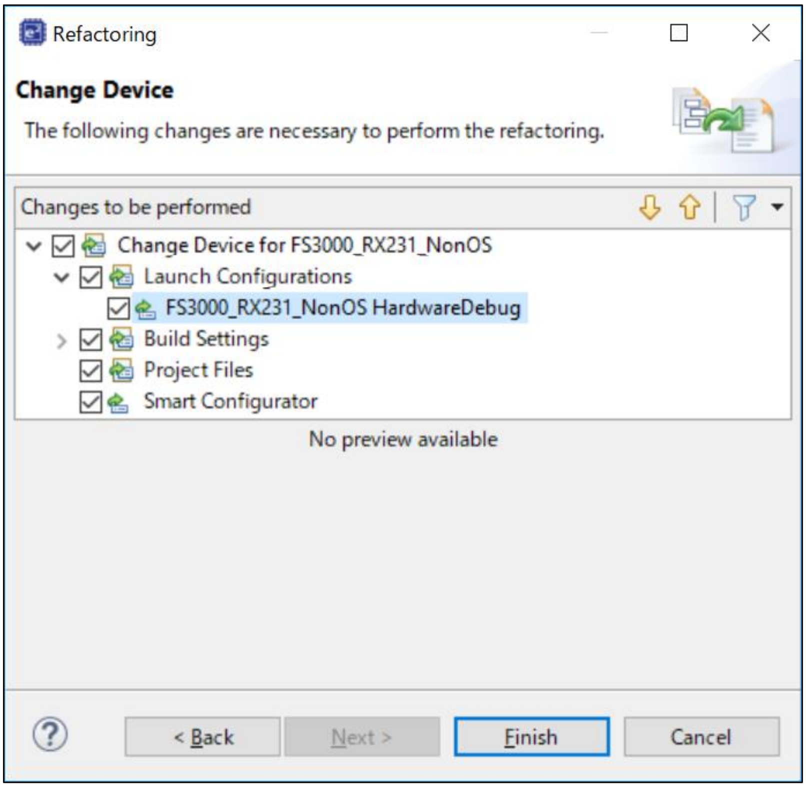
Review the information provided in the list below. Click 'Next >' to view the next item or 'Finish'.

Found problems

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

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 are necessary to perform the refactoring.

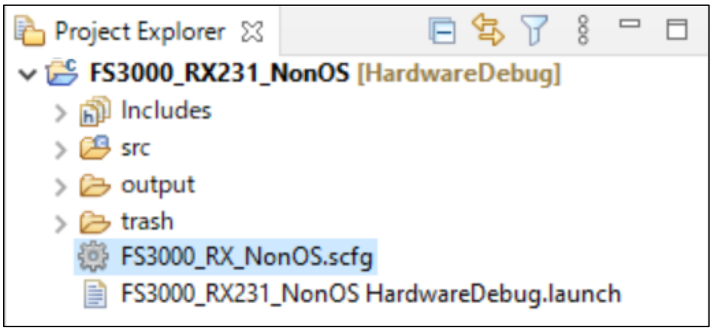
Changes to be performed

- ☑ Change Device for FS3000_RX231_NonOS
  - ☑ Launch Configurations
    - ☑ FS3000_RX231_NonOS HardwareDebug
  - ☑ Build Settings
  - ☑ Project Files
  - ☑ Smart Configurator

No preview available

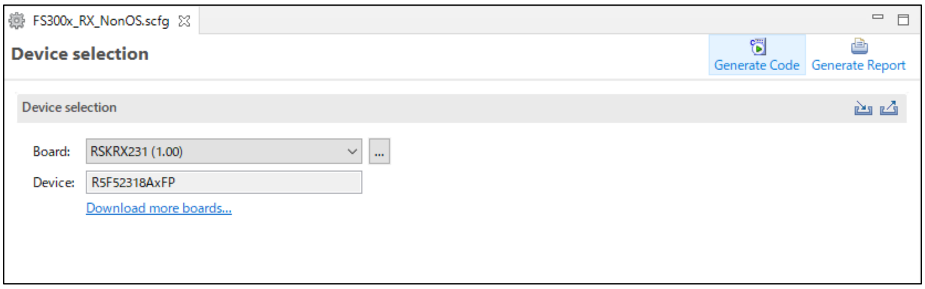
### 6.2.3 Modifying Settings of Smart Configurator

On the project tree, double-click on the .scfg file of the imported project in which the target device has been changed; the Smart Configurator window will open.


Project Explorer
FS3000_RX231_NonOS [HardwareDebug]
Includes
src
output
trash
FS3000_RX_NonOS.scfg
FS3000_RX231_NonOS HardwareDebug.launch

#### (1) Board

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

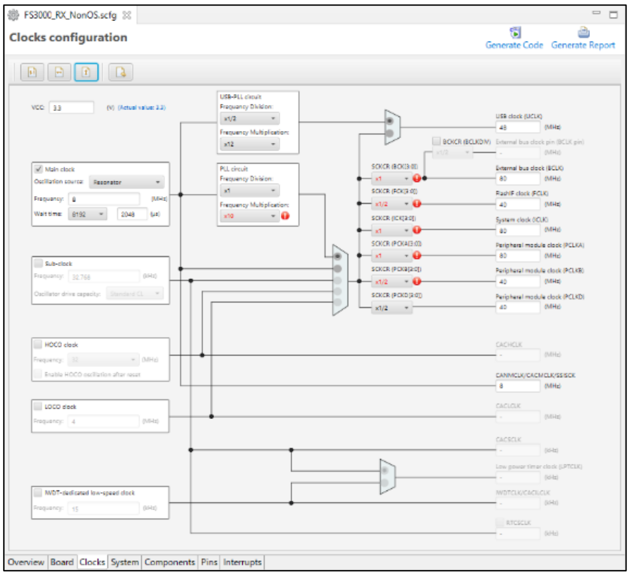


Board: RSKRX231 (1.00)

Device: R5F52318AxFP

#### (2) Clocks

Set up the clocks in the "Clocks" tabbed page according to the specifications of the target board to be used.



VCC: 3.3 (V) (Actual value: 3.3)

USB-PLL circuit: Frequency Division: x1/2; Frequency Multiplication: x12

☑ Main clock: Oscillation source: Resonator; Frequency: 8 (MHz); Wait time: 8192, 2048 (µs)

PLL circuit: Frequency Division: x1; Frequency Multiplication: x10

□ Sub-clock: Frequency: 32.768 (kHz); Oscillator drive capacity: Standard CL

□ HOCO clock: Frequency: 32 (MHz); □ Enable HOCO oscillation after reset

□ LOCO clock: Frequency: 4 (MHz)

□ IWDT-dedicated low-speed clock: Frequency: 15 (kHz)

□ BCKCR (BCLKDIV): x1/2; External bus clock pin (BCLK pin): - (MHz)

SCKCR (BCK[3:0]): x1; External bus clock (BCLK): 80 (MHz)

SCKCR (FCK[3:0]): x1/2; FlashIF clock (FCLK): 40 (MHz)

SCKCR (ICK[3:0]): x1; System clock (ICLK): 80 (MHz)

SCKCR (PCKA[3:0]): x1; Peripheral module clock (PCLKA): 80 (MHz)

SCKCR (PCKB[3:0]): x1/2; Peripheral module clock (PCLKB): 40 (MHz)

SCKCR (PCKD[3:0]): x1/2; Peripheral module clock (PCLKD): 40 (MHz)

USB clock (UCLK): 48 (MHz)

CACHCLK: - (MHz)

CANMCLK/CACMCLK/SSISCK: 8 (MHz)

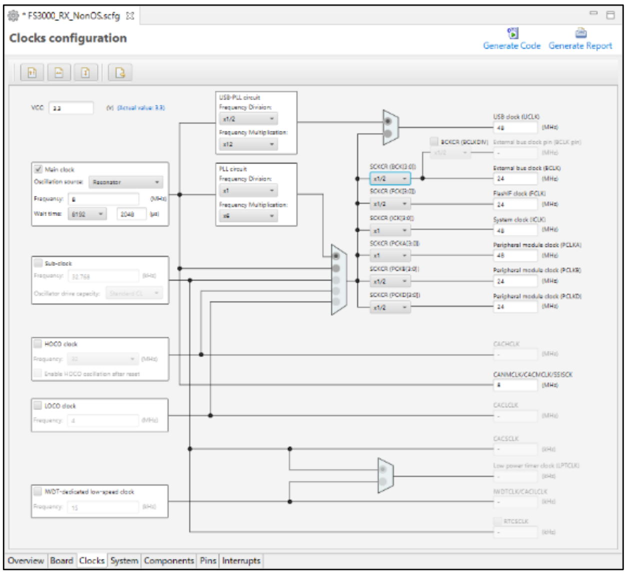
CACLCLK: - (MHz)

CACSCLK: - (kHz)

Low power timer clock (LPTCLK): - (kHz)

IWDTCLK/CACILCLK: - (kHz)

□ RTCSCLK: - (kHz)



VCC: 3.3 (V) (Actual value: 3.3)

USB-PLL circuit: Frequency Division: x1/2; Frequency Multiplication: x12

☑ Main clock: Oscillation source: Resonator; Frequency: 8 (MHz); Wait time: 8192, 2048 (µs)

PLL circuit: Frequency Division: x1; Frequency Multiplication: x6

□ Sub-clock: Frequency: 32.768 (kHz); Oscillator drive capacity: Standard CL

□ HOCO clock: Frequency: 32 (MHz); □ Enable HOCO oscillation after reset

□ LOCO clock: Frequency: 4 (MHz)

□ IWDT-dedicated low-speed clock: Frequency: 15 (kHz)

□ BCKCR (BCLKDIV): x1/2; External bus clock pin (BCLK pin): - (MHz)

SCKCR (BCK[3:0]): x1/2; External bus clock (BCLK): 24 (MHz)

SCKCR (FCK[3:0]): x1/2; FlashIF clock (FCLK): 24 (MHz)

SCKCR (ICK[3:0]): x1; System clock (ICLK): 48 (MHz)

SCKCR (PCKA[3:0]): x1; Peripheral module clock (PCLKA): 48 (MHz)

SCKCR (PCKB[3:0]): x1/2; Peripheral module clock (PCLKB): 24 (MHz)

SCKCR (PCKD[3:0]): x1/2; Peripheral module clock (PCLKD): 24 (MHz)

USB clock (UCLK): 48 (MHz)

CACHCLK: - (MHz)

CANMCLK/CACMCLK/SSISCK: 8 (MHz)

CACLCLK: - (MHz)

CACSCLK: - (kHz)

Low power timer clock (LPTCLK): - (kHz)

IWDTCLK/CACILCLK: - (kHz)

□ RTCSCLK: - (kHz)

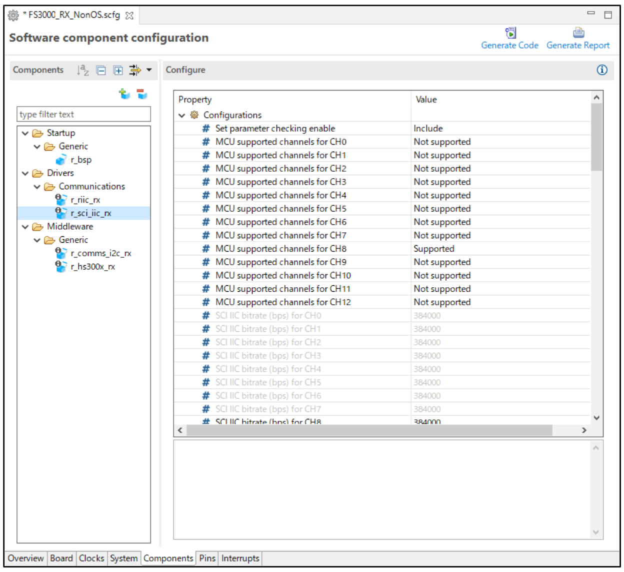
### (3) Components

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

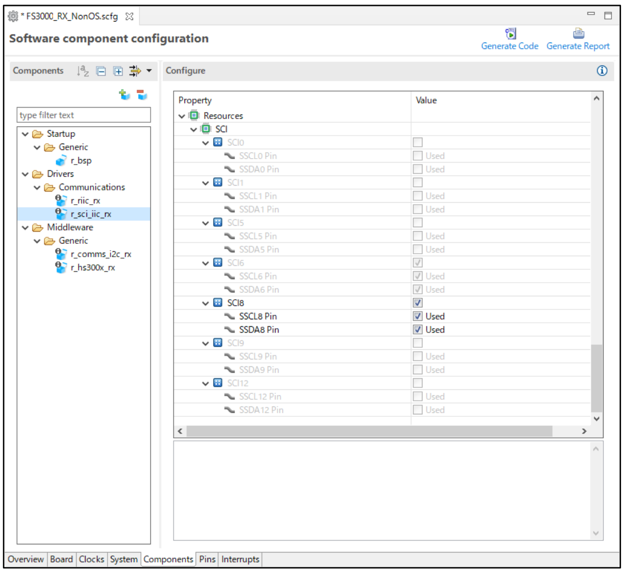
#### (a) Changing I2C Driver Settings

As SCI8 is assigned to Pmod on the RSK-RX231 board, change the setting of "MCU supported channels for CH2" to "Not supported" and "MCU supported channels for CH8" to "Supported" in r_sci_iic_rx.

Check the settings of "SSCL8 Pin" and "SSDA8 Pin" for "SCI8" 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 | Supported |
| MCU supported channels for CH9 | Not supported |
| MCU supported channels for CH10 | Not supported |
| MCU supported channels for CH11 | Not 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 |
| SCI IIC bitrate (bps) for CH4 | 384000 |
| SCI IIC bitrate (bps) for CH5 | 384000 |
| SCI IIC bitrate (bps) for CH6 | 384000 |
| SCI IIC bitrate (bps) for CH7 | 384000 |
| SCI IIC bitrate (bps) for CH8 | 384000 |



| Property | Value |
|---|---|
| Resources | |
| SCI | |
| SCI0 | □ |
| SSCL0 Pin | □ Used |
| SSDA0 Pin | □ Used |
| SCI1 | □ |
| SSCL1 Pin | □ Used |
| SSDA1 Pin | □ Used |
| SCI5 | □ |
| SSCL5 Pin | □ Used |
| SSDA5 Pin | □ Used |
| SCI6 | ☑ |
| SSCL6 Pin | ☑ Used |
| SSDA6 Pin | ☑ Used |
| SCI8 | ☑ |
| SSCL8 Pin | ☑ Used |
| SSDA8 Pin | ☑ Used |
| SCI9 | □ |
| SSCL9 Pin | □ Used |
| SSDA9 Pin | □ Used |
| SCI12 | □ |
| SSCL12 Pin | □ Used |
| SSDA12 Pin | □ Used |

#### (b) Changing COMMS_I2C Settings

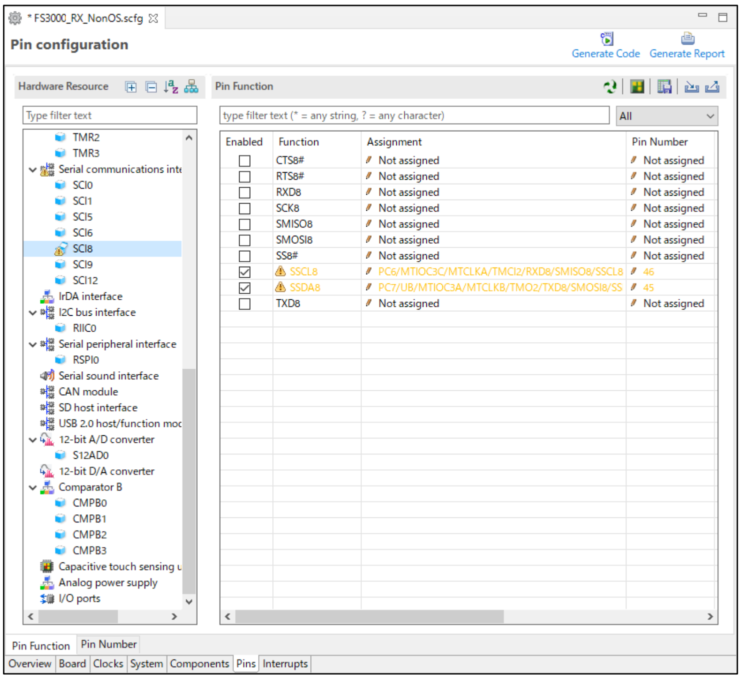
If you have changed the I2C driver or channel, you will need to change this settings.

When used SCI8, change the settings of “Channel No. for I2C Shared Bus0” to “8” in r_comms_i2c_rx.

### (4) Pins

### (a) Changing I2C I/F Pins

Open the "Pins" tabbed page and check that functions are assigned to the SCI8 pins in the "Pin function" panel.



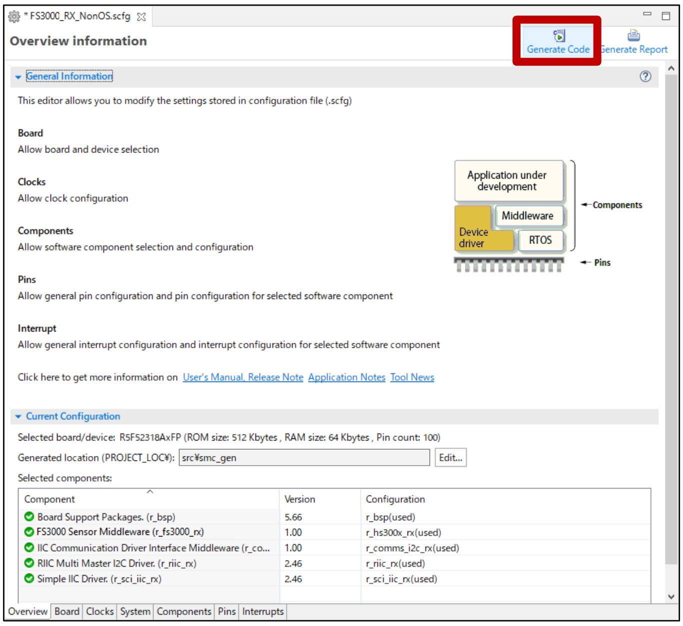
| Enabled | Function | Assignment | Pin Number |
|---|---|---|---|
| □ | CTS8# | Not assigned | Not assigned |
| □ | RTS8# | Not assigned | Not assigned |
| □ | RXD8 | Not assigned | Not assigned |
| □ | SCK8 | Not assigned | Not assigned |
| □ | SMISO8 | Not assigned | Not assigned |
| □ | SMOSI8 | Not assigned | Not assigned |
| □ | SS8# | Not assigned | Not assigned |
| ☑ | ⚠ SSCL8 | PC6/MTIOC3C/MTCLKA/TMCI2/RXD8/SMISO8/SSCL8 | 46 |
| ☑ | ⚠ SSDA8 | PC7/UB/MTIOC3A/MTCLKB/TMO2/TXD8/SMOSI8/SS | 45 |
| □ | TXD8 | Not assigned | Not assigned |

As the use of Pmod Type 2A is specified in the RSK-RX231 board information, a warning message will appear when I2C is used, but this does not produce any problems.

To connect the sensor board, an interposer board that converts Pmod Type 2A to Pmod Type 6A is required.

### (5) Code Generation and Build

Press the [Generate Code] icon to generate code.



Overview information

General Information

This editor allows you to modify the settings stored in configuration file (.scfg)

**Board**
Allow board and device selection

**Clocks**
Allow clock configuration

**Components**
Allow software component selection and configuration

**Pins**
Allow general pin configuration and pin configuration for selected software component

**Interrupt**
Allow general interrupt configuration and interrupt configuration for selected software component

Click here to get more information on User's Manual, Release Note Application Notes Tool News

Current Configuration

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) | 5.66 | r_bsp(used) |
| ✓ FS3000 Sensor Middleware (r_fs3000_rx) | 1.00 | r_hs300x_rx(used) |
| ✓ IIC Communication Driver Interface Middleware (r_co... | 1.00 | r_comms_i2c_rx(used) |
| ✓ RIIC Multi Master I2C Driver. (r_riic_rx) | 2.46 | r_riic_rx(used) |
| ✓ Simple IIC Driver. (r_sci_iic_rx) | 2.46 | 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.

## 6.2.4 Changing Toolchain Setting

If you want to use a toolchain other than the CC-RX toolchain, copy main.c and RX_FS3000.c (Non-OS), or main.c and fs3000_sensor_thread_entry.c (FreeRTOS) from this project to create a new project.

## 6.3 RL78 Sample Project

Use the following procedures to modify a sample project.

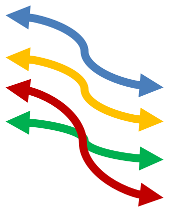
The following explains the change procedure for the following board change example.

- Sample project "FS3000_RL78G23_NonOS":
  PMOD2 (Type 6A: IICA1)
  → Grove (IICA0) of the 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<br>Fast Prototyping Board (Grove) | | |
|---|---|---|
| **Pin** | **Function** | **IICA I/F,<br>Power supply** |
| #1 | SCL | SCLA0 |
| #2 | SDA | SDAA0 |
| #3 | VCC | 3.3V |
| #4 | GND | GND |

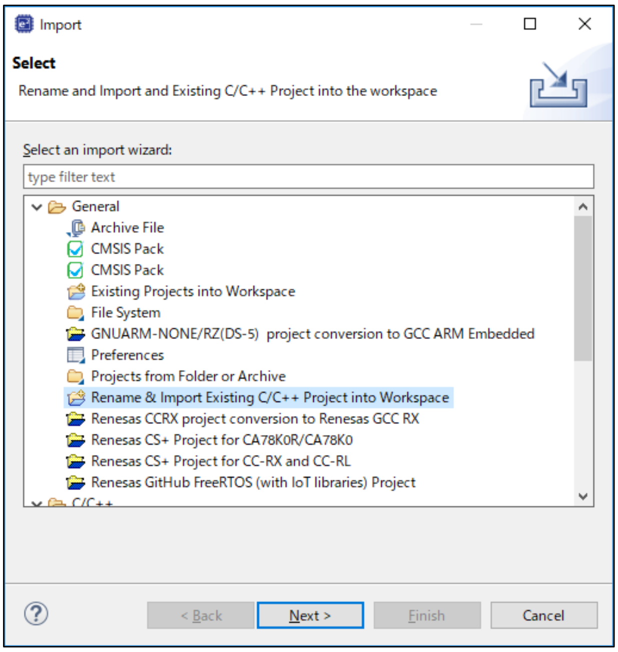


| Renesas Pmod Type 6A<br>Sensor Board | |
|---|---|
| **Pin** | **Function** |
| #1 | IRQ |
| #2 | RESET |
| #3 | IIC_SCL |
| #4 | IIC_SDA |
| #5 | GND |
| #6 | VCC |
| #7 | BUSY# |
| #8 | ENABLE |
| #9 | POWER_ON |
| #10 | GPIO |

### 6.3.1 Creating a New Project

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.



Import

Select

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

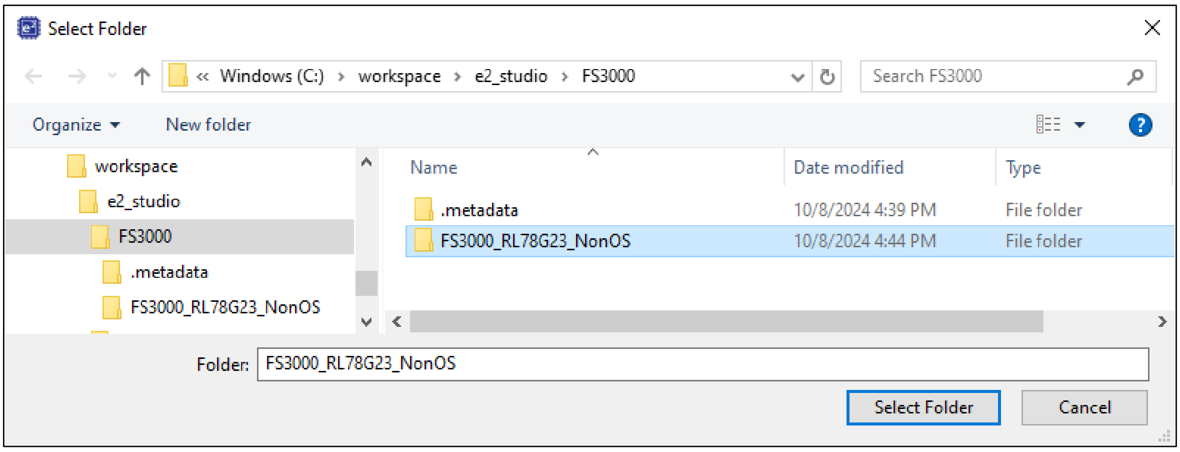
Select an import wizard:

type filter text

- General
  - Archive File
  - CMSIS Pack
  - CMSIS Pack
  - Existing Projects into Workspace
  - File System
  - GNUARM-NONE/RZ(DS-5) project conversion to GCC ARM Embedded
  - Preferences
  - Projects from Folder or Archive
  - Rename & Import Existing C/C++ Project into Workspace
  - Renesas CCRX project conversion to Renesas GCC RX
  - Renesas CS+ Project for CA78K0R/CA78K0
  - Renesas CS+ Project for CC-RX and CC-RL
  - Renesas GitHub FreeRTOS (with IoT libraries) Project
- C/C++

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.



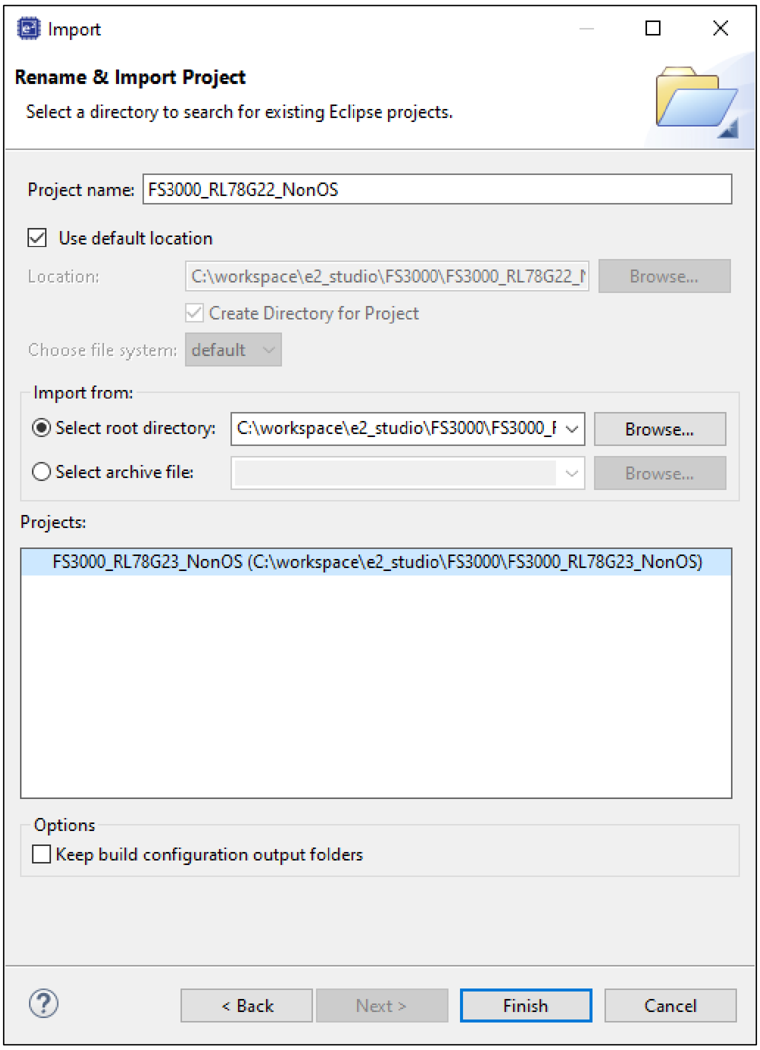
Select Folder

Windows (C:) › workspace › e2_studio › FS3000

| Name | Date modified | Type |
|---|---|---|
| .metadata | 10/8/2024 4:39 PM | File folder |
| FS3000_RL78G23_NonOS | 10/8/2024 4:44 PM | File folder |

Folder: FS3000_RL78G23_NonOS

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



Import

Rename & Import Project

Select a directory to search for existing Eclipse projects.

Project name: FS3000_RL78G22_NonOS

☑ Use default location

Location: C:\workspace\e2_studio\FS3000\FS3000_RL78G22_N

☑ Create Directory for Project

Choose file system: default

Import from:

● Select root directory: C:\workspace\e2_studio\FS3000\FS3000_F

○ Select archive file:

Projects:

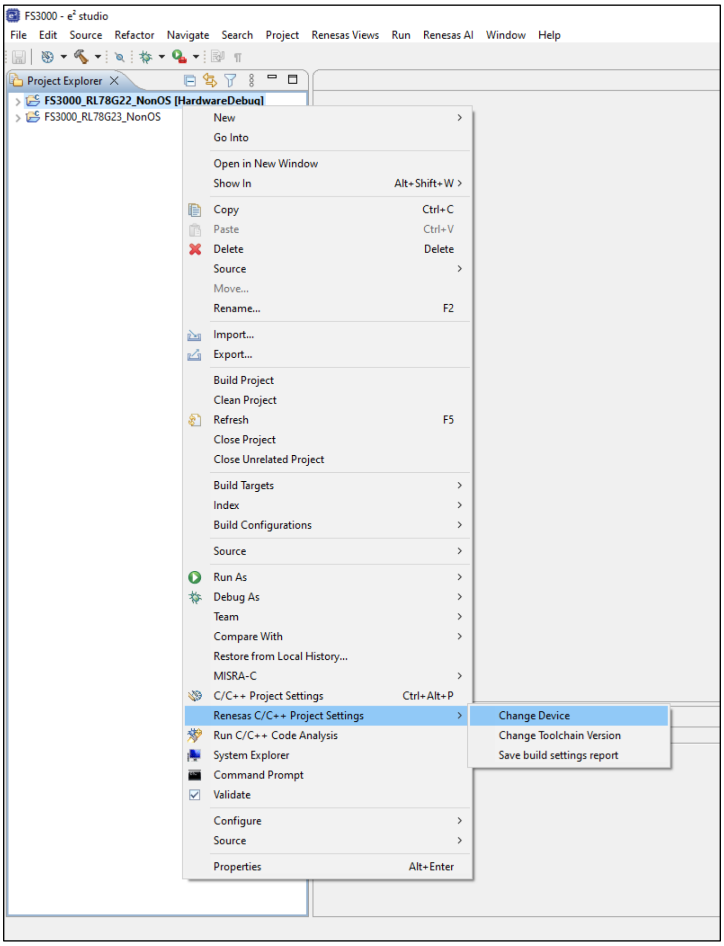
FS3000_RL78G23_NonOS (C:\workspace\e2_studio\FS3000\FS3000_RL78G23_NonOS)

Options

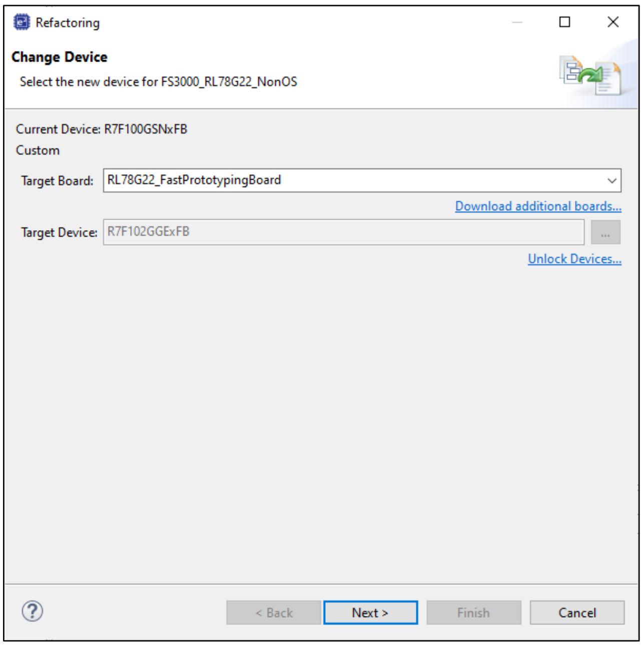
□ Keep build configuration output folders

### 6.3.2 Changing Device

1. Select the imported project from the project tree and right-click on int to open the context menu. Select "Change Device" from the menu.



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


Refactoring

Change Device

Select the new device for FS3000_RL78G22_NonOS

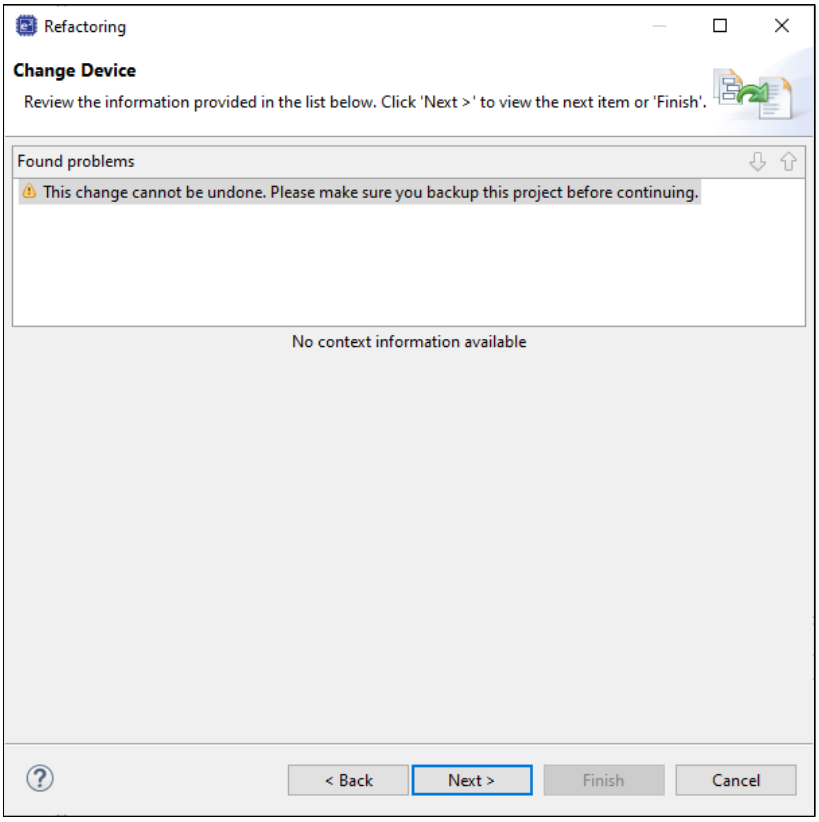
Current Device: R7F100GSNxFB

Custom

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**

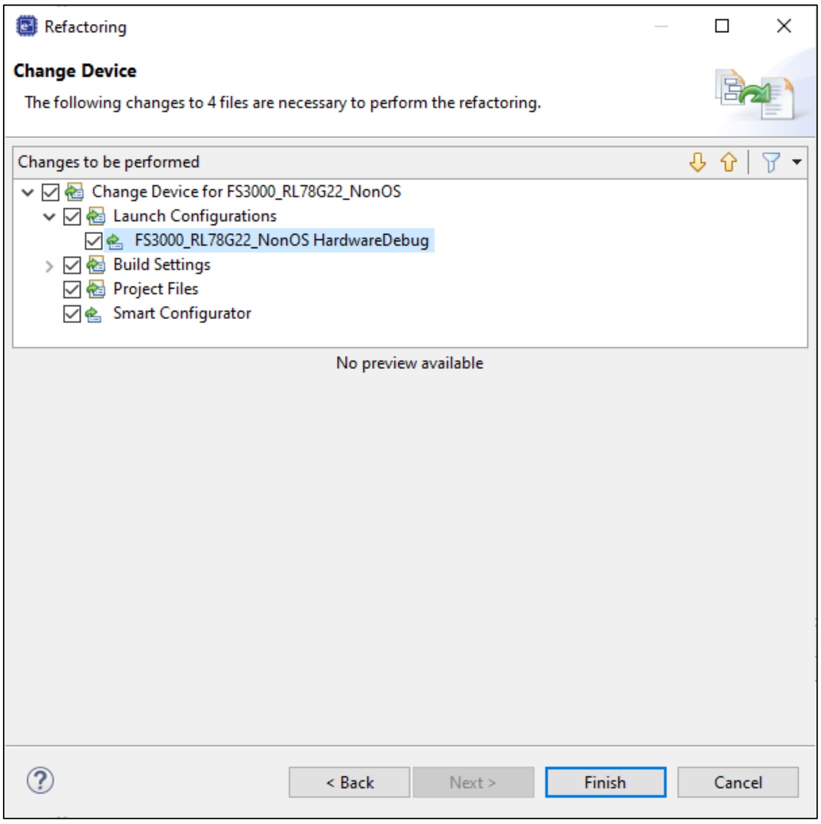
Review the information provided in the list below. Click 'Next >' to view the next item or 'Finish'.

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.

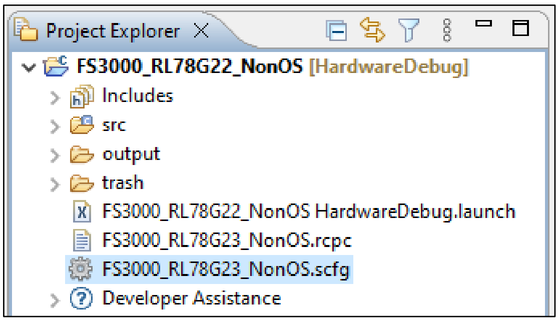
Changes to be performed

- ☑ Change Device for FS3000_RL78G22_NonOS
  - ☑ Launch Configurations
    - ☑ FS3000_RL78G22_NonOS HardwareDebug
  - ☑ Build Settings
  - ☑ Project Files
  - ☑ Smart Configurator

No preview available

### 6.3.3 Modifying Settings of Smart Configurator

On the project tree, double-click on the .scfg file of the imported project in which the target device has been changed; the Smart Configurator window will open.



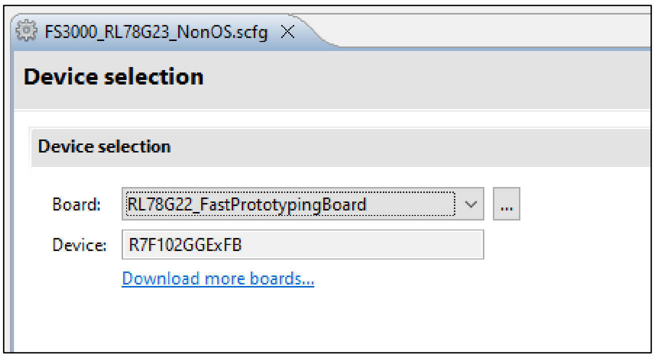
Project Explorer

- FS3000_RL78G22_NonOS [HardwareDebug]
  - Includes
  - src
  - output
  - trash
  - FS3000_RL78G22_NonOS HardwareDebug.launch
  - FS3000_RL78G23_NonOS.rcpc
  - FS3000_RL78G23_NonOS.scfg
  - Developer Assistance

#### (1) Board

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

When the board and device has not been changed, click the [...] button and change the device again in the "Change Device" window.



FS3000_RL78G23_NonOS.scfg

Device selection

Device selection

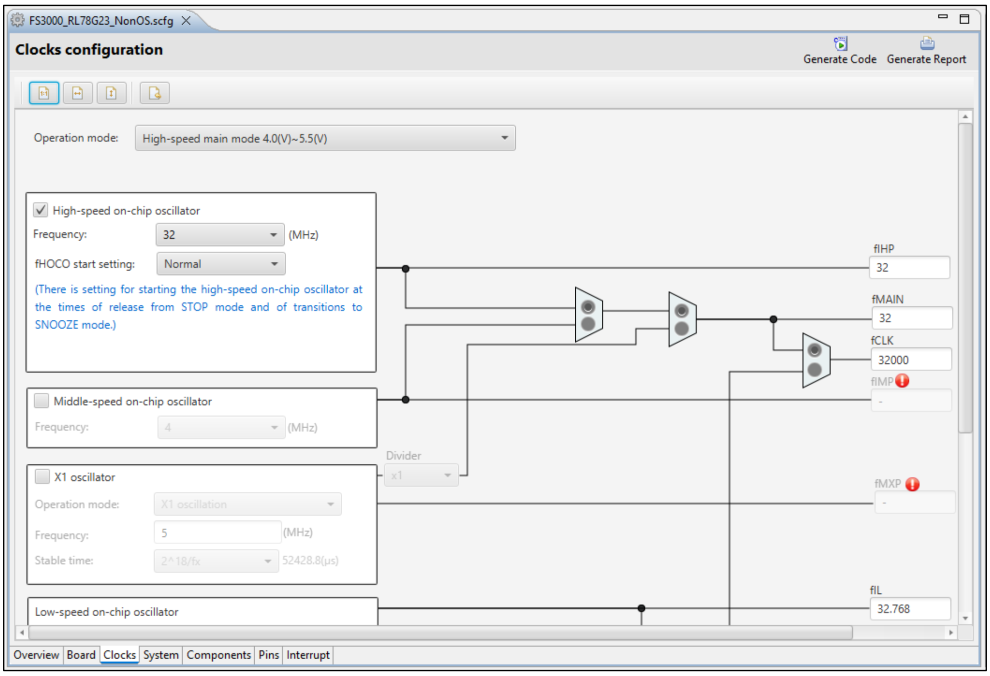
Board: RL78G22_FastPrototypingBoard ...

Device: R7F102GGExFB

Download more boards...

#### (2) Clocks

Set up the clocks in the "Clocks" tabbed page according to the specifications of the target board to be used.



FS3000_RL78G23_NonOS.scfg

Clocks configuration

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

Divider: x1

fIHP: 32

fMAIN: 32

fCLK: 32000

fIMP: -

fMXP: -

fIL: 32.768

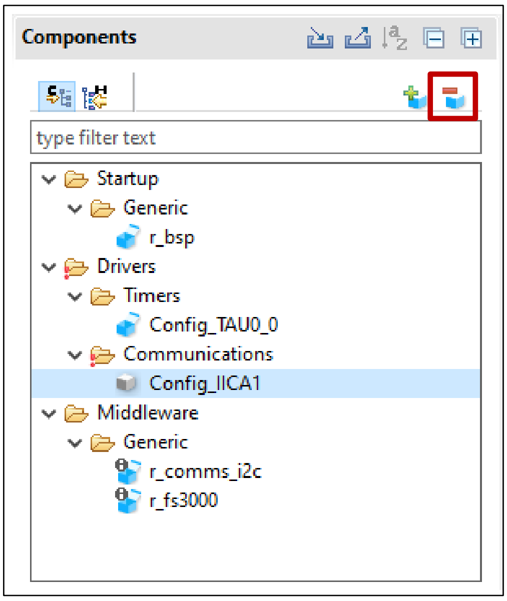
### (3) Components

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

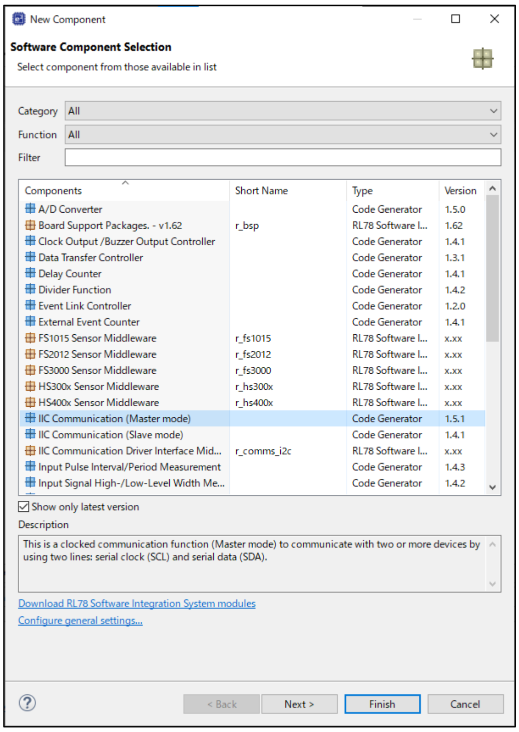
#### (a) Changing I2C Driver Settings

An SAU cannot be used because it does not support the clock stretching function required by FS3000 sensor.

~~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:

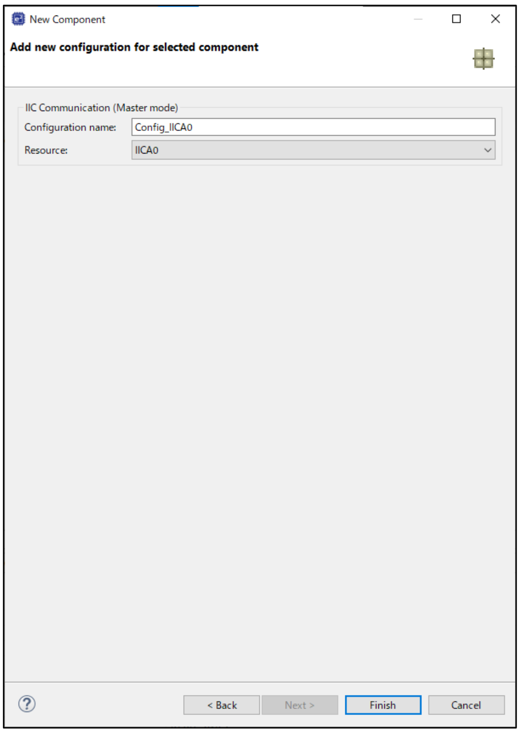
| Components | Short Name | Type | Version |
|---|---|---|---|
| A/D Converter | | Code Generator | 1.5.0 |
| Board Support Packages. - v1.62 | r_bsp | RL78 Software I... | 1.62 |
| Clock Output /Buzzer Output Controller | | Code Generator | 1.4.1 |
| Data Transfer Controller | | Code Generator | 1.3.1 |
| 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).

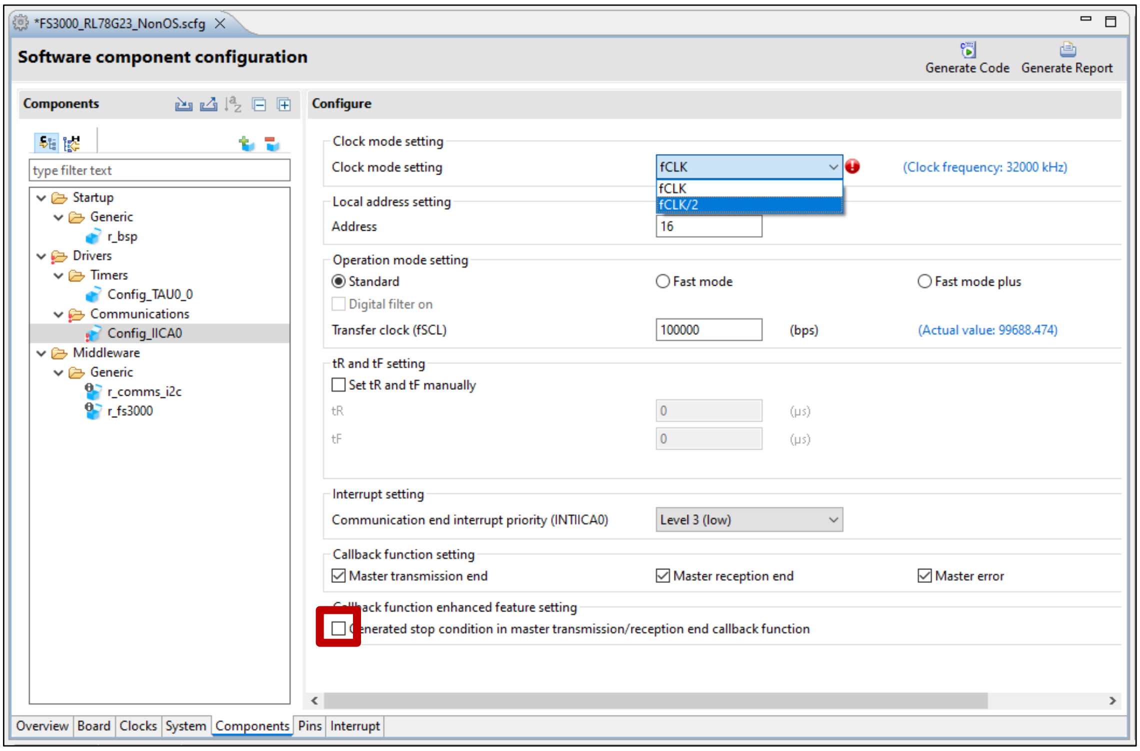
Download RL78 Software Integration System modules
Configure general settings...
< Back
Next >
Finish
Cancel


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".



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

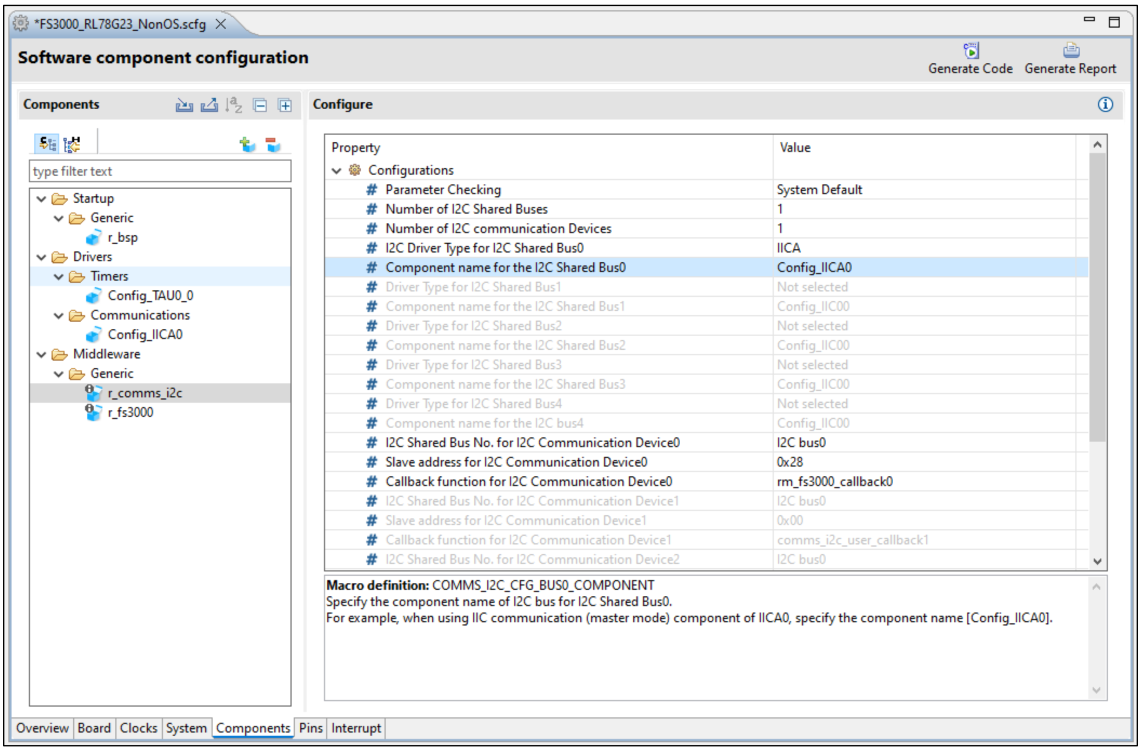
Callback function enhanced feature setting

□ 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.



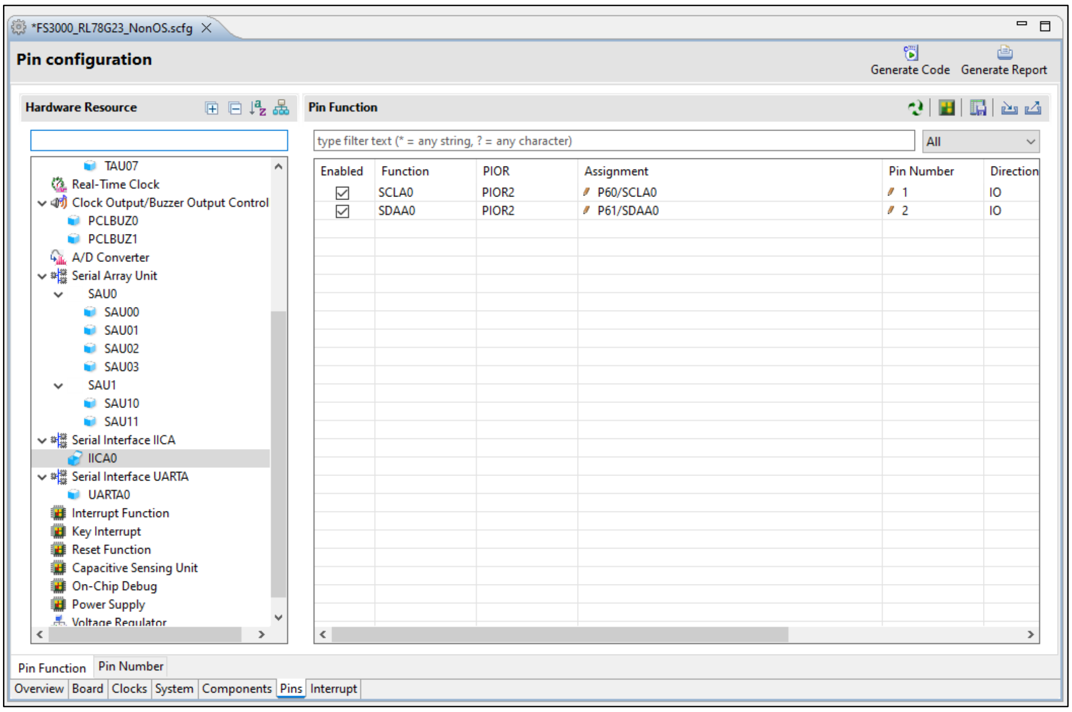
| Property | Value |
|---|---|
| Configurations | |
| Parameter Checking | System Default |
| Number of I2C Shared Buses | 1 |
| Number of I2C communication Devices | 1 |
| I2C Driver Type for I2C Shared Bus0 | IICA |
| Component name for the I2C Shared Bus0 | Config_IICA0 |
| Driver Type for I2C Shared 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 | 0x28 |
| Callback function for I2C Communication Device0 | rm_fs3000_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 |
| I2C Shared Bus No. for I2C Communication Device2 | I2C bus0 |

**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

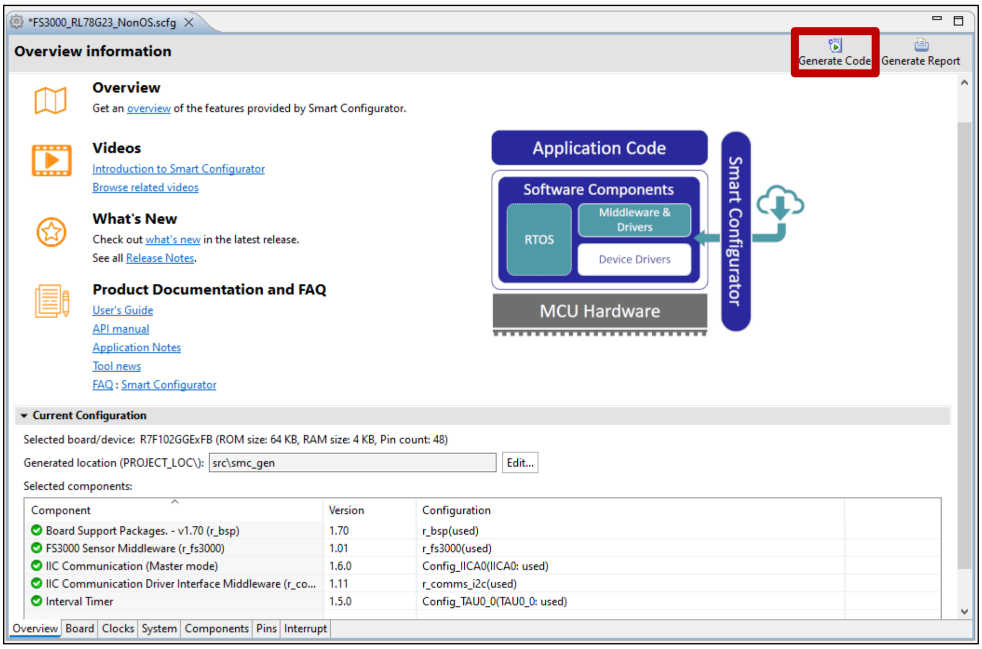
Open the "Pins" tabbed page and check that functions are assigned to the IICA pins in the "Pin function" panel.



| Enabled | Function | PIOR | Assignment | Pin Number | Direction |
|---|---|---|---|---|---|
| ☑ | 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.



**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 and FAQ**

User's Guide

API manual

Application Notes

Tool news

FAQ : Smart Configurator

**Current Configuration**

Selected board/device: R7F102GGExFB (ROM size: 64 KB, RAM size: 4 KB, Pin count: 48)

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

Selected components:

| Component | Version | Configuration |
|---|---|---|
| Board Support Packages. - v1.70 (r_bsp) | 1.70 | r_bsp(used) |
| FS3000 Sensor Middleware (r_fs3000) | 1.01 | r_fs3000(used) |
| IIC Communication (Master mode) | 1.6.0 | Config_IICA0(IICA0: used) |
| IIC Communication Driver Interface Middleware (r_co... | 1.11 | r_comms_i2c(used) |
| Interval Timer | 1.5.0 | Config_TAU0_0(TAU0_0: used) |

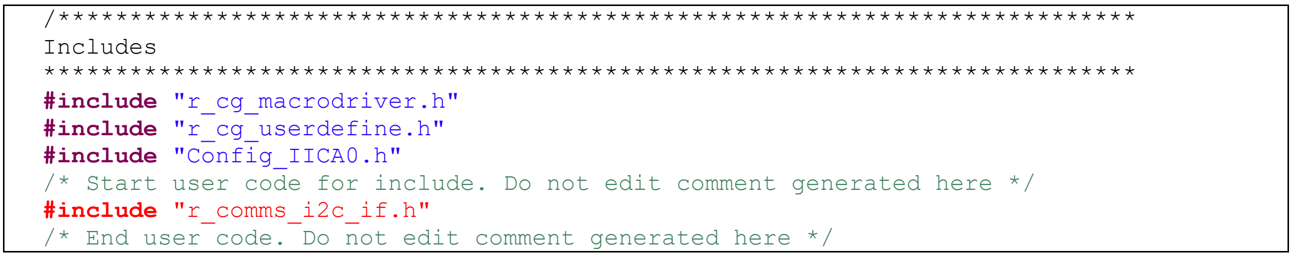
Build the project after implementing “6.3.4 Modifying Generated Code”.

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.

### 6.3.4 Modifying Generated Code

Open Config_IICA0_user.c and add the following code.

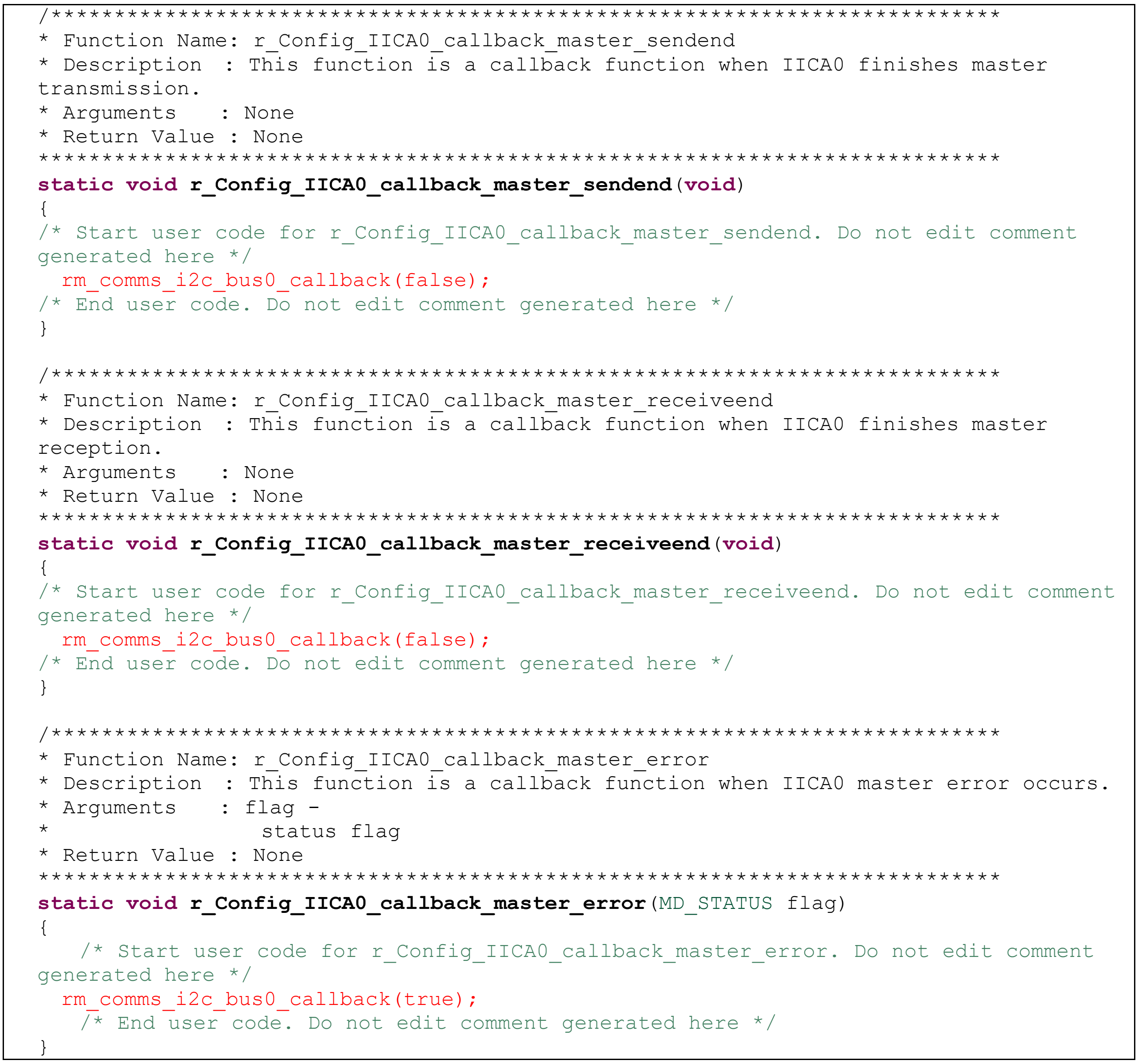
Definition for including r_comms_i2c_if.h:



```
/***************************************************************************
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 */
}
```

### 6.3.5 Changing Toolchain Setting

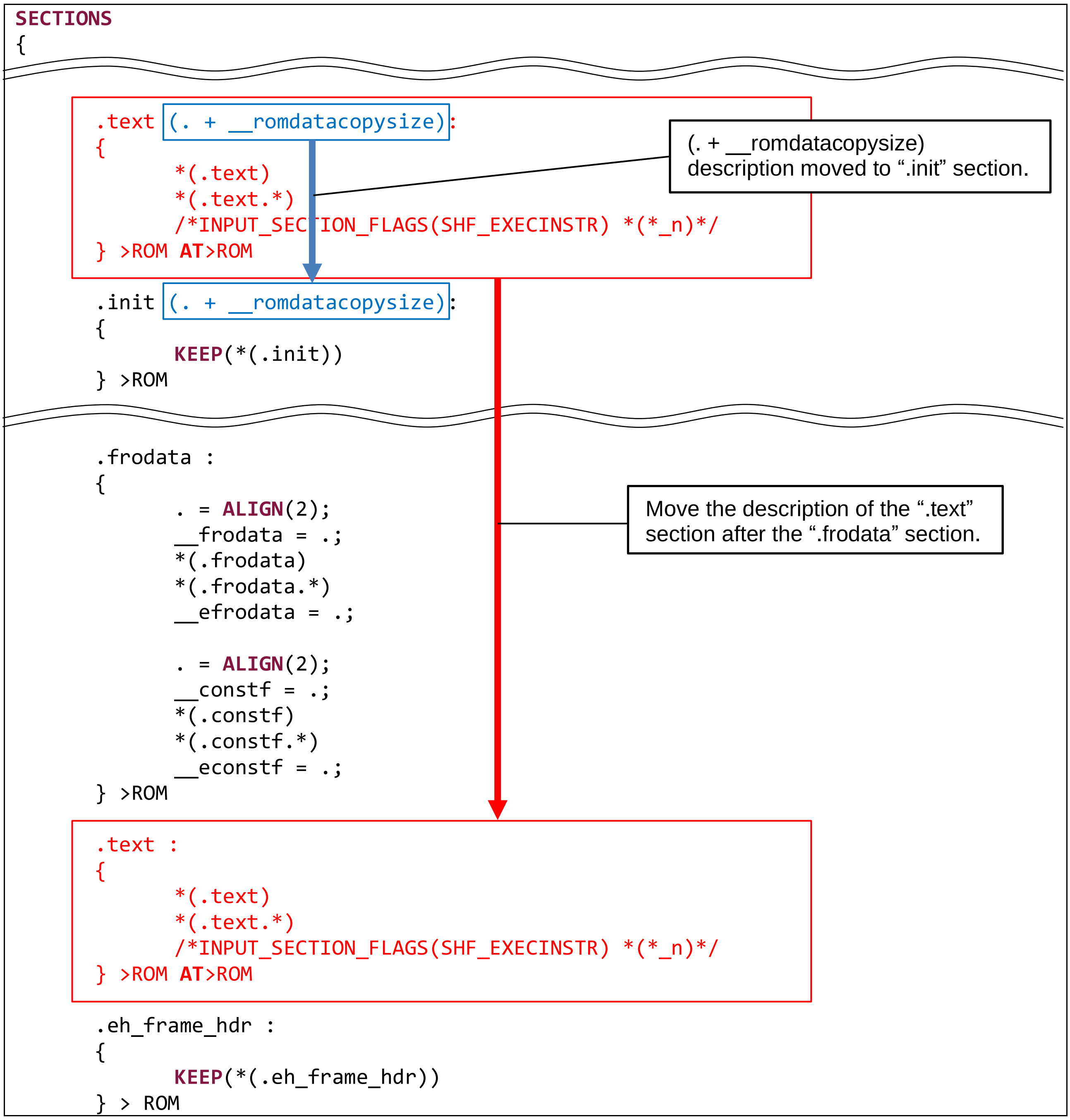
If you want to use a toolchain the LLVM toolchain, use “FS3000_RL78G23_NonOS_LLVM”.

If you want to use a toolchain other than the CC-RL toolchain or LLVM toolchain, copy FS3000_RL78G23_NonOS.c and RL78_FS3000.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 “FS3000_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 = .);
```

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

```
       .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 = .);
```

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

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

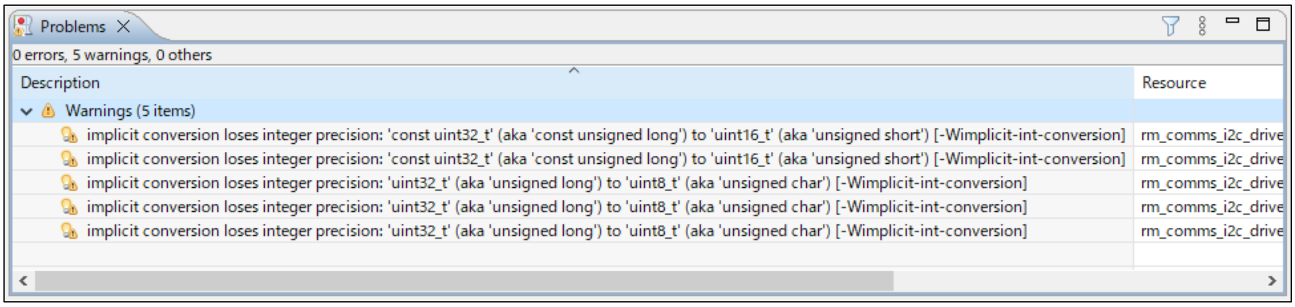
### 6.3.6 Notes for Build on LLVM

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.

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 |

## 6.4 Notes for Interrupt Signal Circuits

FS3000 does not have an Interrupt request signal pin.

Also, since the FS3000 sensor Pmod board does not have the pull-up resistor circuit and interrupt signal circuit, there are no precautions to take when daisy-chaining Renesas sensor Pmod boards.

## 6.5 Notes for RESET Signal Circuits

FS3000 does not have a RESET input signal pin.

Also, since the FS3000 sensor Pmod board does not have the pull-up resistor circuit, there are no precautions to take when daisy-chaining Renesas sensor Pmod boards.

## 6.6 Pull-up Resister 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 Resisters |
|---|---|
| #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 board without the pull-up resisters.

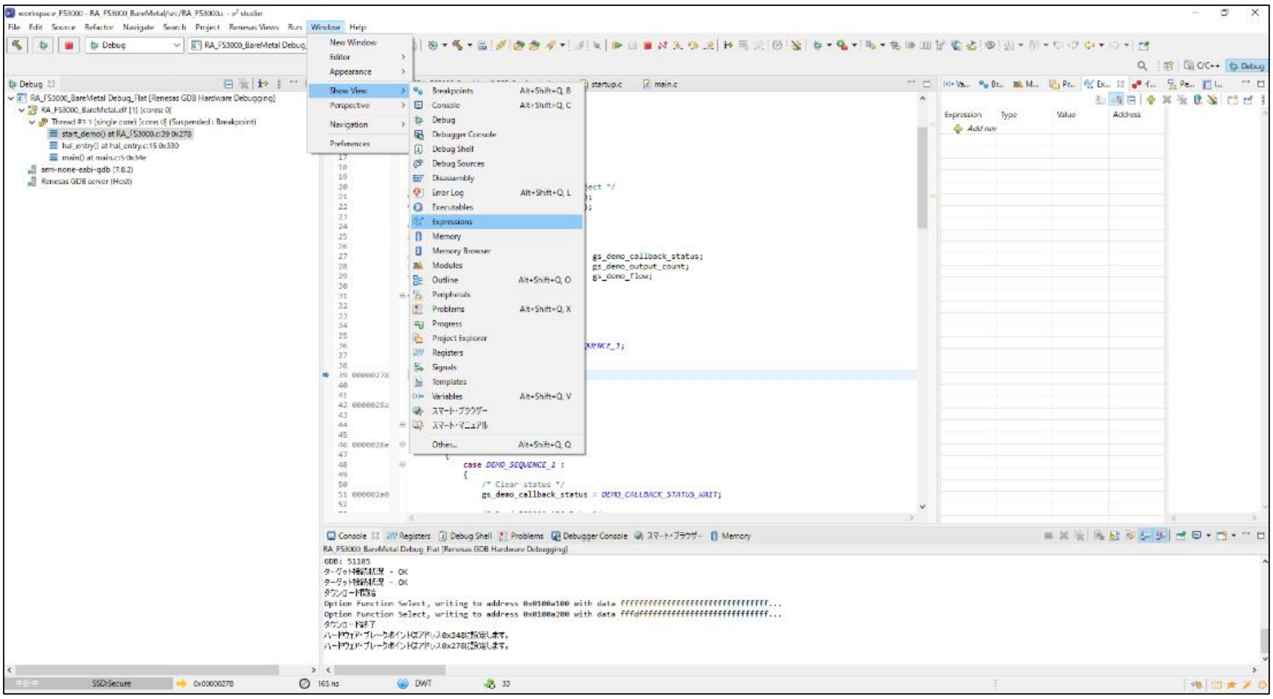
Note 2 Configure when using as an Interrupt signal

# 7. Viewing Air Velocity Data

Use the following procedure to view air velocity data in real time.

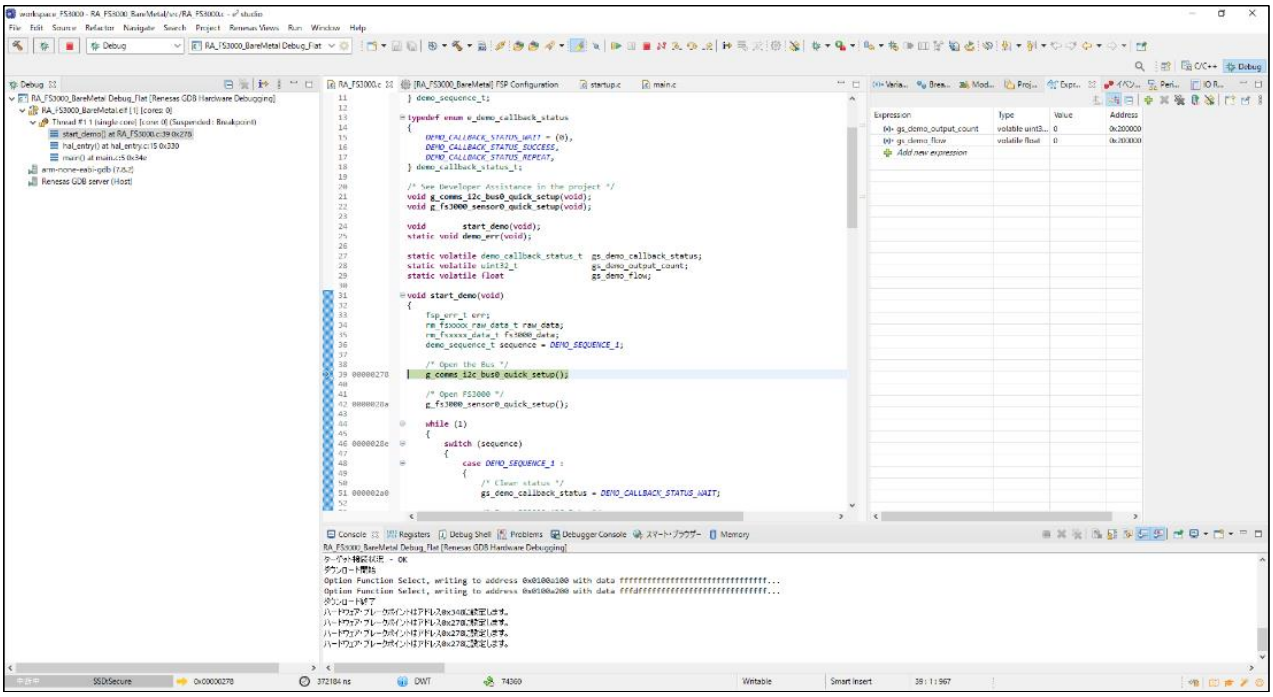
1. After running the Debug, open the “Expressions” window.

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



```
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
GDB: 51185
ターゲット接続状況 - OK
ターゲット接続状況 - OK
ダウンロード開始
Option Function Select, writing to address 0x0100a100 with data ffffffffffffffffffffffffffffffff...
Option Function Select, writing to address 0x0100a200 with data fffdffffffffffffffffffffffffffff...
ダウンロード終了
ハードウェア・ブレークポイントはアドレス0x348に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
```

2. Click “Add new expression” in the “Expressions” and add “gs_fs3000_data”.

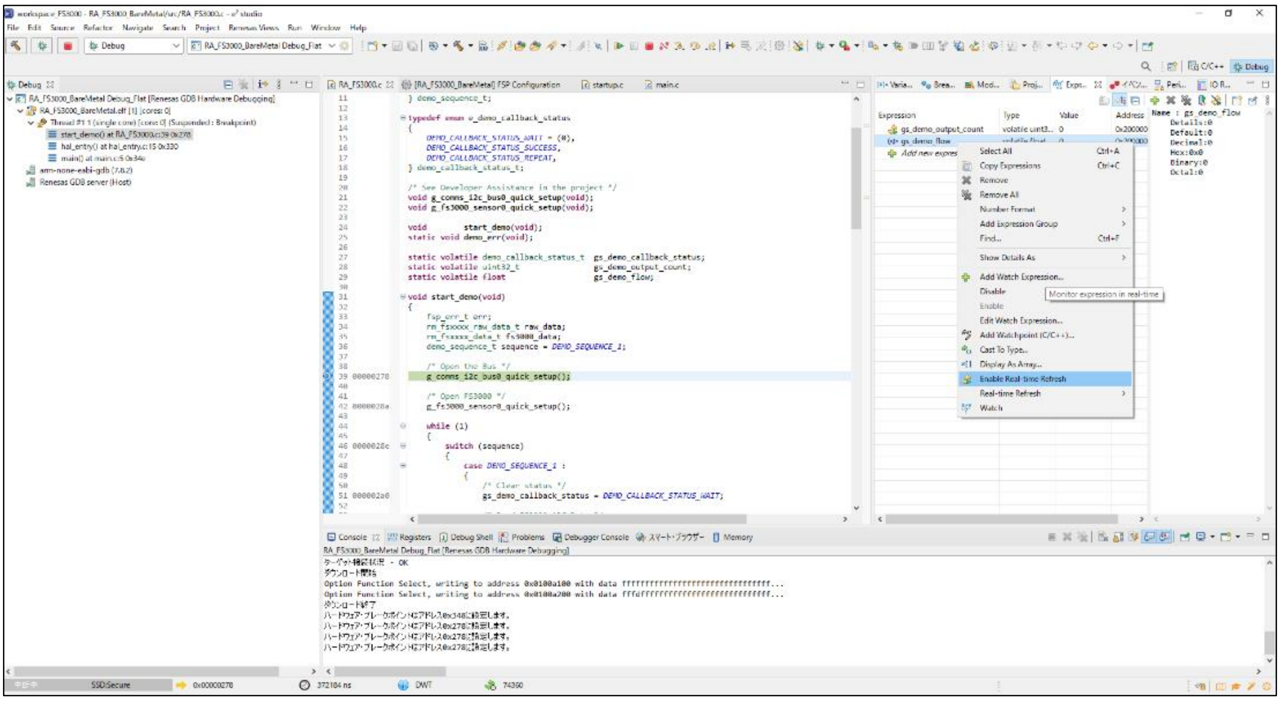


```
} demo_sequence_t;

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

/* See Developer Assistance in the project */
void g_comms_i2c_bus0_quick_setup(void);
void g_fs3000_sensor0_quick_setup(void);

void        start_demo(void);
static void demo_err(void);

static volatile demo_callback_status_t  gs_demo_callback_status;
static volatile uint32_t                gs_demo_output_count;
static volatile float                   gs_demo_flow;

void start_demo(void)
{
    fsp_err_t err;
    rm_fsxxxx_raw_data_t raw_data;
    rm_fsxxxx_data_t fs3000_data;
    demo_sequence_t sequence = DEMO_SEQUENCE_1;

    /* Open the Bus */
00000278    g_comms_i2c_bus0_quick_setup();

    /* Open FS3000 */
0000028a    g_fs3000_sensor0_quick_setup();

    while (1)
    {
0000028e        switch (sequence)
        {
            case DEMO_SEQUENCE_1 :
            {
                /* Clear status */
000002a0                gs_demo_callback_status = DEMO_CALLBACK_STATUS_WAIT;
```

| Expression | Type | Value | Address |
|---|---|---|---|
| (x)= gs_demo_output_count | volatile uint3... | 0 | 0x200000 |
| (x)= gs_demo_flow | volatile float | 0 | 0x200000 |
| Add new expression | | | |

```
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
ターゲット接続状況 - OK
ダウンロード開始
Option Function Select, writing to address 0x0100a100 with data ffffffffffffffffffffffffffffffff...
Option Function Select, writing to address 0x0100a200 with data fffdffffffffffffffffffffffffffff...
ダウンロード終了
ハードウェア・ブレークポイントはアドレス0x348に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
```

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



workspace_FS3000 - RA_FS3000_BareMetal/src/RA_FS3000.c - e² studio
File Edit Source Refactor Navigate Search Project Renesas Views Run Window Help
Debug
RA_FS3000_BareMetal Debug_Flat
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
RA_FS3000_BareMetal.elf [1] [cores: 0]
Thread #1 1 (single core) [core: 0] (Suspended : Breakpoint)
start_demo() at RA_FS3000.c:39 0x278
hal_entry() at hal_entry.c:15 0x330
main() at main.c:5 0x34e
arm-none-eabi-gdb (7.8.2)
Renesas GDB server (Host)
RA_FS3000.c
[RA_FS3000_BareMetal] FSP Configuration
startup.c
main.c
Select All Ctrl+A
Copy Expressions Ctrl+C
Remove
Remove All
Number Format
Add Expression Group
Find... Ctrl+F
Show Details As
Add Watch Expression...
Disable
Enable
Edit Watch Expression...
Add Watchpoint (C/C++)...
Cast To Type...
Display As Array...
Enable Real-time Refresh
Real-time Refresh
Watch
Monitor expression in real-time
SSD:Secure
0x00000278
372184 ns
DWT
74360

```
    } 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_t;

/* See Developer Assistance in the project */
void g_comms_i2c_bus0_quick_setup(void);
void g_fs3000_sensor0_quick_setup(void);

void        start_demo(void);
static void demo_err(void);

static volatile demo_callback_status_t  gs_demo_callback_status;
static volatile uint32_t                gs_demo_output_count;
static volatile float                   gs_demo_flow;

void start_demo(void)
{
    fsp_err_t err;
    rm_fsxxxx_raw_data_t raw_data;
    rm_fsxxxx_data_t fs3000_data;
    demo_sequence_t sequence = DEMO_SEQUENCE_1;

    /* Open the Bus */
00000278    g_comms_i2c_bus0_quick_setup();

    /* Open FS3000 */
0000028a    g_fs3000_sensor0_quick_setup();

    while (1)
    {
0000028e        switch (sequence)
        {
            case DEMO_SEQUENCE_1 :
            {
                /* Clear status */
000002a0                gs_demo_callback_status = DEMO_CALLBACK_STATUS_WAIT;
```

| Expression | Type | Value | Address |
|---|---|---|---|
| gs_demo_output_count | volatile uint3... | 0 | 0x200000 |
| gs_demo_flow | volatile float | 0 | 0x200000 |
| Add new expres |  |  |  |

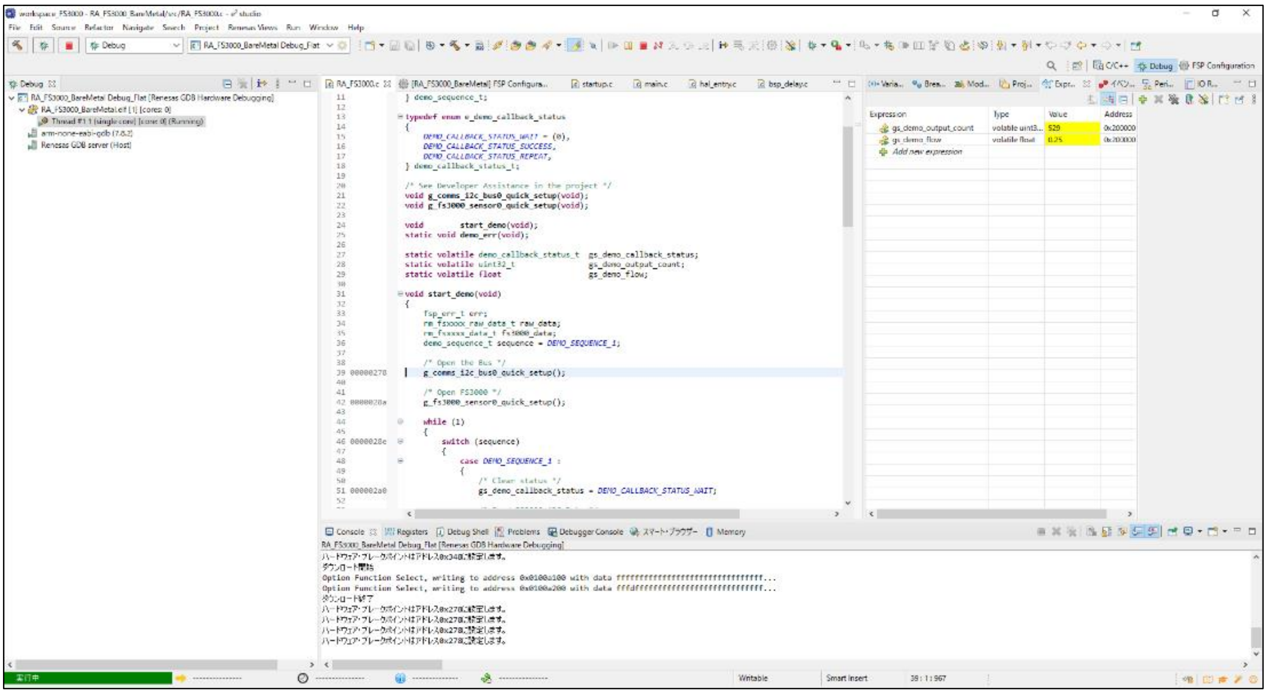
```
Name : gs_demo_flow
    Details:0
    Default:0
    Decimal:0
    Hex:0x0
    Binary:0
    Octal:0
```

Console Registers Debug Shell Problems Debugger Console スマート・ブラウザー Memory

```
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
ターゲット接続状況 - OK
ダウンロード開始
Option Function Select, writing to address 0x0100a100 with data ffffffffffffffffffffffffffffffff...
Option Function Select, writing to address 0x0100a200 with data fffdffffffffffffffffffffffffffff...
ダウンロード終了
ハードウェア・ブレークポイントはアドレス0x348に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
```

4. Start the Debug.

It is possible to check the real-time values.



workspace_FS3000 - RA_FS3000_BareMetal/src/RA_FS3000.c - e² studio
File Edit Source Refactor Navigate Search Project Renesas Views Run Window Help
Debug
RA_FS3000_BareMetal Debug_Flat
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
RA_FS3000_BareMetal.elf [1] [cores: 0]
Thread #1 1 (single core) [core: 0] (Running)
arm-none-eabi-gdb (7.8.2)
Renesas GDB server (Host)
RA_FS3000.c
[RA_FS3000_BareMetal] FSP Configura...
startup.c
main.c
hal_entry.c
bsp_delay.c
C/C++
Debug
FSP Configuration
実行中
Writable
Smart Insert
39 : 1 : 967

| Expression | Type | Value | Address |
|---|---|---|---|
| gs_demo_output_count | volatile uint3... | 529 | 0x200000 |
| gs_demo_flow | volatile float | 0.25 | 0x200000 |
| Add new expression |  |  |  |

```
RA_FS3000_BareMetal Debug_Flat [Renesas GDB Hardware Debugging]
ハードウェア・ブレークポイントはアドレス0x348に設定します。
ダウンロード開始
Option Function Select, writing to address 0x0100a100 with data ffffffffffffffffffffffffffffffff...
Option Function Select, writing to address 0x0100a200 with data fffdffffffffffffffffffffffffffff...
ダウンロード終了
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
ハードウェア・ブレークポイントはアドレス0x278に設定します。
```

Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | June 30, 2022 | - | First Release |
| 1.01 | March 3, 2023 | - | Updated: environments for RL78 |
| 1.02 | March 29, 2023 | - | Updated: Environments for RA, RX, RL78, RZ<br>Updated: Main Processing Flow of Sample Software<br>Updated: Guide for Changing the Target Device |
| 1.03 | September 7, 2023 | - | Updated: Guide for Changing the Target Device<br>Deleted: RE01 items |
| 1.04 | Dec.10.24 | - | Added: FS3000-1015<br>Added: Terms/Abbreviations<br>Updated: Environments for RA, RL78<br>Updated: Sensor Specifications<br>Updated: Sample Software Specifications<br>Updated: Configuration Setting<br>Updated: Guide for Changing the Target Device for RA, RL78 |

# 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.