RX Smart Configurator

User's Guide: IAREW

R20AN0535EJ0140
Rev.1.40
Apr 16, 2024

Introduction

This application note describes the basic usage of the RX Smart Configurator (hereafter called the Smart Configurator), and the procedure for importing its output files to IAR Embedded Workbench.

References to the Smart Configurator and Integrated Development Environment (IDE) in this application note apply to the following versions.

Target device and support compiler

Refer to the following URL for the range of supported devices:

https://www.renesas.com/rx-smart-configurator

Contents

Contents

1. Overview

1.1 Purpose

This application note describes the basic usage of the RX Smart Configurator (hereafter called the Smart Configurator), and the procedure for importing its output files to IAR Embedded Workbench.

Refer to the User's Manual of IAR Embedded Workbench for how to use them.

1.2 Features

The Smart Configurator is a utility for combining software to meet your needs. It handles the following three functions to support the embedding of drivers from Renesas in your systems: importing middleware in the form of Firmware Integration Technology (FIT) modules, generating driver code and making pin settings. Graphical presentation on Smart Configurator, for instance the timing waveform, makes your configuration of middleware and drivers easy.

1.3 Software Components

The Smart Configurator supports two types of software components: Code Generator (CG) and Firmware Integration Technology (FIT). Drivers and middleware supported by each software type are as follows.

- Basic drivers:
  - CG drivers (CMT, A/D Converter, SCI, etc.)
  - FIT modules (CMT, DTC, DMAC, RSPI, SCIFA, etc.)
- Middleware:
  - FIT modules (USB, Ethernet, Flash Memory (programming the on-chip flash memory), etc.)

The basic driver is a control program for peripheral functions of microcomputer such as CMT, A/D converter, SCI, etc. It is convenient to embed a software component (CG driver or FIT module) using code generation function.
In addition, FIT modules can be embedded for using middleware such as USB, Ethernet, and Flash memory (programming the on-chip flash memory) as software components.

2. Installation and uninstallation

This section describes the installation and uninstallation.

2.1 Installing the Smart Configurator

Download the Smart Configurator from the URL below.

https://www.renesas.com/rx-smart-configurator

After activating the installer, install the Smart Configurator and the plug-in by following the procedure of the installer. You will require administrator privileges to do this.

2.2 Uninstalling the Smart Configurator

If you wish to uninstall the Smart Configurator, select "Smart Configurator for RX" from [Programs and Features] in the control panel.

3. Operating the Smart Configurator

3.1 Procedure for Operations

Figure 3-1 Operating Procedure, shows the procedure for generating a source file using Smart Configurator and loading it into IAR Embedded Workbench. For the operation of IAR Embedded Workbench, refer to relevant document of IAR.

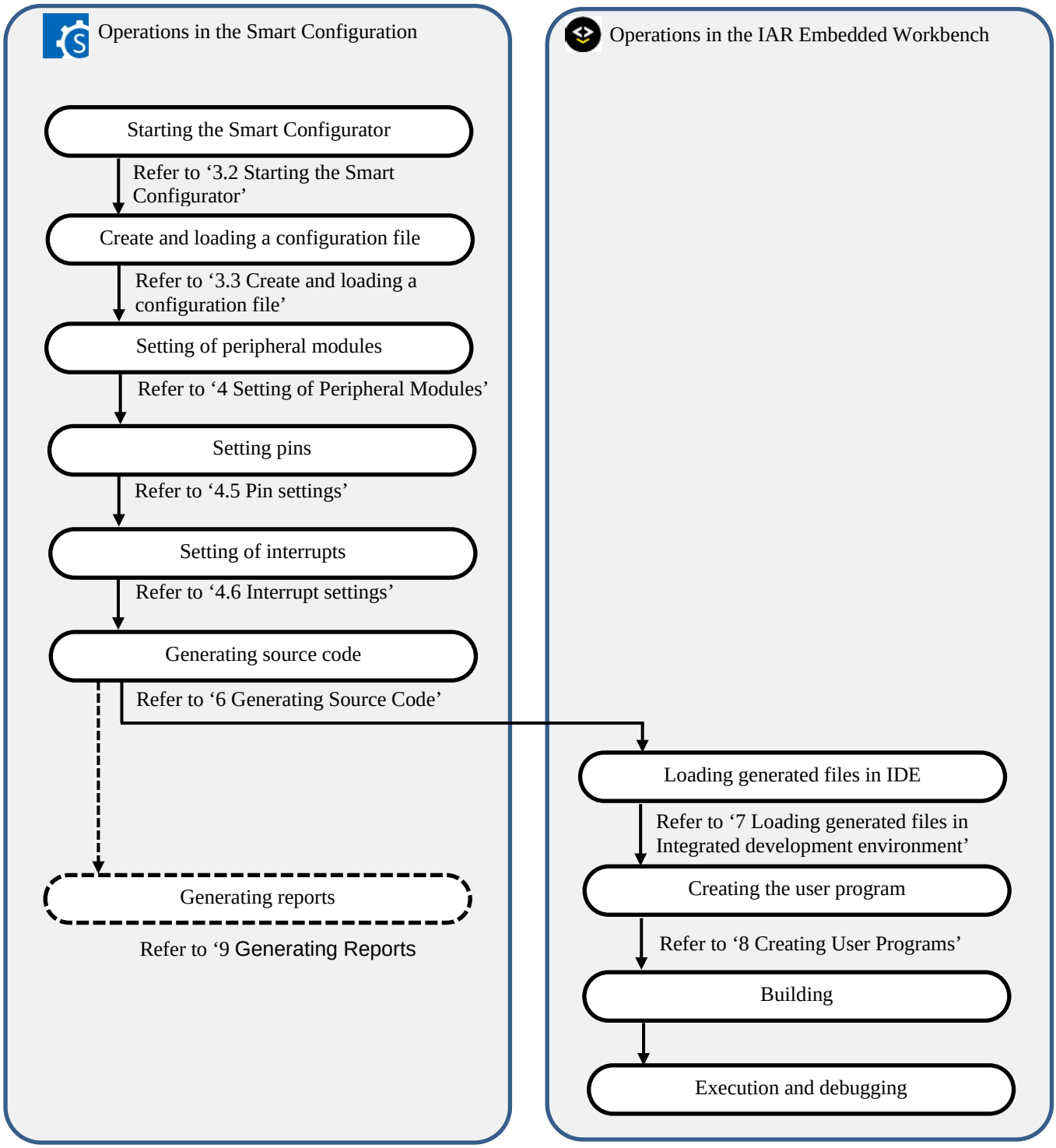

Figure 3-1 Operating Procedure

3.2 Starting the Smart Configurator

Select [Smart Configurator for RX Vx.x.x] of [Renesas Electronics Smart Configurator] from the Windows start menu. The main window of the Smart Configurator will be starting.

Note: Please replace Vx.x.x with your version.

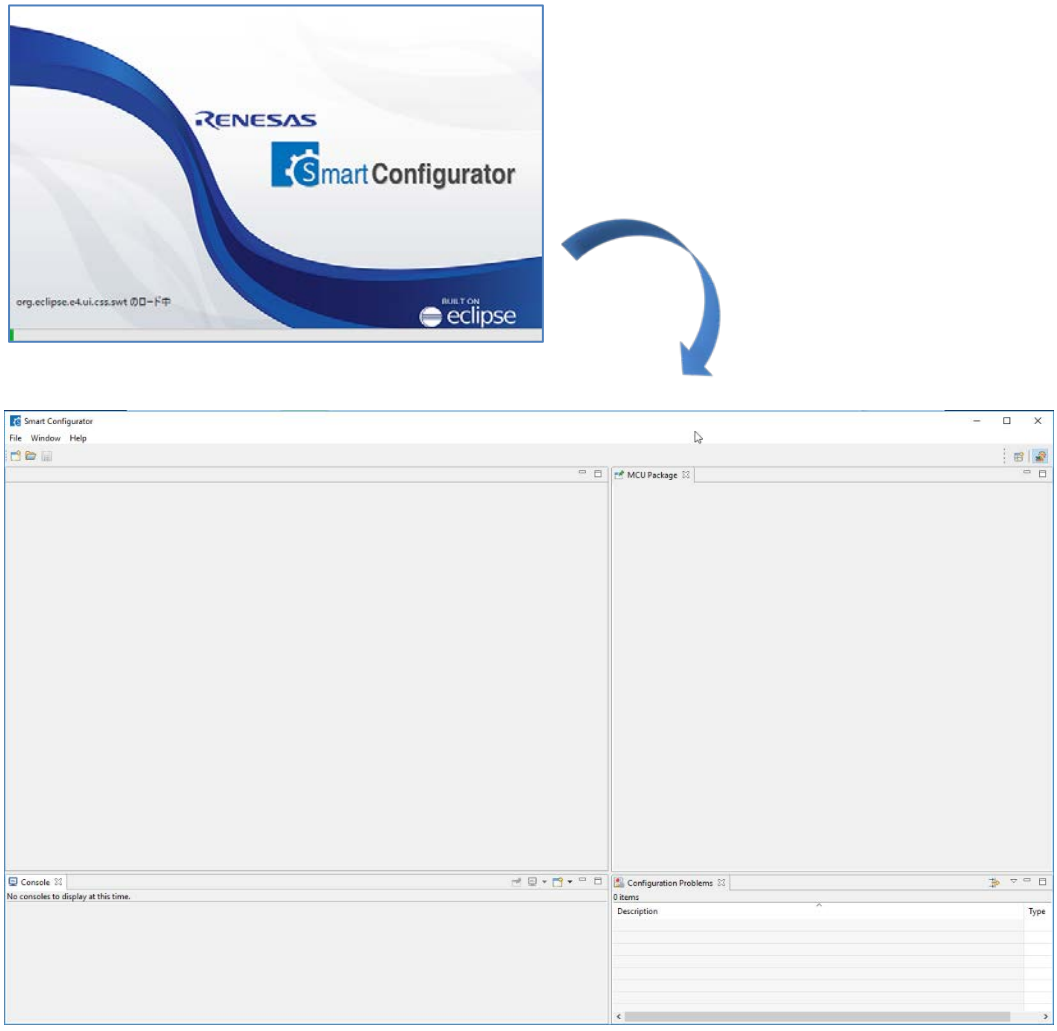

Figure 3-2 Starting of Smart Configurator

3.3 Create and loading a configuration file

Smart Configurator saves and refers to the configuration file (*. scfg) the configuration information of the microcontroller, build tool, peripheral function, pin function etc. used in the project.

3.3.1 Downloading FIT modules

The FIT drivers or middlewares are available from the web page of Renesas Electronics.

Download the files from the following address and unzip them.

https://www.renesas.com/fit

3.3.2 Creating a New Configuration File

On the main window, click the [New Configuration File] button to display the [New Smart Configuration File] dialog box.

(1) In [Platform:], select the device.

(2) In [Toolchain:], select [IAR EWRX Toolchain].

(3) In [File name:], enter the file name.

(4) Confirm [Location:]. If you want to change it, click [Browse] and select the save destination.
Note: The *.eww, *.ewp, *.ewd, main.c and buildinfo.ipcf files will be generated to this location after clicking "Generate Code" button.

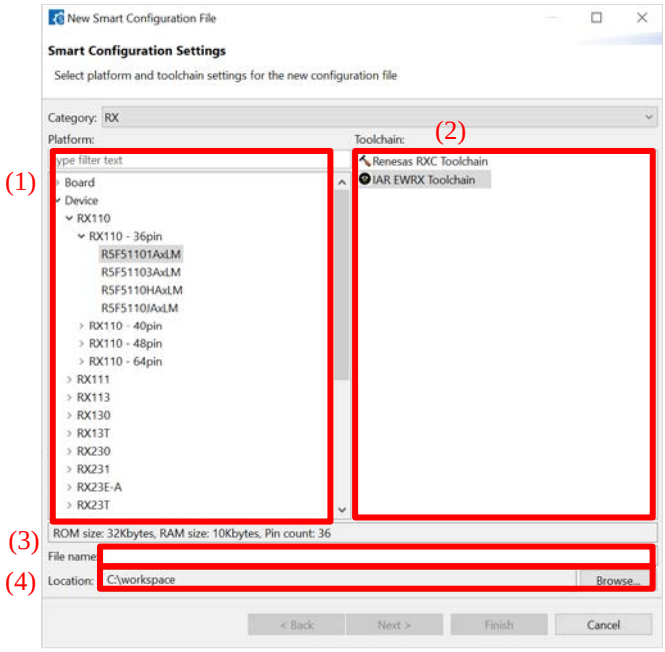

Figure 3-3 Create a Configuration File

(5) If you want to use FIT modules or middleware, click [Next].

a) Configure the language setting (C or C++) through wizard page and when C++ language is selected, main.cpp file will also be generated together with the IAR project file.

b) Configure the bank mode through the wizard page and corresponding Linear/Dual mode device will be automatically configured when loading the IAR project generated by Smart Configurator into IAR EW for RX.

c) Configure the RTOS settings.

d) Click [Browse] and set the path of "FITModules" directory which has been unzipped in chapter '3.3.1 Downloading FIT modules'.

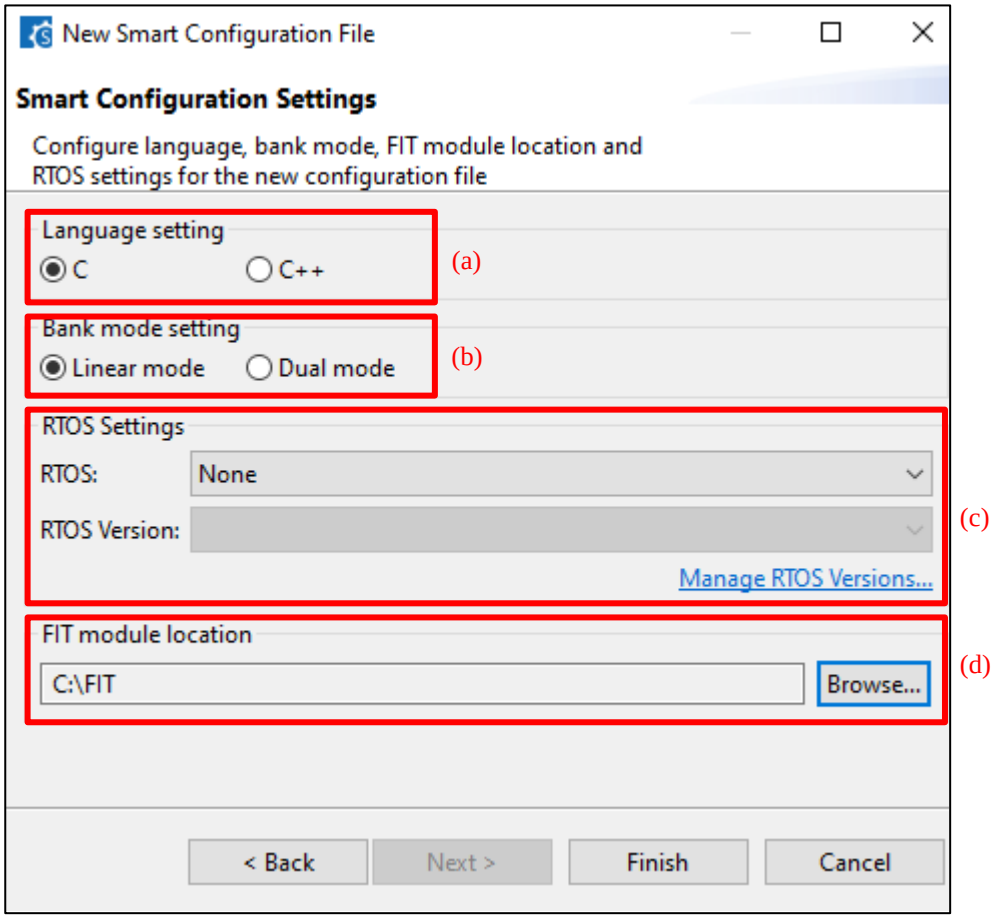

Figure 3-4 Smart Configuration Settings

(6) Click [Finish] to create the configuration file.

(7) Add driver component, configure the setting, generate code, and save the project.
Note: The *.eww, *.ewp, *.ewd and main.c files will be generated only for the first-time code generation, but the buildinfo.ipcf file will always be generated during the code generation.

3.3.3 Opening an Existing Configuration File

On the main window, click the [Opening an Existing Configuration File] button to display the [Open] dialog box. Select the file and click [Open].

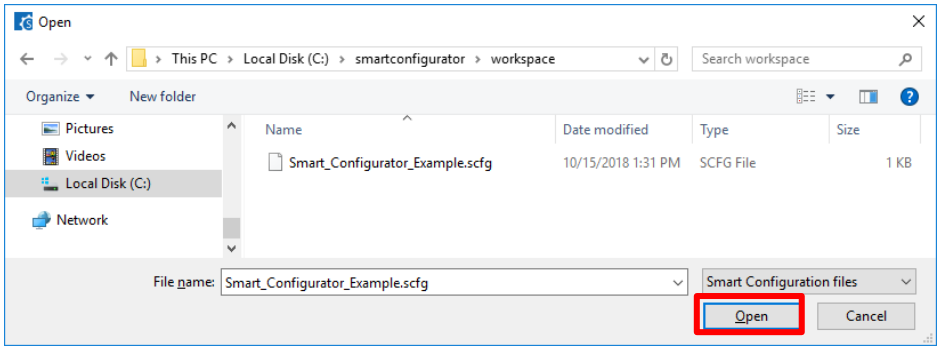

Figure 3-5 Opening an Existing Configuration File

3.4 Window

The main window is displayed when the Smart Configurator is started. The configuration of the window is shown in Figure 3-6, Main Window.

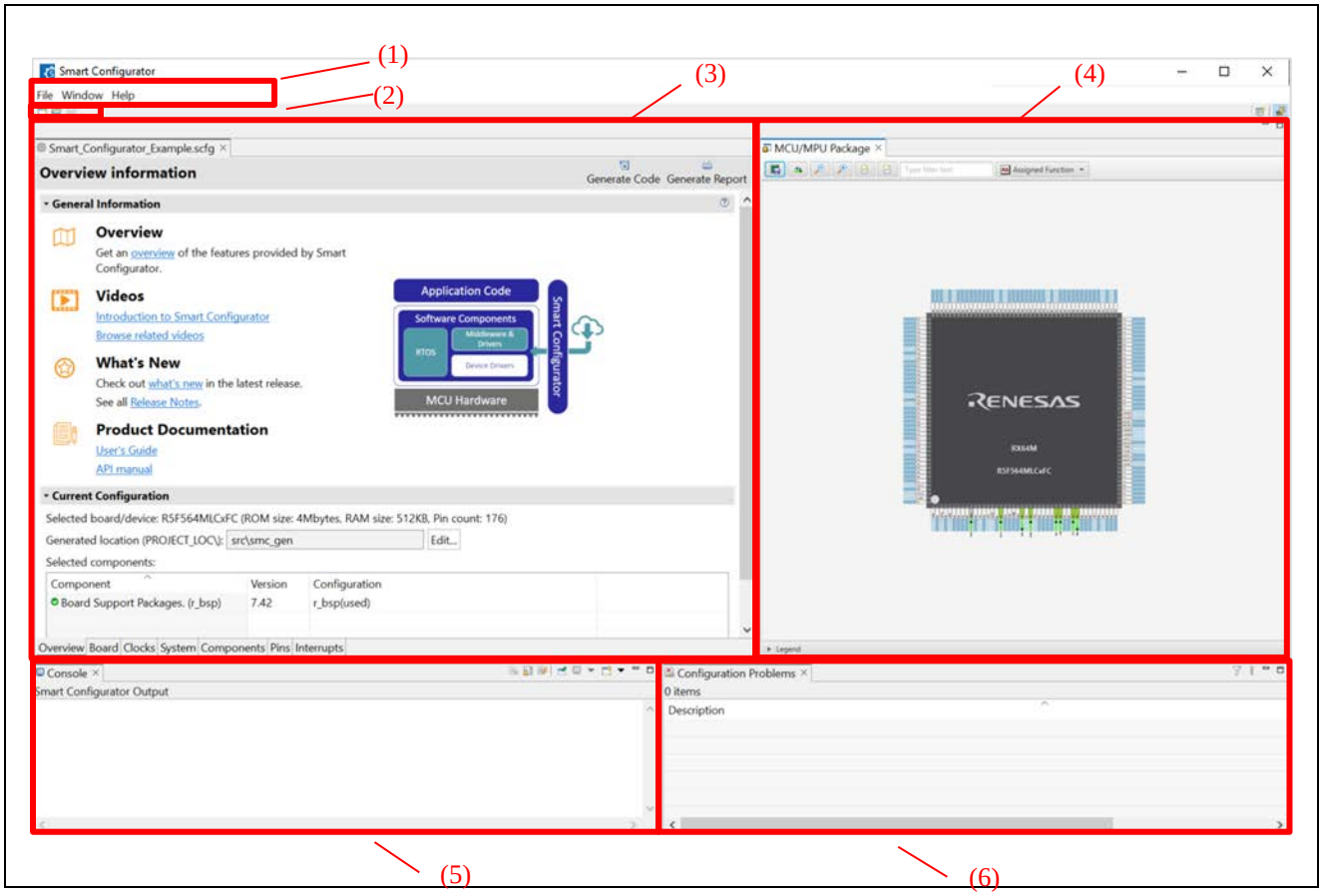

Figure 3-6 Main Window

(1) Menu bar

(2) Main tool bar

(3) Smart Configurator view

(4) MCU/MPU Package view

(5) Console view

(6) Configuration Problems view

3.4.1 Main menu

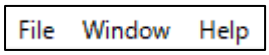

Table 3-1, Main Menu Items, lists the items of the main menu.

Table 3-1 Main Menu Items

Menu		Details
File	New	The dialog box [New Smart Configuration File], which is used to create a new configuration file, is displayed.
	Open	The dialog box [Open], which opens an existing configuration file, is displayed.
	Save	Saves a configuration file with the same name.
	Restart	Smart Configurator is re-started.
	Exit	Execution of the Smart Configurator is terminated.
Window	Preference	The dialog box [Preference], which is used to specify the properties of the configuration file, is displayed.
	Show view	The dialog box [Show view], which is used to set the view of the window, is displayed.
Help	Help Contents	The help menu is displayed.
	Home Page	Open the home page of the Smart Configurator on the Renesas Electronics website.
	Release Notes	Open the release note of the Smart Configurator on the Renesas Electronics website.
	Tool News	Open the tool news of the Smart Configurator on the Renesas Electronics website.
	API Manual	Open the API manual of the Smart Configurator on the Renesas Electronics website.
	About	The version information is displayed.

3.4.2 Toolbar

Some functions of the main menu are allocated to the buttons on the toolbar. Table 3-2, Toolbar Buttons and Related Menu Items, shows the description of those tool buttons.

Table 3-2 Toolbar Buttons and Related Menu Items

Toolbar button	Related menu item
	[File] → [New Smart Configuration File]
	[File] → [Open]
	[File] → [Save]

3.4.3 Smart Configurator view

The Smart Configurator view consists of seven pages: [Overview information], [Board], [Clocks], [System], [Components], [Pins], and [Interrupts]. Select a page by clicking on a tab; the displayed page will be changed.

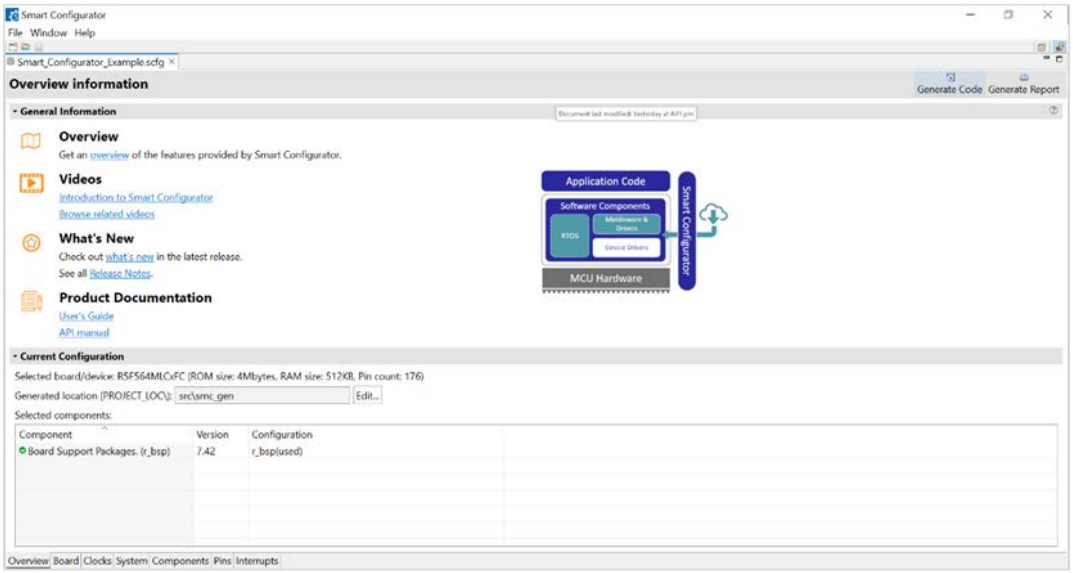

Figure 3-7 Smart Configurator View

3.4.4 MCU/MPU Package view

Display the MCU package. You can save rotation, enlargement, reduction, and MCU/MPU package view of the display to the image file. You can also confirmation pin assignment and change it.

Three types of package view can be switched between [Assigned Function], [Symbolic Name] and [Board Function]. [Assigned Function] displays the assignment status of the pin setting, and [Board Function] displays the initial pin setting information of the board. [Symbolic Name] displays the symbolic name information of the pins. The initial pin setting information of the board is the pin information of the board selected by [Board:] on the [Board] page.

(refer to "chapter 4.1.2 Selecting the board" and "chapter 4.5.6 Pin setting using board pin configuration information").

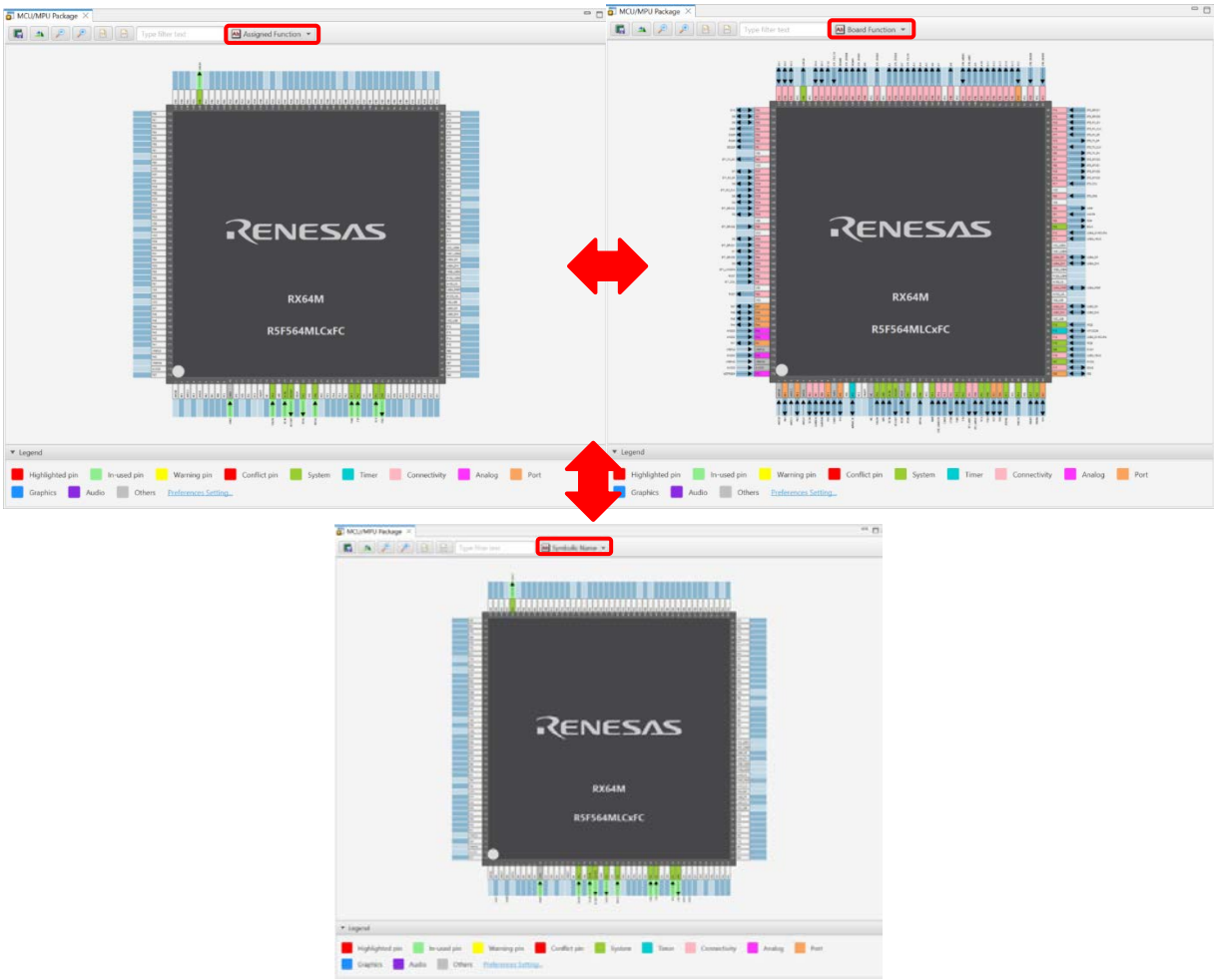

Figure 3-8 MCU/MPU Package View

3.4.5 Console view

The console displays details of changes to the configuration made in the Smart Configurator or MCU/MPU Package view.

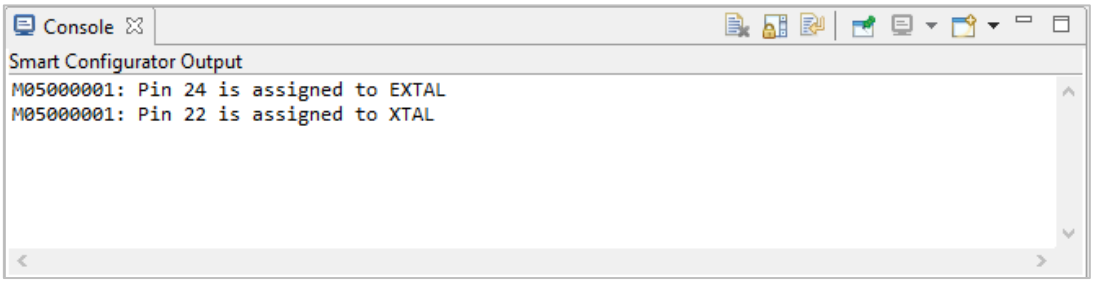

Figure 3-9 Console View

3.4.6 Configuration Problems view

The Configuration Problems view displays problems with peripheral functions, interrupts, and pin conflicts.

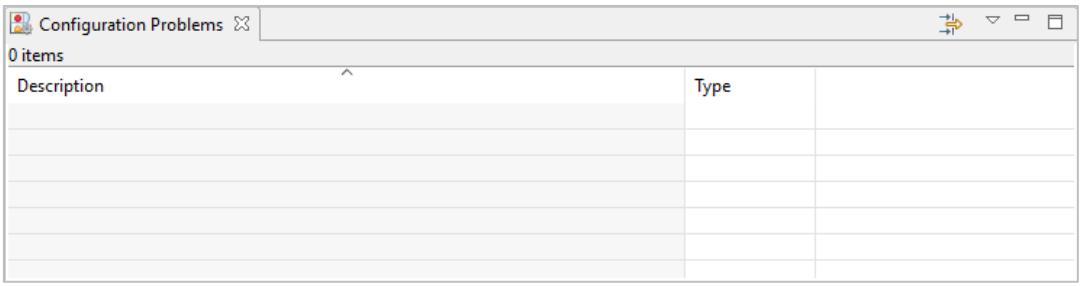

Figure 3-10 Configuration Problems View

4. Setting of Peripheral Modules

You can select peripheral modules from the Smart Configurator view.

4.1 Board setting

On the [Board] page, you can select boards and change devices.

4.1.1 Selecting the device

Click on the [...] button to select a device.

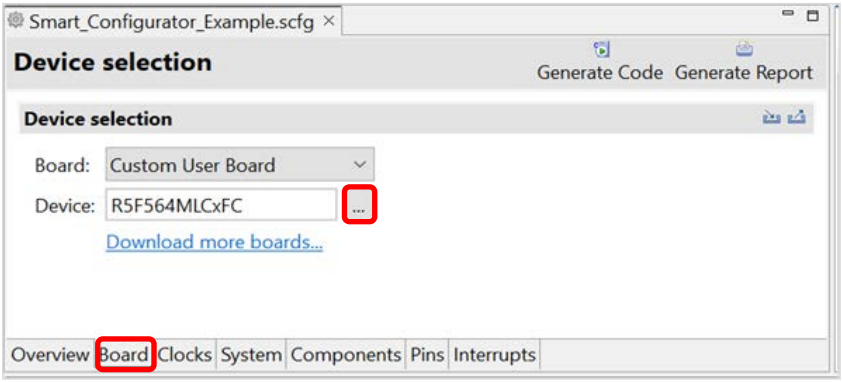

Figure 4-1 Selecting the Device

Note: Device change is not reflected to the device (micro controller) of IAR project.

The following message is displayed when changing the device. For each button operation, refer to "Table 4-1, Device Change Confirmation Operation List".

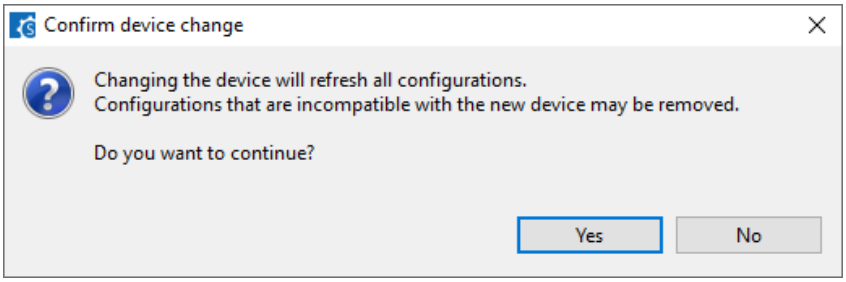

Figure 4-2 Confirm Device Change

Table 4-1 Device Change Confirmation Operation List

Button	Operation explanation
Yes	Change to the selected device.
No	It does not change the device.
Save and continue	After saving the current configuration contents to the configuration file, change to the selected device.
Continue	Changes to the selected device without saving the current configuration contents to the configuration file.
Cancel	It does not change the device.

4.1.2 Selecting the board

Click on the [ ] to select a board from the list. When peripheral functions are configured by board selection, pins are automatically set according to board connection.

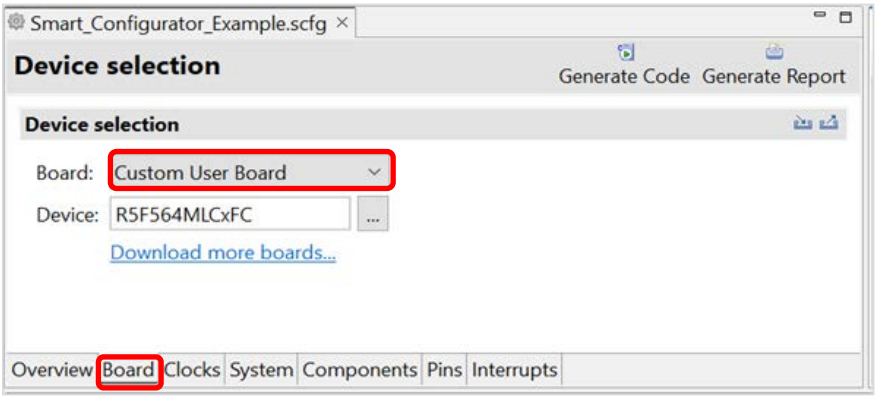

Figure 4-3 Selecting the Board

The following items are changed according to the configuration of the selected board.

- Pin assignment
- Frequency of the main clock
- Frequency of the sub-clock
- Target device

If you change the board, the message shown in "Figure 4-2" or the following message will be displayed. For each button operation, refer to "Table 4-2, Board Change Confirmation Operation List".

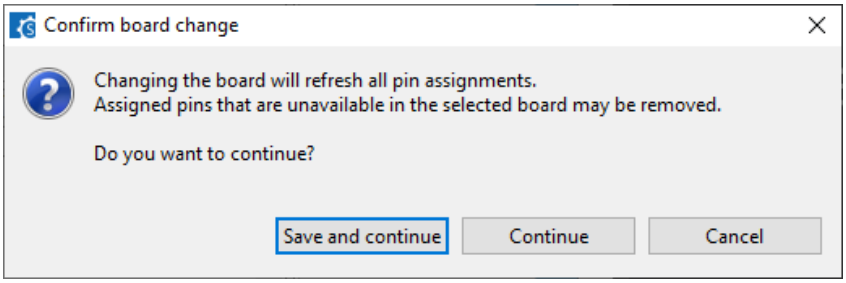

Figure 4-4 Confirm Board Change

Table 4-2 Board Change Confirmation Operation List

Button	Operation explanation
Save and continue	After saving the current configuration contents to the configuration file, change to the selected board.
Continue	Changes to the selected board without saving the current configuration contents to the configuration file.
Cancel	It does not change the board.

Note: Depending on the board selected, the device will change, Device change is not reflected to the device (microcontroller) of IAR project.

4.1.3 Import of board configuration

The board setting is defined in bdf (Board Description File). Follow the procedure below to import board configuration.

(1) Click on the [(Import board setting)] button and select a desired bdf file.

(2) The board of the imported settings is added to the board selection menu.

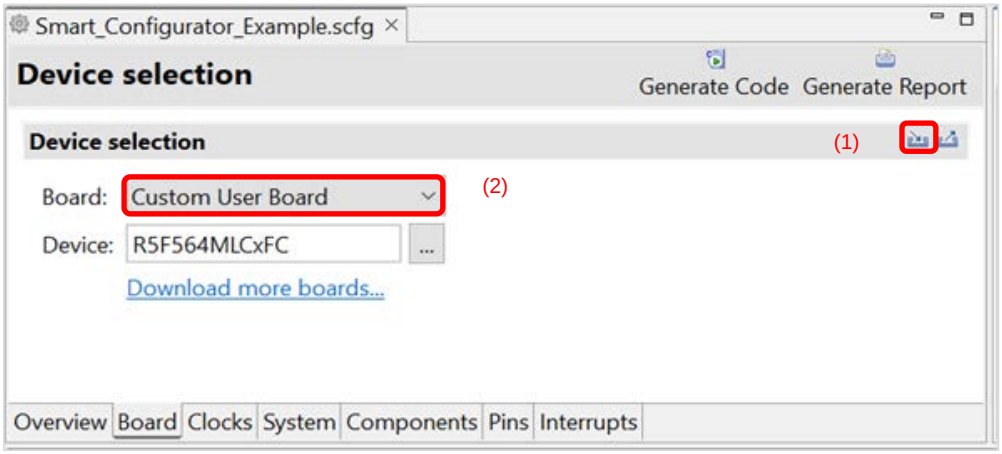

Figure 4-5 Import of Board Configuration (bdf format)

Once a board setting file is imported, the added board is also displayed in the board selection menu of other projects for the same device group.

4.1.4 Export of board configuration

The current main clock frequency, sub clock frequency and pin assignment settings can be exported as board configuration. Follow the procedure below to export the board configuration.

(1) Click on the [(Export board setting)] button on the [Board] tabbed page.

(2) Select the output location and specify a name (Display Name) for the file to be exported.

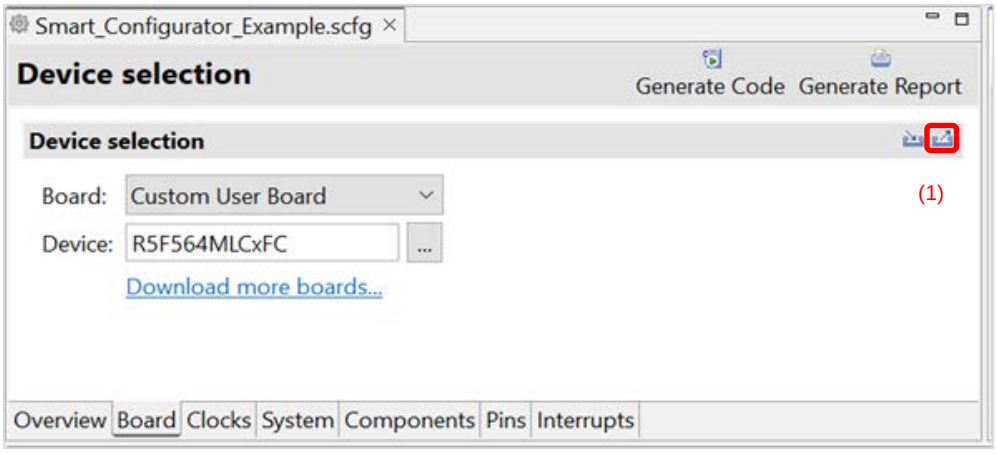

Figure 4-6 Export of Board Configuration (bdf format)

4.2 Clock settings

On the [Clocks] page, set the clock. The [Clocks] page setting is used as the clock source for each component. Set the clock before configuring the component.

The clocks setting is performed in the following procedure.

(1) Set the clock oscillator circuit.

(2) Sets the clock source to be supplied to the CPU and peripheral functions.

(a) When you move the mouse on the screen, the clock signal is displayed in blue.

(b) Click on the screen to select the clock selector.

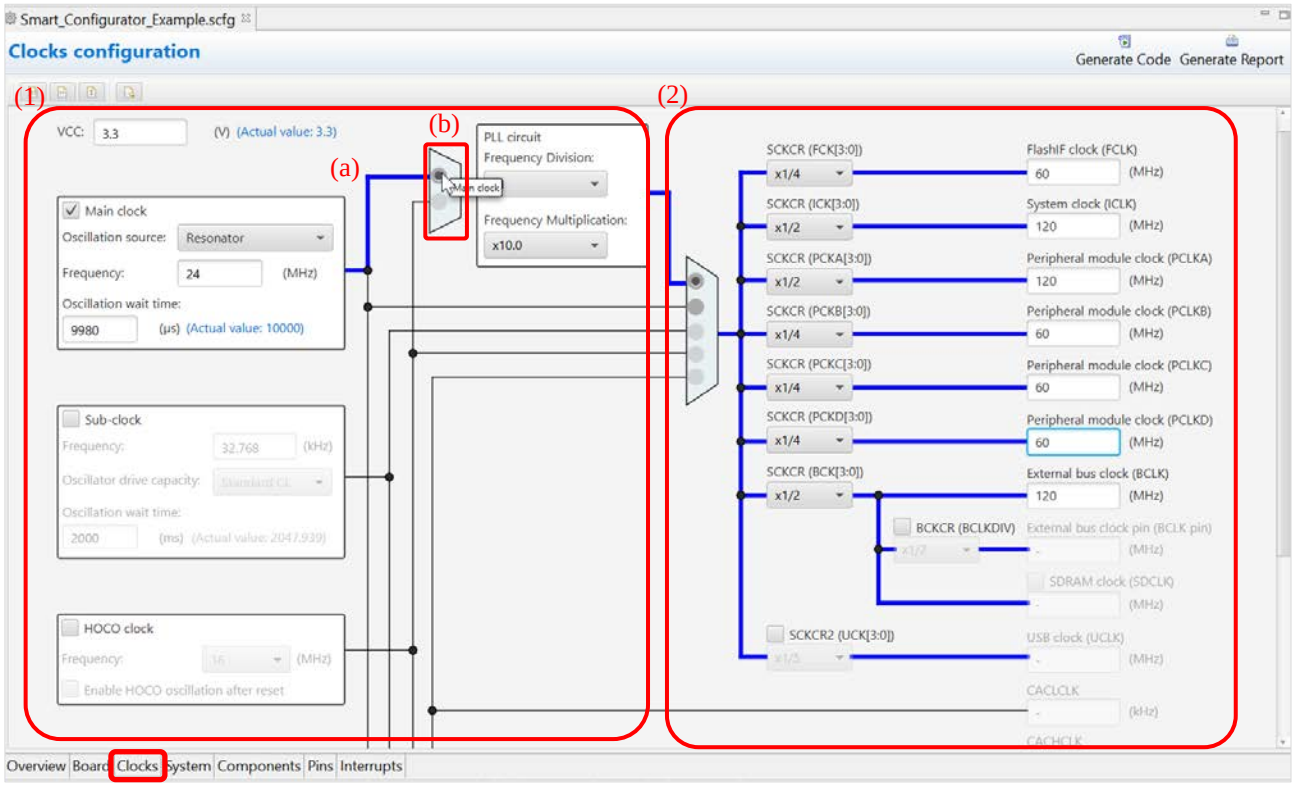

Figure 4-7 Clock Settings

4.3 System Settings

You can set the debug interface pins at [System] tabbed page.

There are 3 types of debug interface available: FINE, JTAG, JTAG (Trace)

You can check the pins configured from Console message or MCU/MPU Package view.

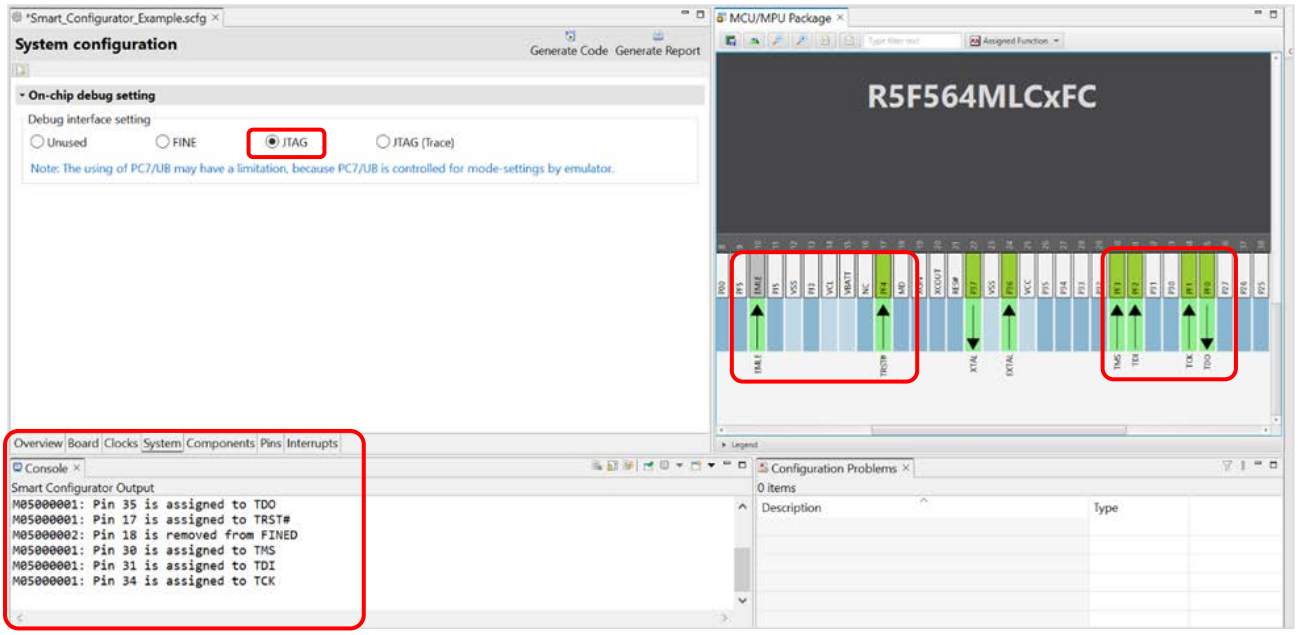

Figure 4-8 Debug Interface Setting at [System] Page

4.4 Software component settings

Drivers can be combined as software components on the [Components] page. Added components are displayed in the component tree at the left of the page.

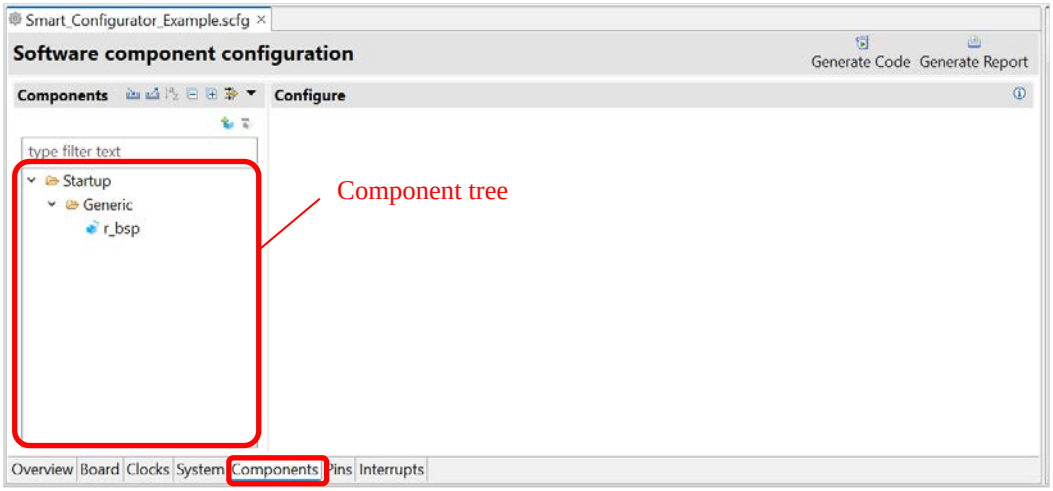

Figure 4-9 Component Page

4.4.1 Adding component

The following describes the procedure for adding a component.

(1) Click on the [(Add component)] icon.

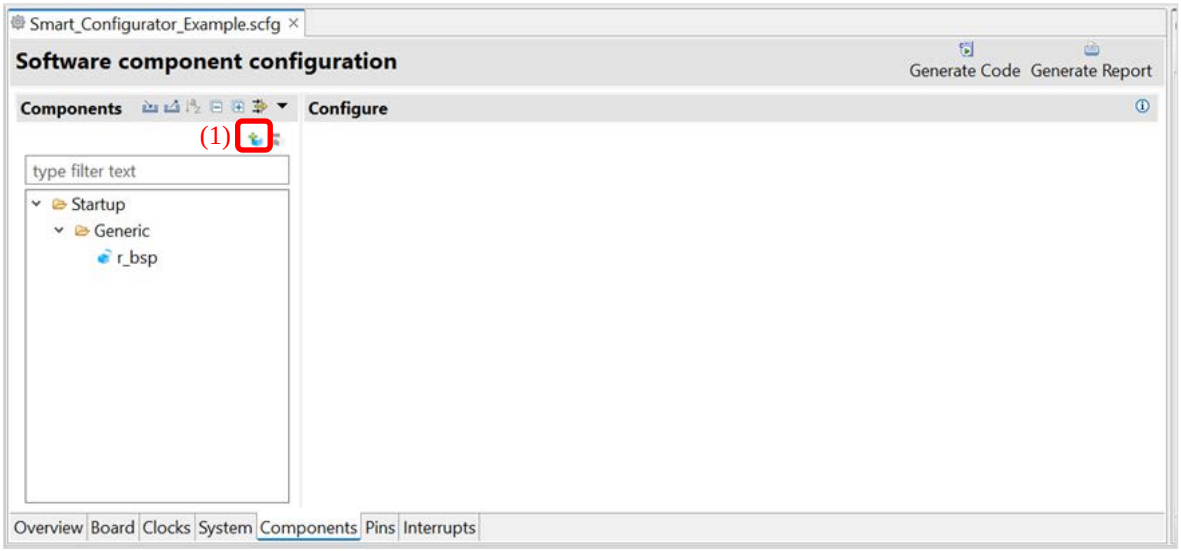

Figure 4-10 Adding Components

(2) Select a component from the list in the [Software Component Selection] page of the [New Component] dialog box (e.g. Single Scan Mode S12AD).

(3) Click on [Next].

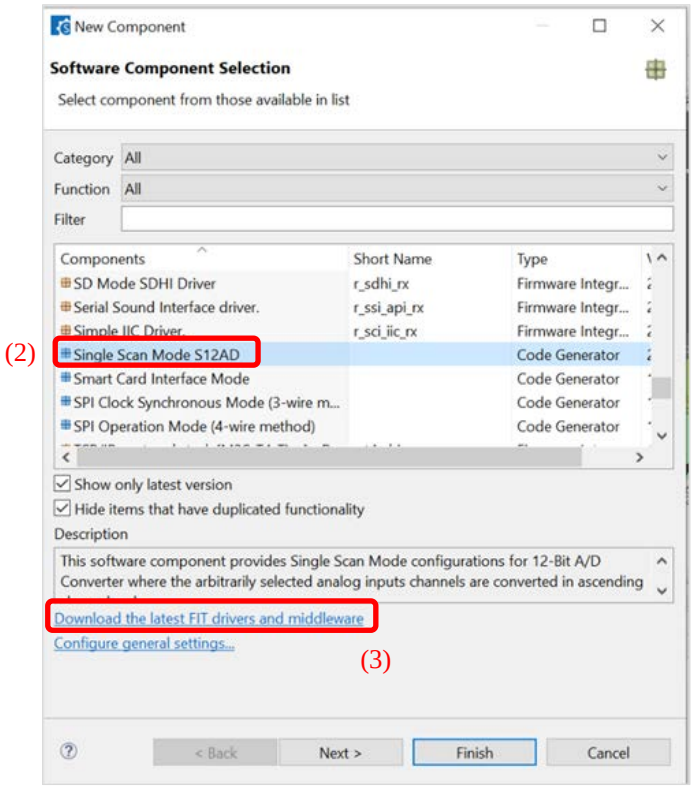

Figure 4-11 Selection of Software Components

(4) Specify an appropriate configuration name in the [Add new configuration for selected component] page or use the default name (e.g. Config_S12AD0).

(5) Select a hardware resource or use the default resource (e.g. S12AD0).

(6) Click on [Finish]. The component is added to the component tree.

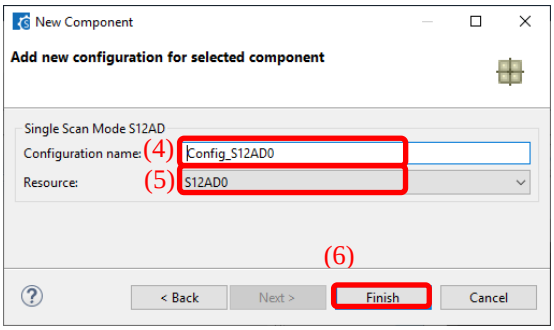

Figure 4-12 Add New Configuration for Selected Component (e.g. S12AD0)

4.4.2 Removing a component

Follow the procedure below to removing a software component.

(1) Select a software component from the Components tree.

(2) Click on the [(Remove component)] icon. The selected software is removed from the component tree. The selected software component will be removed from the Components tree.

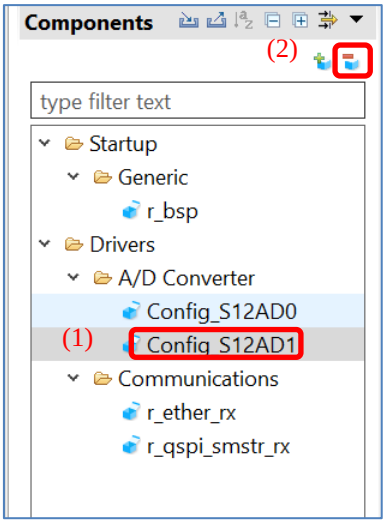

Figure 4-13 Removing a Component

Multiple components can be selected by pressing [Ctrl] and clicking on components. Click on the [(Remove component)] icon. So multiple components can be removed at the same time.

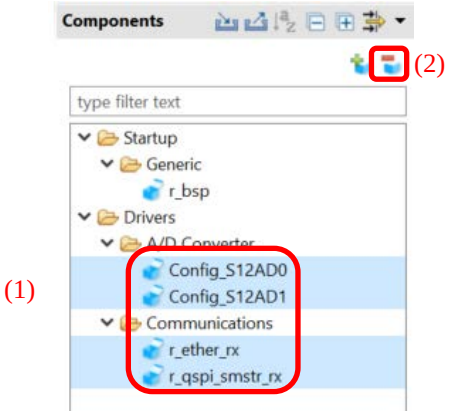

Figure 4-14 Removing Software Components

4.4.3 Switching between the component view and hardware view

The Smart Configurator also provides a function for adding a new component by directly clicking a node in the Components tree. To use this function, you need to switch the view of the Components tree from the component view to the hardware view.

(1) Click on the [(View Menu)] icon and select [Show by Hardware View]. The Components tree will display the components in a hardware resource hierarchy.

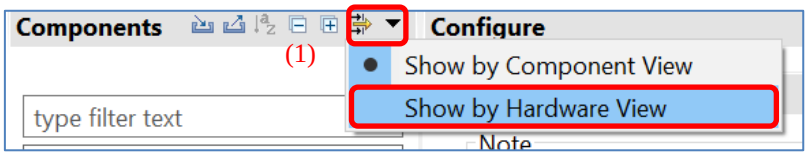

Figure 4-15 Switch to [Show by Hardware View]

(2) Double-click on a hardware resource node (e.g. S12AD1 under 12-bit A/D converter) to open the [New Component] dialog box.

(3) Select a component from the list (e.g. Single Scan Mode S12AD) to add a new configuration as described in "chapter 4.4.1Adding component".

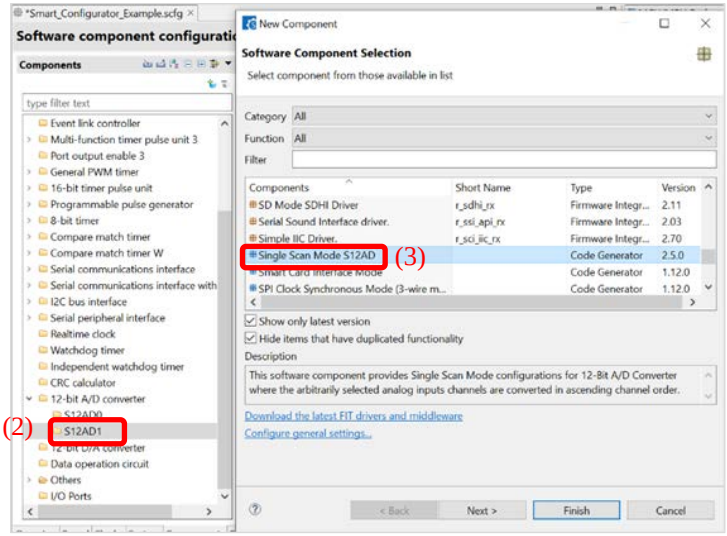

Figure 4-16 Adding CG Components form the Hardware View

4.4.4 Component configuration settings

Follow the procedure below to setting the component configuration.

(1) Click the component in the component tree. (e.g. Config_S12AD0).

(2) Configure the driver in the [Configure] panel to the right of the Components tree. The Figure 4-17 is an example.

a. Select AN000.

b. Select [A/D conversion start trigger pin] under [Conversion start trigger setting].

c. Click on [Advance setting] to expand the view.

d. Select [Discharge] for [Charge setting].

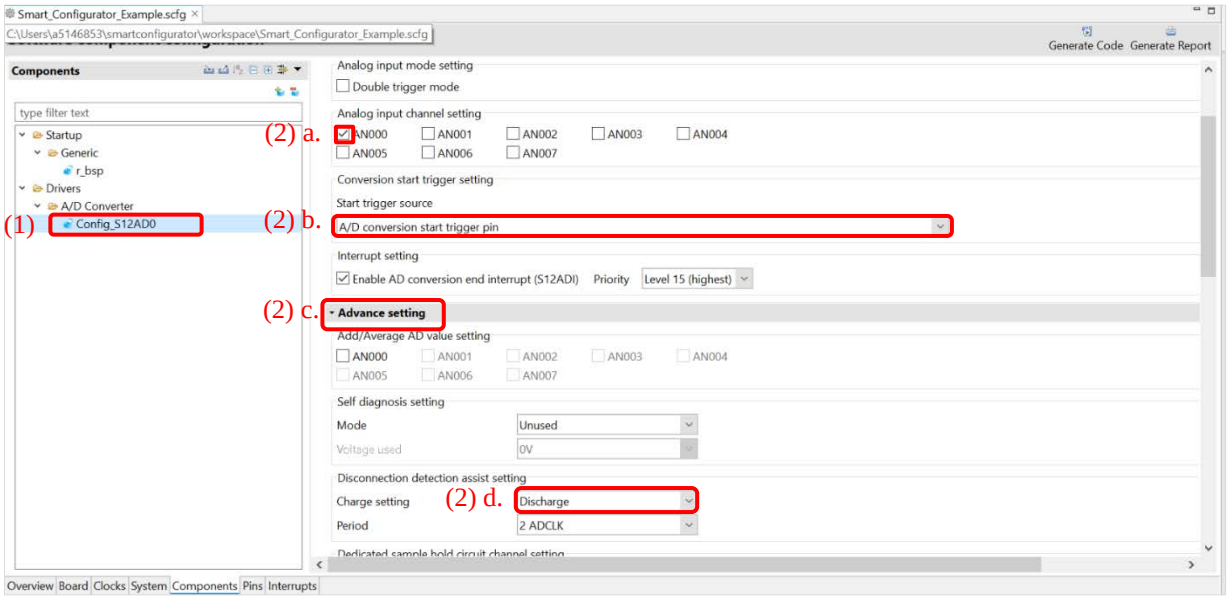

Figure 4-17 Component Configuration Settings

The code generation of the component is set to enabled by default.

Right click on the component and click [✓ Generate code], it changes to [Generate code] and no code is generated.

Clicking [Generate code] will change to [✓ Generate code] and generate code.

4.4.5 Component resource change

You can change the resource of the component (e.g. change from S12AD0 to S12AD1). Compatible configurations can be migrated from the current resource to the newly selected resource.

Follow the procedure below to change the resource.

(1) Right-click on a component (e.g. Config_S12AD0).

(2) Select [Change resource] from the context menu.

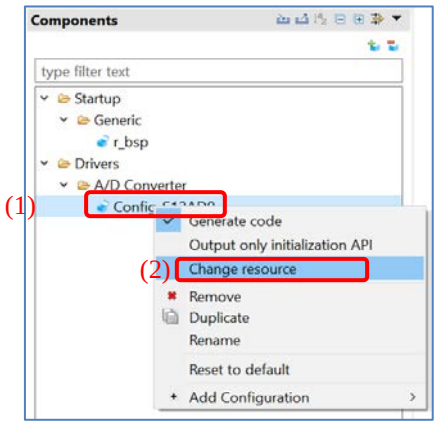

Figure 4-18 Resource Change

(3) Select a new resource in the [Resource Selection] dialog box (e.g. S12AD1).

(4) The [Next] button will be active; click on it.

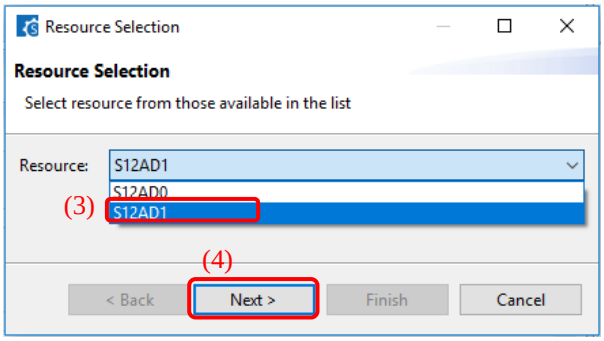

Figure 4-19 Select a New Resource

(5) The configuration information is displayed on the [Configuration setting selection] page of the [Select Resource] dialog.

(6) Check the portability of the settings.

(7) Select whether to use the listed or default settings.

(8) Click on [Finish].

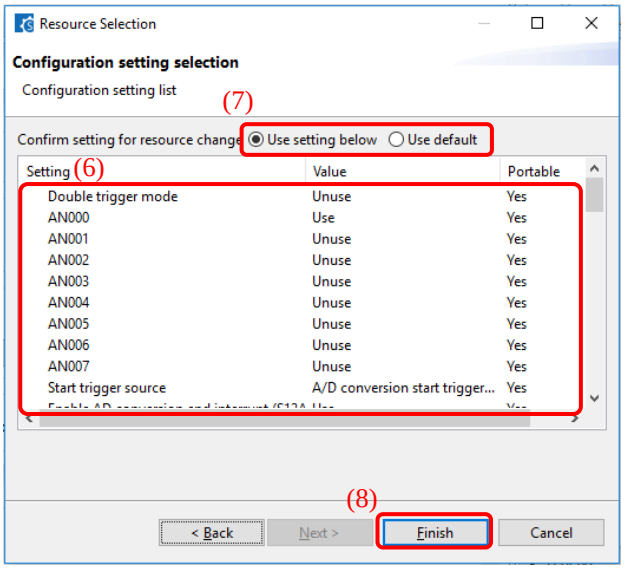

Figure 4-20 Confirm New Resource Settings

The resource is automatically changed (e.g. changed from S12ADI0 to S12ADI1).

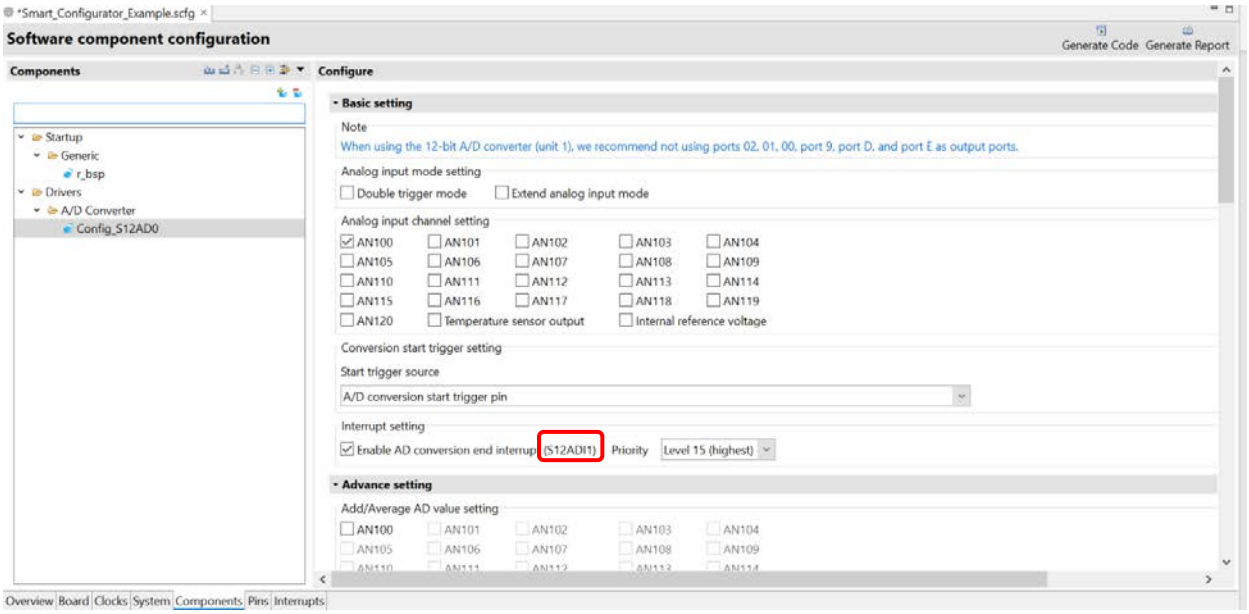

Figure 4-21 Resource Changed Automatically

To change the configuration name, follow the procedure below.

(9) Right-click on the component.

(10) Select [Rename] to rename the configuration (e.g. change Config_S12AD0 to Config_S12AD1).

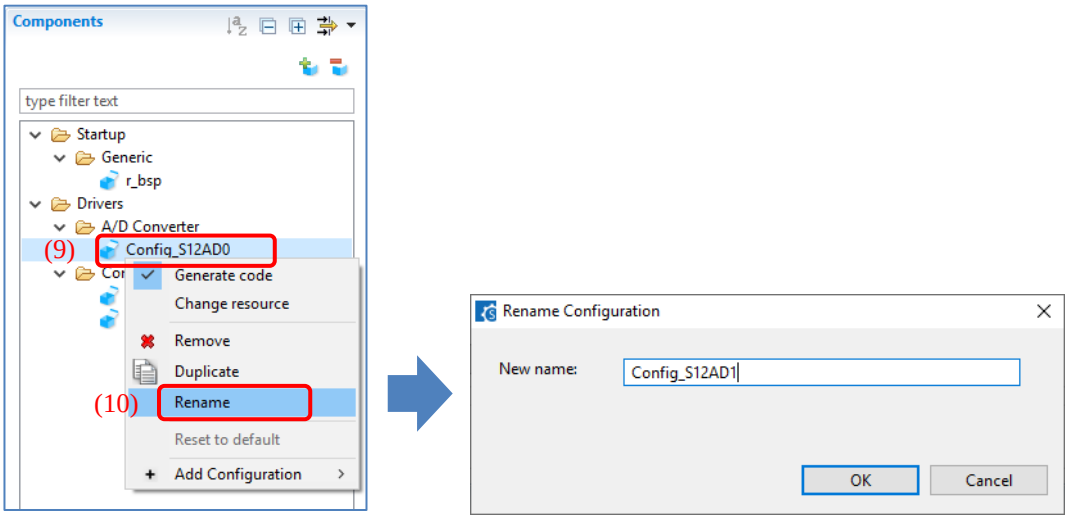

Figure 4-22 Renaming the Configuration

4.4.6 Adding FIT drivers or middleware

The following describes the procedure for adding a FIT driver or a middleware.

(1) Click on the [(Add component)] icon.

(2) Select components from the list in the [Software Component Selection] page of the [New Component] dialog box (e.g. r_ether_rx and r_qspi_smstr_rx). Two or more components can be selected by clicking with the Ctrl key pressed.

(3) Check that [Type] for the selected components is [FIT].

(4) Click on [Finish].

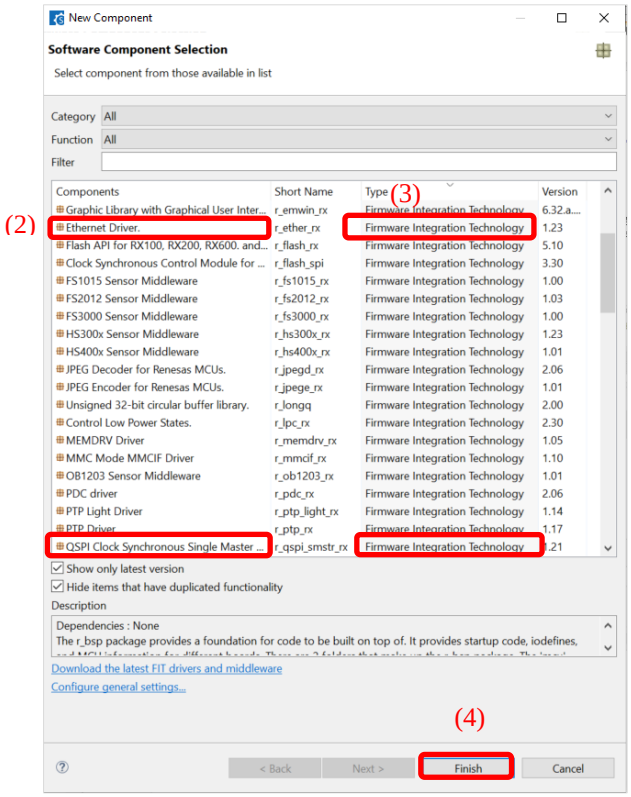

Figure 4-23 Adding FIT Modules

4.4.7 Setting of the FIT software components

The FIT drivers or middleware are available by setting configuration options.

The way of setting depends on each component.

- Set on the panel and settings will be generated to configuration file of FIT module automatically at each time of code generation action
- Set with modifying the configuration file of the FIT module

The configuration file is generated in the r_config folder after source code generation.
See the "chapter 7.1 Adding Custom Code of FIT" to set the configuration options.

In addition, some components provide pin setting on the Configure panel. Followings are examples of pin setting on Configure panel.

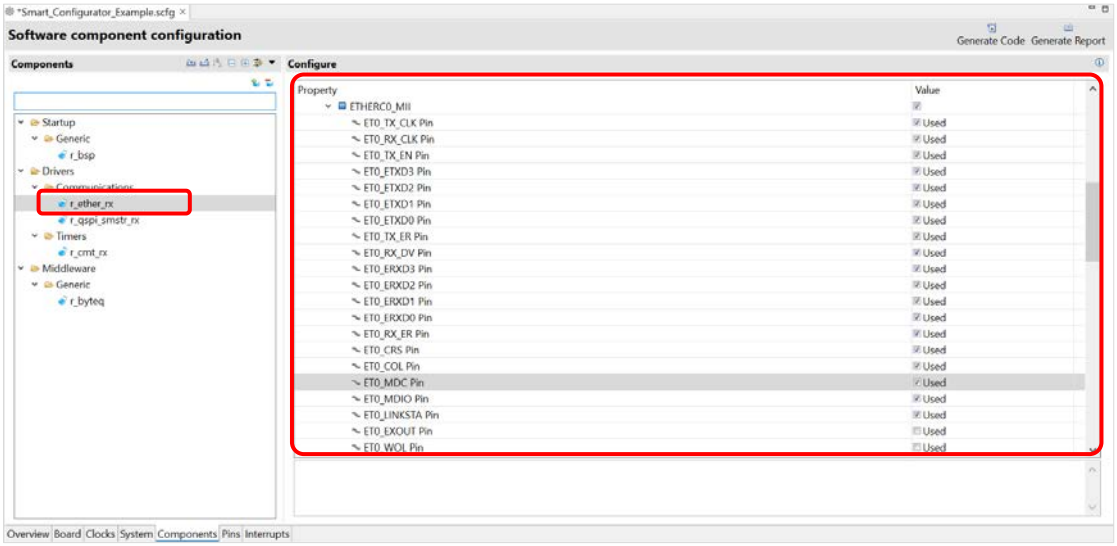

Figure 4-24 Pin setting of r_ether_rx

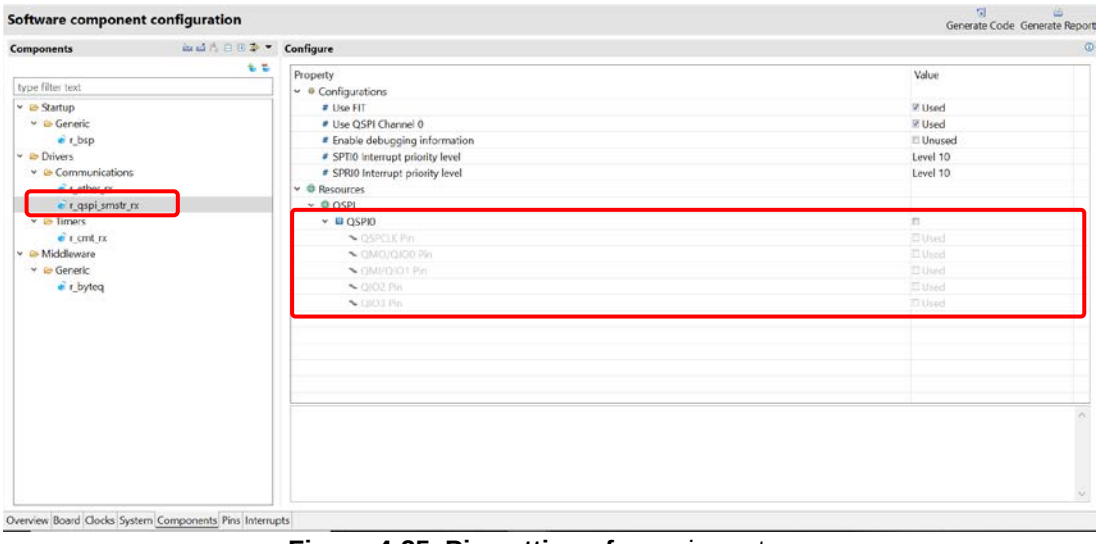

Figure 4-25 Pin setting of r_qspi_smstr_rx

4.4.8 Changing the version of the FIT software components

Change the version of the FIT software component as follows.

(1) Right click the FIT software component in the component tree.

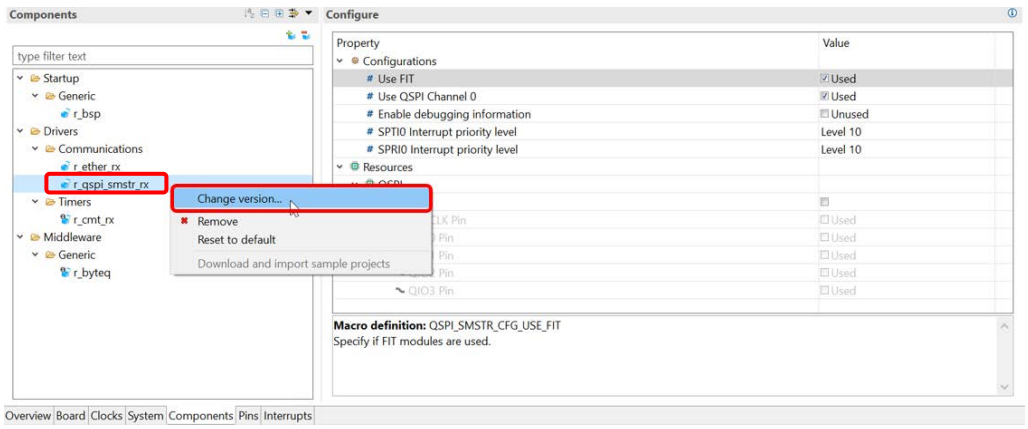

Figure 4-26 Changing the version of FIT software component

(2) Select the [Change version…] in the context menu.

(3) If supported version is selected, the [Next] button will be clickable.
Otherwise, the message "Selected version doesn't support current device or toolchain" is displayed.

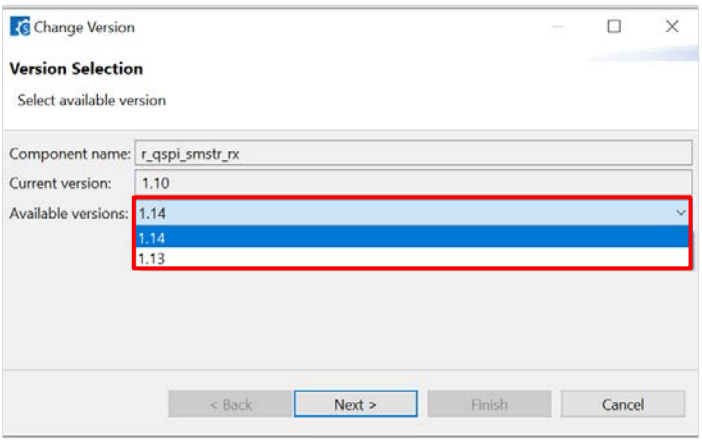

Figure 4-27 Select version of FIT software component

(4) Click the [Next] button.

(5) The setting change items are displayed.
Click the [Finish] button if there are no problems with changing the settings.

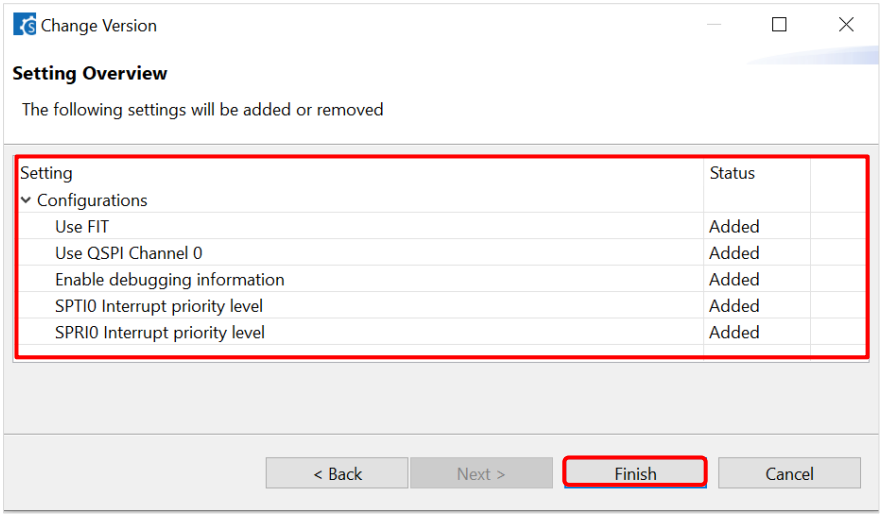

Figure 4-28 Confirm setting change item

(6) The message "Confirm to change version and proceed to generate code" is displayed.
Click [Yes] button if there are no problems.

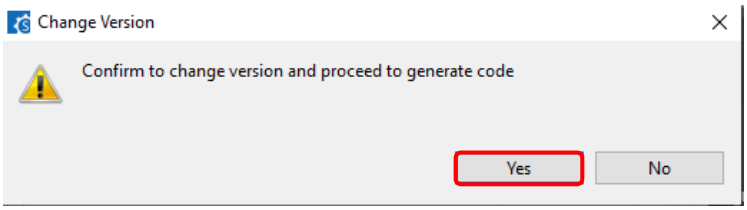

Figure 4-29 Confirm setting change item

(7) The version of software component has been changed then the source codes are generated automatically.

4.4.9 Solving the grey-out component

When a component version is not available, it will be greyed out. Follow the procedure below to fix a greyed-out component.

(1) From the component tree, right-click the greyed out component and select [Change version…]. Refer to chapter "chapter 4.4.8 Changing the version of the FIT software components" to change to an available version.

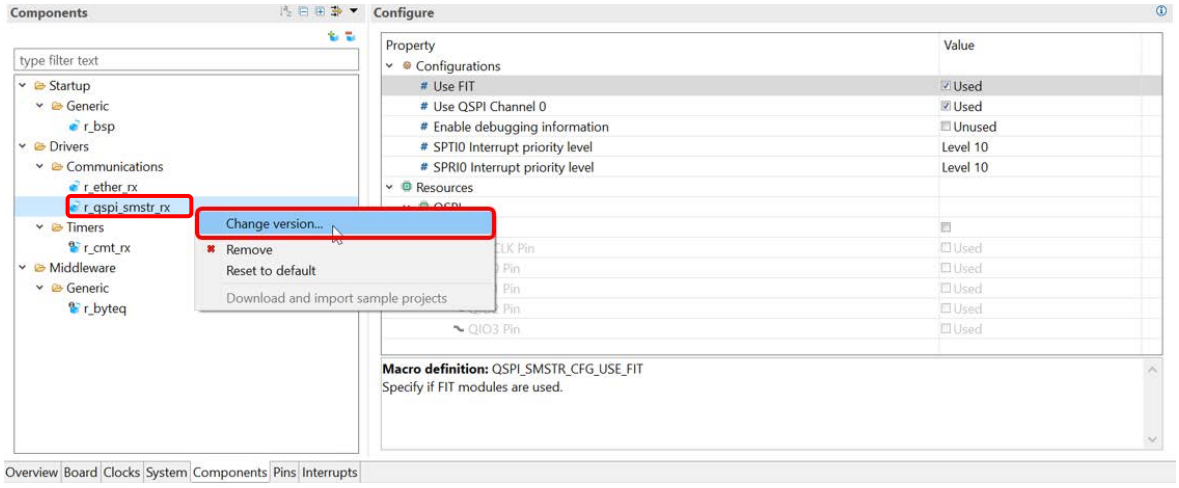

Figure 4-30 Change version of a greyed out component

(2) If there is no available version for this component, refer to chapter "chapter 3.3.1 Downloading FIT modules" to download this component from Renesas website.

4.4.10 "i" mark on FIT modules icon

If the icon of FIT driver or middleware has an "i" mark [ ], there are two meanings.

(1) Sample project of this FIT module is available

When there is sample project available for the FIT module, an "i" mark will appear to inform you.

Download the sample program files from Renesas Electronics home page.

https://www.renesas.com/fit

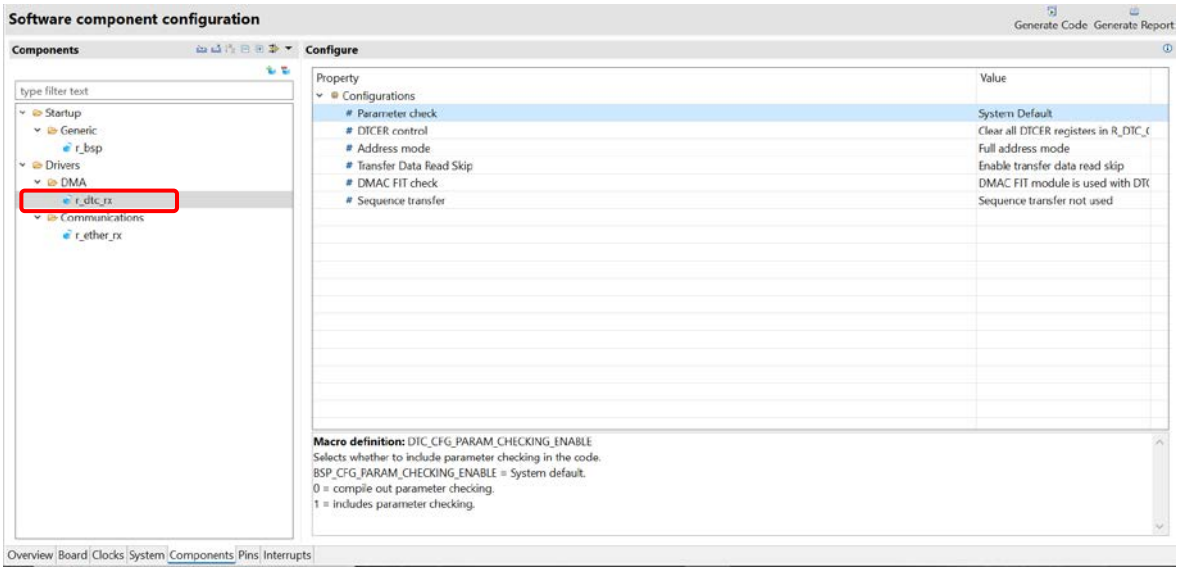

Figure 4-31 FIT driver with sample project available for download in Renesas homepage

(2) Higher version of this FIT module is available in the computer

When the project is using an older version of FIT module, an "i" mark will appear if a higher version is available in the computer. A message will also appear when you mouse-over the FIT module.

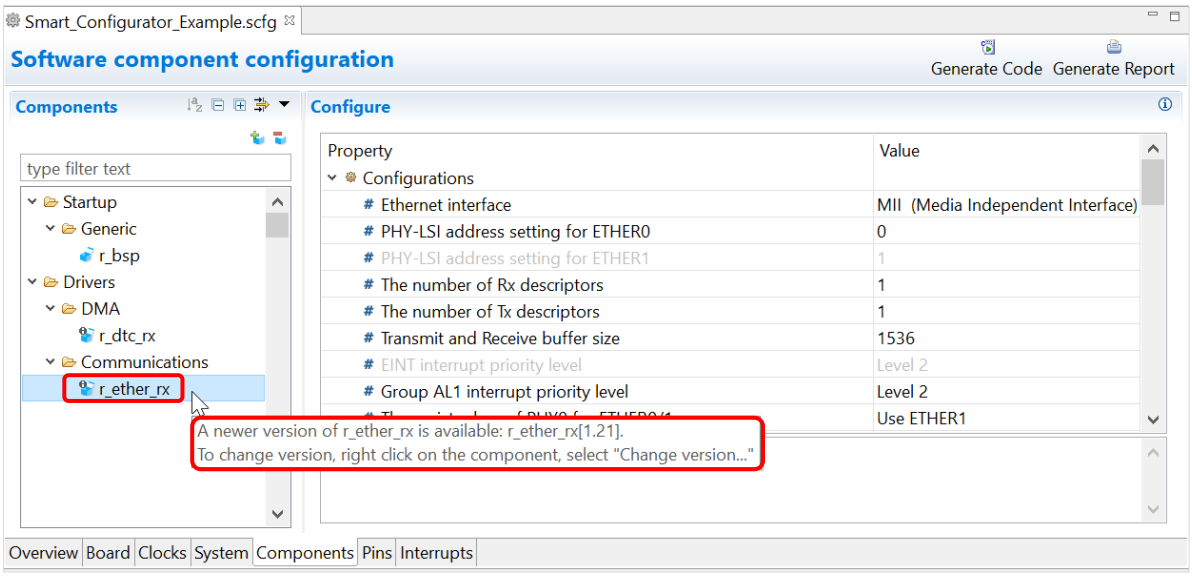

Figure 4-32 A newer version of FIT module is available in the computer

You can right-click the FIT module to change from current version to a higher version. Please refer to chapter "4.4.8 Changing the version of the FIT software components".

4.4.11 Configure Analog Front End component

The RX23E-A group microcontrollers are equipped with an analog front end (AFE) that can measure temperature, pressure, flow, and weight with less than 0.1% precision without calibration, making it ideal for high-precision sensing, test and measurement equipment.

When creating project for RX23E-A, you can use the AFE configuration tool for:

- Easy setting AFE on GUI
- Easy checking pins confliction
- Easy checking analog multiplexer connection

This chapter will describe how to use analog multiplexer connection:

(1) In RX23E-A project, open smart configurator, select [Components] tab and add new component "Analog Front End" and "Continuous Scan Mode DSAD"

(2) Select [Config_DSAD0] from the Components Tree. Perform setting as following:

- Analog input channel setting: enable channel 0
- [Channel setting] > [Channel 0] > Positive input signal: AIN1
- [Channel setting] > [Channel 0] > Negative input signal: AIN3

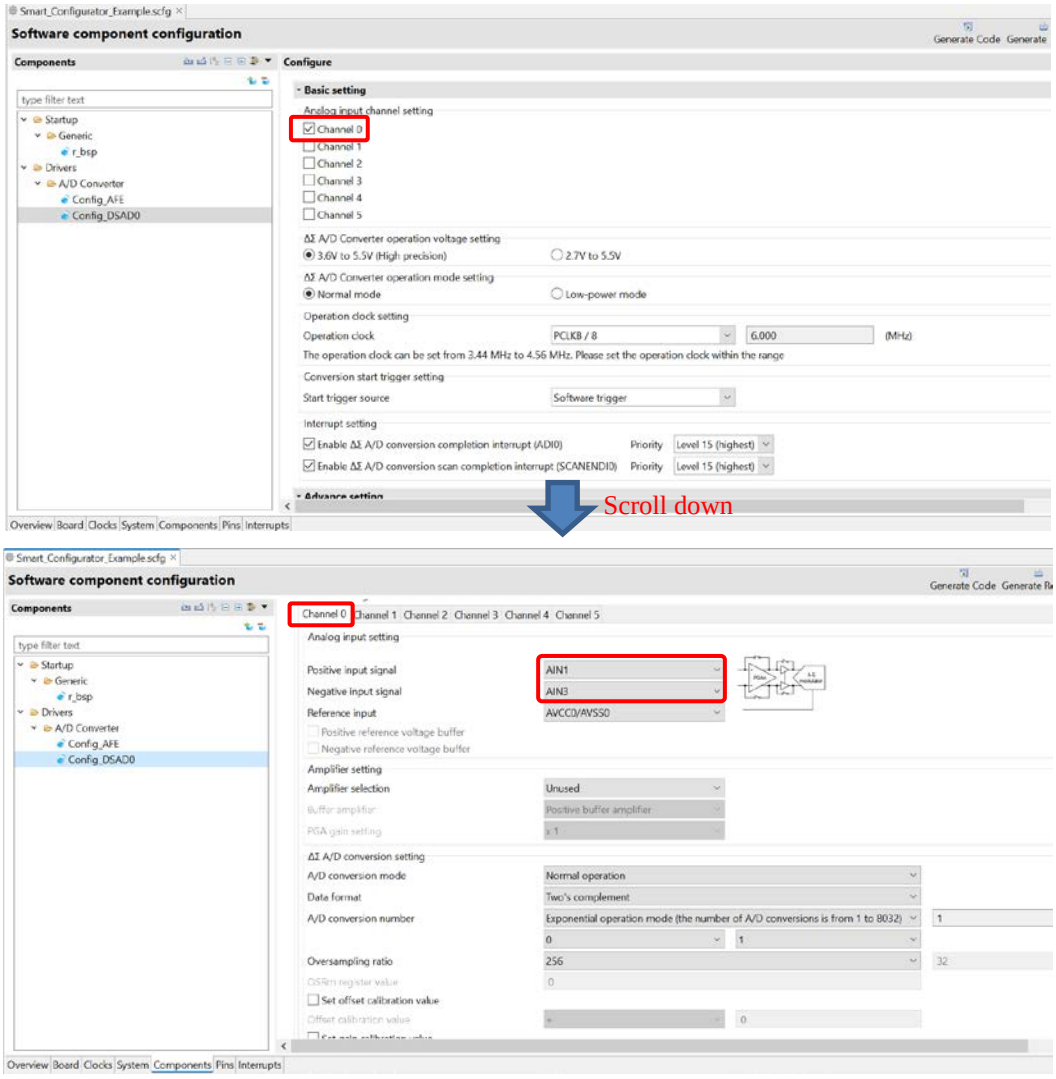

Figure 4-33 Config_DSAD0 Setting

(3) Select [Config_AFE] from the Components Tree. In the [AFE setting] tab, change the [Bias output setting] as follows:

- Enable bias voltage output: checked.
- AIN1 pin output: checked
- AIN3 pin output: checked

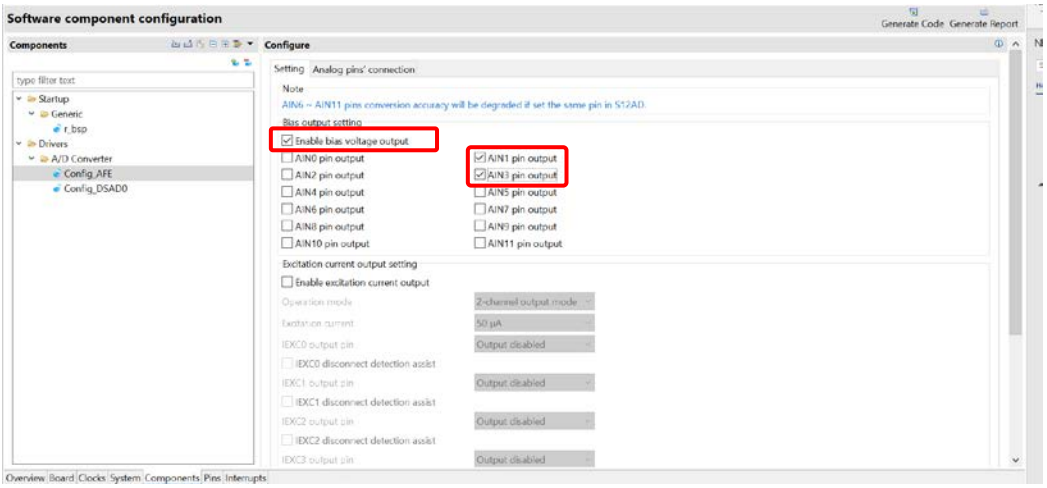

Figure 4-34 Config_AFE setting

(4) Select [Analog pin's connection] tab, you can see the block diagram of the AFE multiplexed pin connection. The active connection of analog multiplexer is highlighted. So, you can check the analog multiplexer connection easily and confirm the configuration.

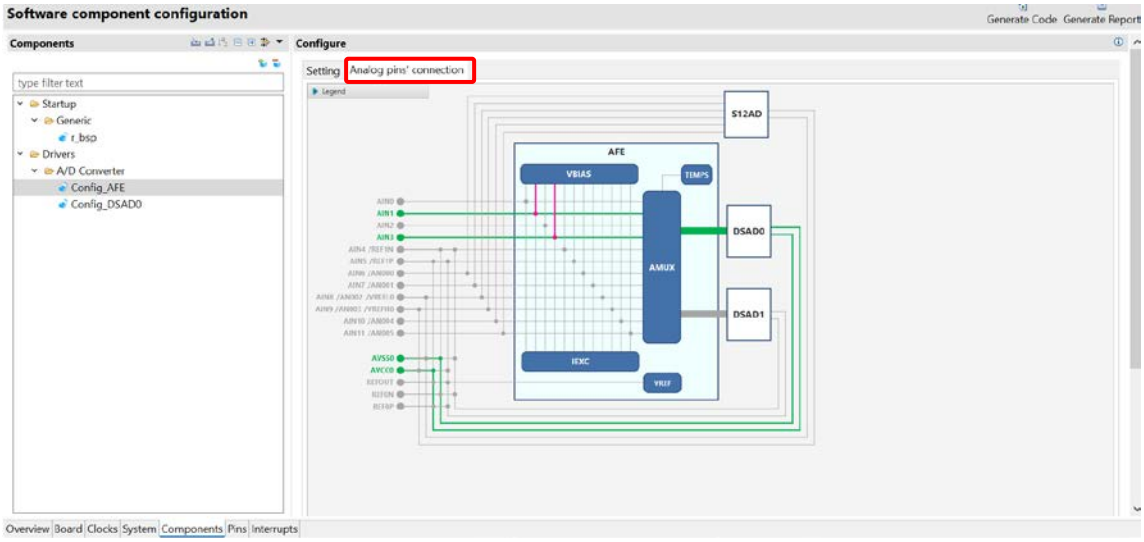

Figure 4-35 Block diagram of the AFE multiplexed pin connection

4.4.12 Configure Motor Component

Motor Driver Generator is a utility tool to generate drivers for all peripheral functions used for motor control from one GUI setting.

Note: The supported devices are RX13T, RX23T, RX24T, RX24U, RX26T, RX66T, RX72T, and RX72M.

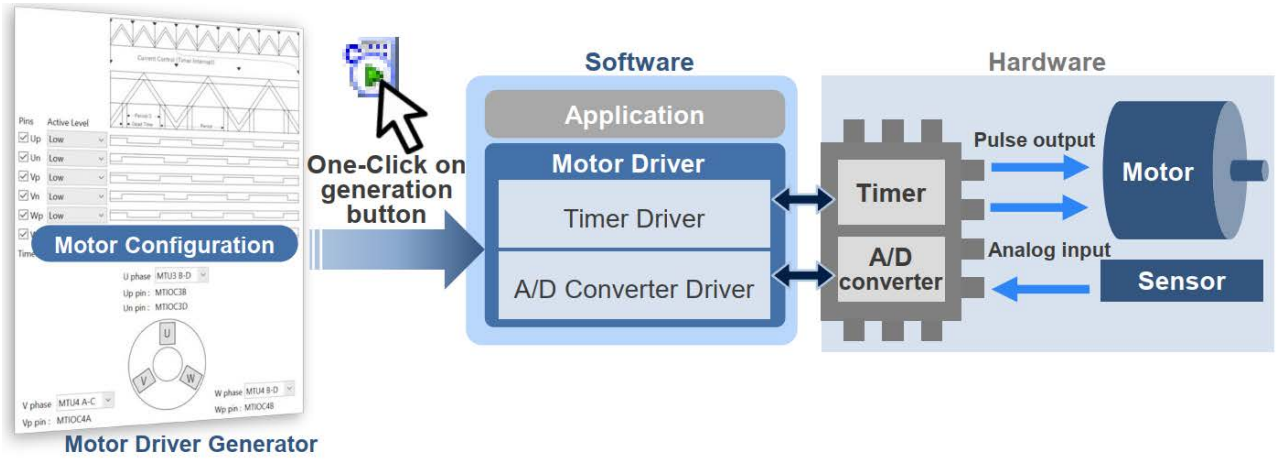

Figure 4-36 Motor Driver Generator

This chapter will describe how to use Motor Driver Generator:

(1) In the project of supported device (for e.g. RX24T), open Smart Configurator, select [Components] tab and add new component "Motor". In the [New Component] dialog, select the Motor type as you wish and click [Finish] button.

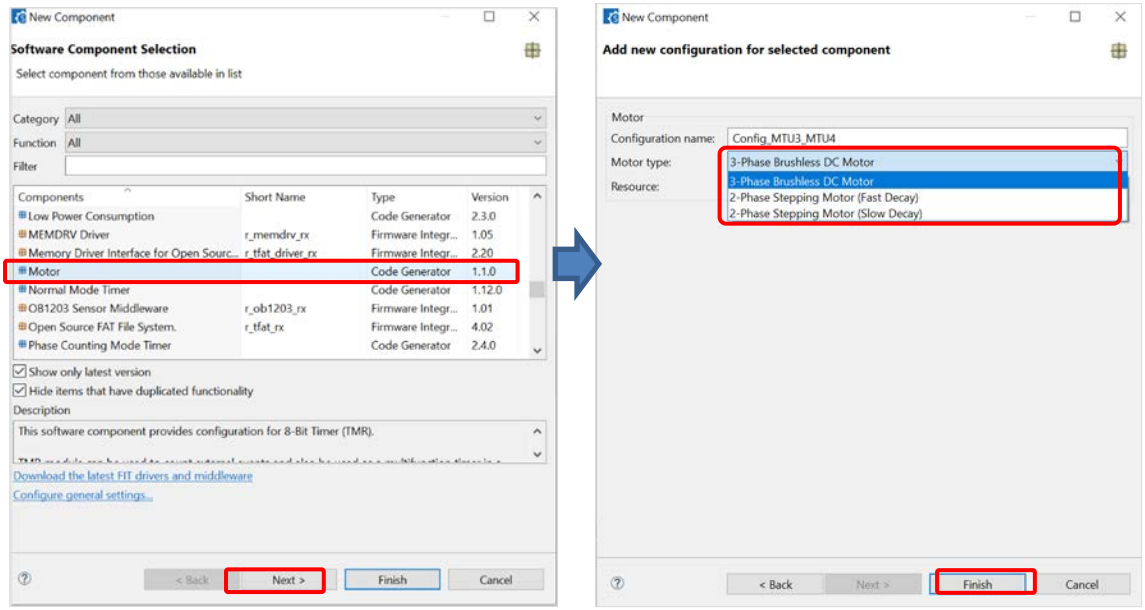

Figure 4-37 Add Motor component

(2) Select "Config_MTU3_MTU4" in the component tree, on the Configure panel, select the [Timer Setting] tab. In this tab, you will be able to use the GUI for Timer driver setting, including: Period Setting, Output Level Setting, Output Pin Select and Timer Interrupt Setting.

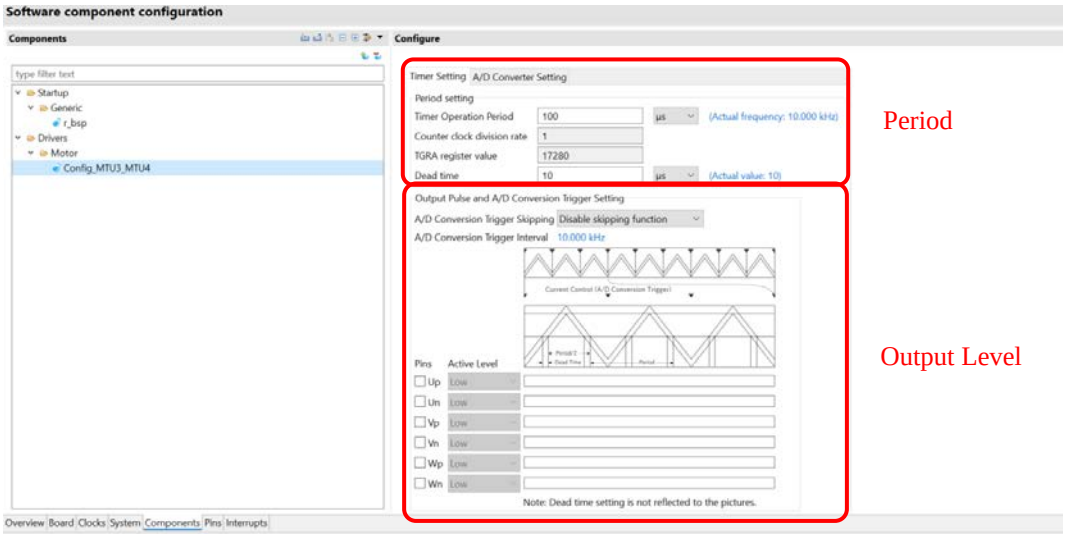

Figure 4-38 Timer Driver Setting (1)

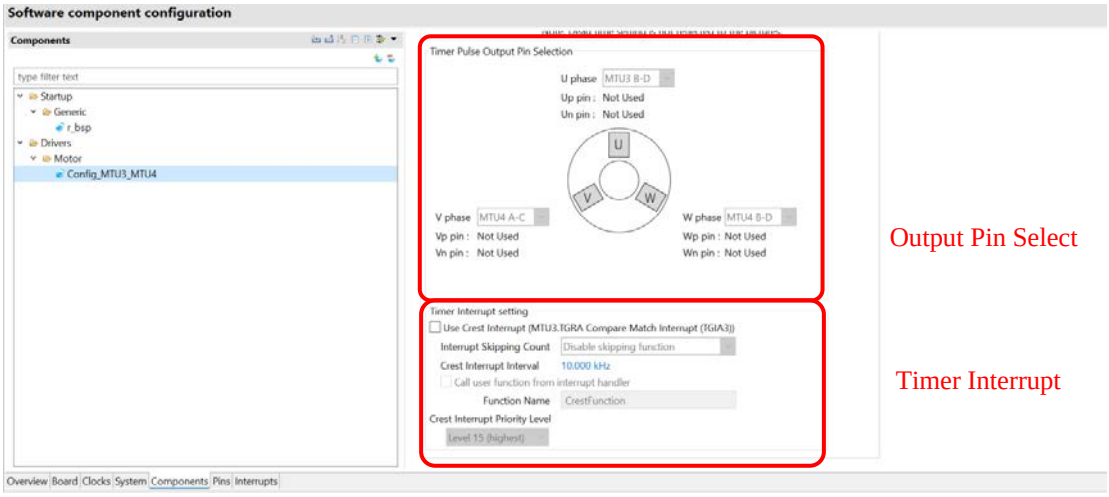

Figure 4-39 Timer Driver Setting (2)

(3) Select the [A/D Converter Setting] tab. In this tab, you will be able to use the GUI for A/D Converter driver setting, including: Analog Pin Select, A/D Interrupt Setting.

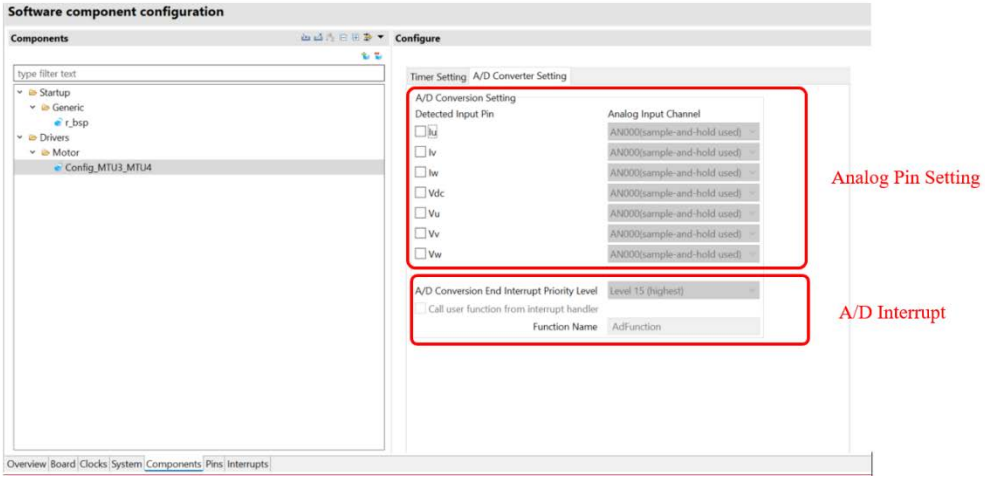

Figure 4-40 A/D Converter Driver Setting

(4) GPT peripheral is supported in some devices (for e.g., RX26T). Add new component "Motor". In the [New
Component] dialog, select the [Triangle_GPT] or [GPT0_GPT1_GPT2], [GPT4_GPT5_GPT6]resource and click
[Finish] button.

Note: The [GPT0_GPT1_GPT2] and [GPT4_GPT5_GPT6] resources are exclusively designed for GPT Complementary Mode, which is accessible only in RX26T

The [Triangle_GPT] resource is intended for GPT Triangle PWM Mode. With this resource, users can modify GPT channel configurations for both Master and Slave channels, utilizing the currently available GPT channels. The [Triangle_GPT] resource is available on RX24T, RX24U, RX26T, RX66T, RX72M, RX72T

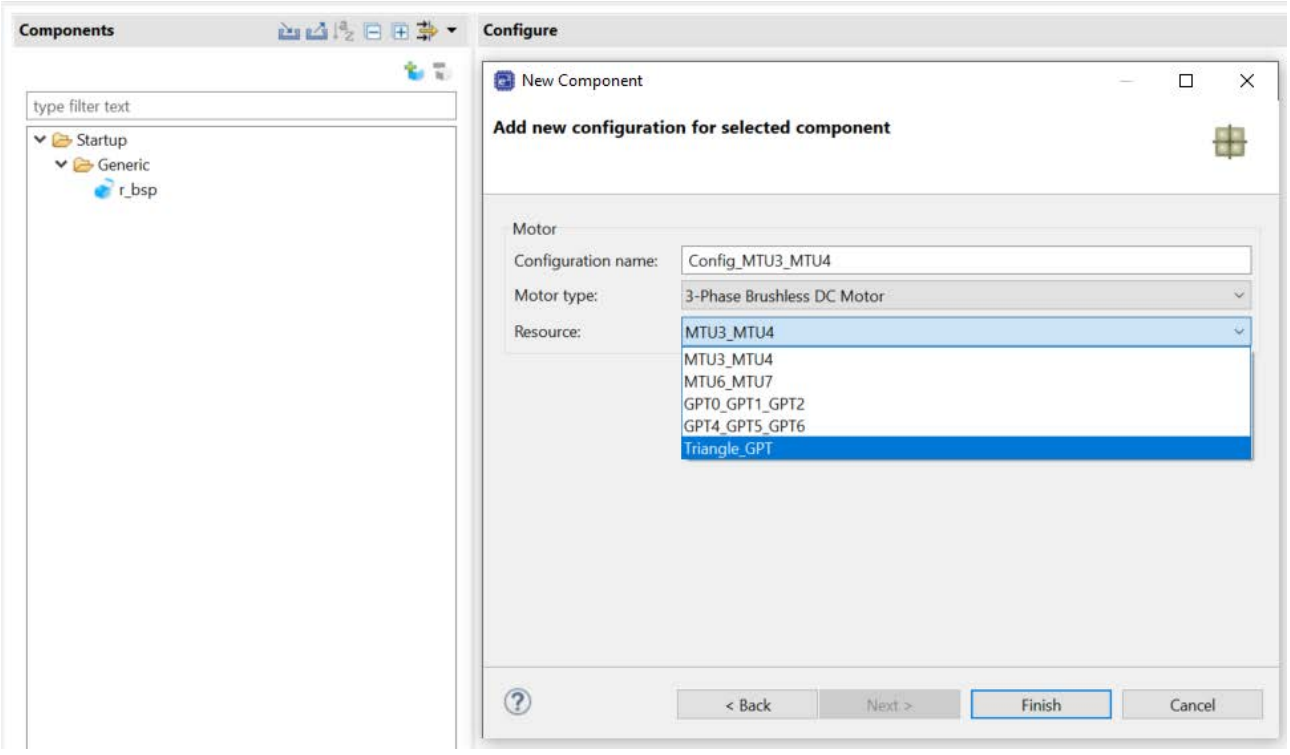

Figure 4-41 Select MOTOR resource

(5) GPT resource device has some differences between MTU resources

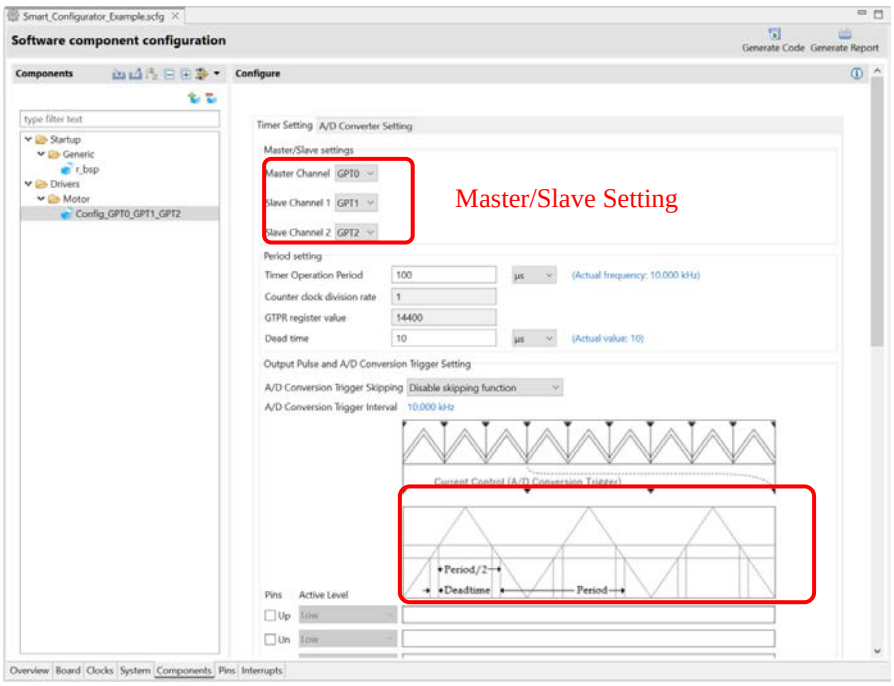

Figure 4-42 Timer Driver Setting(1)

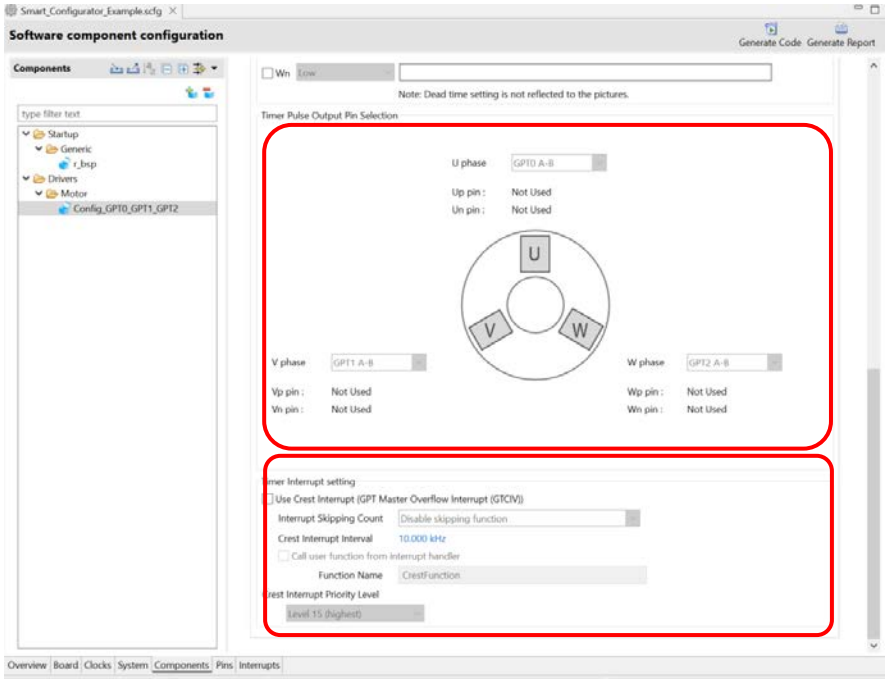

Figure 4-43 Timer Driver Setting (2)

4.4.13 Configure general setting of component

The general setting of the component, such as code generation component settings, FIT(RX) component settings, dependency settings and location settings, can be configured inside the [Preferences] dialog.

If you want to change the settings, click the [Configure general settings...] link on the [Software Component Selection] page displayed in the [New Component] dialog (Figure 4-11), and the [Preferences] dialog will be displayed, as shown in Figure 4-44.

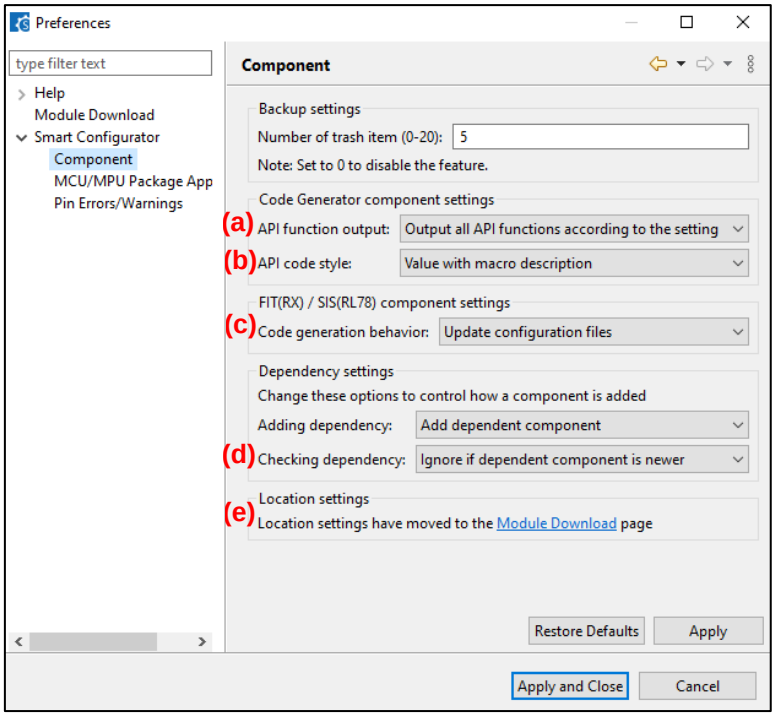

Figure 4-44 Configure General Setting of Component

(a) The API function output has two options: "Output all API functions according to the setting" and "Output only initialization API function". "Output all API functions according to the setting" is the default selection.

If "Output all API functions according to the setting" is selected, all API functions will be generated.

If "Output only initialization API function" is selected, only initialization API function will be generated. (Only void R_{ConfigurationName}_Create (void), void R_{ConfigurationName}_Create_UserInit (void) in *.h *, *c * are generated out.)

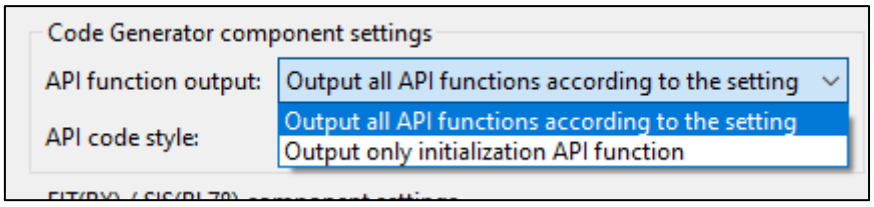

Figure 4-45 Updates of "API function output"

Output only initialization API feature is supported for individual configuration (Code Generator component). For using this feature, please right-click the selected component and select the "Output only initialization API" from the context menu.

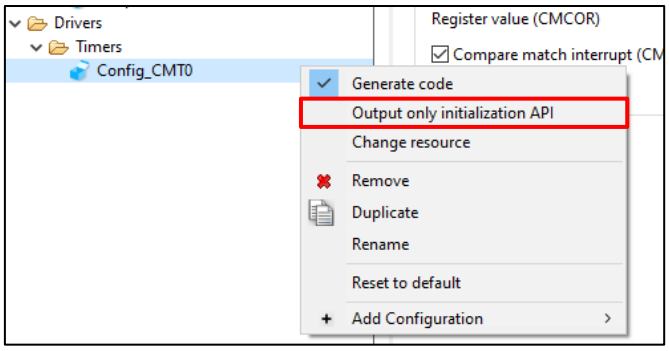

Figure 4-46 Output only initialization API

(b) The API code style has two options: "Value with macro description" and "Value without macro description (raw HEX)". "Value with macro description" is the default selection.

If "Value with macro description" is selected, all API with macro description will be generated.

If "Value without macro description (raw HEX)" is selected, code with HEX value will be generated.

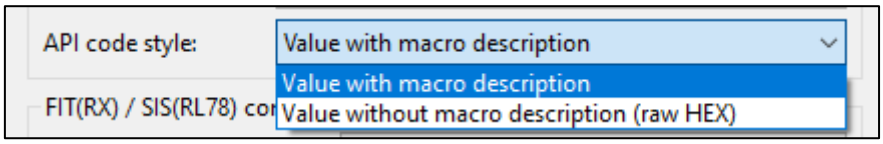

Figure 4-47 Updates of "API code style"

(c) The code generation behavior has two options: "Update configuration files" and "Re-generate all component files". "Update configuration files" is the default selection.

If "Update configuration files" is selected and generate code, Smart Configurator will check whether the files are existing inside the user project. If the file exists, the file will not be overwritten. However, configuration files (e.g., xxx_config.h) will still be refreshed when code is generated.

If "Re-generate all component files" is selected and generate code, Smart Configurator does not check the existence of the file and the file will always be overwritten.

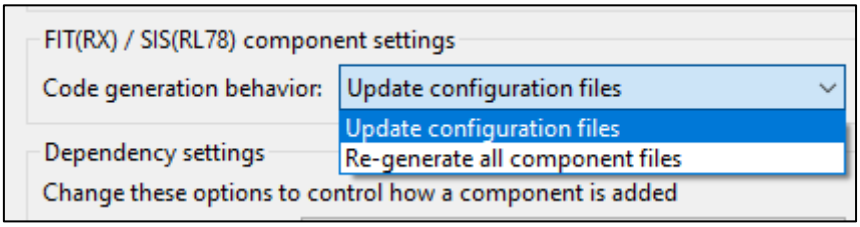

Figure 4-48 Updates of "Code generation behavior"

(d) Checking dependency has three options: "Do not check for dependent component", "Ignore if dependent component is newer" and "Strict check for dependent component". "Ignore if dependent component is newer" is the default selection.

If the version of the module and its dependency do not match, a warning message with code W04020011 will be displayed. If the user checks the revision history of the module and its dependencies and determines that there is no need to change the module being used, they can ignore this warning.

To clear this warning, select "Do not check for dependent components" in the "Checking dependency" list box in component preferences and then click "Apply".

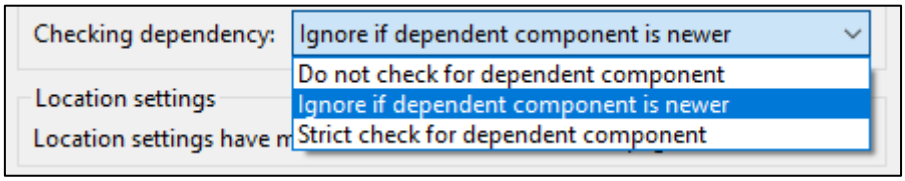

Figure 4-49 Updates of "Checking dependency"

(e) If you want to change the location of components, click [Module Download] link, then find [Location (RX)], click [Browse…] and select another folder.

Figure 4-50 Updates of "Location settings"

4.4.14 Export configuration of component

To export the current configuration of a component, click the [Export Configuration] button on the [Components] tab page. The configuration will be saved as an *.xml file.

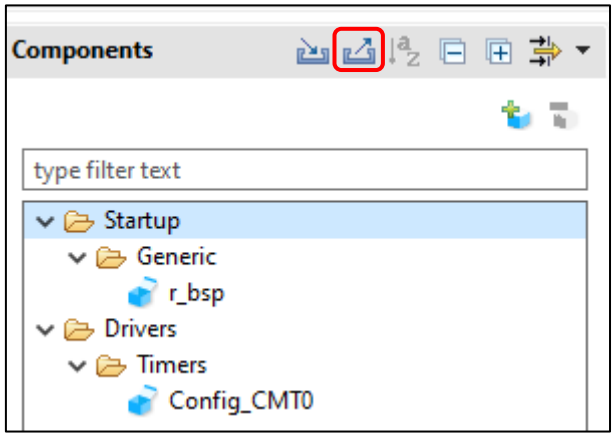

Figure 4-51 Export Configuration (xml format)

4.4.15 Import configuration of component

To import the configuration of a component, click the [Import Configuration] button and select the exported XML file.

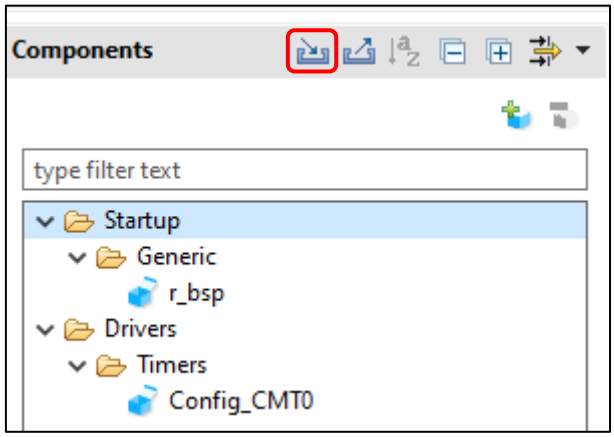

Figure 4-52 Import Configuration (xml format)

4.5 Pin settings

The [Pins] page is used for allocating pin functions. You can switch the display by clicking on the [Pin Function] and [Pin Number] tabs. The [Pin Function] list shows the pin functions for each of the peripheral functions, and the [Pin Number] list shows all pins in order of pin number.

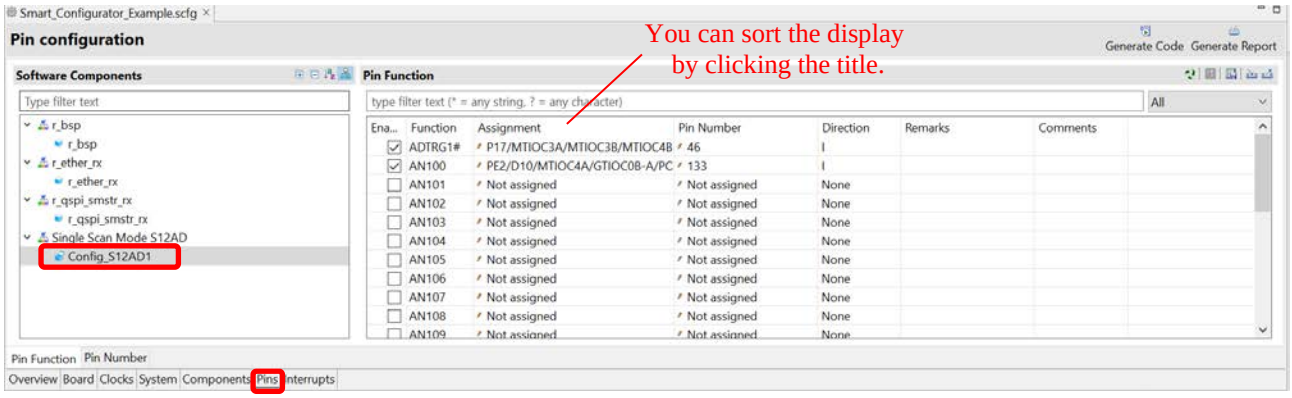

Figure 4-53 [Pins] Page ([Pin Function])

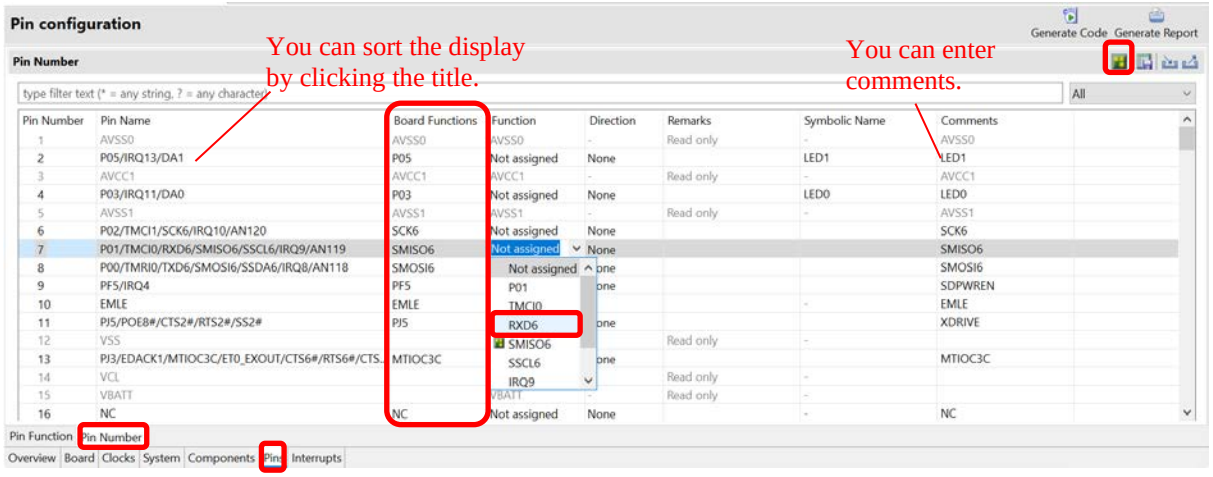

Figure 4-54 [Pins] Page ([Pin Number])

When you select a board on the [Board] page, the initial pin setting information of the board is displayed in [Default Function]. In addition, the [ ] icon displayed in the [Function] selection list indicates the initial pin function of the board.

4.5.1 Assign pins to resources

In the Pins page, assign pin to the resource used by the component. Pin assignment can be done in either [Pin Function] list or [Pin Number] list.

The procedure for pin assignment in the [Pin Function] list is described below.

(1) Click on [(Show by Hardware Resource or Software Components)] to switch to the software component view.

(2) Select the target software component (e.g. Config_S12AD1).

(3) Click the [Enabled] header to sort by pins used.

(4) Pin assignment is performed with the [Assignment], [Pin Number] column or [(Next group of pins for the selected resource)] button.

(a) Click [Assignment] or [Pin Number] and assign a terminal from the list (e.g. change from P17 to P13).

(b) Click the [(Next group of pins for the selected resource)] button and change the pin assignment. Each time you click, the pin with the function switches.

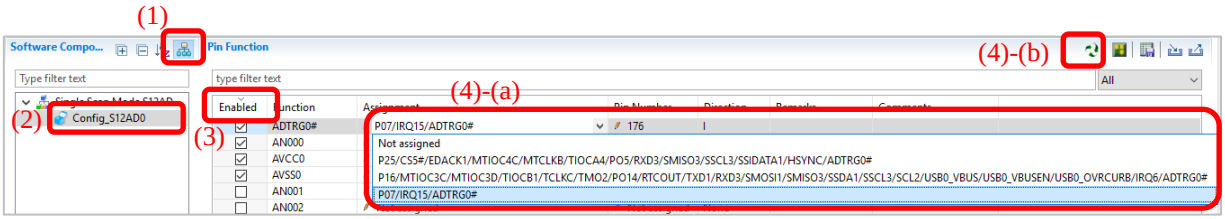

Figure 4-55 Pin Assignments in the [Pin function] List

When the component is set, the check box in the [Enabled] column is checked. Pin assignment is possible even when the component is not set. If pin assignment is done without component being set, we will display "No component is using this pin" in the [Remarks] column.

4.5.2 Pin setting using MCU/MPU package

Follow the procedure below to assign pins in the MCU/MPU Package view.

(1) Zoom in to the view by clicking the [(Zoom in)] button or scrolling the view with the mouse wheel.

(2) Right-click on the target pin.

(3) Select the signal to be assigned to the pin.

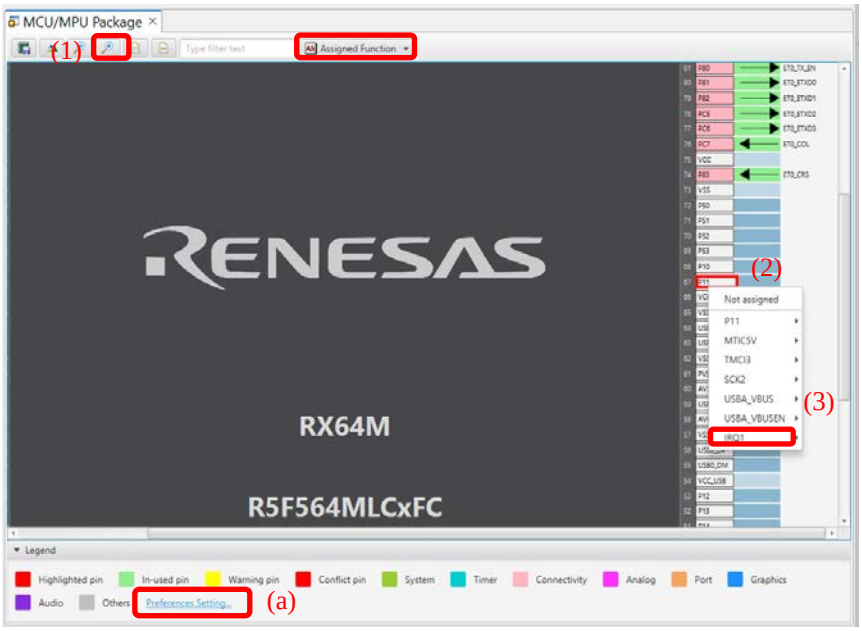

Figure 4-56 Assigning Pins Using the MCU/MPU Package View

(a) The color of the pins can be customized through [Preference Setting...].

4.5.3 Show pin number from pin functions

You can go to the pin number associated with a pin function.

Follow the procedure below to jump to pin number from a pin function.

(1) In the [Pin Function] tab, right click on a Pin Function to open the pop up menu.

(2) Select "Jump to Pin Number"

(3) The [Pin Number] tab is opened with a Pin Number being selected. This is the pin number of the pin function

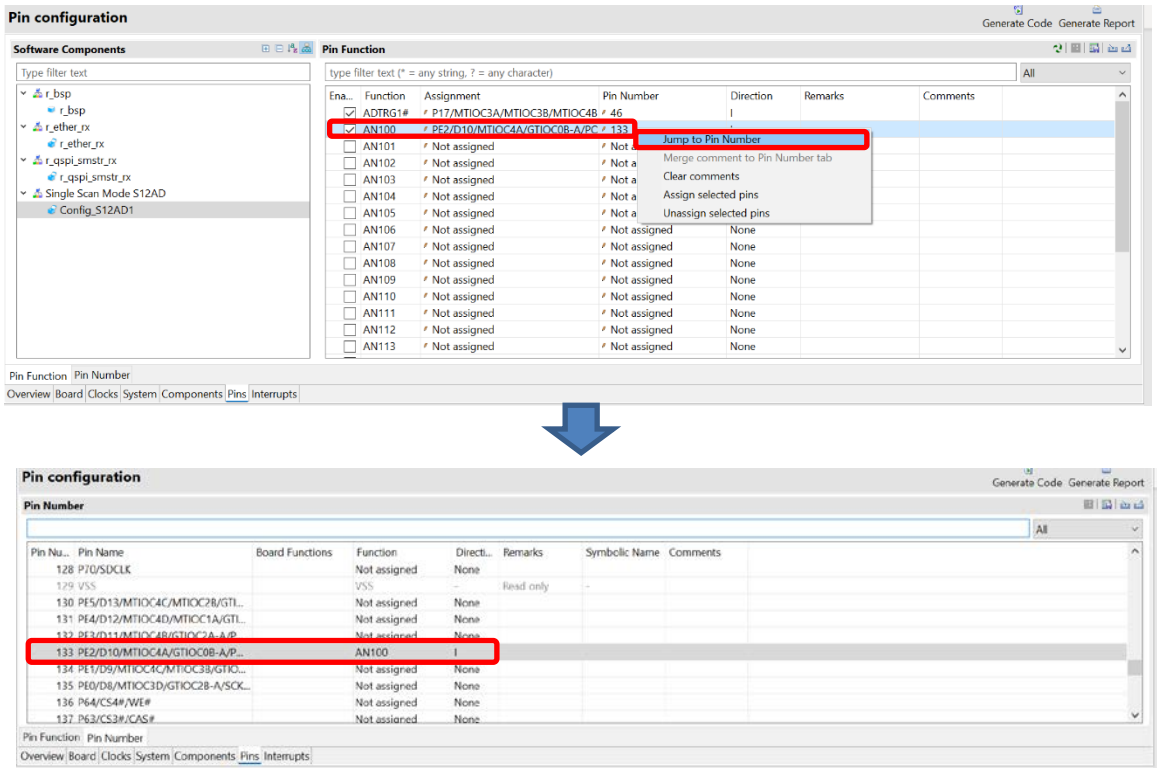

Figure 4-57 Jump to pin number

4.5.4 Export pin settings

You can export pin assignment settings in XML format. Exported files can be imported into projects of the same device family. Follow the procedure below to export the pin settings.

(1) Click on the [(Export board setting)] button on the [Pins] page.

(2) In the [Export] dialog, enter the file name to export.

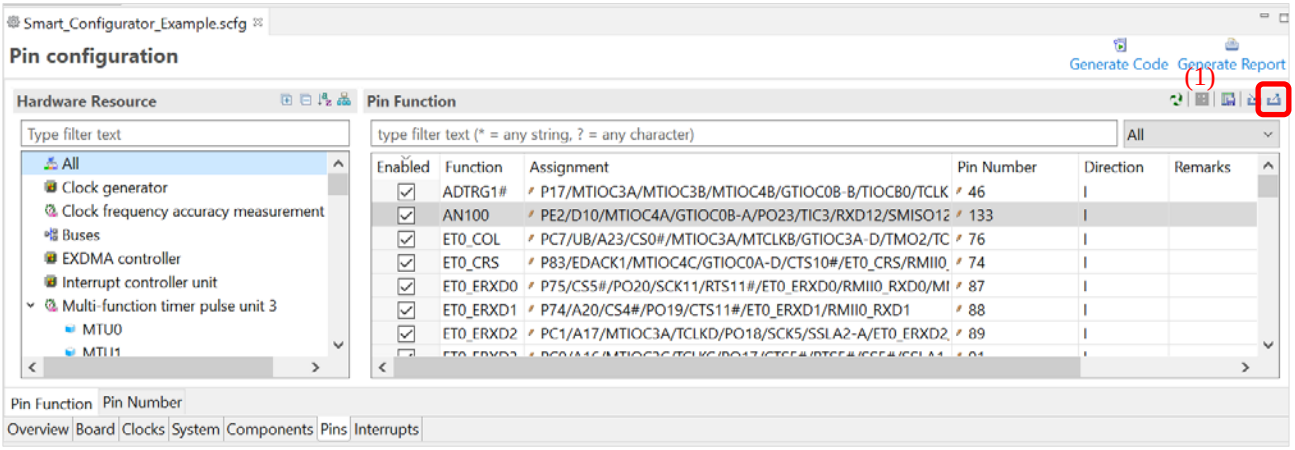

Figure 4-58 Export Pin Settings (XML format)

The Smart Configurator can also export the pin settings to a CSV file. Click the [(Save the list to .csv file)] button on the [Pins] page.

4.5.5 Import pin settings

You can import XML format files including pin assignment settings. When you import a file, the terminal assignment is reflected. Follow the procedure below to import the pin settings.

(1) Click on the (Import board setting)] button on the [Pins] page.

(2) In the [Import] dialog, enter the file name to import.

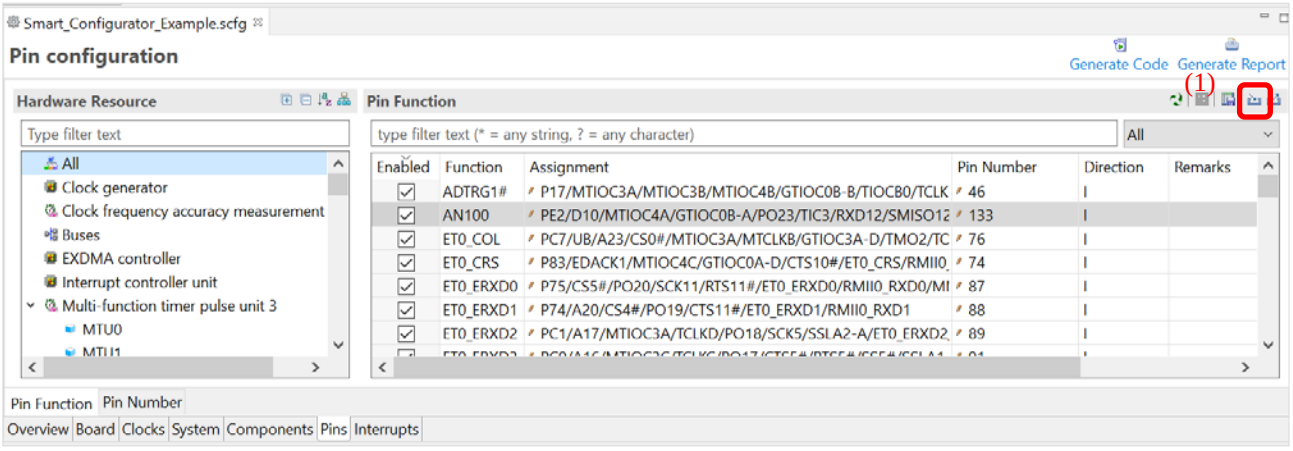

Figure 4-59 Import Pin Settings (XML format)

4.5.6 Pin setting using board pin configuration information

You can set the initial pin configuration according to the Renesas board that you selected to use. You can check the board that selected to use in [Board] tabbed page.

The following describes the procedure for collective setting of pins.

(1) Select [Board] in the MCU/MPU Package. (The initial pin configuration of the board can be referred.)

(2) Open the [Pin Configuration] page and click the [(Assign default board pins)] button.

(3) When [Assign default board pins] dialog opens, click [Select all].

(4) Click [OK].

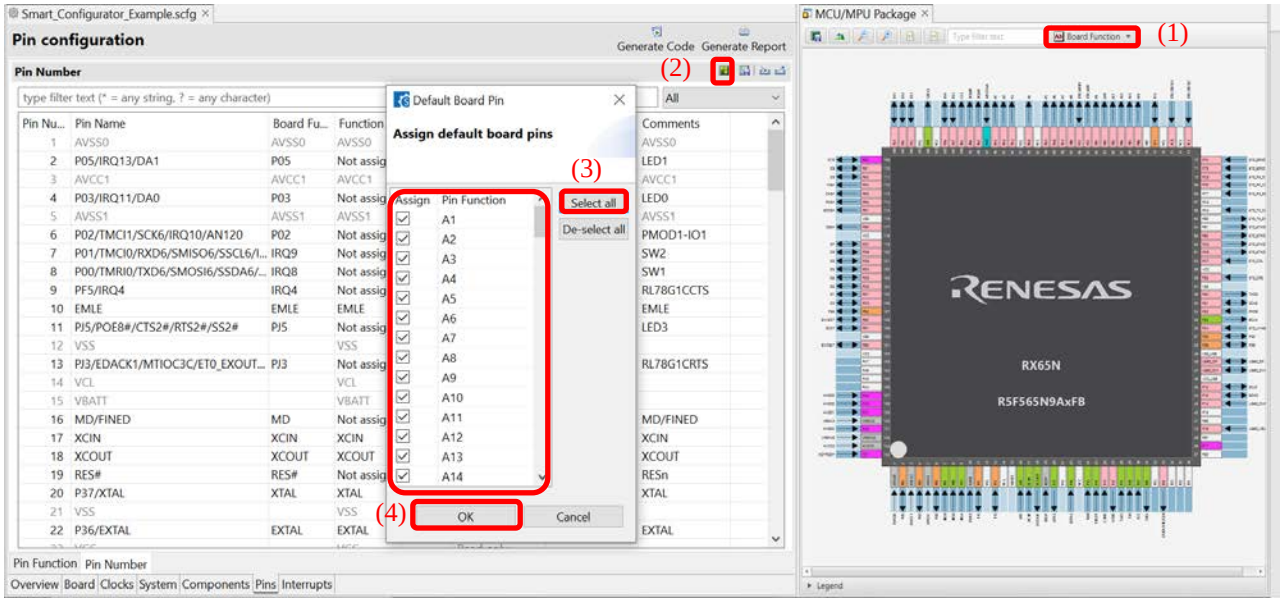

Figure 4-60 Setting for Initial Pin Configuration

If you do not set pin settings all at once, specify them individually in procedure (3).

4.5.7 Pin filter feature

By specifying the filter range on the [Pin Function] tab and [Pin Number] tab on the [Pins] page, you can refer to it more easily.

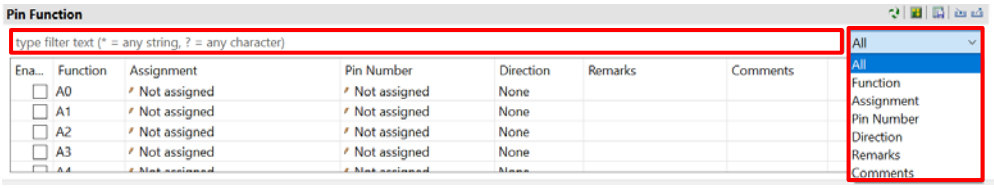

Figure 4-61 Filter for [Pin Function] Tab

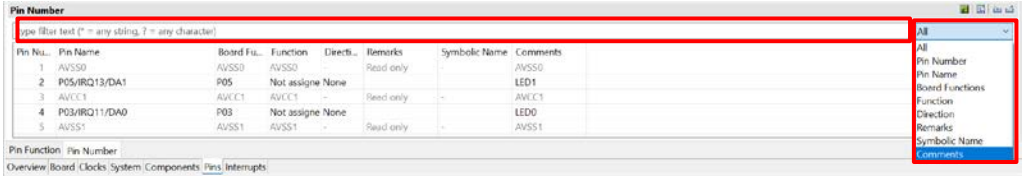

Figure 4-62 Filter for [Pin Number] Tab

4.5.8 Pin Errors/Warnings setting

You can control how pin problem is displayed on Configuration Problems view by using the Pin Errors/Warnings setting. If you want to control it, on the [New Component] dialog, click the [Configure general settings...] link to display the [Preferences] dialog. Then select [Smart Configurator] > [Pin Errors/Warnings] and use the combo boxes to change the errors/warning setting.

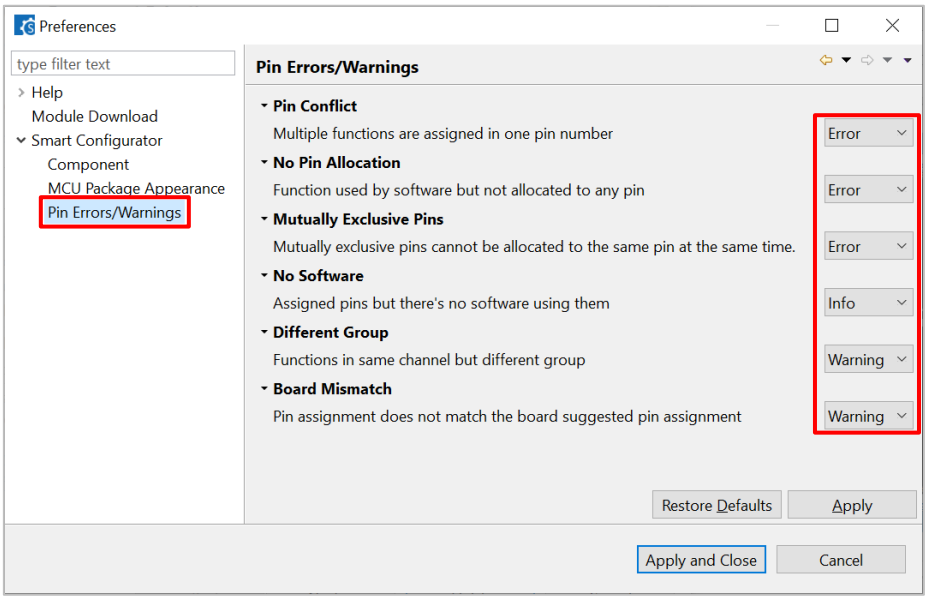

Figure 4-63 Pin Errors/Warnings settings at Preferences

Example: Change "No Software" setting from "Info" to "Error"

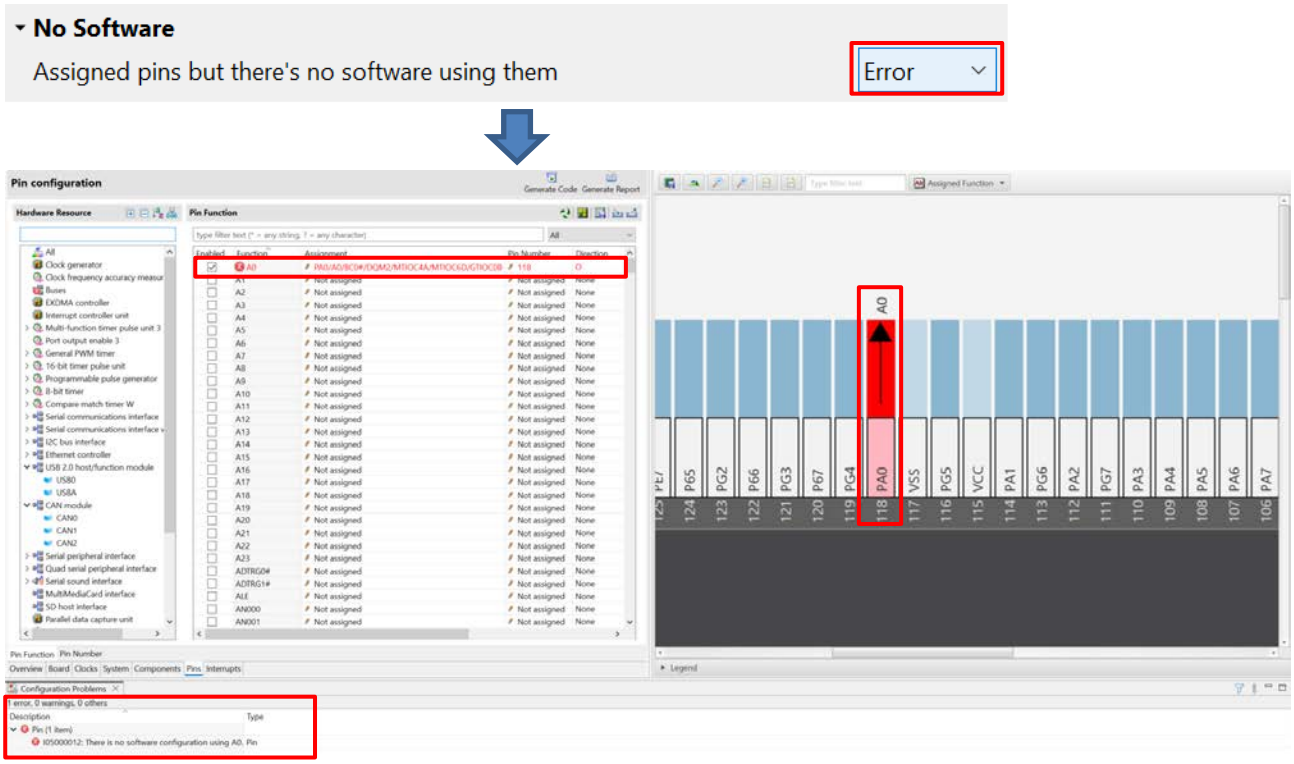

Figure 4-64 Change "No Software" setting from "Info" to "Error"

4.5.9 Symbolic name setting

[Symbolic Name] is an attribute of pins and can be found in [Pin Number] page and [MCU/MPU Package] page. It allows users to utilize their own symbols. The use of symbolic names in the user's application allows the source code to remain unchanged even when the MCU is changed, and pin assignments remapped. When a symbolic name is entered into the Pin page or the MCU/MPU Package view for any port pin, a macro definition will be generated in the Pin.h file

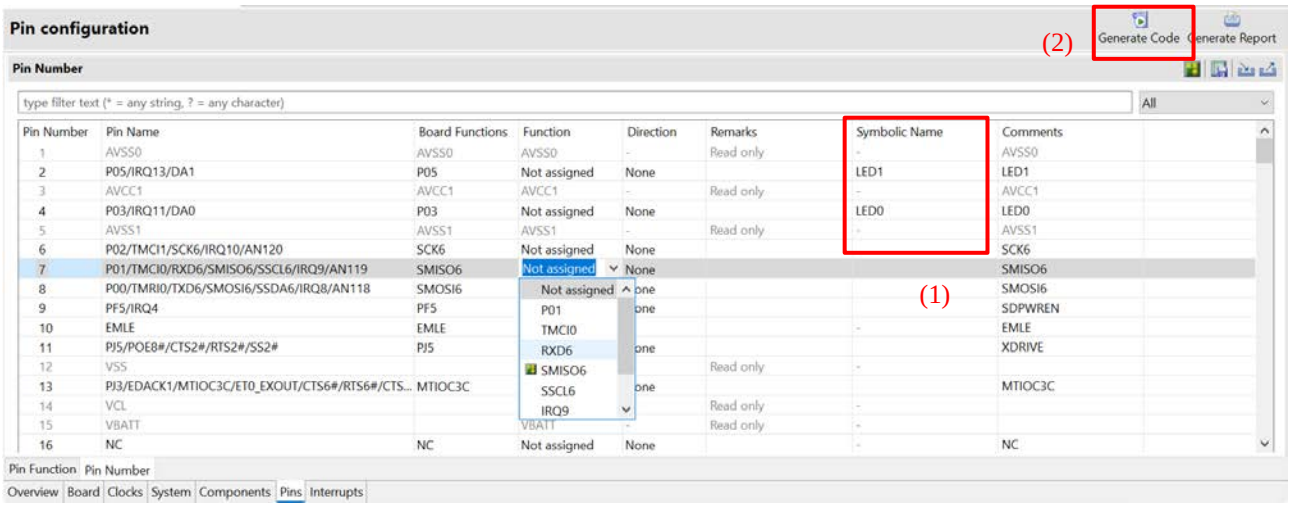

Figure 4-65 pin setting of symbolic name

After generating code check at pin.h file.

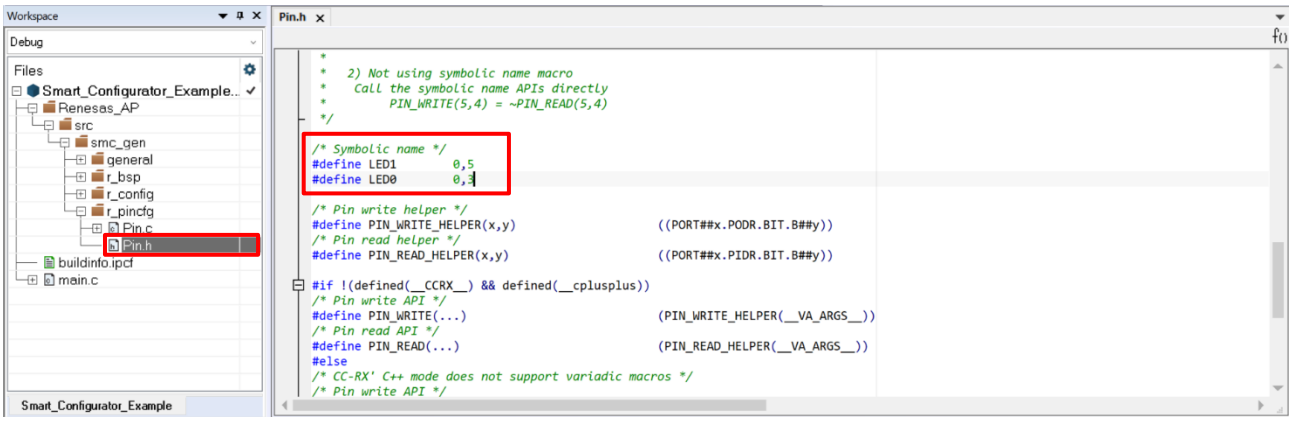

Figure 4-66 Symbolic Name in generated code

```
void main(void)
{
     // Set port direction
    PORTB.PDR.BYTE = 0x08U;
    PORTE.PDR.BYTE = 0x20U;

    //Init LED status
    PIN_WRITE(LED1) = 1U;
    PIN_WRITE(LED0) = 0U;

    //Toggle LEDs
    while(1){
        PIN_WRITE(LED0) = ~PIN_READ(LED1);
        PIN_WRITE(LED1) = ~PIN_READ(LED0);

        //Toggle LEDs
        for (int i = 0;i < 100000;i++){
            nop();
        }
    }
}
```

Figure 4-67 Using Symbolic Name in main function

4.6 Interrupt settings

Check and set the interrupts of the peripheral modules that have been selected on the [Components] page. The interrupts are displayed for each of the vector numbers. Set the interrupt priority levels, the source of the fast interrupt, or a dynamic interrupt vector number.

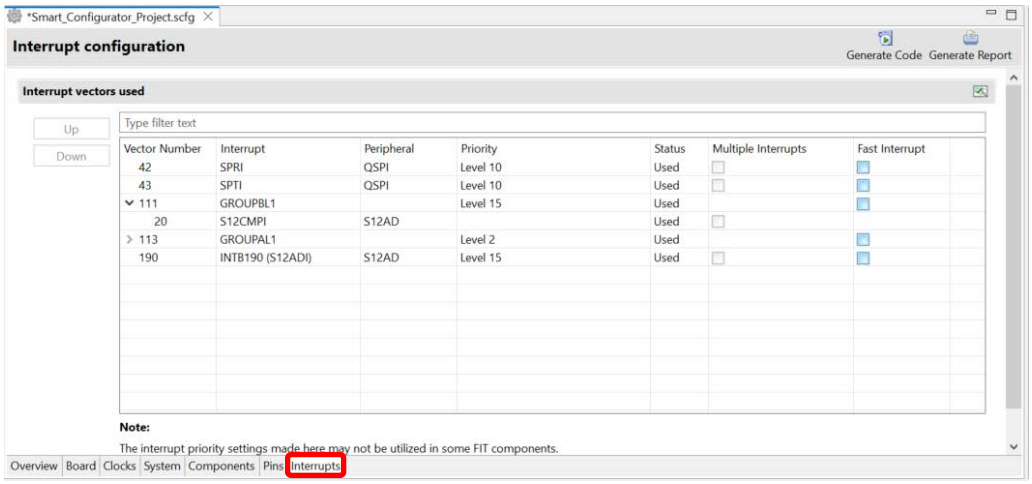

Figure 4-68 [Interrupts] Page

4.6.1 Changing the interrupt priority level and fast interrupt setting

When an interrupt is used in a configuration on the [Components] page, the status of the interrupt will be changed to "Used". To display the used interrupts only, click on the [(Show used interrupts)] button.

(1) You can change the interrupt priority level on the [Interrupts] page.

(2) To use an interrupt as a fast interrupt, tick the checkbox in the [Fast Interrupt] column. Only one interrupt can be specified as a fast interrupt among all interrupts and components used.

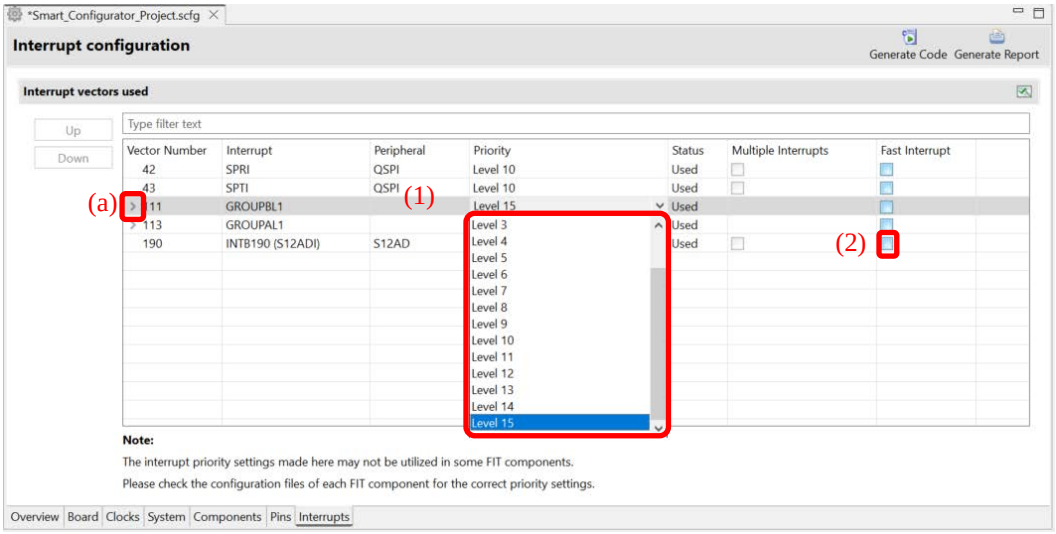

Figure 4-69 Interrupt Settings

(a) Group interrupts are collapsed in the interrupt table. Click on the [(Open)] button to expand the view and see the interrupts in the group interrupt list.

4.6.2 Changing the interrupt priority level and fast interrupt setting

The [Interrupt configuration] page enables you to change the vector numbers of software configurable interrupts A and B.

(1) Select a desired software configurable interrupt.

(2) The [Up] and [Down] buttons will be enabled. Click on a button to change the vector number.

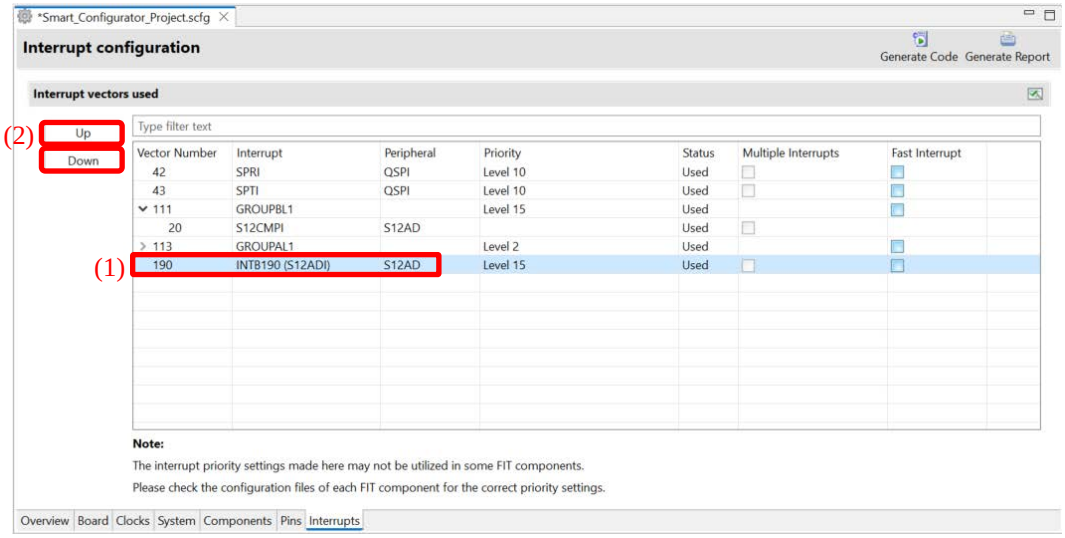

Figure 4-70 Changing the Vector Number of Software Configurable Interrupt A or B

4.6.3 Multiple interrupts setting

The multiple interrupt feature on the RX MCU allows the processing of another interrupt while the current interrupt is running. The setting of multiple interrupts can be configurated from both the Interrupt page and the Component configuration.

(1) Select a component(supported multiple interrupt) and enable its multiple interrupts settings.

(2) Multiple interrupts setting is bidirectional synchronization in [Interrupts] page.

(3) Open generated file in project explorer, generated code can be found.

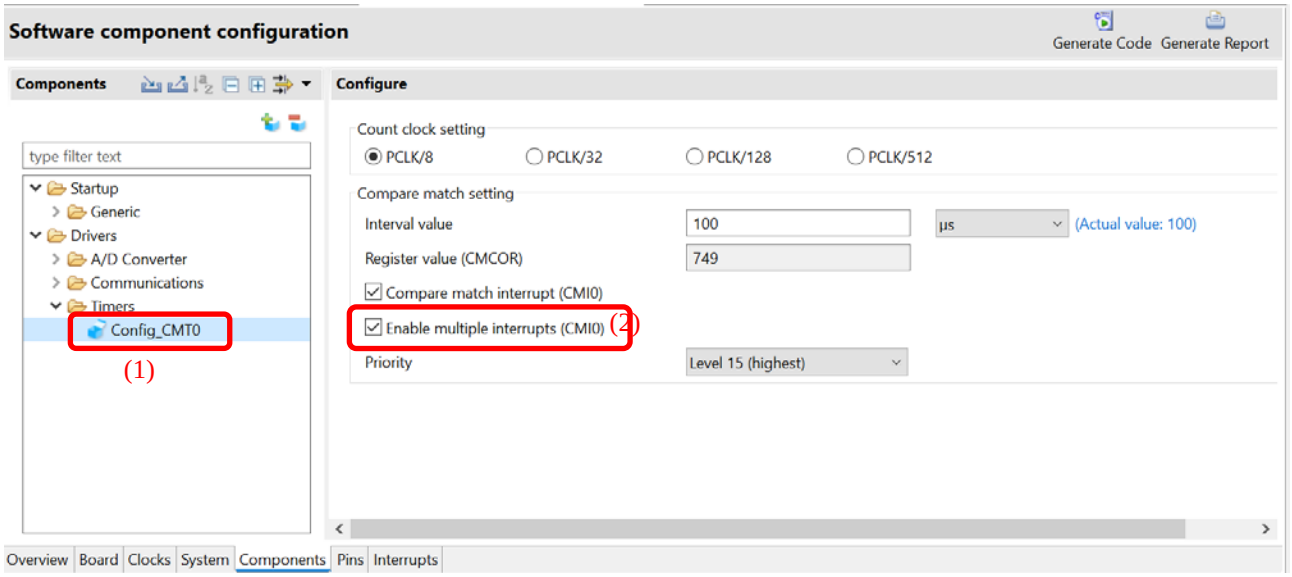

Figure 4-71 Multiple interrupts in component page

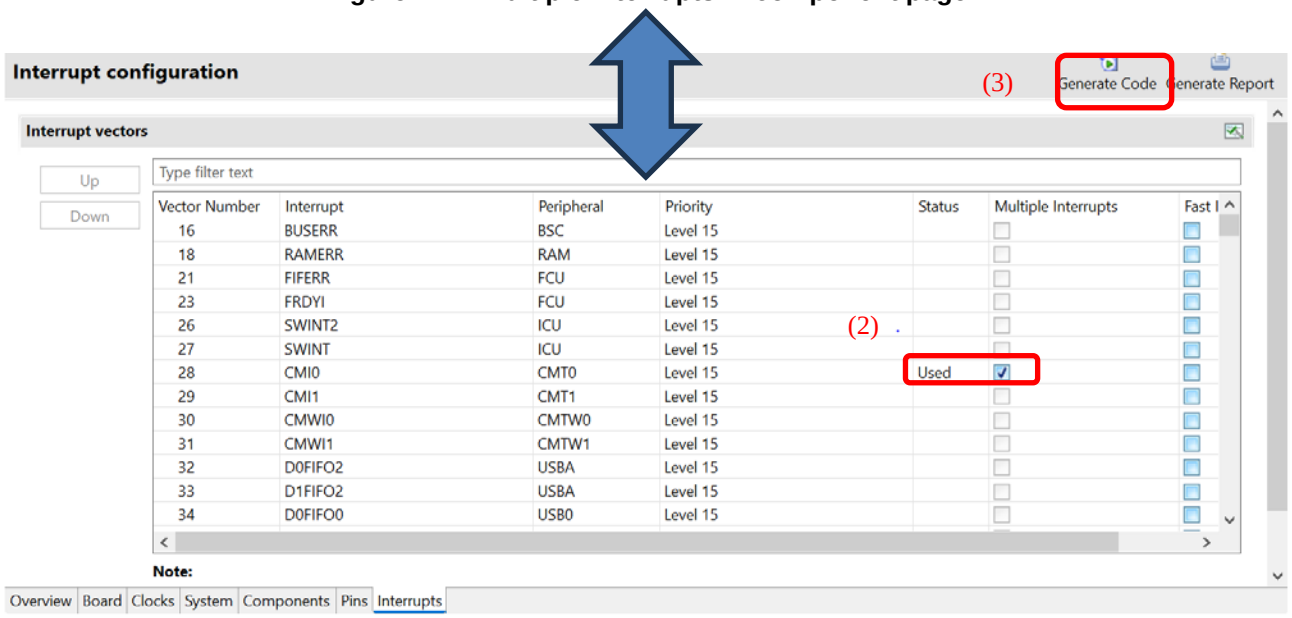

Figure 4-72 Multiple interrupts in component in Interrupts page

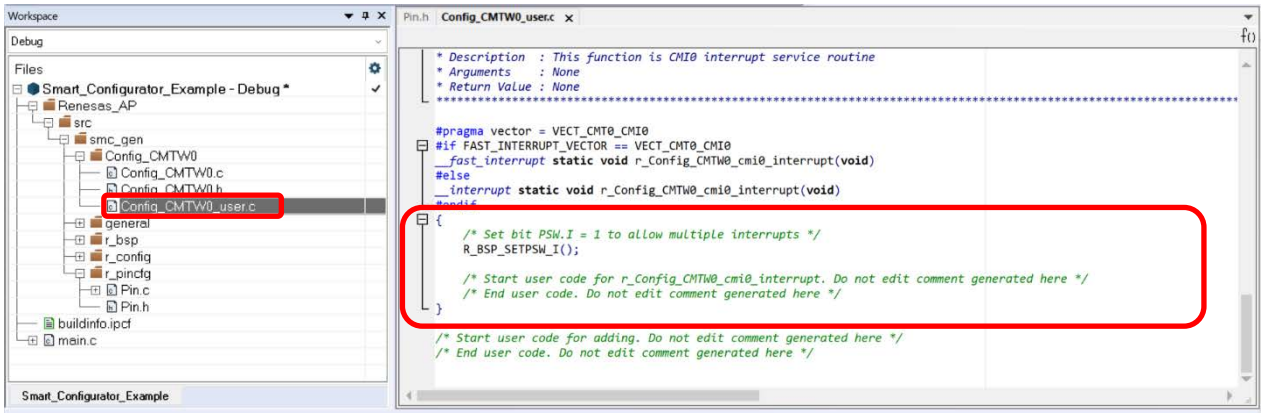

Figure 4-73 Multiple interrupts in generated code

5. Managing Conflicts

Adding components, setting pins and interrupts may cause problems related to resource mismatch. This information will be displayed in the **Configuration Problems** view. You can refer to the information displayed to fix the conflict issues.

5.1 Resource conflicts

When two software components are configured to use the same resource (e.g. S12AD1), an error mark ( ) will be displayed in the [Components tree].

The **[**Configuration Problems view] will display messages on peripheral conflicts to inform you in which software configurations peripheral conflicts have been detected.

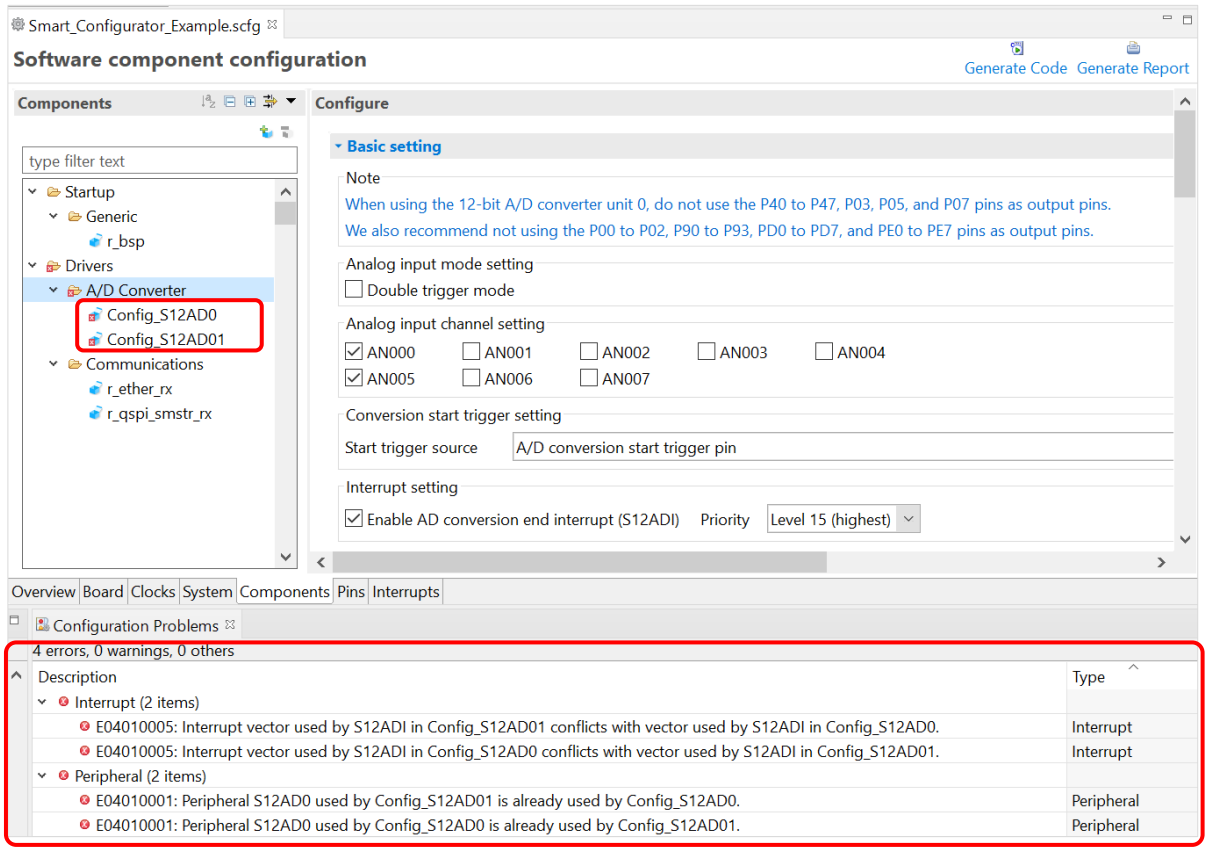

Figure 5-1 Resource Conflicts

5.2 Resolving pin conflicts

When multiple pin functions are assigned to the same pin, an error mark is displayed in the tree and [Pin Function] list on the [Pins] page.

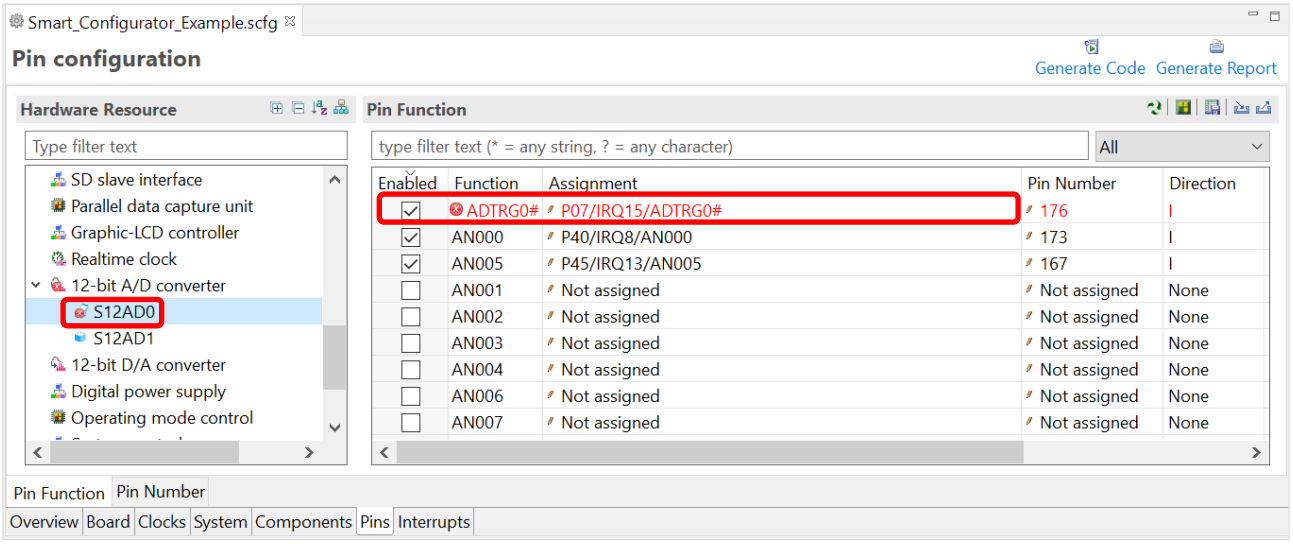

Figure 5-2 Pin Conflicts

The detailed information regarding conflicts is displayed in the [Configuration Problems view].

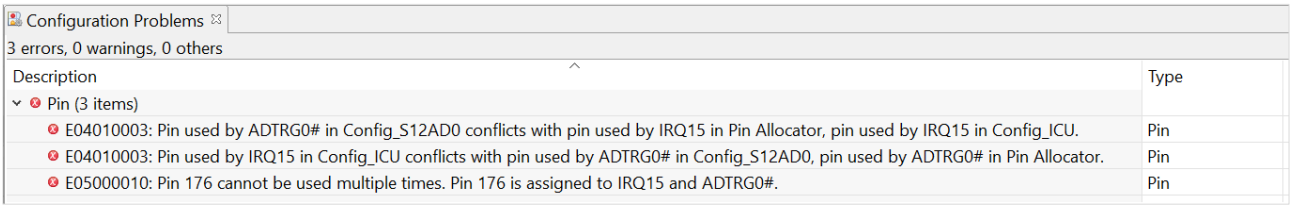

Figure 5-3 Pin Conflict Message

To resolve a conflict, right-click on the node with an error mark on the tree and select [Resolve conflict]. The pins of the selected node will be re-assigned to other pins.

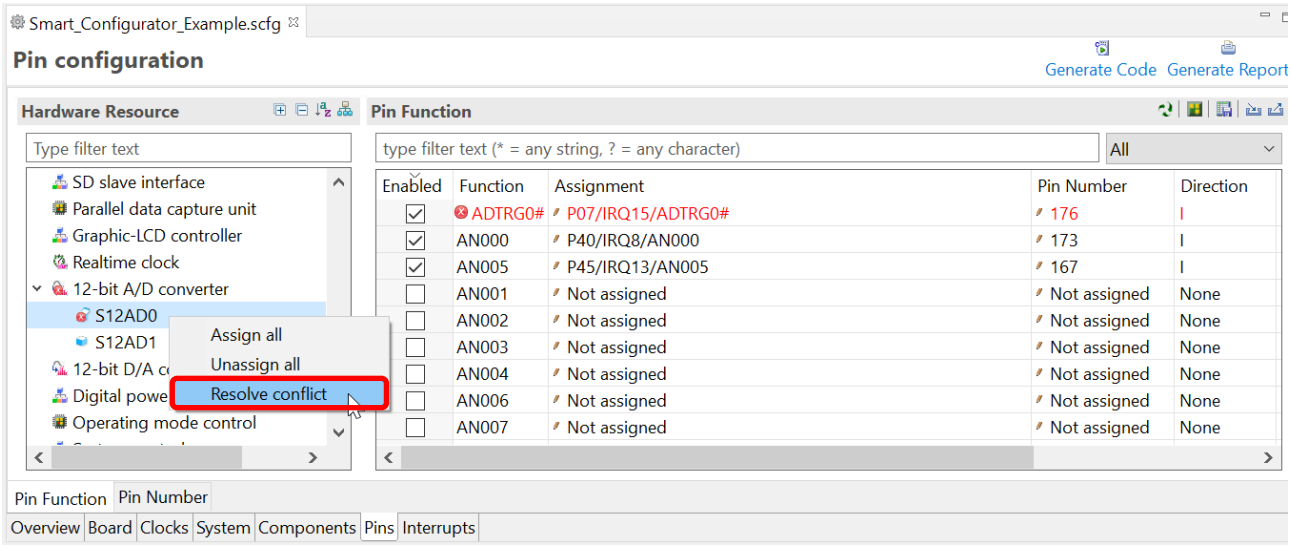

Figure 5-4 Resolving Pin Conflicts

6. Generating Source Code

6.1 Generating Source Code File

Output a source file for the configured details by clicking on [(Generate Code)] button in the Smart Configurator view.

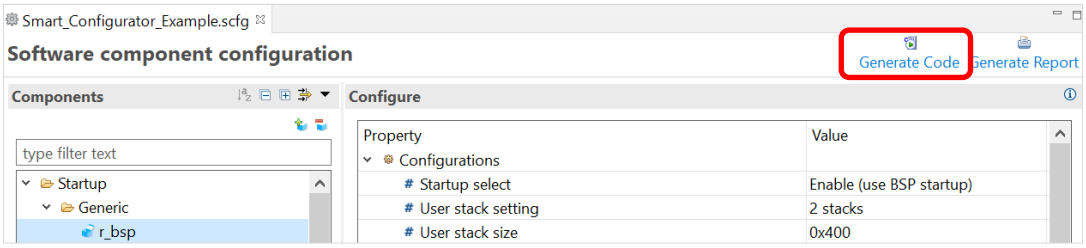

Figure 6-1 Generating a Source File

The Smart Configurator generates a source file in <ConfigurationFileDir>\src\smc_gen. If your Smart Configurator has already generated a file, a backup copy of that file is also generated (refer to the section 6.6, Backing up Generated Source Code).

6.2 Configuration of Generated Files and File Names

The Figure 6-2 below, shows the folders and files output by the Smart Configurator. "ConfigName" indicates the configuration name set in the component.

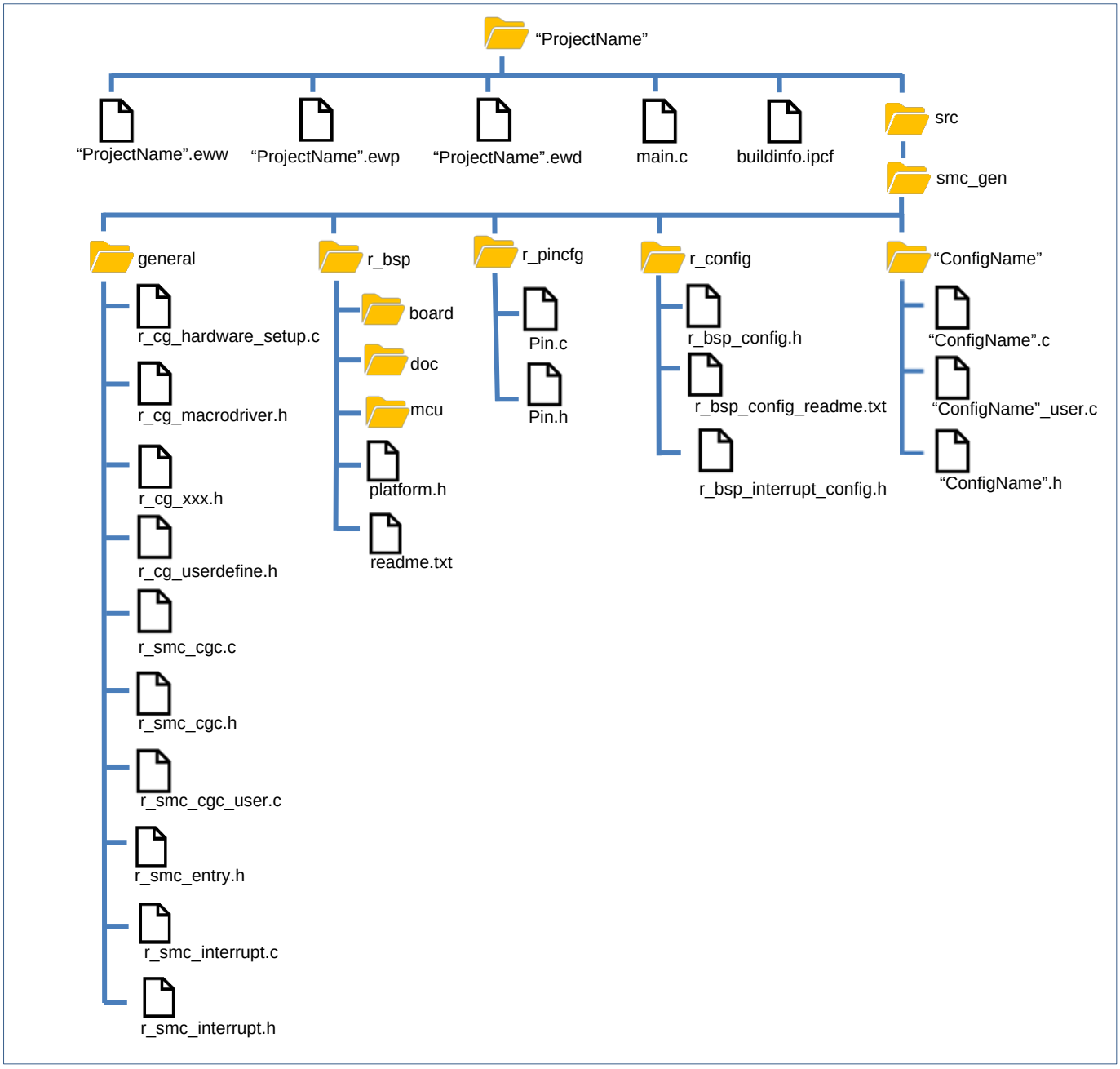

Figure 6-2 Configuration of Generated Files and File Names

Folder	File	Description
{ProjectName}	{ProjectName}.eww	This file is generated only once during the first code generation. {ProjectName}.ewp file path is specified in this file.
	{ProjectName}.ewp	This file is generated only once during the first code generation. It appends the "buildinfo.ipcf" and "main.c" files at the end of this file.
	{ProjectName}.ewd	This file is generated only once during the first code generation. It is the same as the default *.ewd file that is generated by IAR Embedded Workbench.
	main.c	This file is generated only once in the first code generation. It contains main () function.
	buildinfo.ipcf	This file is always generated. It contains source file registration information. From Smart Configurator for RX V2.15.0, the name of .ipcf file is updated into "buildinfo.ipcf". If user loads a project that was created before Smart Configurator for RX V2.15.0, the .ipcf file will be regenerated as "buildinfo.ipcf" but the original .ipcf file ({ProjectName}.ipcf) will still exist in the folder and will not be removed.
general	-	This folder is always generated. It contains header files and source files commonly used by drivers of the same peripheral function.
	r_cg_hardware_setup.c	This file is always generated.It contains R_Systeminit that calls all driver initialization functions with the name R_ConfigName_Create. R_Systeminit also calls the functions for initializing clocks other than the clock source, fast interrupt, and group interrupts.
	r_cg_macrodriver.h	This file is always generated. This header file contains common macro definitions used in drivers.
	r_cg_xxx.h(Note*1)	These files are always generated.The files contain macro definitions for setting SFR registers.
	r_cg_userdefine.h	This file is always generated. User can add macro definitions in the dedicated user code areas.
	r_smc_cgc.c	This file is always generated. It contains the initialization of clock sources other than the clock source selected in the [Clocks] page.
	r_smc_cgc.h	This file is always generated. This header file contains macro definitions to initialize clocks other than the selected clock source.
	r_smc_cgc_user.c	This file contains functions to be added to R_CGC_Create after the CGC initialization. User can add codes and functions in the dedicated user code areas.
	r_smc_entry.h	This file is always generated. This file includes the header files of CG drivers that are added to the project. When using functions of CG drivers in source files added by user, including this file is necessary.
	r_smc_interrupt.c	This file is always generated. It contains fast interrupt and group interrupt initialization (depending on hardware specification).

	r_smc_interrupt.h	This file is always generated. It contains macro definitions for fast interrupt and group interrupt initialization. It also contains the priority level of all interrupts that are configured in the [Interrupts] tabbed page. User can use these macro definitions in application codes.
r_bsp		This folder is always generated. It consists of multiple subfolders (board, doc, mcu) with: - Initialization codes to start up the MCU before entering main () - Definitions of all SFR registers in iodefine.h (mcu folder) - Application note of r_bsp (doc folder) It also contains platform.h that will include r_bsp.h of the device used in the project.
r_pincfg	Pin.c	This file is always generated. It is a reference of pin function initialization for all peripherals configured in the [Pins] tabbed page (except I/O Ports).
	Pin.h	This file is always generated. It contains the function prototypes of pin settings in Pin.c
r_config	r_bsp_config.h	The file is always generated. It contains the configuration of BSP.
	r_bsp_interrupt_config.h	This file is always generated. It contains mapping of the software configurable interrupts A and B (depending on hardware specification).
{ConfigName}	-	This folder is generated for the added component. API functions in this folder are named after the ConfigName (configuration name).
	{ConfigName}.c	This file contains functions to initialize driver (R_ConfigName_Create) and perform operations that are driver-specific, e.g. start (R_ConfigName_Start) and stop (R_ConfigName_Stop).
	{ConfigName}_user.c	This file contains interrupt service routines and functions for user to add code after the driver initialization (R_ConfigName_Create). User can add codes and functions in the dedicated user code areas.
	{ConfigName}.h	This is header file for {ConfigName}.c and {ConfigName}_user.c.

Note *1: xxx is the name of a component.

6.3 Initializing Clocks

Configurations of clock source in [Clocks] page are generated in \src\smc_gen\r_config folder.

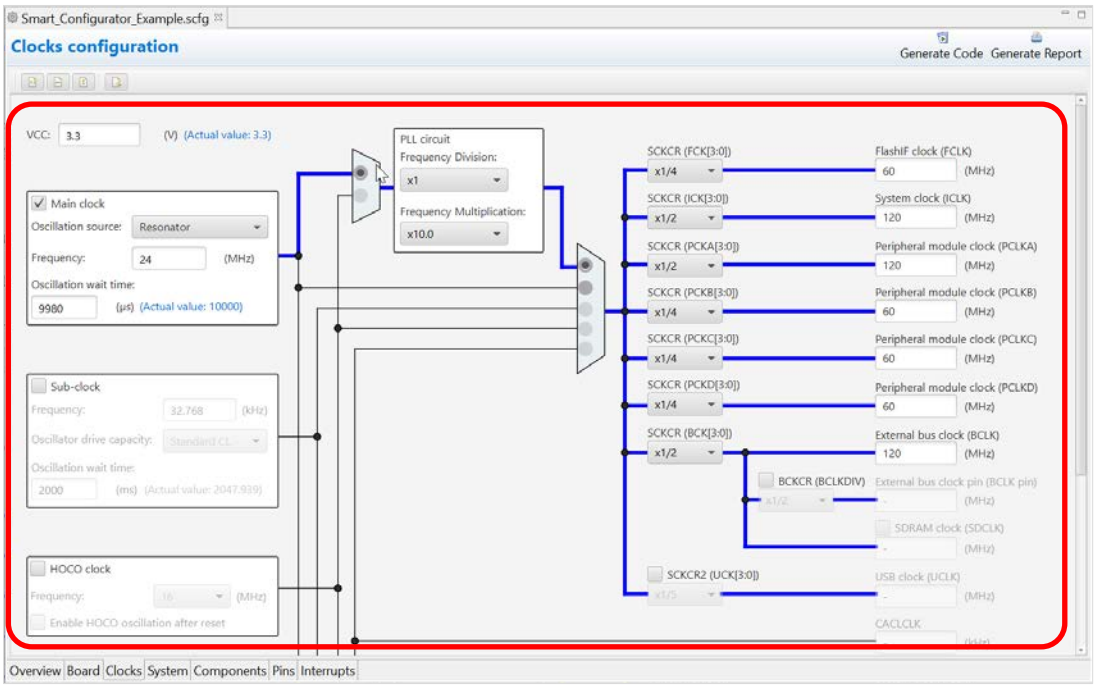

Figure 6-3 Clocks Source Configuration

Table 6-1 Clock Source File Description

Folder	File	Macros/Functions	Description
general	*r_cg_cgc.c*	*R_CGC_Create*	This API function initializes clocks other than the selected clock source. *R_Systeminit* in *r_cg_hardware_setup.c* will call this function before entering *main()* function.
	r_cg_cgc.h	Macros related to clocks	These macros are for clock initialization in *R_CGC_Create*.
	r_cg_cgc_user.c	*R_CGC_Create_UserInit*	This API function is used to add code to *R_CGC_Create* after the CGC initialization.

6.4 Initializing Pins

Pin configuration settings are generated by the component into source files as shown in (1) and (2) below.

(1) Pins initialization for drivers with {*ConfigName*}

The pin function is initialized with *R_ConfigName_Create* of *src**smc_gen**{ConfigName}**{ConfigName}.c*.

Table 6-2 File to Initialize Pins

Folder	File	Function	Description
{ConfigName}	*{ConfigName}.c*	*R_ConfigName_Create*	This API function initializes pins used by this component. *R_Systeminit* in *r_cg_hardware_setup.c* will call this function before entering *main()* function.

(2) Reference pins initialization codes

Refer to *Pin.c* in the *src**smc_gen**r_pincfg* folder for the initialization code of all pin functions set on the [Pins] page (except I/O ports).

Table 6-3 Reference File for Initialization of All Pins

Folder	File	Function	Description
r_pincfg	*Pin.c*	*R_Pins_Create*	This function contains the initialization codes of all pins function configured at [Pins] page except I/O ports.

6.5 Initializing Interrupts

Configurations in [Interrupt] page are generated in few source files.

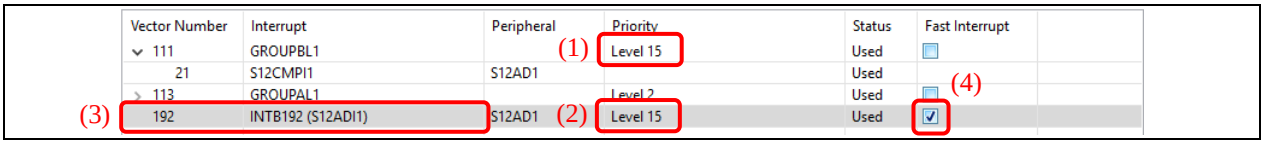

Figure 6-4 Interrupt Configuration

Table 6-4 Interrupt Generation File Description

No	Item	Folder	File	Description
(1)	Priority	**general**	*r_smc_interrupt.c*	This interrupt priority level setting is for group interrupts *(Note2)*. It is initialized in *R_Interrupt_Create* of this file. *R_Systeminit* in *r_cg_hardware_setup.c* will call this function before entering *main()* function.
(2)	Priority	**{ConfigName}**	*{ConfigName}.c*	This interrupt priority level setting is for normal interrupts and software configurable interrupts A and B *(Note2)*. It is initialized in *R_ConfigName_Create* of this file. *R_Systeminit* in *r_cg_hardware_setup.c* will call this function before entering *main()* function.
(1) (2)	Priority	**general**	*r_smc_interrupt.h*	Priority level of all interrupts configured in the [Interrupts] tabbed page is defined in this file. User can use these macro definitions in the application codes.
(3)	Vector Number	**r_config**	*r_bsp_interrupt_config.h*	Vector number of software configurable interrupts A and B *(Note2)* in the [Interrupts] tabbed page will be mapped in this file and handled by *r_bsp*.
(4)	Fast Interrupt	**general**	*r_smc_interrupt.c*	Fast interrupt setting will be initialized in *R_Interrupt_Create* of this file. *R_Systeminit* in *r_cg_hardware_setup.c* will call this function before entering *main()* function.
			r_smc_interrupt.h	Priority level of all interrupts configured in the [Interrupts] tabbed page is defined in this file. User can use these macro definitions in the application codes.

Note *2: The type of interrupt depends on hardware specifications.

6.6 Backing up Generated Source Code

The smart configurator has a source code backup function.

<ConfigurationFileDir>\trash\<Date-and-Time>

The Smart Configurator generates a backup folder for the previously generated source code when new code is generated b icking on [ ]. <Date-and-Time> indicates the date and time when the backup folder is created after code generation.

7. Loading generated files in Integrated development environment

Load source code outputted by Smart Configurator on Integrated Development Environment Platform.

7.1 Adding Custom Code of FIT

When [FIT] is selected as the component type, the configuration options are set in r_xxx_config.h in the folder r_config. For the settings of the configuration options, refer to the application note (in the doc folder) on the FIT module (r_xxx) which you have added to the project tree.

If the target file already exists, the existing contents of the file are protected when source code is output.

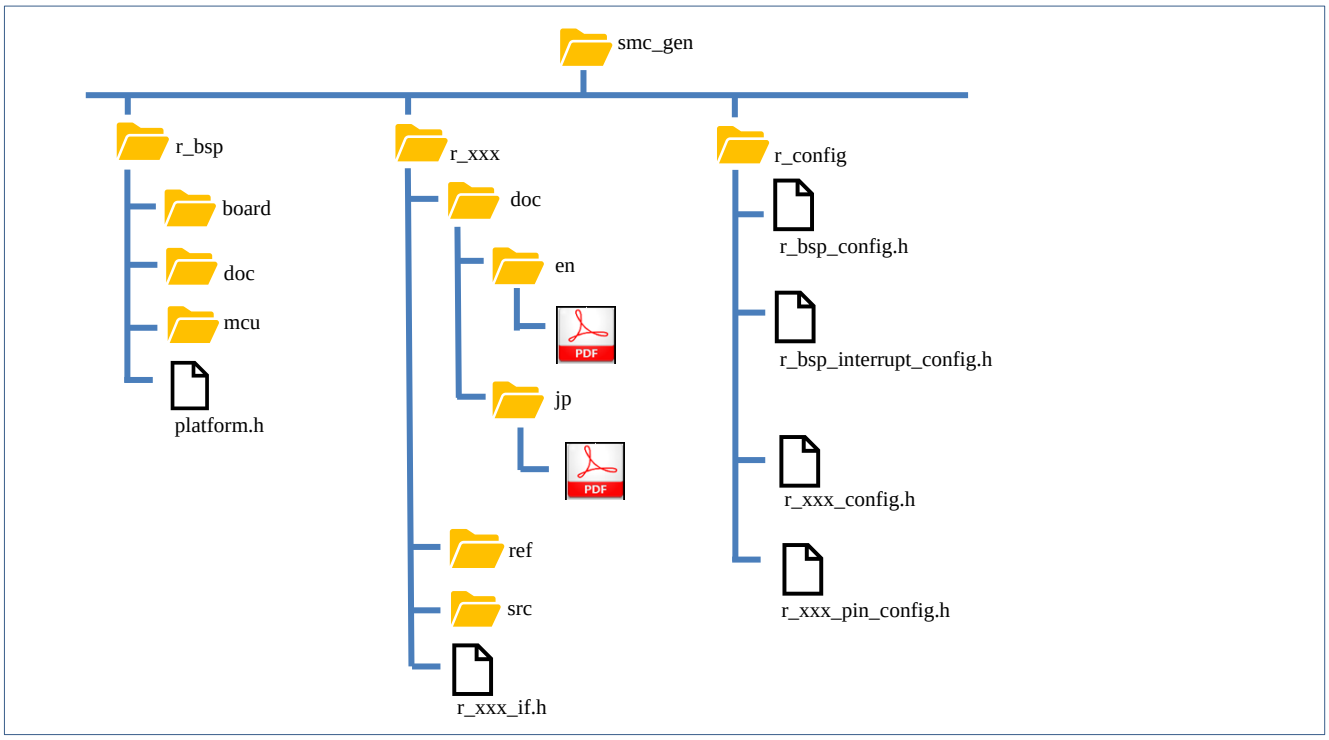

Figure 7-1 Tree Structure of Directories and Files for a FIT Module

7.2 Loading in IAR Embedded Workbench

When IAR environment is selected for the compiler to be used, Smart Configurator outputs the related files (.eww/.ewp/.ewd/main.c) together with the source file. It is not necessary for the user to create project files in IAR Embedded Workbench.

The usage procedure is as follows.

(1) Select "Open Workspace…" from the "File" menu of IAR Embedded Workbench.

(2) In the "Open Workspace" dialog box, browse to the folder where the project file is saved, select the project file (.eww), and click the "Open" button.

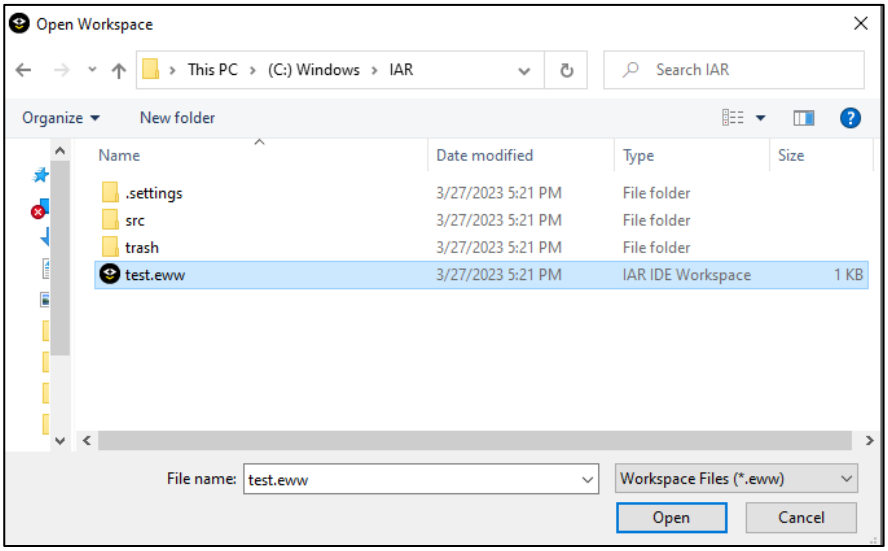

Figure 7-2 Load *.eww file

(3) The source file output by the Smart Configurator is added to the IAR C project workspace

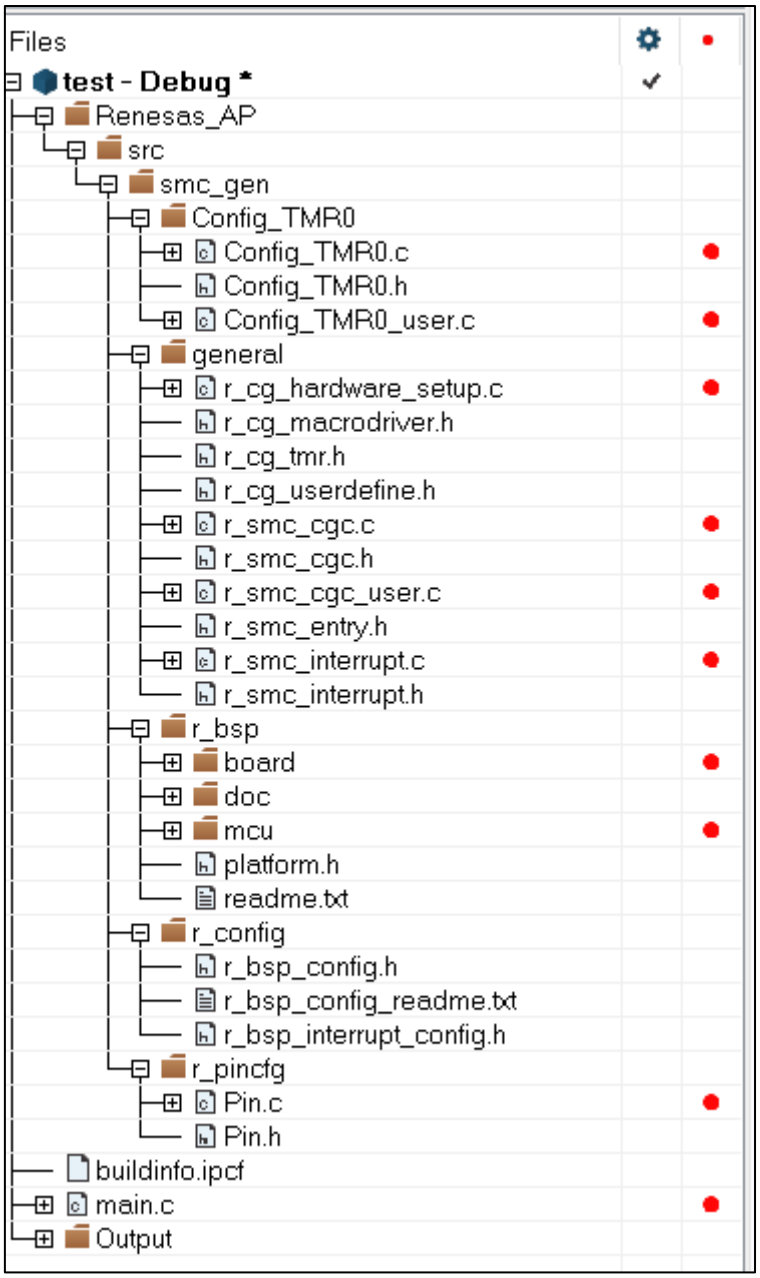

Figure 7-3 New files added to IAR workspace

(4) Select "Options…" from the "Project" menu of IAR Embedded Workbench.

(5) Change the target device to match with the target device selected when creating Smart Configurator's configuration file inside the "Options for node" dialog box.

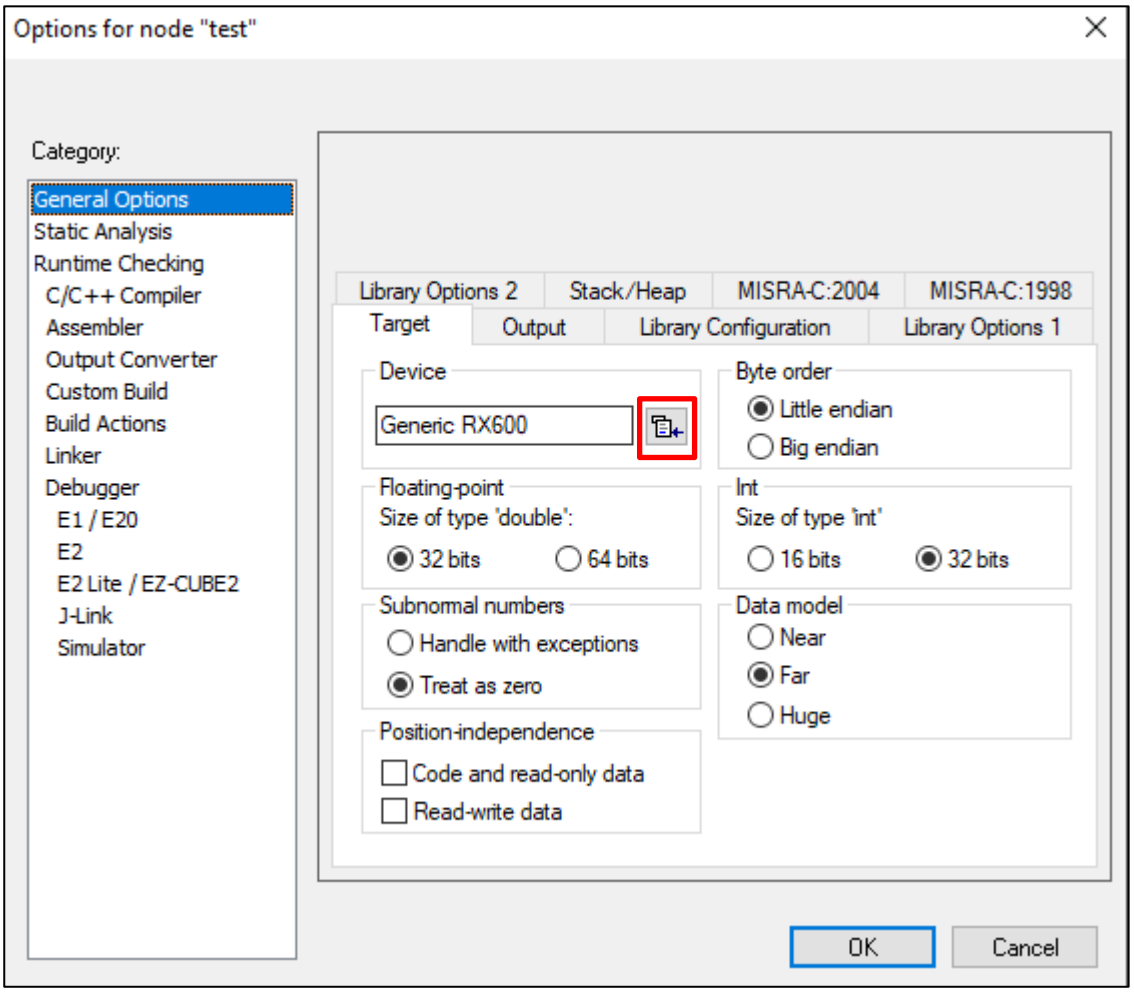

Figure 7-4 Options for node

7.3 Build IAR Project File

After successfully loading the Smart Configurator project file to IAR Embedded Workbench, user can right-click on the project name and select [Rebuild All] from the context menu. This will execute the build operation successfully.

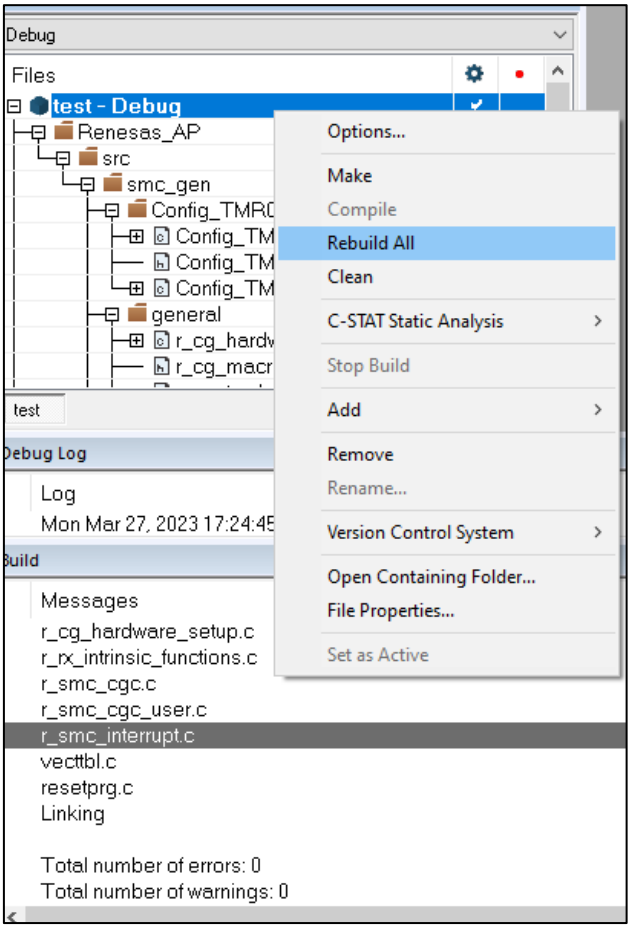

Figure 7-5 Build C Project File in IAR

8. Creating User Programs

Create a user program in the IDE. This chapter describes how to add custom code to the source file generated by the SC.

8.1 Adding Custom Code in the Case of Code Generator

When [Code Generator] is selected as the component type, if files which have the same name already exist, new code will be merged only with the existing code that is between the comments below.

```
/* Start user code for xxxx. Do not edit comment generated here */

/* End user code. Do not edit comment generated here */
```

In the case of [Code Generator], three files are generated for each of the specified peripheral functions. The file names are "Config_xxx.h", "Config_xxx.c", and "Config_xxx_user.c" as the default, with "xxx" representing the name of the peripheral module. For example, "xxx" will be "CMT3" for the compare-match timer (resource CMT3). The comments to indicate where to add custom code are at the start and end of each of the three files. Comments to indicate where to add user code are also added to the interrupt function for the peripheral module corresponding to Config.xxx_user.c. The following examples are for CMT3 (Config_CMT3_user.c).

```
/***********************************************************************************
Pragma directive
***********************************************************************************/
/* Start user code for pragma. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
Includes
***********************************************************************************/
#include "r_cg_macrodriver.h"
#include "r_cg_userdefine.h"
#include "Config_CMT3.h"
/* Start user code for include. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
Global variables and functions
***********************************************************************************/
/* Start user code for global. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
* Function Name: R_Config_CMT3_Create_UserInit
* Description  : This function adds user code after initializing the CMT3 channel
* Arguments    : None
* Return Value : None
***********************************************************************************/

void R_Config_CMT3_Create_UserInit(void)
{
    /* Start user code for user init. Do not edit comment generated here */
    /* End user code. Do not edit comment generated here */
}
```

```
/***********************************************************************************
* Function Name: r_Config_CMT3_cmi3_interrupt
* Description  : This function is CMI3 interrupt service routine
* Arguments    : None
* Return Value : None
***********************************************************************************/

#if FAST_INTERRUPT_VECTOR == VECT_PERIB_INTB129
#pragma interrupt r_Config_CMT3_cmi3_interrupt(vect=VECT(PERIB,INTB129),fint)
#else
#pragma interrupt r_Config_CMT3_cmi3_interrupt(vect=VECT(PERIB,INTB129))
#endif
static void r_Config_CMT3_cmi3_interrupt(void)
{
    /* Start user code for r_Config_CMT3_cmi3_interrupt. Do not edit comment generated here */
    /* End user code. Do not edit comment generated here */
}

/* Start user code for adding. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */
```

8.2 Using Generated Code in user application

To use the generated code of FIT and Code Generator, follow the below steps:

1) Open the *main.c* file, add code to include the header files of the modules you want to use.

   In case of FIT, it is <r_xxx_if.h>.

   In case of Code Generator, it is <r_smc_entry.h>.

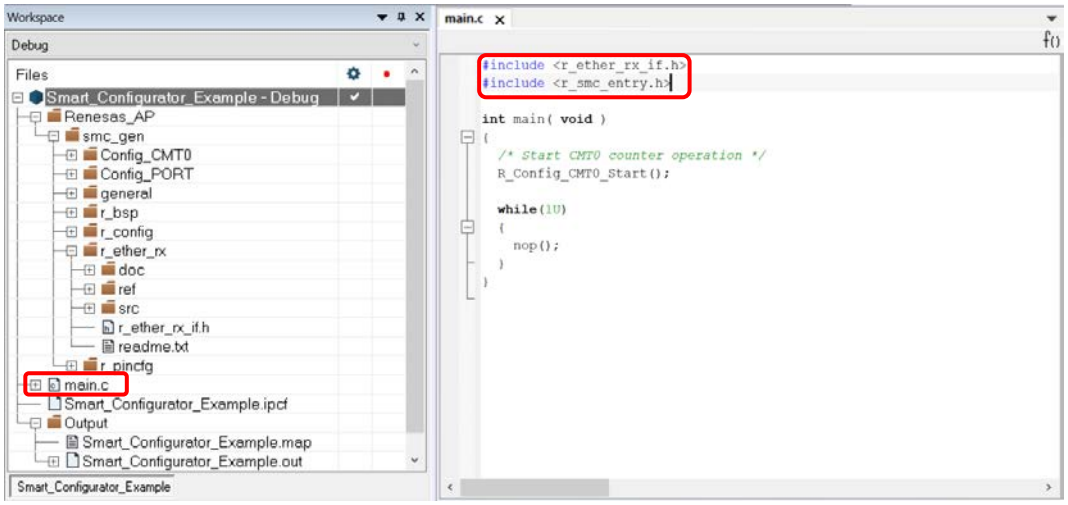

Figure 8-1 Add header files

2) In the main function, call the functions generated and add application codes.

   In case of Code Generator, driver initialization functions (*R_ConfigName_Create*) including initialization of pins have been called in *R_Systeminit* function of *r_cg_hardware_setup.c* by default. You just need to add application codes to perform operations that are driver-specific, for e.g. start (*R_ConfigName_Start*) and stop (*R_ConfigName_Stop*).

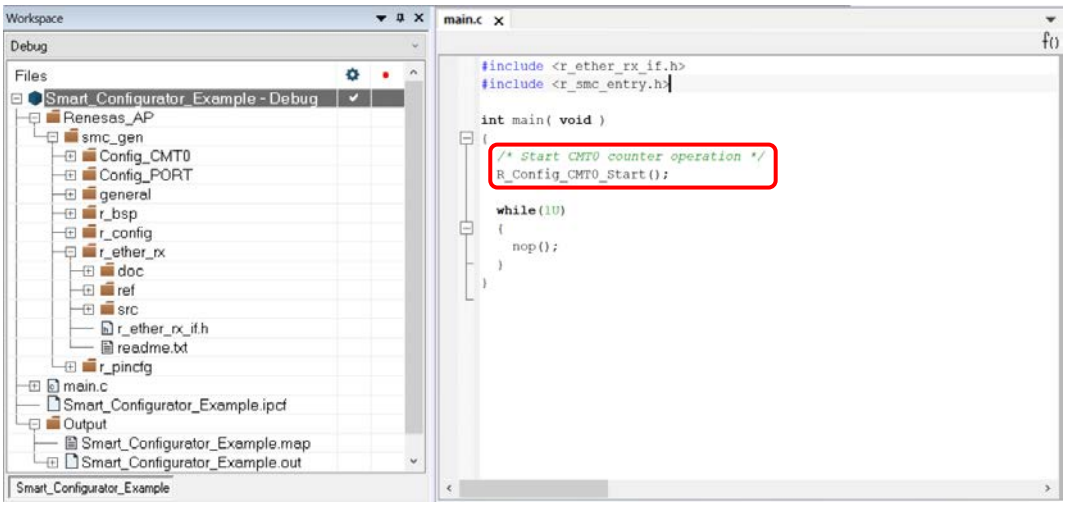

Figure 8-2 Call Code Generator functions

In case of FIT module, refer to the examples provided in the "API Functions" chapter of corresponding Application Note. You can find the Application Note in [doc] folder under each FIT module.

9. Generating Reports

The Smart Configurator can output the configuration information of the project to the report. Follow the procedure below to generate a report.

9.1 Report on Configuration

A report is output in response to clicking on the [(Generate Report)] button in the Smart Configurator view. Two selections of output files are available (PDF, Text).

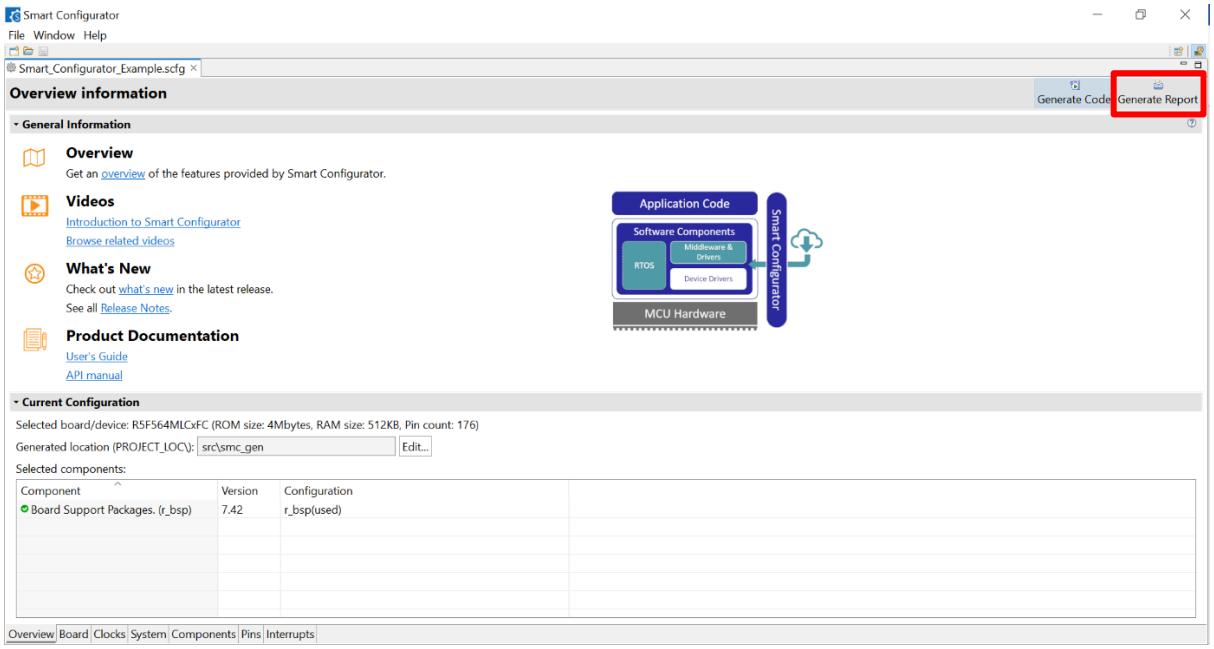

Figure 9-1 Output of a Report on the Configuration

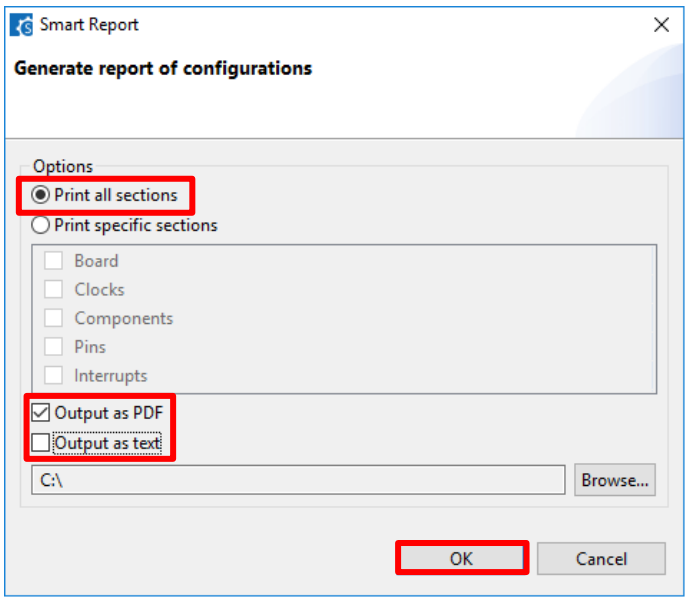

Figure 9-2 Dialog Box for Output of a Report (Example is selecting "Output as PDF")

9.2 Configuration of Pin Function List and Pin Number List (in csv Format)

A list of the configuration of pin functions and pin numbers (whichever is selected at the time) is output in response to clicking on [(Save the list to .csv file)] on the [Pins] page of the Smart Configurator view.

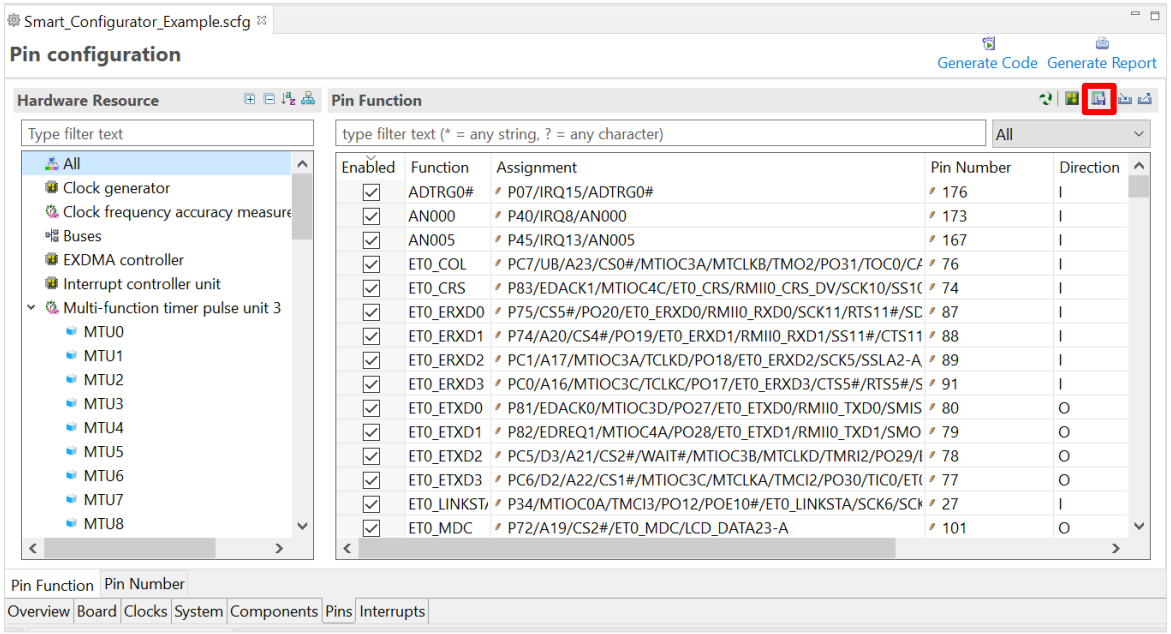

Figure 9-3 Output of a List of Pin Functions or Numbers (in csv Format)

9.3 Image of MCU/MPU Package (in png Format)

An image of the MCU/MPU package is output in response to clicking on the [(Save Package View to external image file)] button of the [MCU/MPU Package] view.

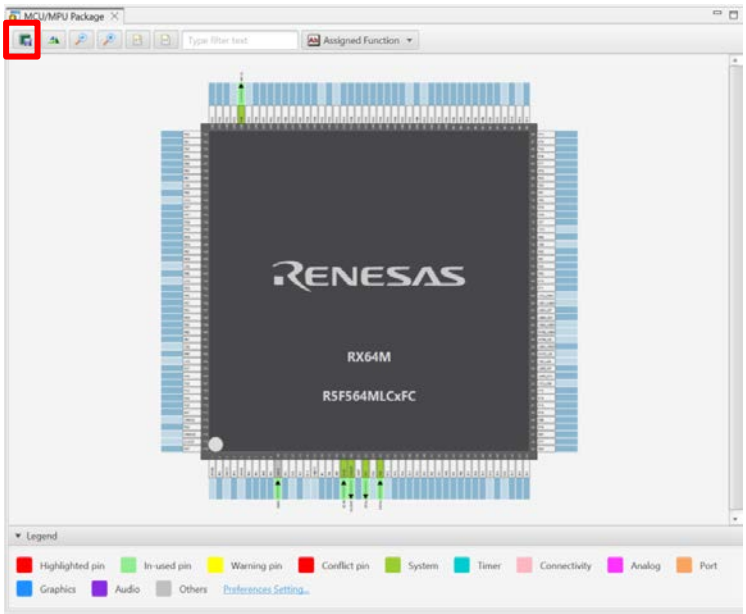

Figure 9-4 Outputting a Figure of MCU/MPU Package (in png Format)

10. User code protection feature for Smart Configurator Code Generation component

The Smart Configurator for RX V2.16.0 and the later version now incorporates an enhanced user code protection feature. This feature empowers users to insert codes to any location in the generated codes by utilizing the specific tags, as shown in Figure 10-1. After the next code generation, the inserted user codes will be protected and automatically merged into the generated files.

The user code protection feature will only be supported on the files that are generated by the "Code Generation component".

10.1 Specific tags for the user code protection feature

When using the user code protection feature, please insert /* Start user code */ and /* End user code */ as shown in Figure 10-1 and add the user codes between these tags. If the specific tags do not match exactly, the inserted user code will not be protected after the code generation.

```
/* Start user code */

User code can be added between the specific tags

/* End user code */
```

Figure 10-1 Specific tags for user code protection feature

10.2 Examples of using user code protection feature to add new user code

Figure 10-2 shows an example of adding new user code into the Create API of CMT module by using the specific tags shown in Figure 10-1. After updating the configuration in the CMT GUI and re-generating the codes, the inserted user codes will be automatically merged into the new generated file.

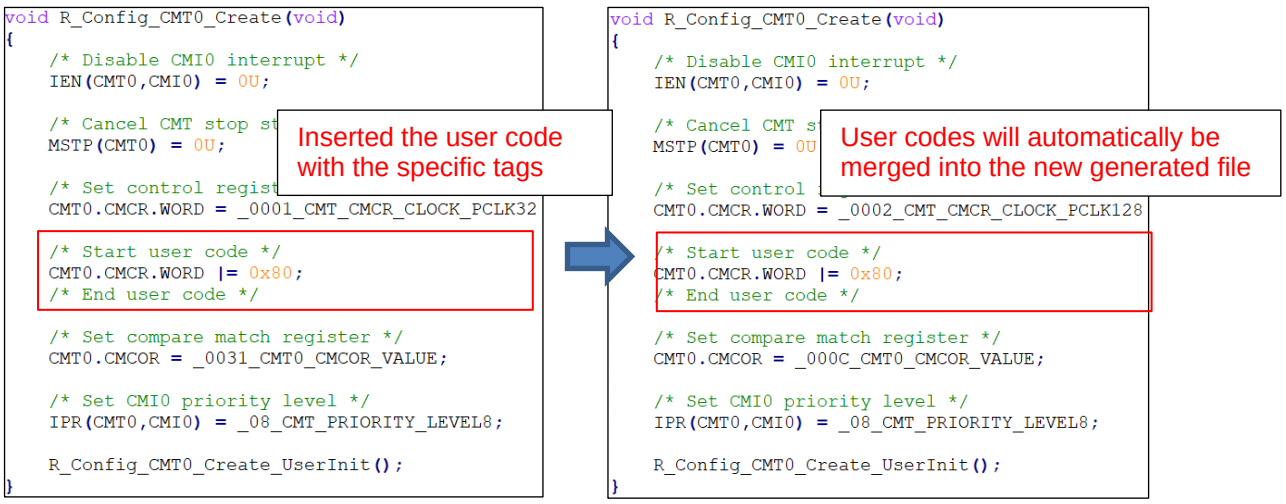

Figure 10-2 User code protection with auto merge

10.3 What to do when merge conflict occurs

10.3.1 What is Merge conflict

When the lines of generated codes before and after the inserted user codes are updated due to changes in GUI configuration or the version update of Smart Configurator, merge conflict codes will be generated out, as shown in Figure 10-3.

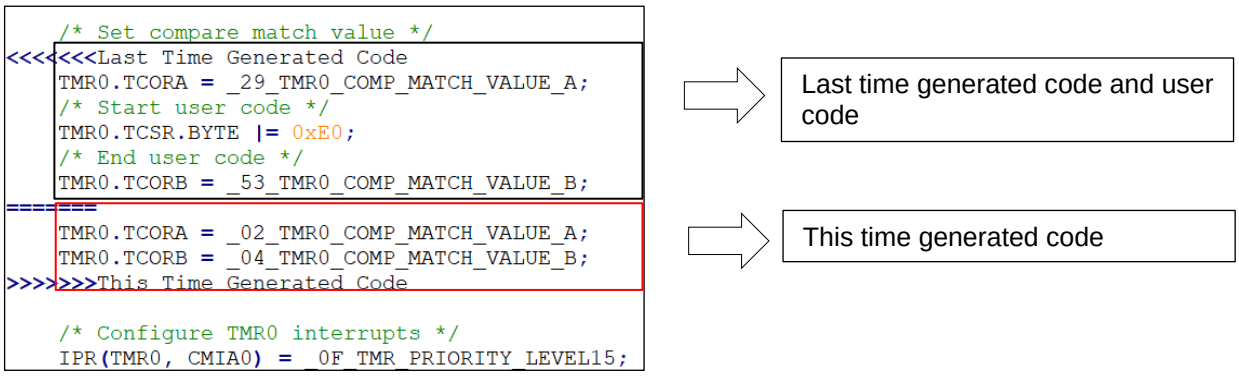

Figure 10-3 User code protection with merge conflict

If the merge conflict occurs, conflict message will be displayed in the Smart Configurator console, as shown in Figure 10-4.

Figure 10-4 The merge conflict message outputted in the Smart Configurator console

10.3.2 Steps for resolving the merge conflict

To resolve this merge conflict, open the highlighted conflict files and follow the steps below to solve the merge conflicts manually.

1) Copy the user code from "Last Time Generated Code" and paste it into the new position in "This Time Generated Code" as shown in Figure 10-5.

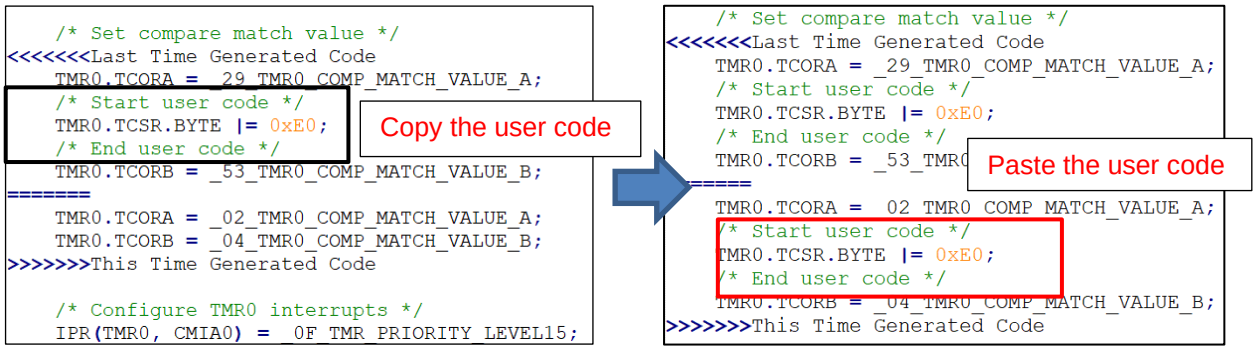

Figure 10-5 Generated conflict code

2) Remove last time generated code and the conflicts commend (<<<<<<<Last Time Generated Code, ======= and >>>>>>>This Time Generated Code) as shown in Figure 10-6**.**

```
/* Set compare match value */
TMR0.TCORA = _02_TMR0_COMP_MATCH_VALUE_A;
/* Start user code */
TMR0.TCSR.BYTE |= 0xE0;
/* End user code */
TMR0.TCORB = _04_TMR0_COMP_MATCH_VALUE_B;
```

Figure 10-6 The codes after resolving the merge conflict

Another way to solve merge conflict:

1) Click this console message to open the compare view.

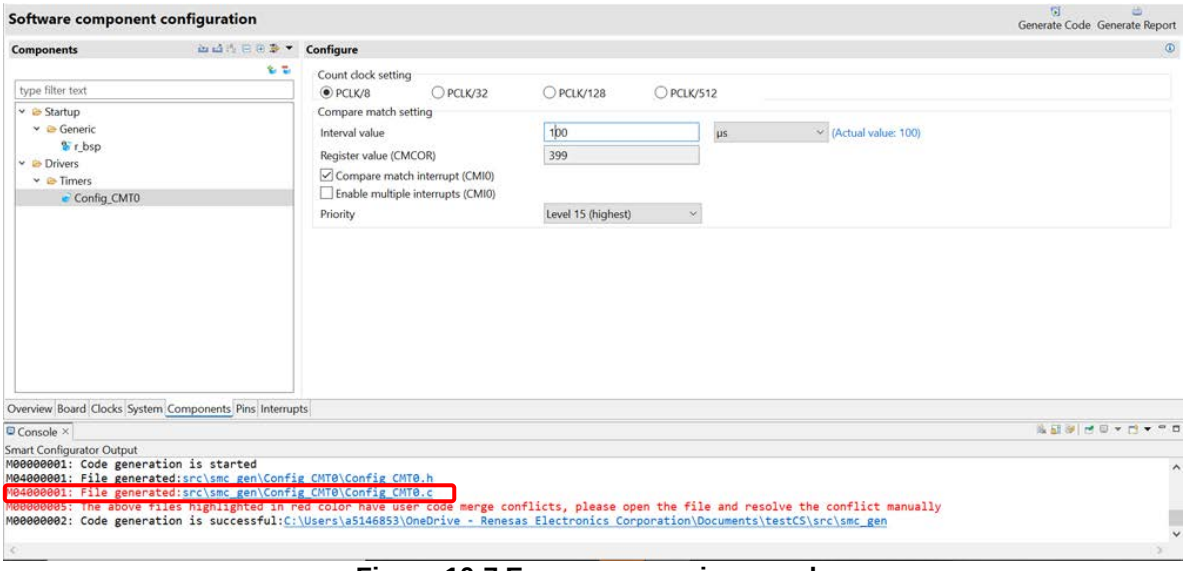

Figure 10-7 Error message in console

2) After compare view is opened, user can apply left change to the right. Or user can edit right side code manually.

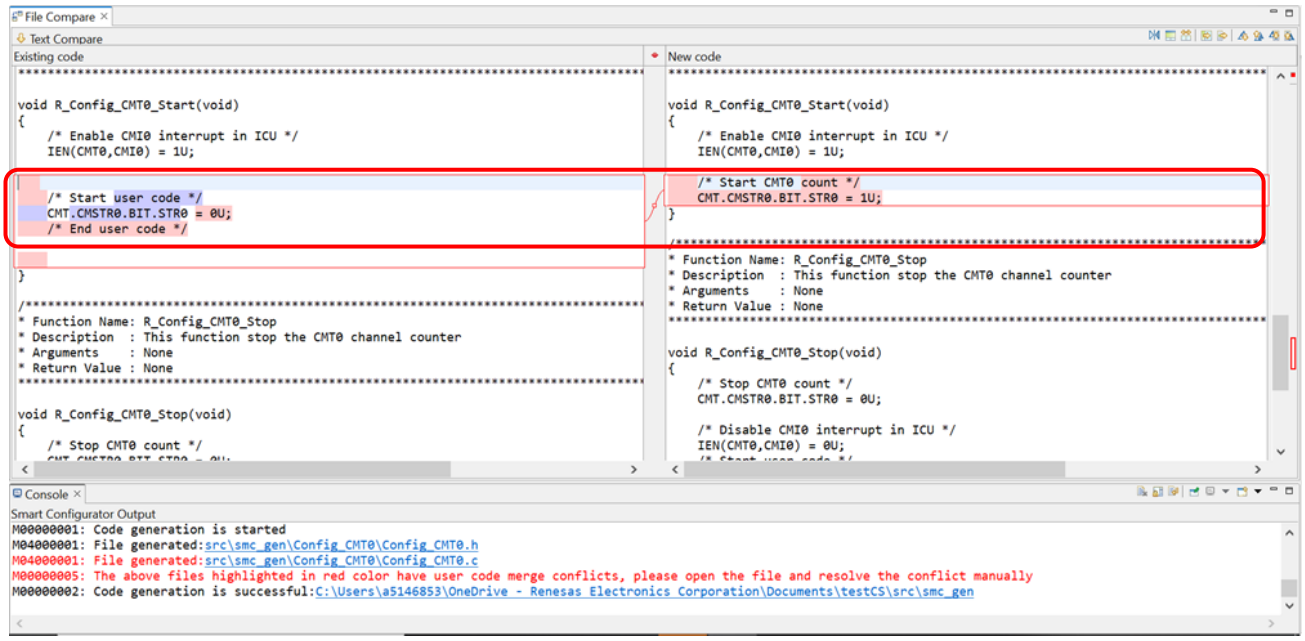

Figure 10-8 Compare Viewer for conflict code

11. Help

11.1 Help

Refer to the help system for detailed information on the Smart Configurator.

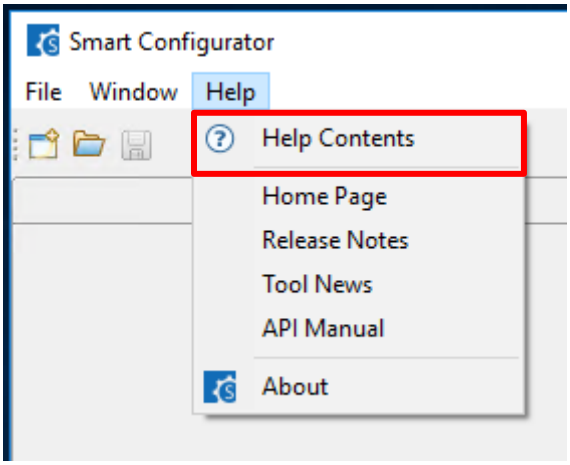

Figure 11-1 Help Menu

The help system can also be activated from the [Overview information] page by clicking button.

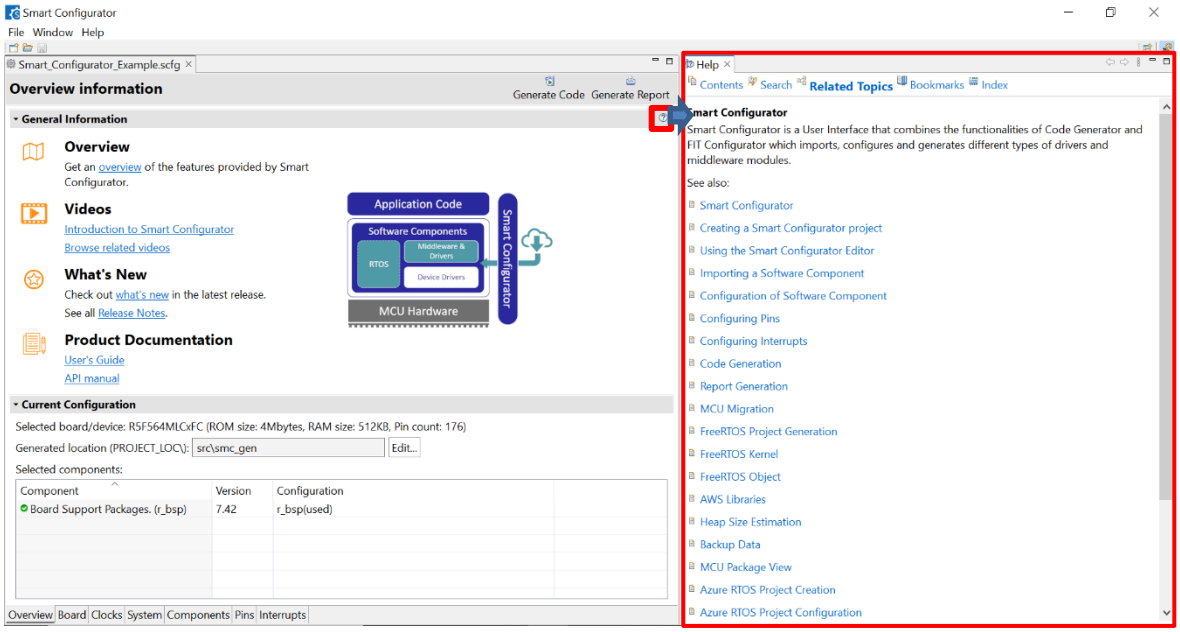

Figure 11-2 Smart Configurator Quick Start information

12. Documents for Reference

User's Manual: Hardware

Obtain the latest version of the manual from the web site of Renesas Electronics.

Technical Update/Technical News

Obtain the latest information from the web site of Renesas Electronics.

User's Manual: Development Environment

Obtain the latest version of the manual from each company web site.

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Jan 25, 2019	-	First edition issued
1.10	Sep 09, 2019	4	1.2 Features updated, 1.3 Software Components added
		8	3.3.1 Downloading FIT modules added 3.3.2 Creating a New Configuration File updated
		12	3.4.1 Main menu updated
		29	Adding FIT drivers or middleware added
		30	4.4.7 Setting of the FIT software components added
		31	4.4.8 Changing the version of the FIT software components added
		31	4.4.9 Solving the grey-out component added
		32	4.4.10 “i” mark on FIT modules icon added
		33 - 34	4.4.11 Configure Analog Front End component added
		35 – 37	4.4.12 Configure Motor Component added
		38	4.4.13 Configure general setting of component added
		45	7.1 Adding Custom Code of FIT added
		50	11.1 Help updated
1.20	Jun 20, 2021	All	Smart Configurator images updated
		13	3.4.4 MCU/MPU Package view updated
		18	4.3 System Settings added
		30	4.4.9 Solving the grey-out component added
		31	4.4.10 “i” mark on FIT modules icon added
		32	4.4.11 Configure Analog Front End component added
		34	4.4.12 Configure Motor Component added
		37	4.4.13 Configure general setting of component updated
		40	4.5.2 Pin setting using MCU/MPU package updated
		41	4.5.3 Show pin number from pin functions added
		44	4.5.8 Pin Errors/Warnings setting added
		57	8.2 Using Generated Code in user application added
		59	9.3 Image of MCU/MPU Package (in png Format) updated
		60	10 Help updated
1.30	Apr 16,2023	1	Updated the URL for RX Smart Configurator
		5	2.1 Updated the URL for RX Smart Configurator
		8 - 9	3.3.2 Updated the steps to create a New Configuration File
		39 – 41	4.4.13 Updated the general setting of component
		42	4.4.14 Added the export configuration of component
		42	4.4.15 Added the Import configuration of component
		55 – 57	6.2 Update the file structure and added explaination for the table
		62 – 64	7.2 Update the flow of loading in IAR Embedded Workbench
		65	7.3 Added the build of IAR project file
		71 - 73	10 Added new chapter 10 User code protection feature for Smart Configurator Code Generation Component
1.40	Apr 16, 2024	-	Figures are updated to RX Smart Configurator V2.20
		14	3.4.4 MCU/MPU Package View updated
		23	4.4.2 Removing a component updated
		37 - 40	4.4.12 Configure Motor Component updated
		41	4.4.13 Configure general setting of component updated
		52	4.5.9 Symbolic name setting added
		56	4.6.3 Multiple interrupts setting added
		79	10.3.2 Steps for resolving the merge conflict updated.

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

Trademarks

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

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.