

Smart Configurator

R01AN4422EJ0120

Rev.1.20

Guide on Sample Projects for RH850/F1KM Devices

02 Mar, 2020

Introduction

This document describes, with the use of sample projects, how to build files output from the Smart Configurator for RH850/F1KM devices in the various integrated development environments.

Target Devices

RH850/F1KM-S1, RH850/F1KM-S4

Contents

1. Overview	3
1.1 Purpose	3
1.2 Operating Environment	3
2. Outline of the Sample Projects.....	4
2.1 List of Sample Projects	4
2.2 Notes on the Sample Projects	5
3. Basic Procedures for Operating the Smart Configurator	6
3.1 CS+ Environment	6
3.2 MULTI Environment	7
3.3 IAREW Environment	8
4. Description of the Sample Project for CS+ and CCRH	9
4.1 Configuration of the Sample Project for CS+ and CCRH.....	9
4.2 Basic Operating Procedure.....	10
4.3 Procedure for Changing the Device	13
4.4 Settings in the Sample Project for CS+ and CCRH	16
5. Description of the Sample Project for CS+ and GHS CCRH850	17
5.1 Configuration of the Sample Project for CS+ and GHS CCRH850	17
5.2 Basic Operating Procedure.....	18
5.3 Procedure for Changing the Device	23
5.4 Settings in the Sample Project for CS+ and GHS CCRH850	27
6. Description of the Sample Project for CS+, CCRH, and RI850V4.....	28
6.1 Configuration of the Sample Project for CS+, CCRH, and RI850V4.....	28
6.2 Basic Operating Procedure.....	29

6.3	Procedure for Changing the Device	32
6.4	Settings in the Sample Project for CS+, CCRH, and RI850V4	35
7.	Description of the Sample Project for MULTI and GHS CCRH850	37
7.1	Configuration of the Sample Project for MULTI and GHS CCRH850	37
7.2	Basic Operating Procedure.....	38
7.3	Procedure for Changing the Device	41
7.4	Settings in the Sample Project for MULTI and GHS CCRH850.....	45
8.	Description of the Sample Project for IAREW and IAR ICC.....	46
8.1	Configuration of the Sample Project for IAREW and IAR ICC.....	46
8.2	Basic Operating Procedure.....	47
8.3	Procedure for Changing the Device.....	49
8.4	Settings in the Sample Project for IAREW and IAR ICC	53
9.	Operations in the Smart Configurator.....	54
9.1	Setting the Peripheral Modules (Software Components)	54
9.2	Generating Drivers	57
9.3	Adding the Application Code to the User Code Area.....	57
9.4	Adding the Application Code to main()	58

1. Overview

1.1 Purpose

This document describes, with the use of sample projects, how to build files output from the Smart Configurator for RH850/F1KM devices in the various integrated development environments.

When applying this application note to a microcontroller, change the contents in accord with the specifications of the microcontroller you are using and validate the correct operation of the sample projects.

1.2 Operating Environment

Install the Smart Configurator and tools to be used in order to create or build programs in each integrated development environment based on the source files generated by the Smart Configurator with the use of the sample projects. For details on how to use your integrated development environment, refer to the user's manual for the integrated development environment you are using.

Table 1-1 Operating Environment

Type	Name	Abbreviation in This Manual
IDE	CS+ for CC V7.00.00 or later	CS+
Toolchain	C Compiler Package for RH850 Family	CCRH
Toolchain	Real-time OS for RH850 Family [RI850V4] V2	RI850V4
IDE	MULTI v7.1.6 or later	MULTI
Toolchain	Green Hills Compiler	GHS CCRH850
IDE	IAR Embedded Workbench for RH850 V1.40.5 or later	IAREW
Toolchain	IAR C/C++ Compiler	IAR ICC

Green Hills is a registered trademark of Green Hills Software, Inc. in the United States and other countries.

IAR Embedded Workbench is a registered trademark of IAR Systems.

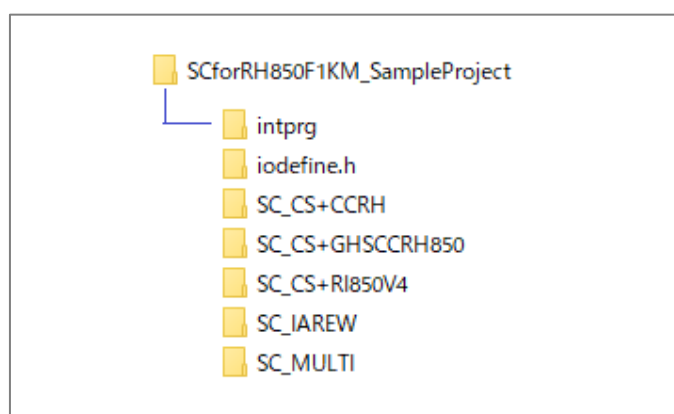
2. Outline of the Sample Projects

The Smart Configurator for RH850/F1KM devices outputs a main function and source files that initialize peripheral modules that were set by components of the Smart Configurator. After the microcontroller has been reset, the initialization processing that is to be performed before execution of the main function and the startup routine that starts the main function and handles other processing are not output.

Therefore, we have prepared sample projects that include startup of the sample projects so that code for peripheral modules according to the settings in the Smart Configurator and user applications employing this code can be built immediately.

2.1 List of Sample Projects

The Smart Configurator for RH850/F1KM devices provides the following sample projects.



For details on the sample projects, see the descriptions in the relevant sections.

Table 2-1 Sample Projects

Explanatory Section	Folder Name	Target Device	Description
Section 2	intprg	—	File defining the EI maskable interrupt vector table for RH850/F1KM-S1 and RH850/F1KM-S4 devices
Section 2	iodefine.h	—	Renesas CCRH header file defining the registers for RH850/F1KM-S1 and RH850/F1KM-S4 devices
Section 4	SC_CS+CCRH	R7F701651 (RH850/F1KM-S4)	Project for CS+ for CC and CCRH compiler
Section 5	SC_CS+GHSCCRH850	R7F701649 (RH850/F1KM-S4)	Project for CS+ for CC and GHS CCRH850 compiler
Section 6	SC_CS+RI850V4	R7F701649 (RH850/F1KM-S4)	Project for CS+ for CC, CCRH compiler, and RI850V4
Section 7	SC_MULTI	R7F701645 (RH850/F1KM-S4)	Project for MULTI and GHS CCRH850 compiler
Section 8	SC_IAREW	R7F701645 (RH850/F1KM-S4)	Project (workspace) for IAREW and IAR ICC compiler

2.2 Notes on the Sample Projects

1. The Smart Configurator outputs the register descriptors according to `iodef.h` for the Renesas CCRH compiler. Though header files of register definitions are also prepared in the GHS CCRH850 and IAR ICC compilers, include `iodef.h` for the Renesas CCRH compiler when building files generated by the Smart Configurator.
2. The Smart Configurator uses interrupts with the table lookup method as the method for selecting the interrupt handler addresses. The address where the table starts is set as `0x00000200` in the sample projects.
3. The definition of the interrupt vector table of peripheral modules that was set in the Smart Configurator is reflected in `smc/general/r_cg_intvector.c`, which is output by the Smart Configurator. The file `sc_intprg-S1.c` or `sc_intprg-S4.c` in the `intprg` folder defines the vector table of EI maskable interrupt sources, which is not set by the Smart Configurator.
4. Settings of files and sections provided with the sample projects are examples. They should be changed or created newly by yourself to match the specifications of the microcontroller in use and the customer's system.

3. Basic Procedures for Operating the Smart Configurator

This section describes the basic operating procedures when building a user application with the files output from the Smart Configurator for RH850/F1KM devices.

The basic operating procedures in each integrated development environment without the use of the sample projects are described here. For the operating procedure when using a sample project, see the relevant section among sections 4 to 8, in accordance with the sample project you will be using.

3.1 CS+ Environment

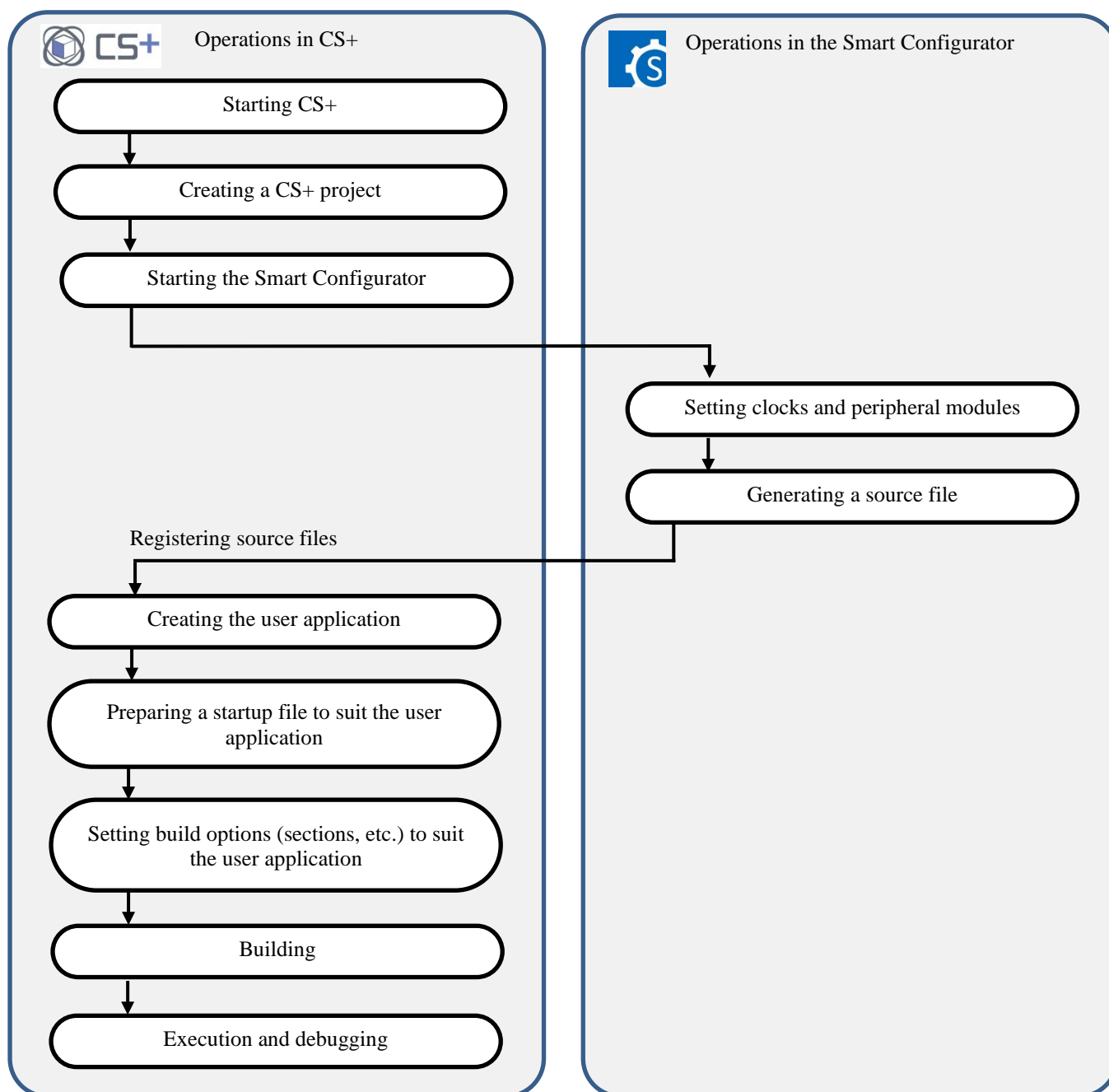


Figure 3.1 Basic Operating Procedure in the CS+ Environment

3.2 MULTI Environment

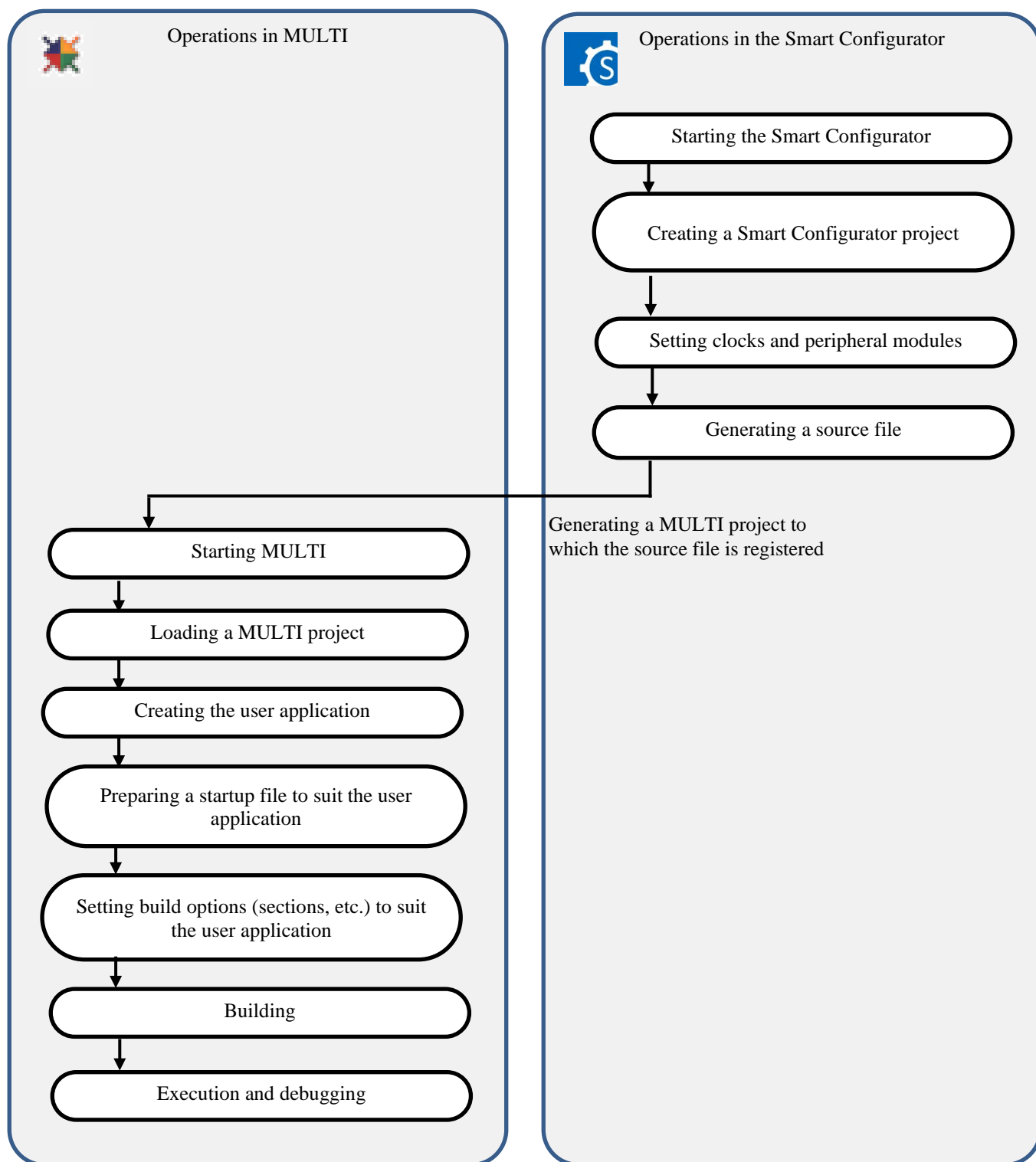


Figure 3.2 Basic Operating Procedure in the MULTI Environment

3.3 IAREW Environment

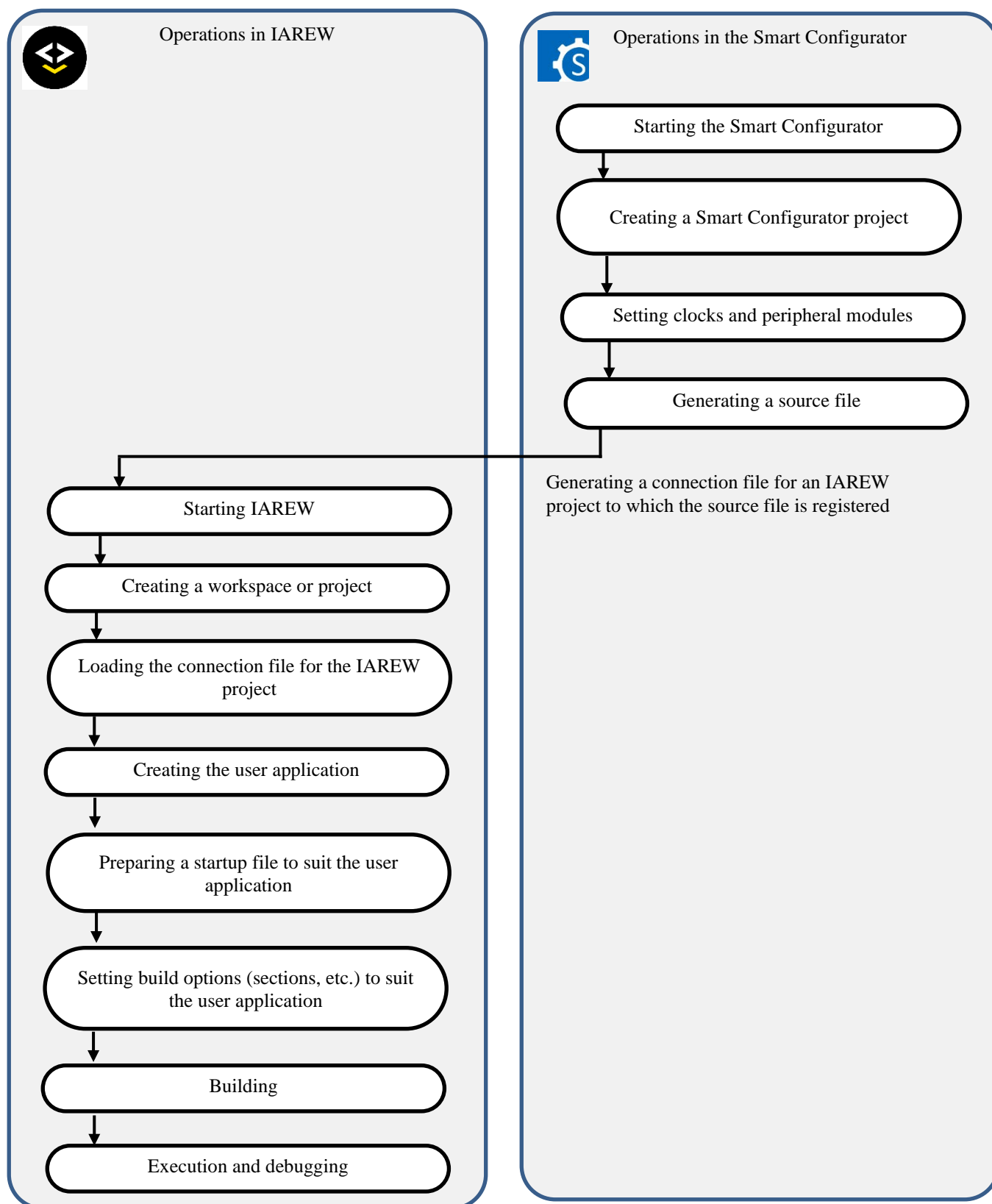


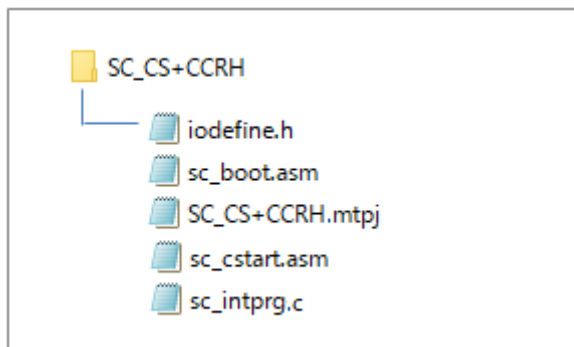
Figure 3.3 Basic Operating Procedure in the IAREW Environment

4. Description of the Sample Project for CS+ and CCRH

4.1 Configuration of the Sample Project for CS+ and CCRH

The following shows the configuration of the sample project.

Table 4-1 File Configuration of the Sample Project for CS+ and CCRH



File Name	Outline of File
SC_CS+CCRH.mtpj	Project file for CS+
sc_boot.asm	Definition of the processing from a reset up to branching to the startup routine and definition of the interrupt vector table
sc_cstart.asm	Definition of the startup routine to be executed until branching to the main function
sc_intprg.c	Definition of the EI-level maskable interrupt vector table
iodefine.h	Definitions of registers

The Smart Configurator does not output the above files.

4.2 Basic Operating Procedure

Figure 4.1 shows the operating procedure when using the Smart Configurator based on a sample project.

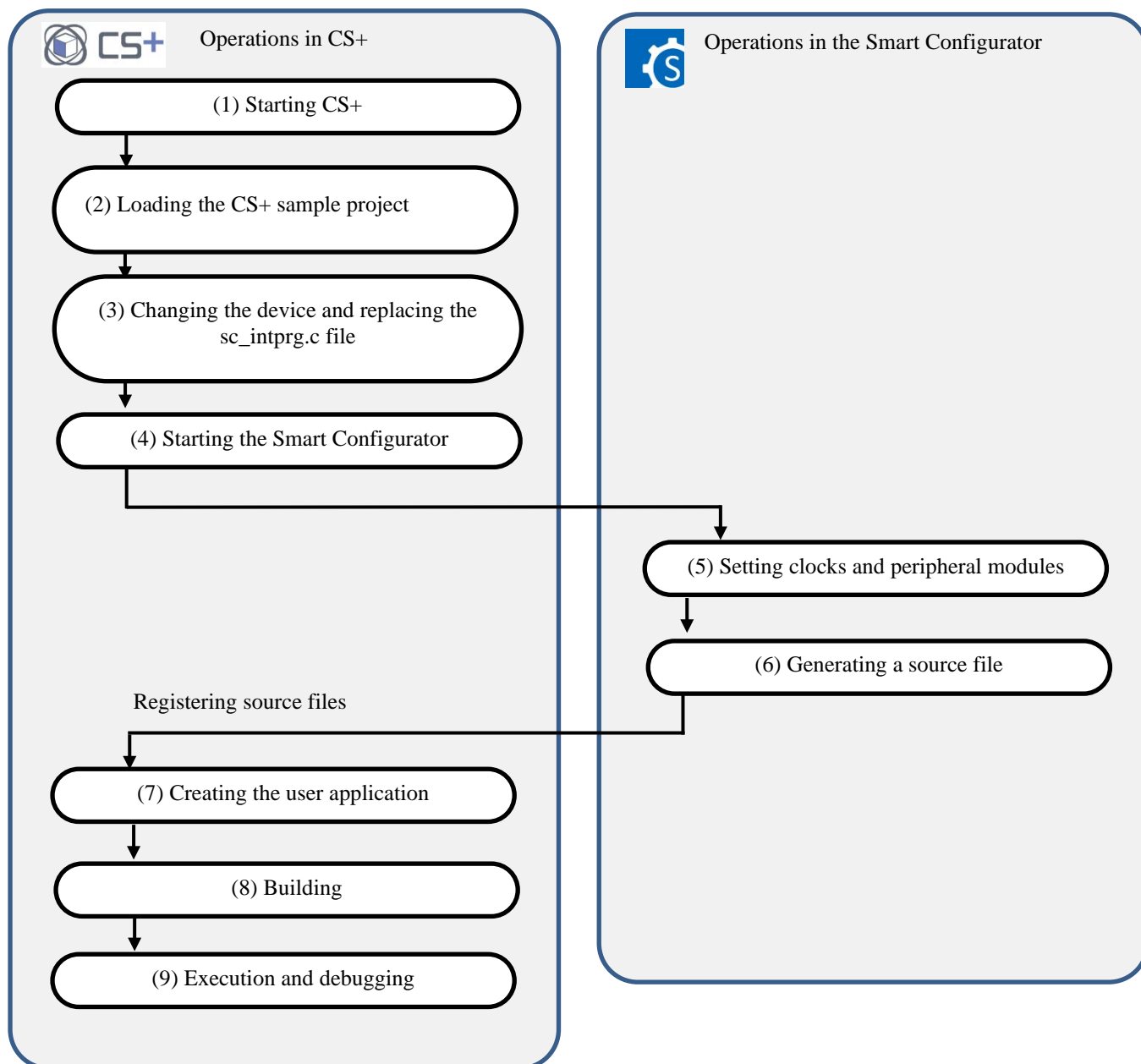


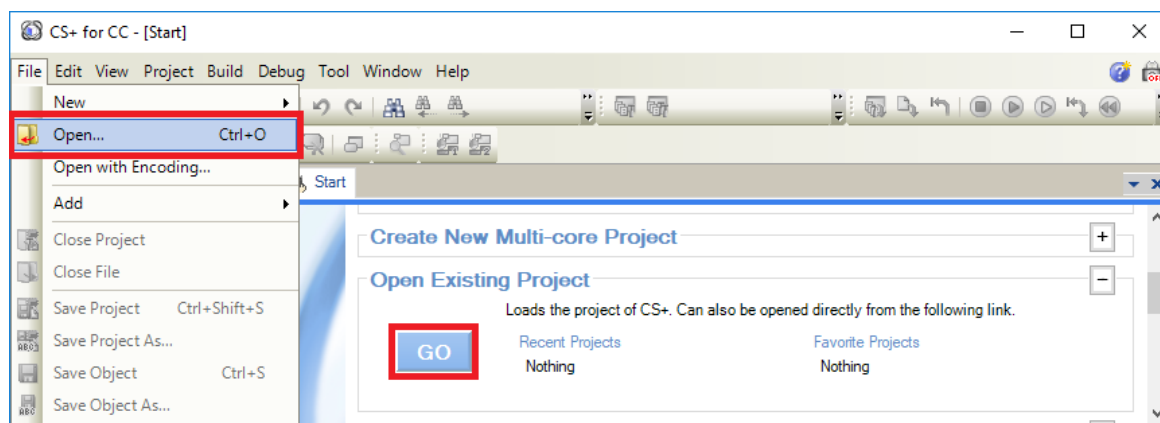
Figure 4.1 Operating Procedure for the Sample Project for CS+ and CCRH

(1) Starting CS+

In the [Start] menu of Windows, select [Renesas Electronics CS+] → [CS+ for CC(RL78,RX,RH850)].

(2) Loading the CS+ sample project

From the [Open...] item of the [File] menu or [Open Existing Project] of CS+, select "SC_CS+CCRH.mtpj".

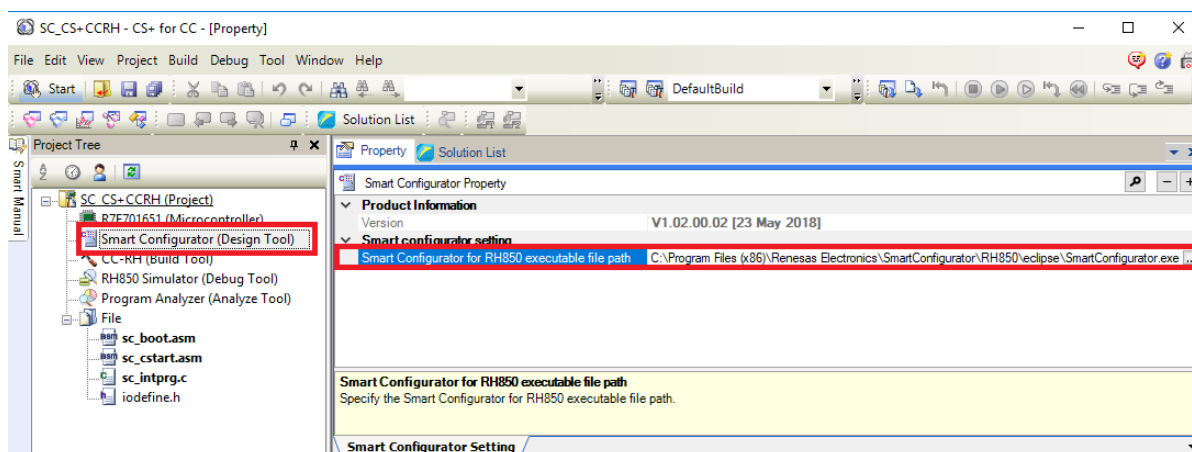


(3) Changing the device

The R7F701651 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the target device and file to be used with reference to section 4.3, "Procedure for Changing the Device". If the device does not require changing, proceed to step (4).

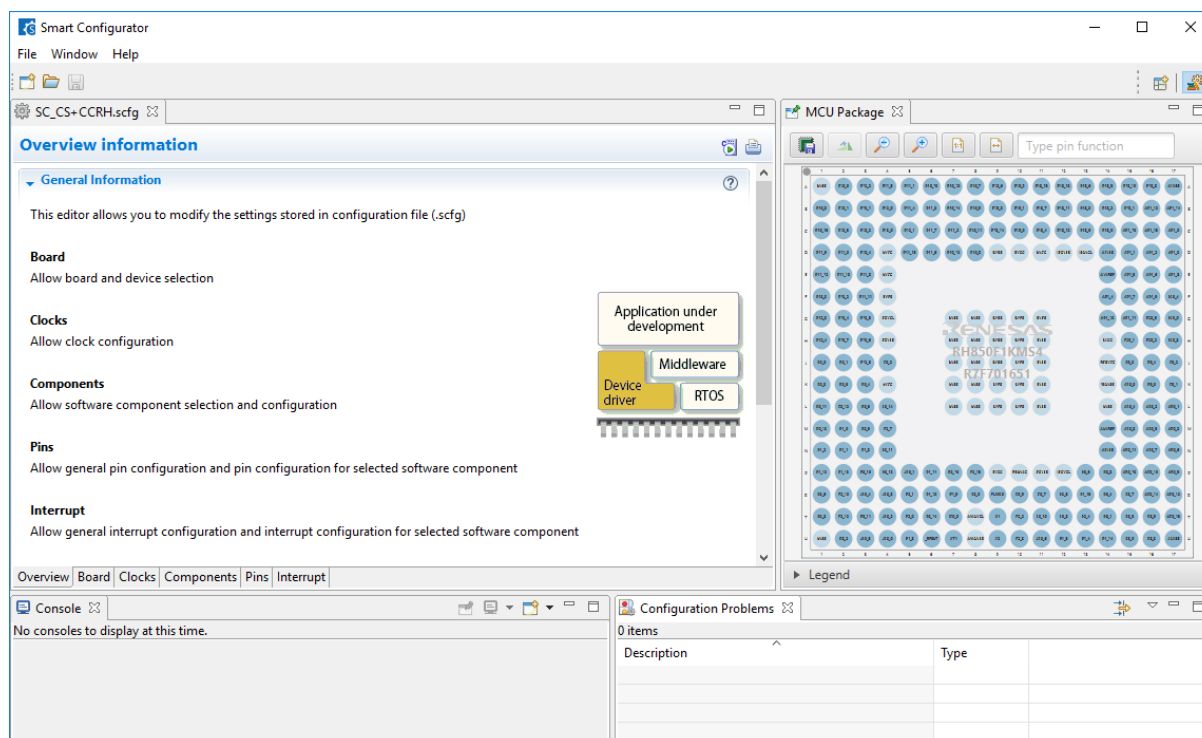
(4) Starting the Smart Configurator

- (4)-1. Confirm the setting of the path for the Smart Configurator for RH850/F1KM devices. In the Project Tree panel, select [Smart Configurator (Design Tool)] and open the [Property] panel. Confirm that the path in which the Smart Configurator for RH850/F1KM devices was installed is set in [Smart Configurator for RH850 executable file path].



- (4)-2. Start the Smart Configurator for RH850/F1KM devices by double-clicking on [Smart Configurator (Design Tool)] in the Project Tree panel.

For steps (5) to (7), the procedure for setting clocks and components and generating a driver in the Smart Configurator, see section 9, "Operations in the Smart Configurator".



(8) Building

Build the driver and application code. Select [Build Project] from the [Build] menu or click on the [Builds the project. (F7)] button  in the toolbar of CS+.

(9) Execution and debugging

For program execution and debugging in the emulator, refer to *CS+ V7.00.00 Integrated Development Environment User's Manual: RH850 Debug Tool* (R20UT4299).

4.3 Procedure for Changing the Device

When the target device of the sample project in Table 2-1, "Sample Projects" differs from the device that is actually to be used, the target device or file to be used must be changed according to the following procedure.

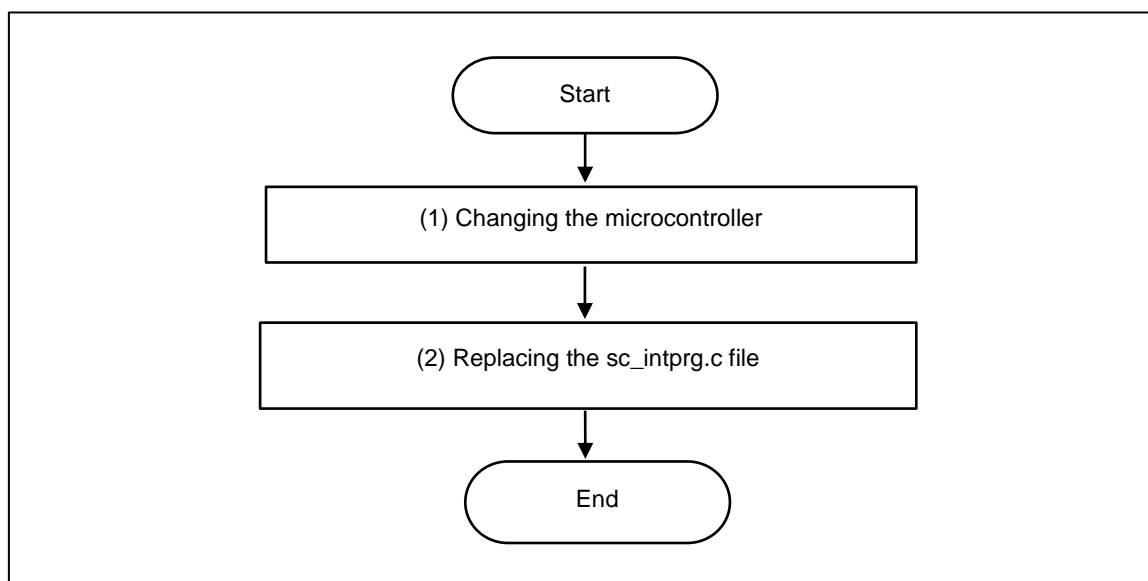
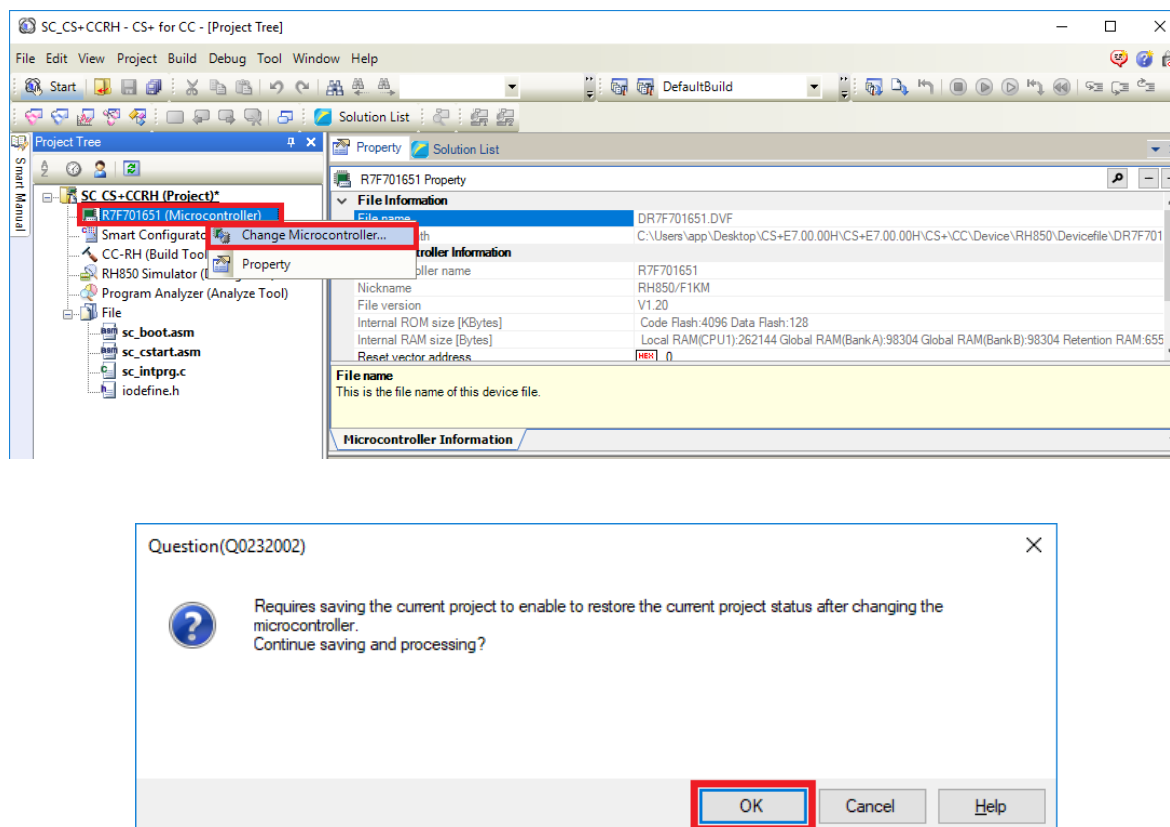


Figure 4.2 Changing the Target Device of the Sample Project (CS+ Project)

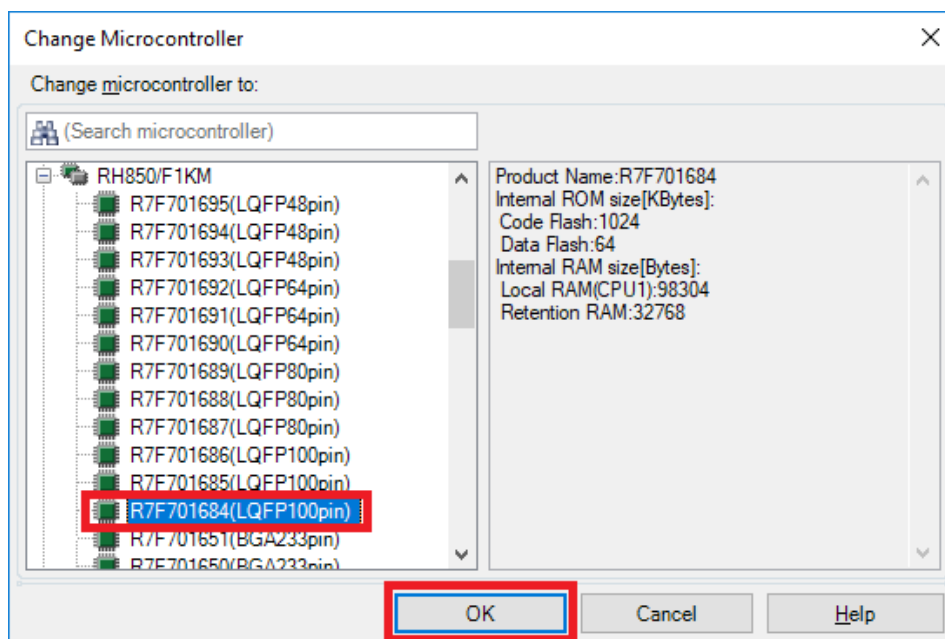
(1) Changing the microcontroller

- (1)-1. Select "R7F701651 (Microcontroller)" and then select [Change Microcontroller...] from the context menu. Click on the [OK] button in the [Question] dialog box that appears.

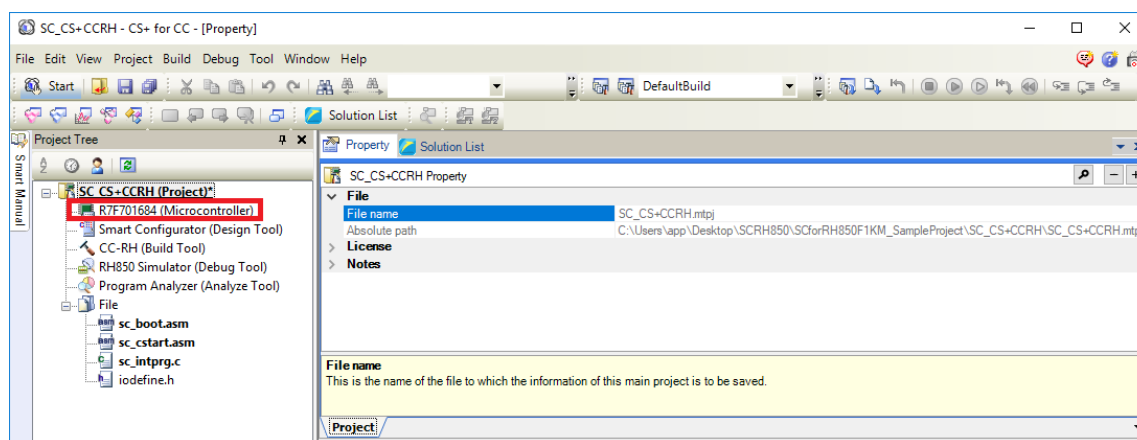


- (1)-2. In the [Change Microcontroller] dialog box, select the RH850/F1KM-S1 or RH850/F1KM-S4 device to be used.

Example: Changing from R7F701651 (RH850/F1KM-S4) to R7F701684 (RH850/F1KM-S1)



- (1)-3. Confirm that the microcontroller displayed in the Project Tree panel has become the device after the change.

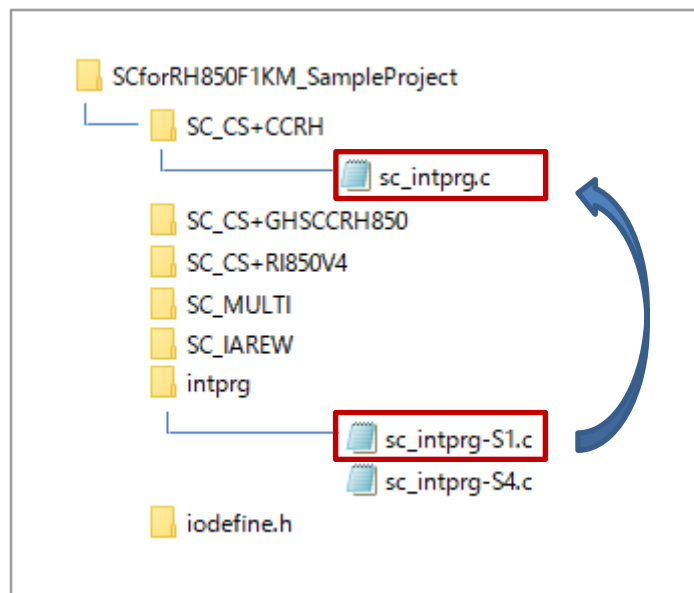


- (1)-4. Save the project by selecting [Save Project] from the [File] menu.

(2) Replacing the sc_intprg.c file

This step is only necessary when a device of the RH850/F1KM-S4 Group was changed to a device of the RH850/F1KM-S1 Group in step (1).

- (2)-1. Since the sc_intprg.c file included in the sample project is for RH850/F1KM-S4 devices, it should be replaced with a file for RH850/F1KM-S1 devices. In Windows Explorer, rename sc_intprg-S1.c in the intprg folder "sc_intprg.c" and replace sc_intprg.c of the sample project with this file.



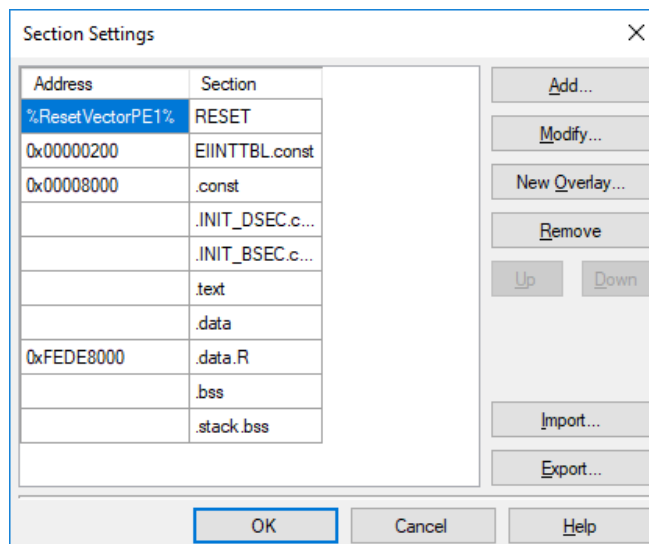
(Files to which the description does not apply are omitted from this figure.)

4.4 Settings in the Sample Project for CS+ and CCRH

The sample project is created in CS+ as an [Empty Application(CC-RH)] project. The include path is added and settings of the following options are changed.

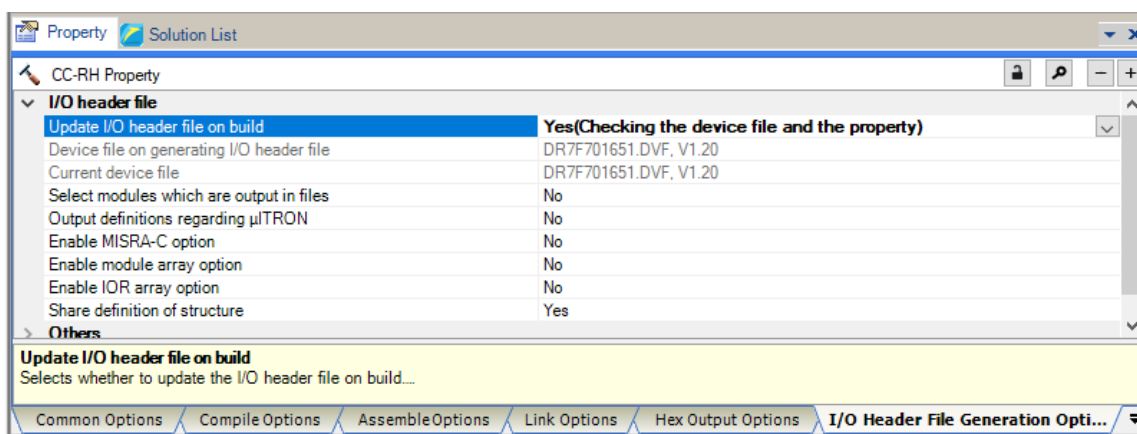
- (a) [Property] panel from [CC-RH (Build Tool)] → [Link Options] tab → [Section] → [Section start address]

[Settings in the sample project]



- (b) [Property] panel from [CC-RH (Build Tool)] → [I/O Header File Generation Options] tab → [I/O Header File] → [Update I/O header file on build]

[Settings in the sample project]

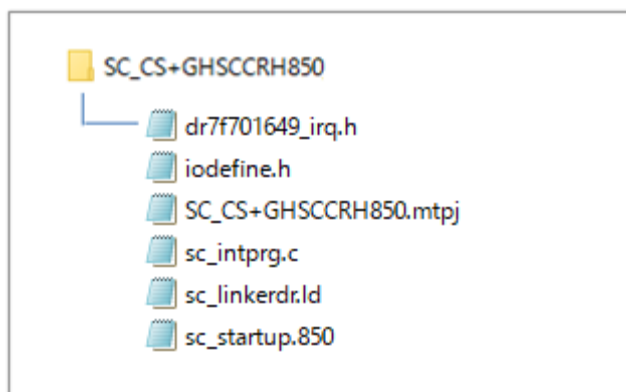


5. Description of the Sample Project for CS+ and GHS CCRH850

5.1 Configuration of the Sample Project for CS+ and GHS CCRH850

The following shows the configuration of the sample project.

Table 5-1 File Configuration of the Sample Project for CS+ and GHS CCRH850



File Name	Outline of File
SC_CS+GHSCCRH850.mtpj	Project file for CS+
sc_startup.850	Definition of the processing from a reset up to branching to the given application project, definition of the exception interrupt vector table, and definition of the startup routine up to branching to the main function
dr7f701649_irq.h	Definition of the exception interrupt vector table (macro definitions)
sc_linkerdr.ld	Linker directive file
sc_intprg.c	Definition of the EI-level maskable interrupt vector table
iodefine.h	Definitions of registers

The Smart Configurator does not output the above files.

5.2 Basic Operating Procedure

Figure 5.1 shows the operating procedure when using the Smart Configurator based on a sample project.

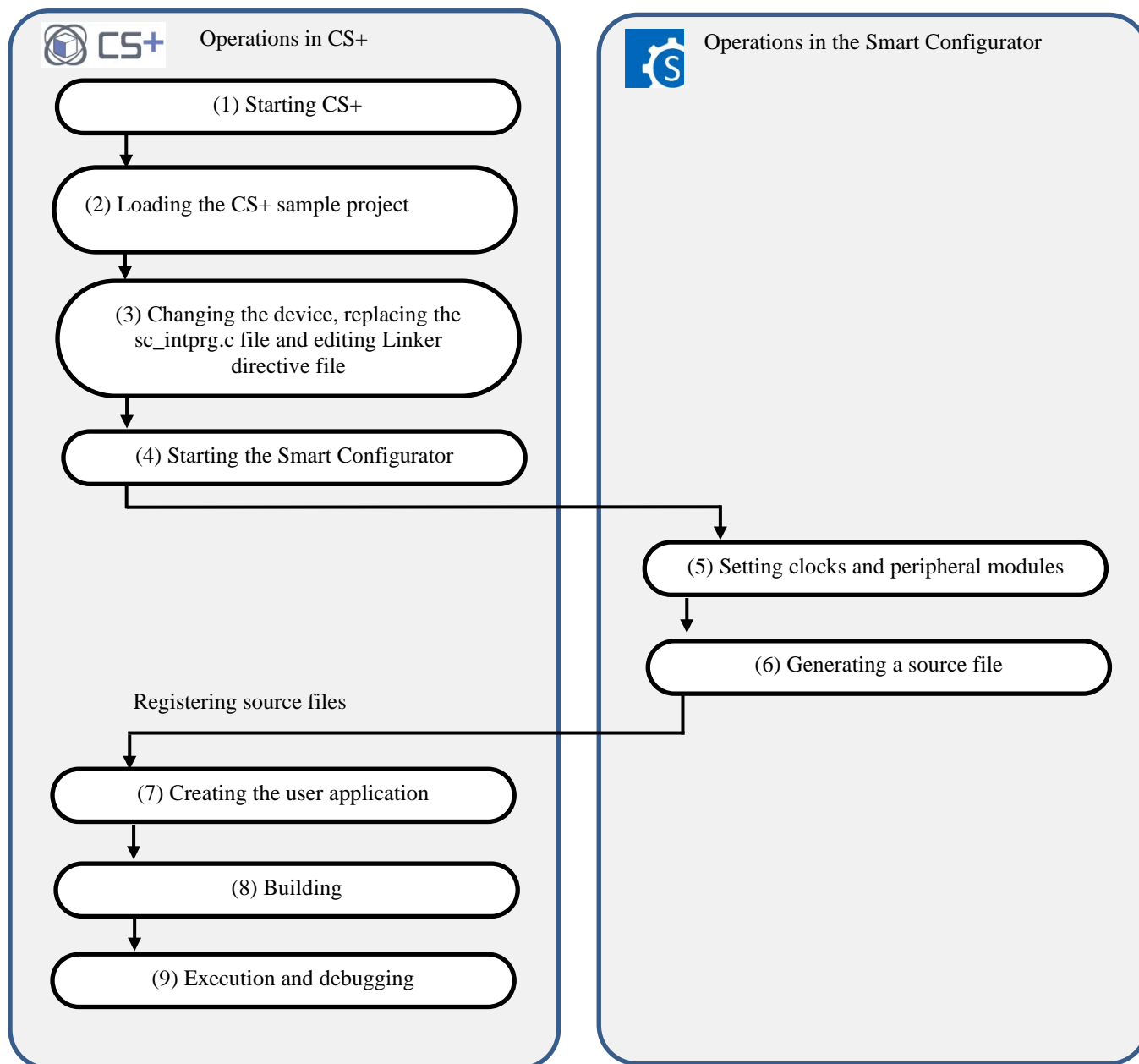


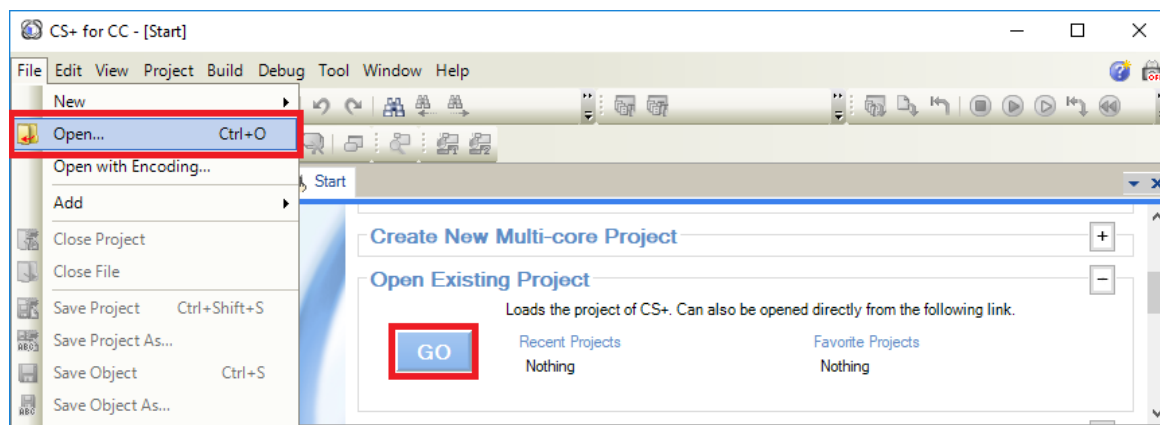
Figure 5.1 Operating Procedure for the Sample Project for CS+ and GHS CCRH850

(1) Starting CS+

In the [Start] menu of Windows, select [Renesas Electronics CS+] → [CS+ for CC(RL78,RX,RH850)].

(2) Loading the CS+ sample project

From the [Open...] item of the [File] menu or [Open Existing Project] of CS+, select "SC_CS+GHSCCRH850.mtpj".

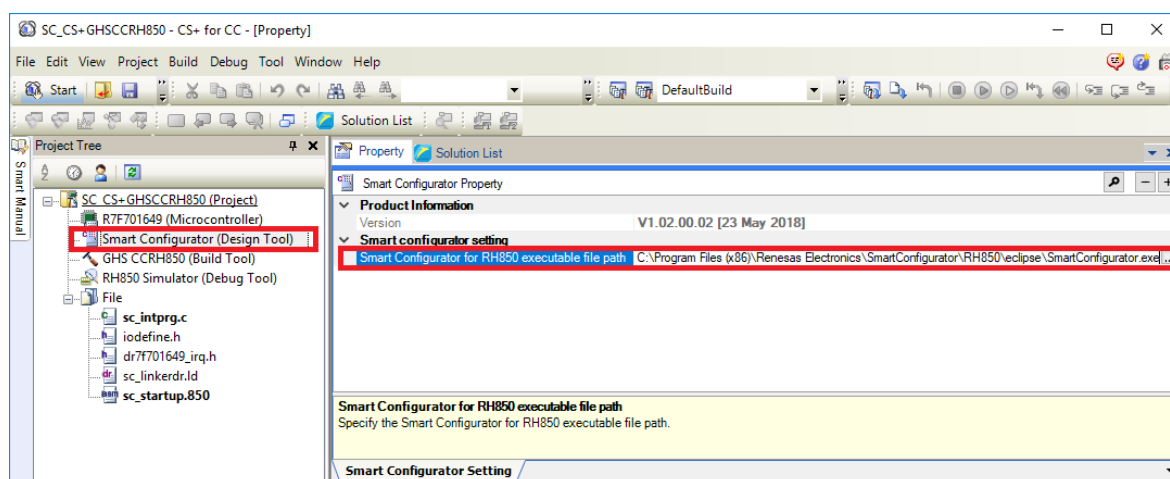


(3) Changing the device

The R7F701649 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the target device and file to be used with reference to section 5.3, "Procedure for Changing the Device". If the device does not require changing, proceed to step (4).

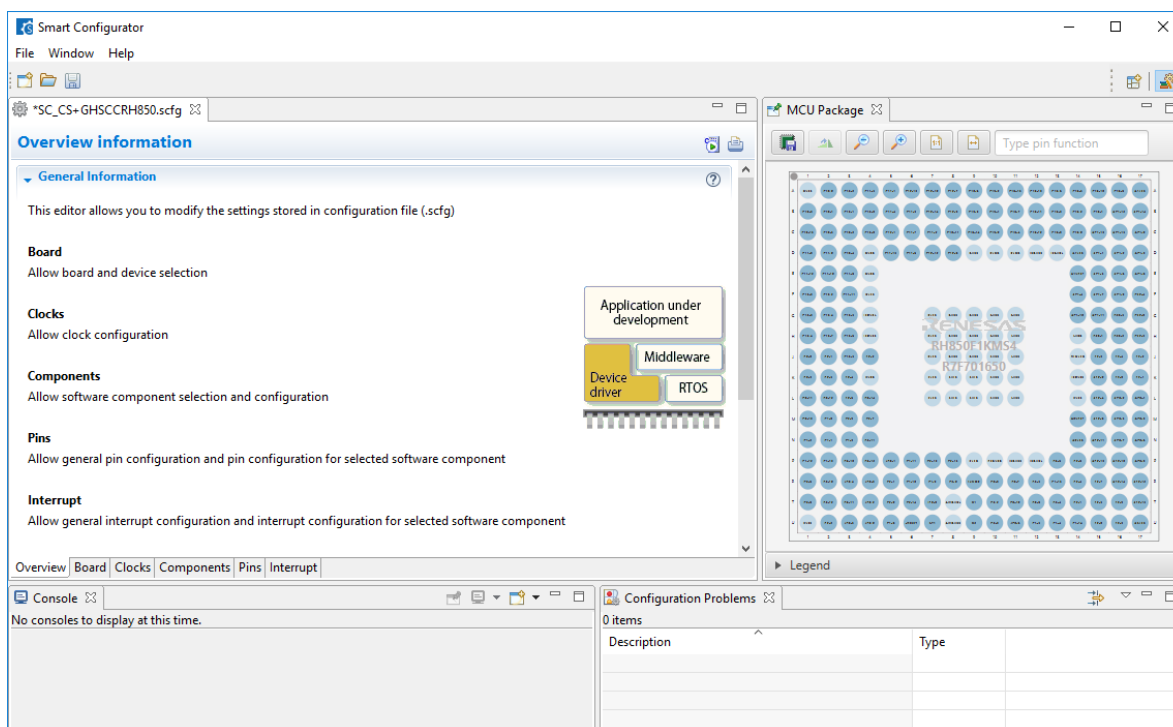
(4) Starting the Smart Configurator

- (4)-1. Confirm the setting of the path for the Smart Configurator for RH850/F1KM devices. In the Project Tree panel, select [Smart Configurator (Design Tool)] and open the [Property] panel. Confirm that the path in which the Smart Configurator for RH850/F1KM devices was installed is set in [Smart Configurator for RH850 executable file path].



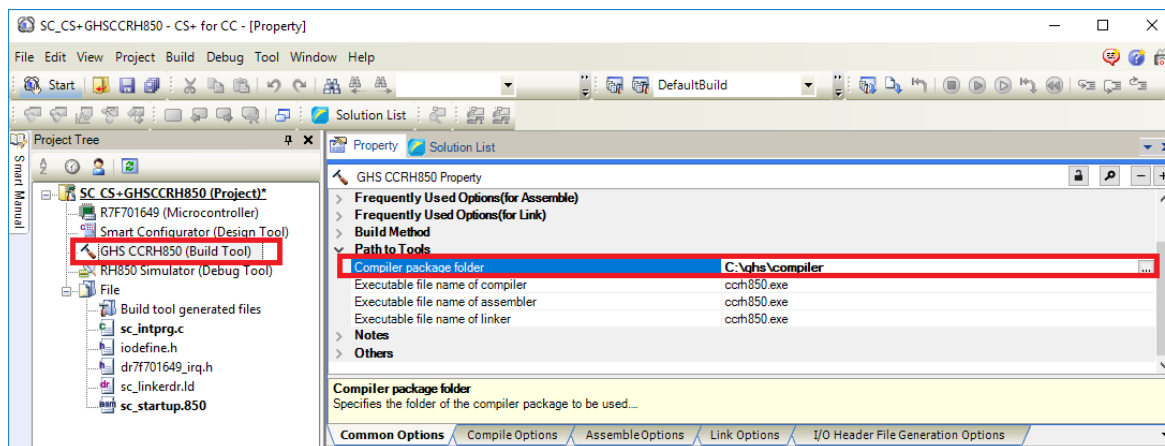
- (4)-2. Start the Smart Configurator for RH850/F1KM devices by double-clicking on [Smart Configurator (Design Tool)] in the Project Tree panel.


For steps (5) to (7), the procedure for setting clocks and components and generating a driver in the Smart Configurator, see section 9, "Operations in the Smart Configurator".

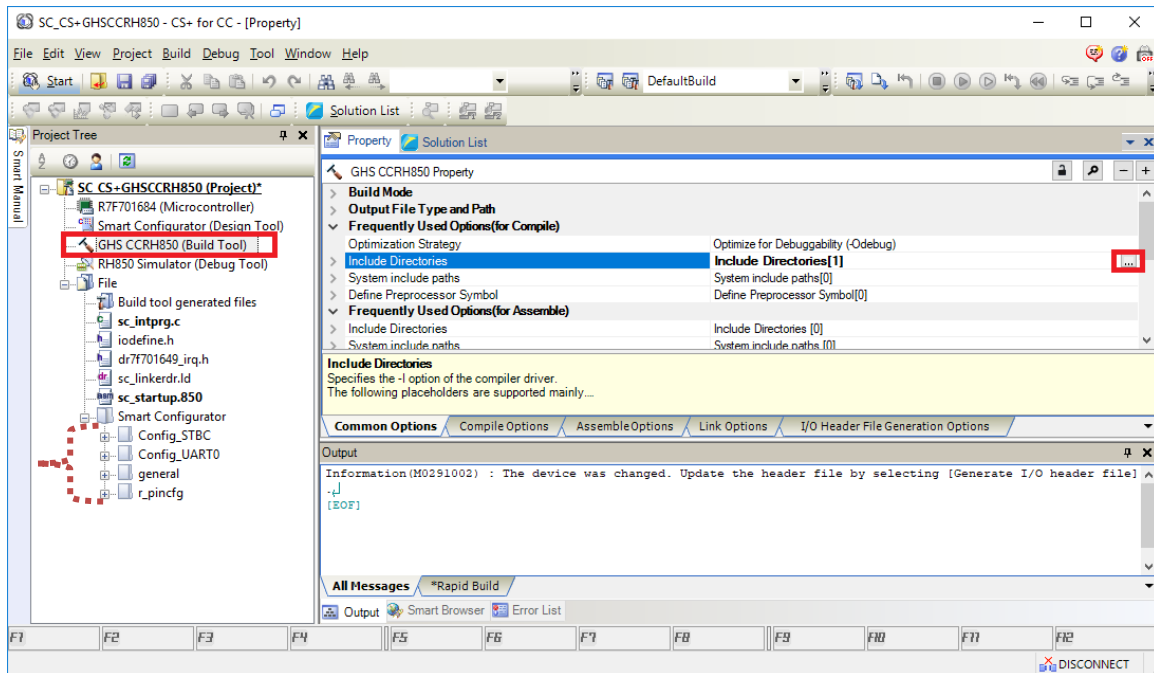


(8) Building

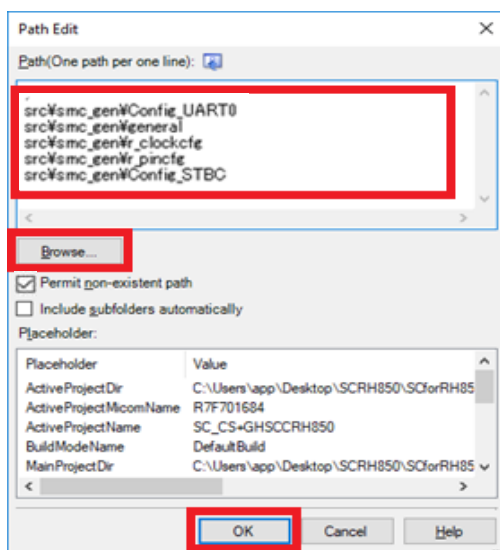
- (8)-1. Set the path for the GHS CCRH850 compiler. In the Project Tree panel, select [GHS CCRH850 (Build Tool)] and open the [Property] panel. Set the path to where the GHS CCRH850 compiler was installed in [Compiler package folder] on the [Common Options] tabbed page.




- (8)-2. Add the folder containing the files generated by the Smart Configurator for RH850/F1KM devices to the include path. In the Project Tree panel, select [GHS CCRH850 (Build Tool)] and open the [Property] panel. Click on the  button in [Include Directories] on the [Common Options] tabbed page.



Click on the [Browse...] button in the [Path Edit] dialog box and add iodefne.h and the folder containing the files generated by the Smart Configurator.



Remarks: Smart Configurator creates “src/smc_gen” folder under the folder where the sample project is located and generates files in “src/smc_gen”. When adding the path in the [Path Edit] dialog box, related subfolders under “src/smc_gen” are added by specified “src/smc_gen” after checking [Include subfolders automatically].

- (8)-3. Build the driver and application code. Select [Build Project] from the [Build] menu or click on the [Builds the project. (F7)] button  in the toolbar of CS+.

(9) Execution and debugging

For program execution and debugging in the emulator, refer to *CS+ V7.00.00 Integrated Development Environment User's Manual: RH850 Debug Tool* (R20UT4299).

5.3 Procedure for Changing the Device

When the target device of the sample project in Table 2-1, "Sample Projects" differs from the device that is actually to be used, the target device or file to be used must be changed according to the following procedure.

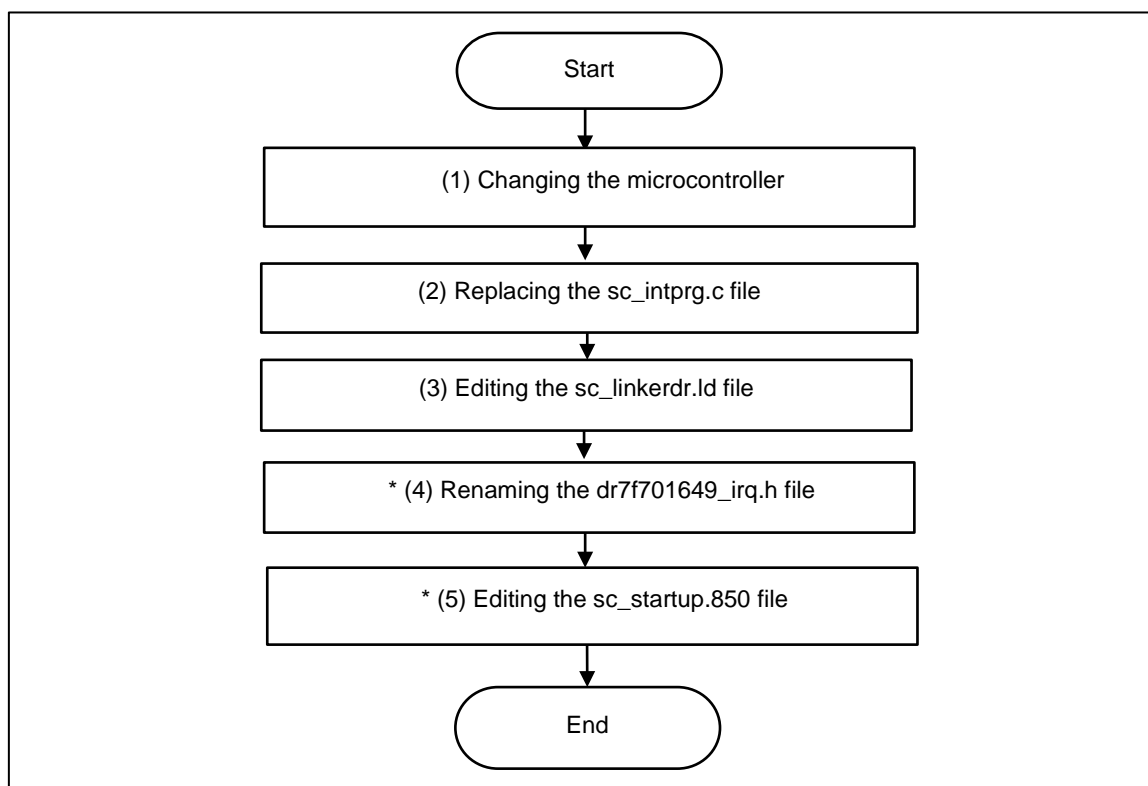
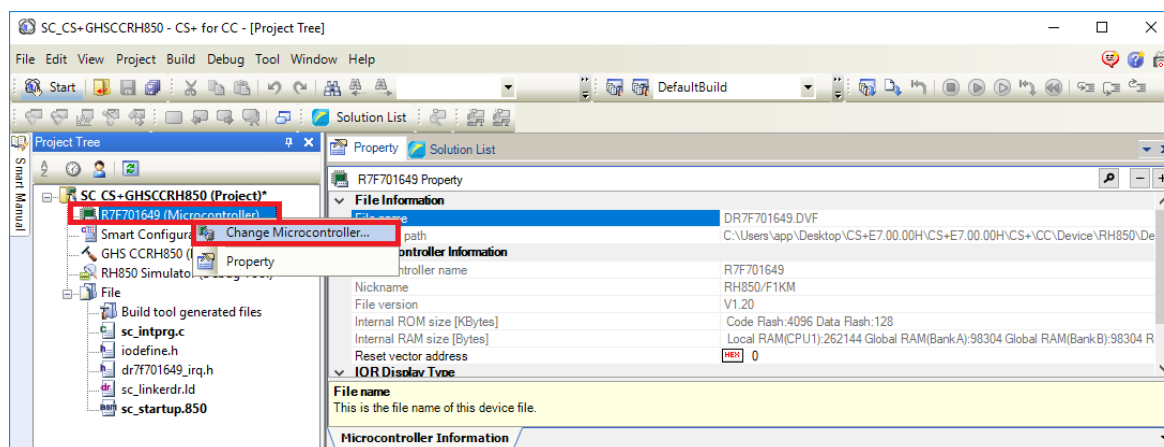


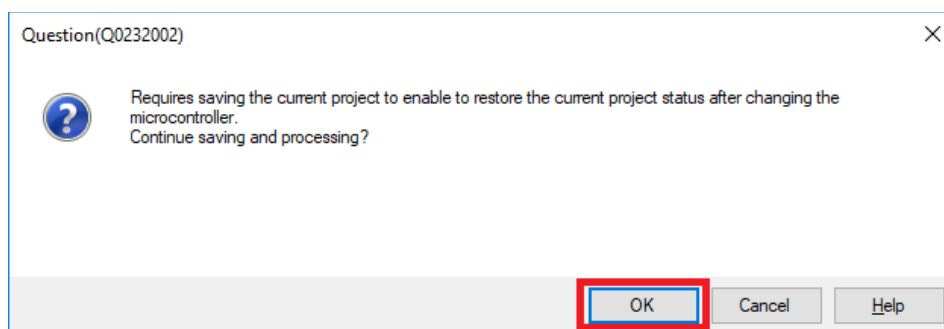
Figure 5.2 Changing the Target Device of the Sample Project (CS+ Project)

*: Steps (4) and (5) are optional.

(1) Changing the microcontroller

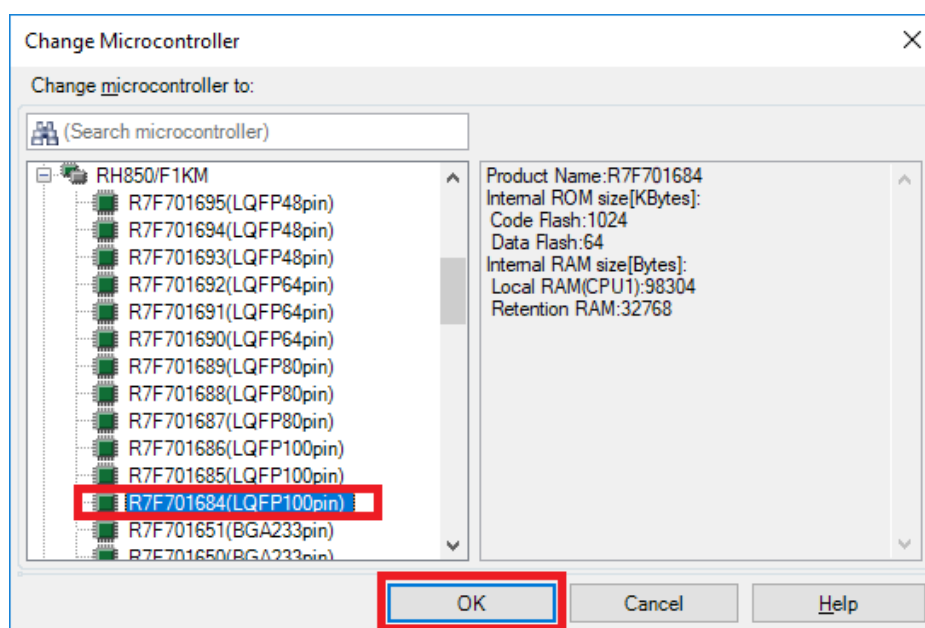
- (1)-1. Select "R7F701649 (Microcontroller)" and then select [Change Microcontroller...] from the context menu. Click on the [OK] button in the [Question] dialog box that appears.



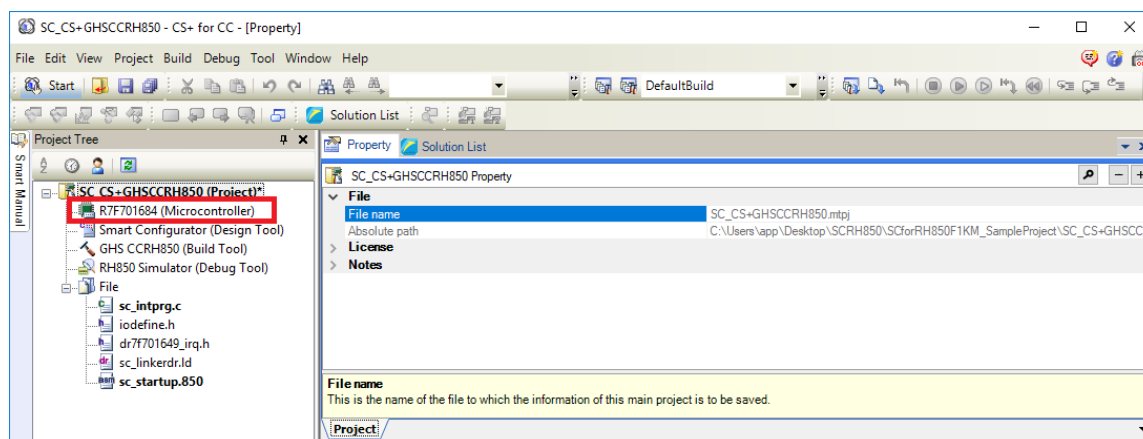


(1)-2. In the [Change Microcontroller] dialog box, select the RH850/F1KM-S1 or RH850/F1KM-S4 device to be used.

Example: Changing from R7F701649 (RH850/F1KM-S4) to R7F701684 (RH850/F1KM-S1)



(1)-3. Confirm that the microcontroller displayed in the Project Tree panel has become the device after the change.

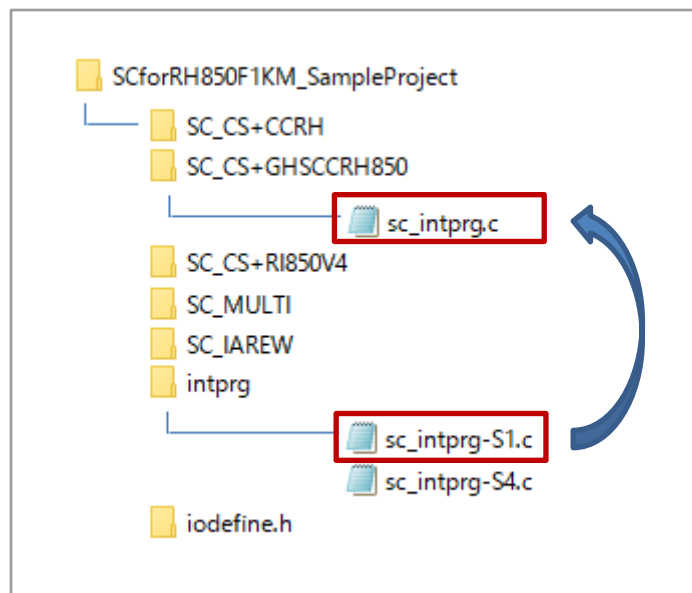


(1)-4. Save the project by selecting [Save Project] from the [File] menu.

(2) Replacing the sc_intprg.c file

This step is only necessary when a device of the RH850/F1KM-S4 Group was changed to a device of the RH850/F1KM-S1 Group in step (1).

- (2)-1. Since the sc_intprg.c file included in the sample project is for RH850/F1KM-S4 devices, it should be replaced with a file for RH850/F1KM-S1 devices. In Windows Explorer, rename sc_intprg-S1.c in the intprg folder "sc_intprg.c" and replace sc_intprg.c of the sample project with this file.



(Files to which the description does not apply are omitted from this figure.)

(3) Editing the sc_linkerdr.ld file

- (3)-1. Open the "sc_linkerdr.ld" in any editor to edit it.

- (3)-2. Change the start address and size of memories to match the specifications of the microcontroller in use.
(Renaming microcontroller name in the file header is optional.)

```

44 /*
45 /*=====
46 /* Environment:
47 /*      Device: R7F701849
48 /*      IDE: GH8 Multi for V800 V6.xx or late
49 /*=====
50
51 CONSTANTS
52 {
53     zero_start = 0xffff8000
54 }
55
56 MEMORY
57 {
58     iROM_0      : ORIGIN = 0x00000000, LENGTH = 4096k
59     iROM_1      : ORIGIN = 0x01000000, LENGTH = 32k
60     iRAM_0      : ORIGIN = 0xFEB00000, LENGTH = 256k
61     iRAM_R      : ORIGIN = 0xFE000000, LENGTH = 64k
62 }

```

- (3)-3. Change the "EIINTTBL_end" that is the end address of EIINTTBL section to match the specifications of the microcontroller in use.

```

64 SECTIONS
65 {
66
67 /* Start of internal ROM area (iROM_0) */
68
69 /* original */
70 /* .intvect          :>iROM_0 /* start of inter
71 /* .intvect_end 0x07E4 :>. /* end of inter
72 /* */
73
74 /* Changed for sample project */
75 .intvect          :>iROM_0 /* start of interr
76 .intvect_end 0x01FF :>. /* end of interr
77 EIINTTBL align (4) :>iROM_0 /* start of EI mask
78 EIINTTBL_end 0x07E4 :>. /* end of EI mask
79 /* */

```

Reference value:

RH850/F1KM-S1: 0x0798

RH850/F1KM-S4: 0x07E4

The following steps (4) and (5) are optional. Build of the sample project is possible without steps (4) and (5).

(4) Renaming the dr7f701649_irq.h file

(4)-1. In Windows Explorer, rename from “dr7f01649_irq.h” to “dr7f0xxx_irq.h” to match the microcontroller name in use.

(5) Editing the sc_startup.850 file

(5)-1. Open the “sc_startup.850” in any editor to edit it.

(5)-2. Change the include file name to the file name changed in step (4) and save the sc_startup.850 file. (Renaming microcontroller name in the file header is optional.)

```

45 -----
46 -- Environment:
47 --      Device: R7F701649
48 --      IDE:    GHS Multi for
49 -----
50
51 -----
52 ----- Selection of external interrupt
53 ----- User modifiable section
54 ----- Please uncomment the required
55 -----
56 #include "dr7f701649_IRQ.h"
57

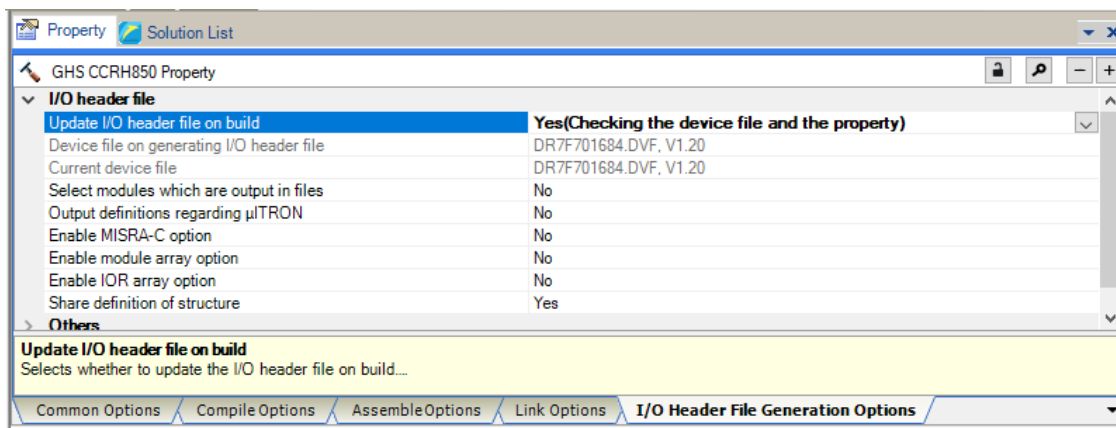
```

5.4 Settings in the Sample Project for CS+ and GHS CCRH850

The sample project is created in CS+ as an [Empty Application(GHS CCRH850)] project. The include path is added and settings of the following options are changed.

- (a) [Property] panel from [GHS CCRH850 (Build Tool)] → [I/O Header File Generation Options] tab → [I/O Header File] → [Update I/O header file on build]

[Settings in the sample project]

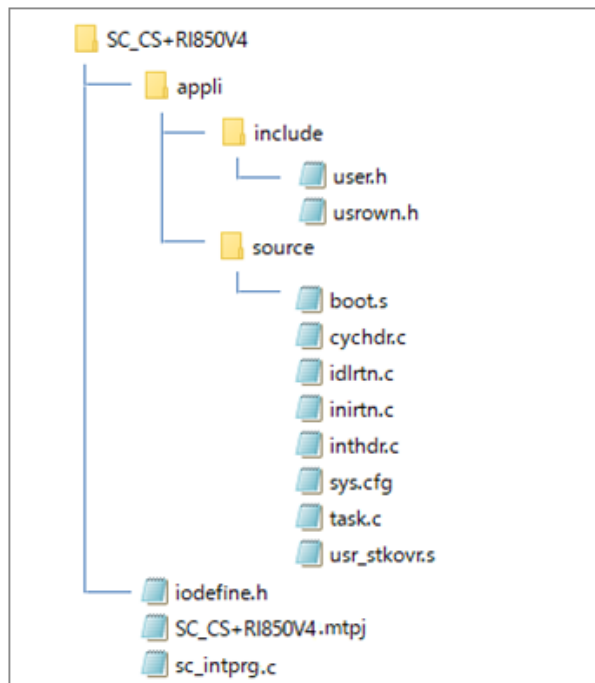


6. Description of the Sample Project for CS+, CCRH, and RI850V4

6.1 Configuration of the Sample Project for CS+, CCRH, and RI850V4

The following shows the configuration of the sample project.

Table 6-1 File Configuration of the Sample Project for CS+, CCRH, and RI850V4



File Name	Outline of File
SC_CS+RI850V4.mtpj	Project file for CS+
user.h	System-dependent information (for the user-coded part)
usr_rown.h	System-dependent information (for the user-coded part)
boot.s	Boot processing
cychdr.c	Cyclic handler
idlrtn.c	Idle routine
inirtn.c	Initialization routine
inthdr.c	Interrupt handler
task.c	Task
usr_stkovr.s	Overflow post-processing
sys.cfg	System configuration file
iodefine.h	Definitions of registers
sc_intprg.c	Definition of the EI-level maskable interrupt vector table

The Smart Configurator does not output the above files.

For details on the RI850V4 related files, refer to *RI850V4 V2 Real-Time Operating System User's Manual: Coding* (R20UT2889).

6.2 Basic Operating Procedure

Figure 6.1 shows the operating procedure when using the Smart Configurator based on a sample project.

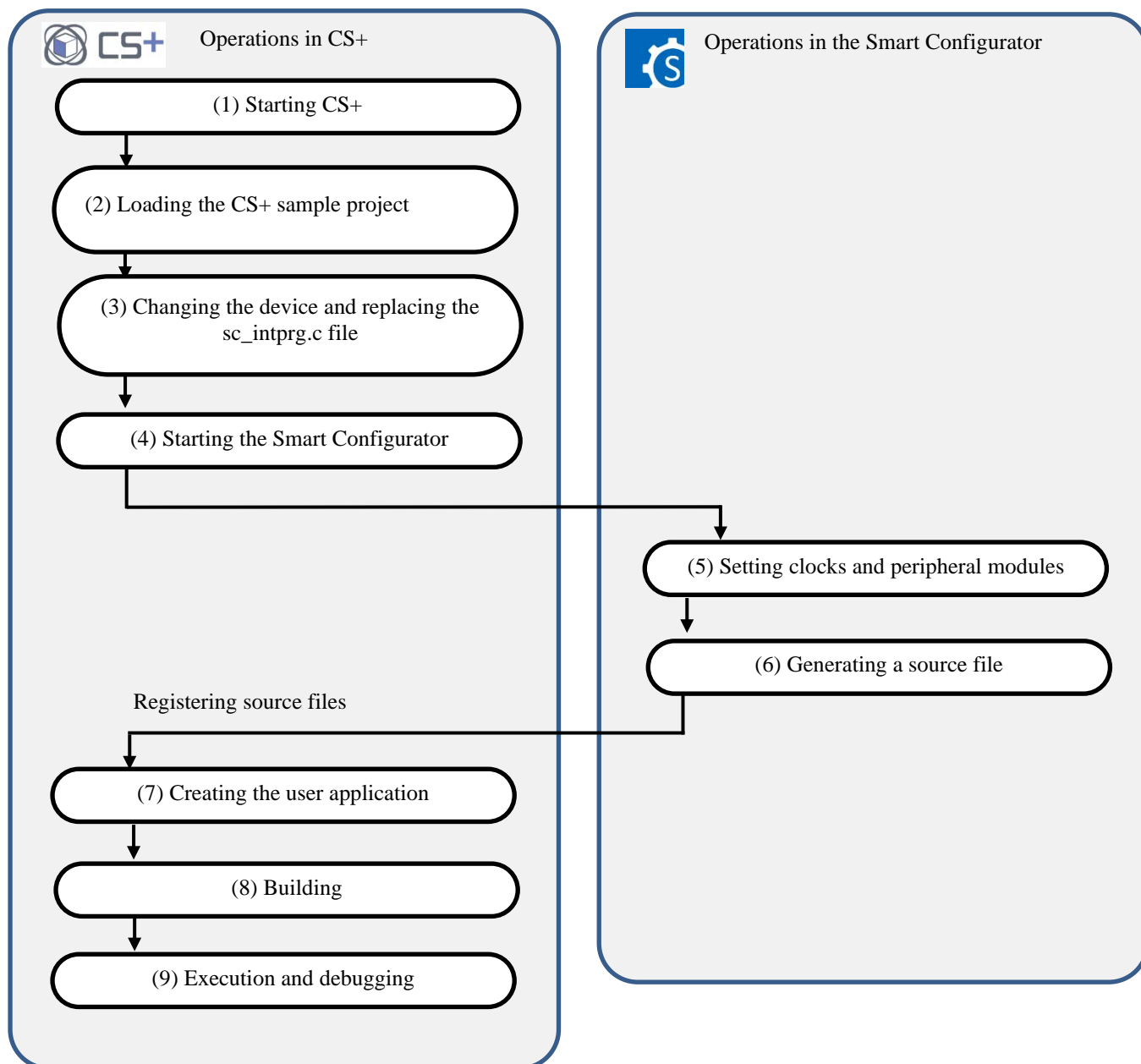


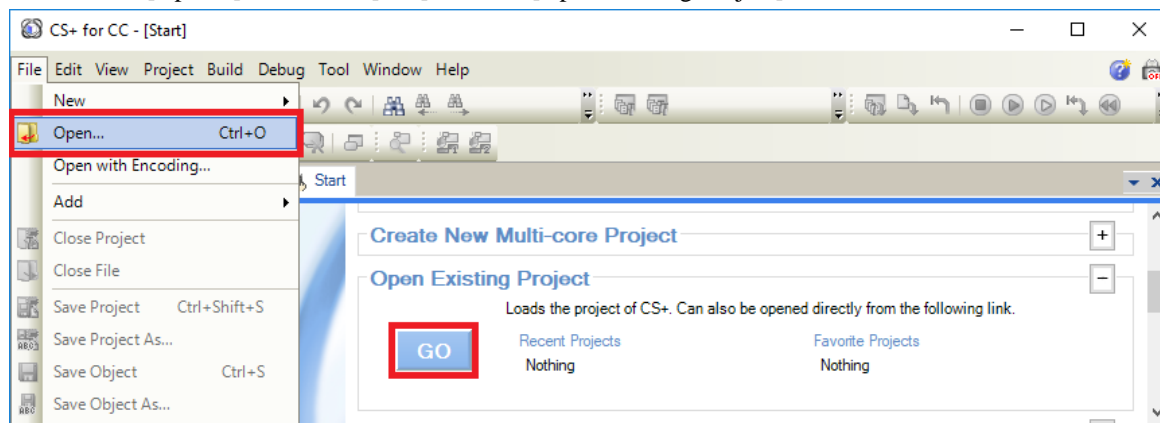
Figure 6.1 Operating Procedure for the Sample Project for CS+, CCRH, and RI850V4

(1) Starting CS+

In the [Start] menu of Windows, select [Renesas Electronics CS+] → [CS+ for CC(RL78,RX,RH850)].

(2) Loading the CS+ sample project

From the [Open...] item of the [File] menu or [Open Existing Project] of CS+, select "SC_CS+RH850V4.mtpj".

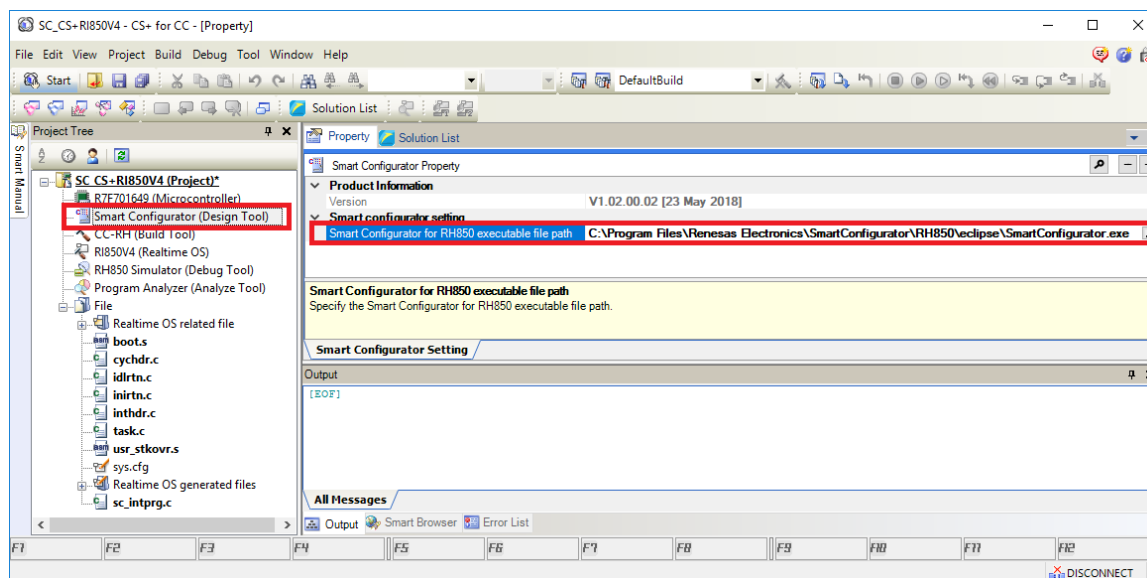


(3) Changing the device

The R7F701649 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the target device and file to be used with reference to section 6.3, "Procedure for Changing the Device". If the device does not require changing, proceed to step (4).

(4) Starting the Smart Configurator

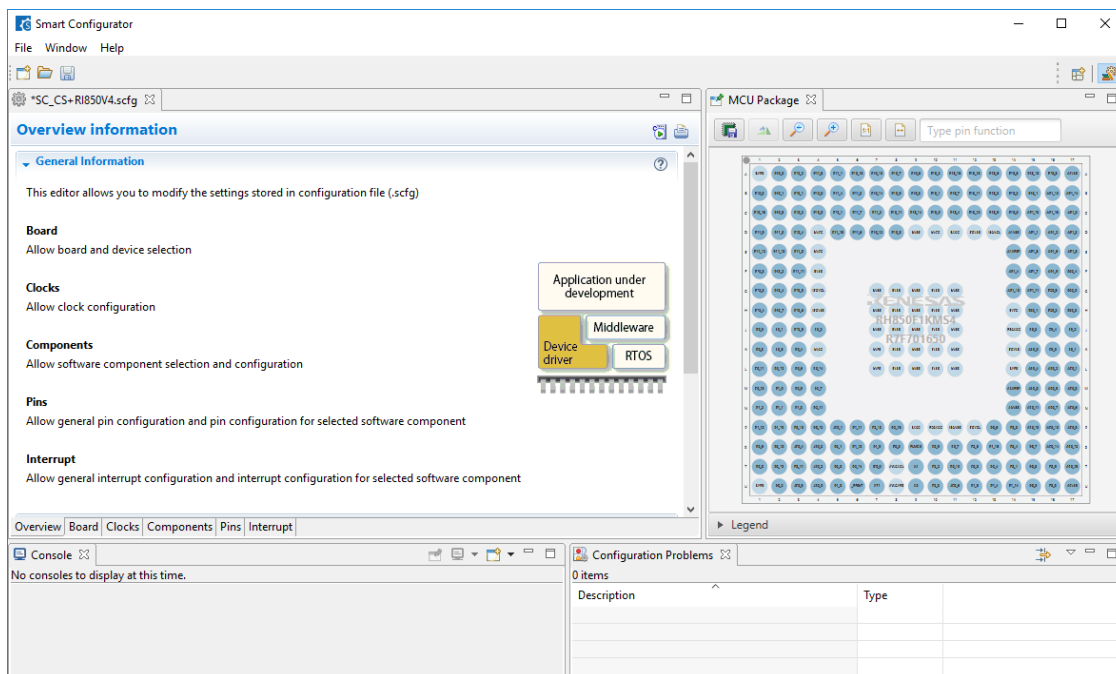
- (4)-1. Confirm the setting of the path for the Smart Configurator for RH850/F1KM devices. In the Project Tree panel, select [Smart Configurator (Design Tool)] and open the [Property] panel. Confirm that the path in which the Smart Configurator for RH850/F1KM devices was installed is set in [Smart Configurator for RH850 executable file path].



- (4)-2. Start the Smart Configurator for RH850/F1KM devices by double-clicking on [Smart Configurator (Design Tool)] in the Project Tree panel.

For steps (5) to (7), the procedure for setting clocks and components and generating a driver in the Smart Configurator, see section 9, "Operations in the Smart Configurator".

Note: Do not use the OS timer (OSTM) in the component settings of the Smart Configurator. This is because the file "appli/source/inirtn.c" of the sample project contains the code for setting the OS timer.



(8) Building

Build the driver and application code. Select [Build Project] from the [Build] menu or click on the [Builds the project. (F7)] button  in the toolbar of CS+.

(9) Execution and debugging

For program execution and debugging in the emulator, refer to *CS+ V7.00.00 Integrated Development Environment User's Manual: RH850 Debug Tool (R20UT4299)*.

6.3 Procedure for Changing the Device

When the target device of the sample project in Table 2-1, "Sample Projects" differs from the device that is actually to be used, the target device or file to be used must be changed according to the following procedure.

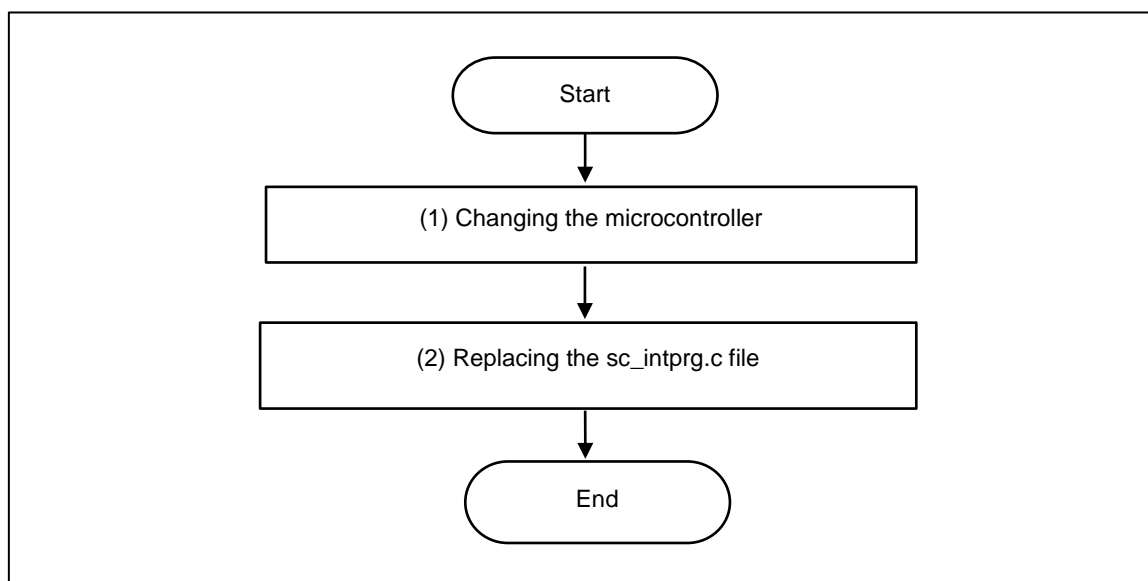
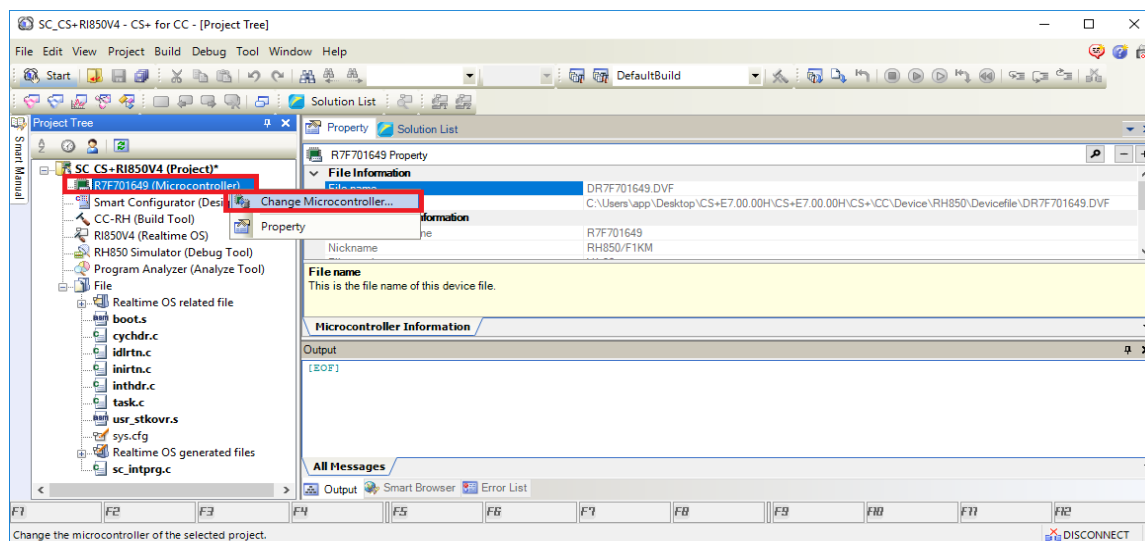
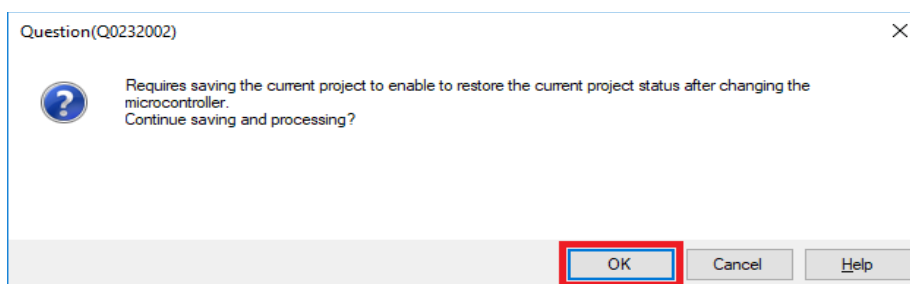


Figure 6.2 Changing the Target Device of the Sample Project (CS+ Project)

(1) Changing the microcontroller

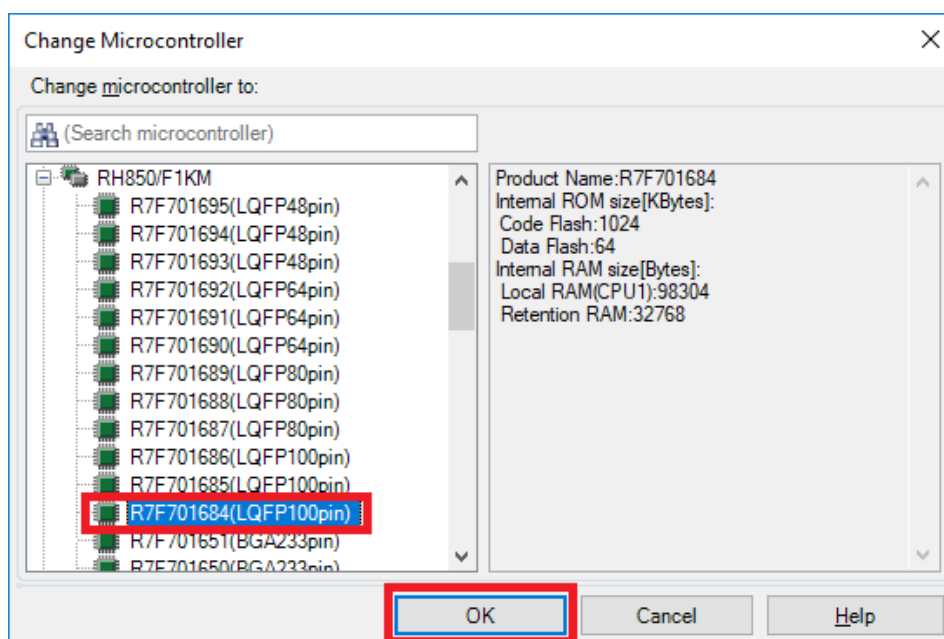
- (1)-1. Select "R7F701649 (Microcontroller)" and then select [Change Microcontroller...] from the context menu. Click on the [OK] button in the [Question] dialog box that appears.



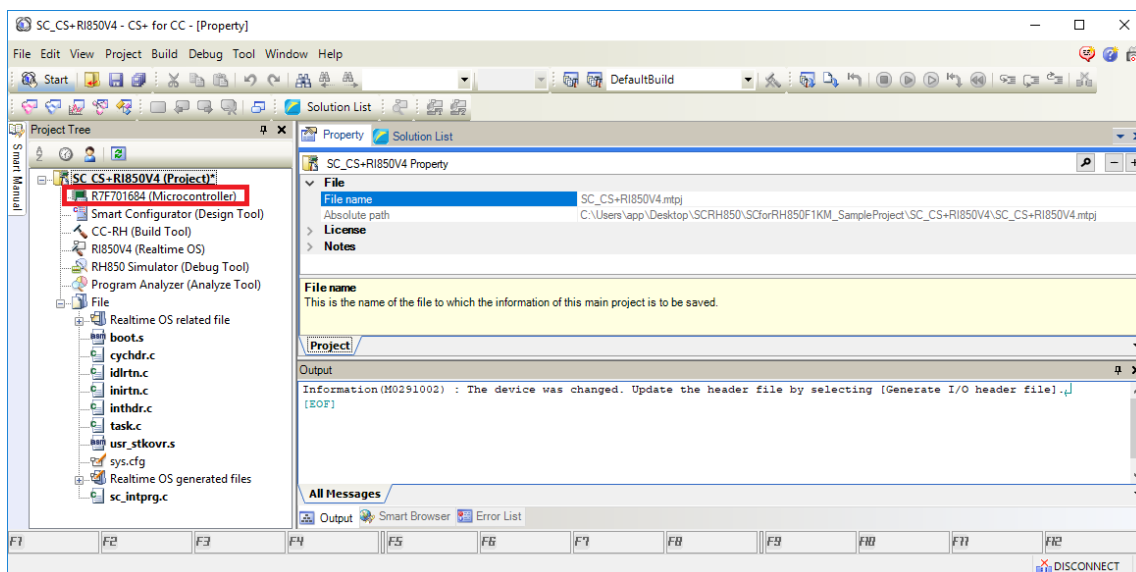


- (1)-2. In the [Change Microcontroller] dialog box, select the RH850/F1KM-S1 or RH850/F1KM-S4 device to be used.

Example: Changing from R7F701649 (RH850/F1KM-S4) to R7F701684 (RH850/F1KM-S1)



(1)-3. Confirm that the microcontroller displayed in the Project Tree panel has become the device after the change.

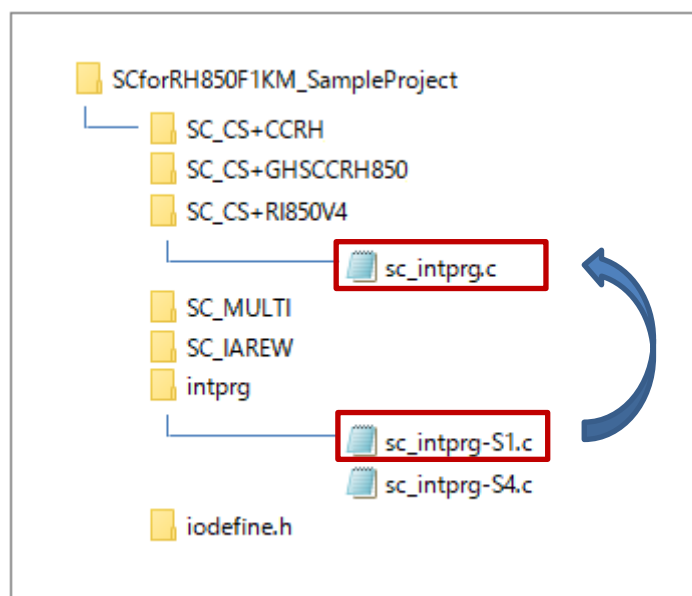


(1)-4. Save the project by selecting [Save Project] from the [File] menu.

(2) Replacing the sc_intprg.c file

This step is only necessary when a device of the RH850/F1KM-S4 Group was changed to a device of the RH850/F1KM-S1 Group in step (1).

(2)-1. Since the sc_intprg.c file included in the sample project is for RH850/F1KM-S4 devices, it should be replaced with a file for RH850/F1KM-S1 devices. In Windows Explorer, rename sc_intprg-S1.c in the intrpg folder "sc_intprg.c" and replace sc_intprg.c of the sample project with this file.



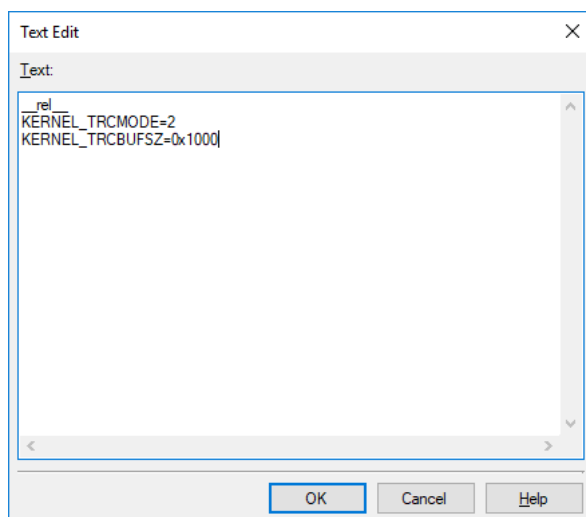
(Files to which the description does not apply are omitted from this figure.)

6.4 Settings in the Sample Project for CS+, CCRH, and RI850V4

The sample project is created in CS+ as an [Application(RI850V4,CC-RH)] project. The include path is added and settings of the following options are changed.

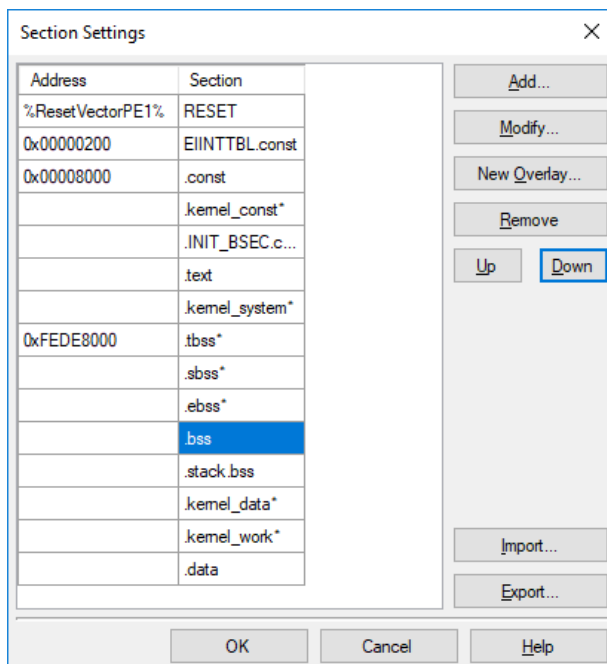
- (a) [Property] panel from [CC-RH (Build Tool)] → [Common Options] tab → [Frequently Used Options(for Compile)] → [Macro definition]

[Settings in the sample project]



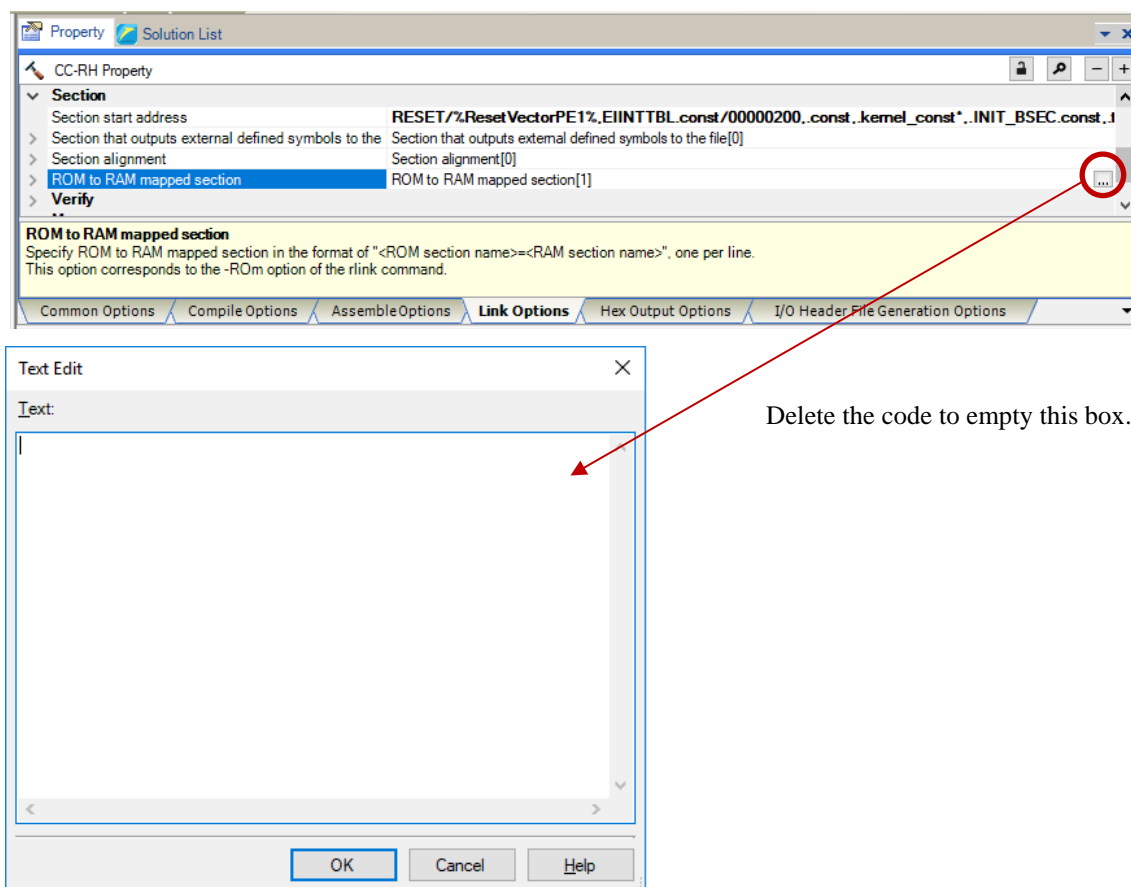
- (b) [Property] panel from [CC-RH (Build Tool)] → [Link Options] tab → [Section] → [Section start address]

[Settings in the sample project]



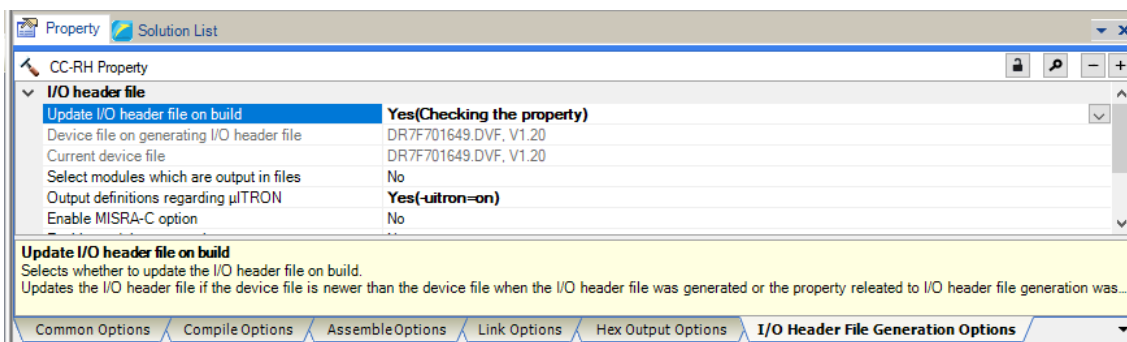
- (c) [Property] panel from [CC-RH (Build Tool)] → [Link Options] tab → [Section] → [ROM to RAM mapped section]

[Settings in the sample project]



- (d) [Property] panel from [CC-RH (Build Tool)] → [I/O Header File Generation Options] tab → [I/O Header File] → [Update I/O header file on build]

[Settings in the sample project]

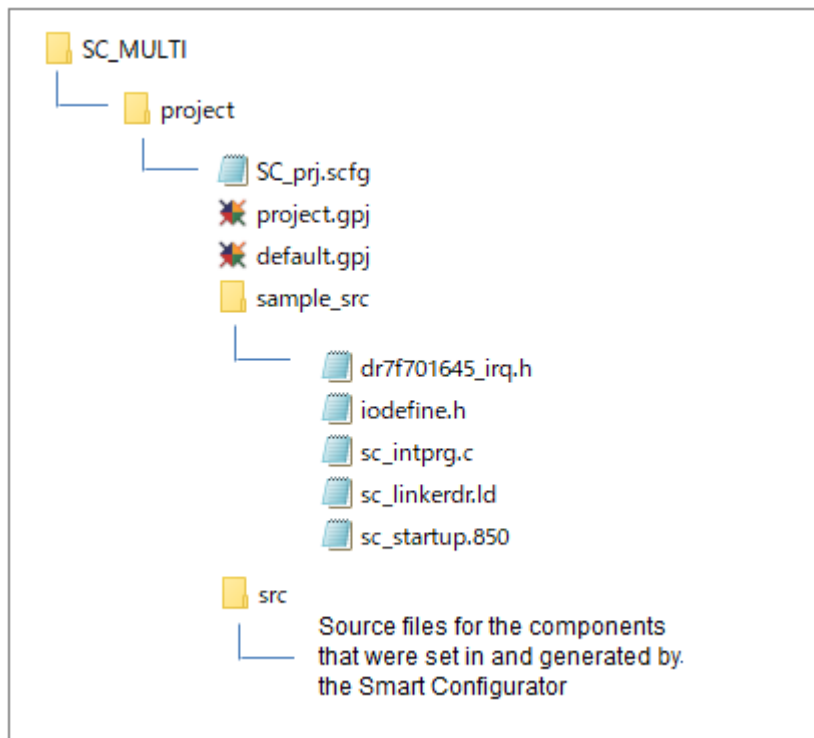


7. Description of the Sample Project for MULTI and GHS CCRH850

7.1 Configuration of the Sample Project for MULTI and GHS CCRH850

The following shows the configuration of the sample project.

Table 7-1 File Configuration of the Sample Project for MULTI and GHS CCRH850



File Name	Outline of File
SC_prj.scfg	Project file for the Smart Configurator
default.gpj	Top project file for MULTI
project.gpj	Project file for MULTI
sc_startup.850*	Definition of the processing from a reset up to branching to the given application project, definition of the exception interrupt vector table, and definition of the startup routine up to branching to the main function
dr7f701645_irq.h*	Definition of the exception interrupt vector table (macro definitions)
sc_linkerdr.ld*	Linker directive file
sc_intprg.c*	Definition of the EI-level maskable interrupt vector table
iodefine.h*	Definitions of registers

*: The Smart Configurator does not output these files.

7.2 Basic Operating Procedure

Figure 7.1 shows the operating procedure when using the Smart Configurator based on a sample project.

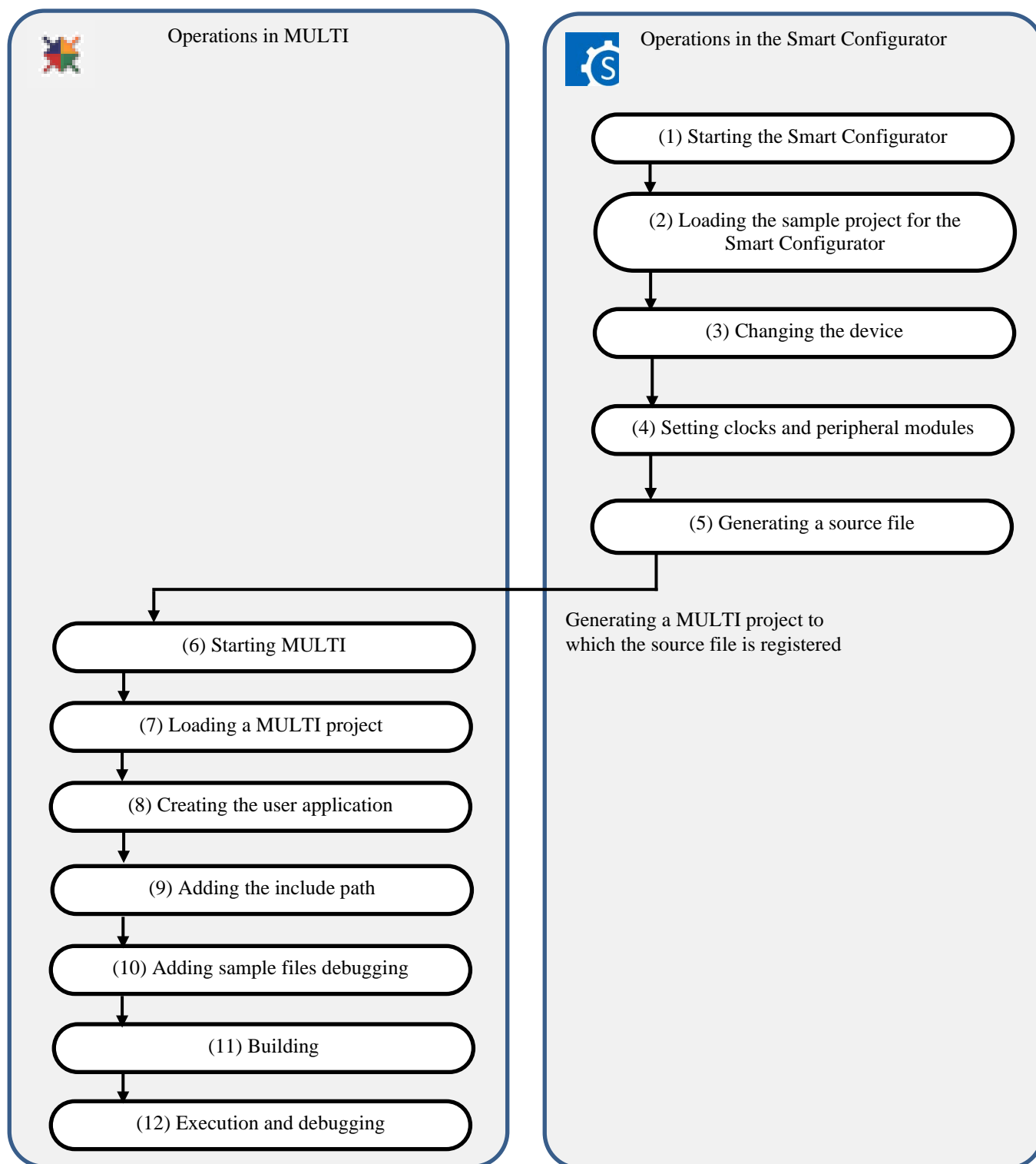



Figure 7.1 Operating Procedure for the Sample Project for MULTI and GHS CCRH850

(1) Starting the Smart Configurator

In the [Start] menu of Windows, select [Renesas Electronics Smart Configurator] → [Smart Configurator for RH850].

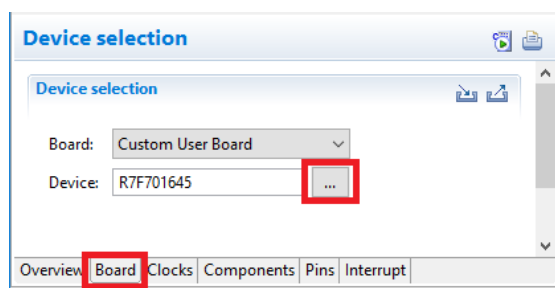
(2) Loading the sample project for the Smart Configurator

From the [Open...] item of the [File] menu or with the [Open Existing Configuration File] toolbar button  of the Smart Configurator, select "SC_prj.scfg".

(3) Changing the device

The R7F701645 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the target device in the Smart Configurator and change the file to be used with reference to section 7.3, "Procedure for Changing the Device". If the device does not require changing, proceed to step (4).

- Changing the target device in the Smart Configurator
Change the device in [Device selection] on the [Board] page.



For steps (4) and (5), the procedure for setting clocks and components and generating a driver in the Smart Configurator, see section 9, "Operations in the Smart Configurator".

(6) Starting MULTI

In the [Start] menu of Windows, select [Green Hills Software] → [MULTI].

(7) Loading a MULTI project

Select the [Open Project Manager...] item of the [Components] menu of MULTI. Then select "default.gpj" in the [Select a project to open] dialog box.

(8) Creating the user application

For the procedure for creating the user application, see section 9, "Operations in the Smart Configurator".

(9) Adding the include path

Add an include path to refer iodef.h.

Repeat step (9) if step (5) is done after adding the include path by step (9).

(9)-1. Right click on the "project.gpj" in MULTI and select the [Set Build options...].

(9)-2. In the [Basic Options] tab of the [Build Options for project.gpj] dialog box, double click the [Project]-[Include Directories].

(9)-3. In the [Edit List Option] dialog box, add the path to iodefine.h.

(10) Adding sample files

Add the sample startup file and the others as the build target.

Repeat step (10) if step (5) is done after adding files as the build target by step (10).

(10)-1. Right click on the “project.gpj” in MULTI and select the [Add File into project.gpj...]

(10)-2. In the [Choose file(s) to add:] dialog box, add all files under “sample_src” folder.

(11) Building

Select [Build] → [Build Program Project] of MULTI.

(12) Execution and debugging

For program execution and debugging in the emulator, refer to the user’s manual for MULTI.

7.3 Procedure for Changing the Device

When the target device of the sample project in Table 2-1, "Sample Projects" differs from the device that is actually to be used, the target device or file to be used must be changed according to the following procedure.

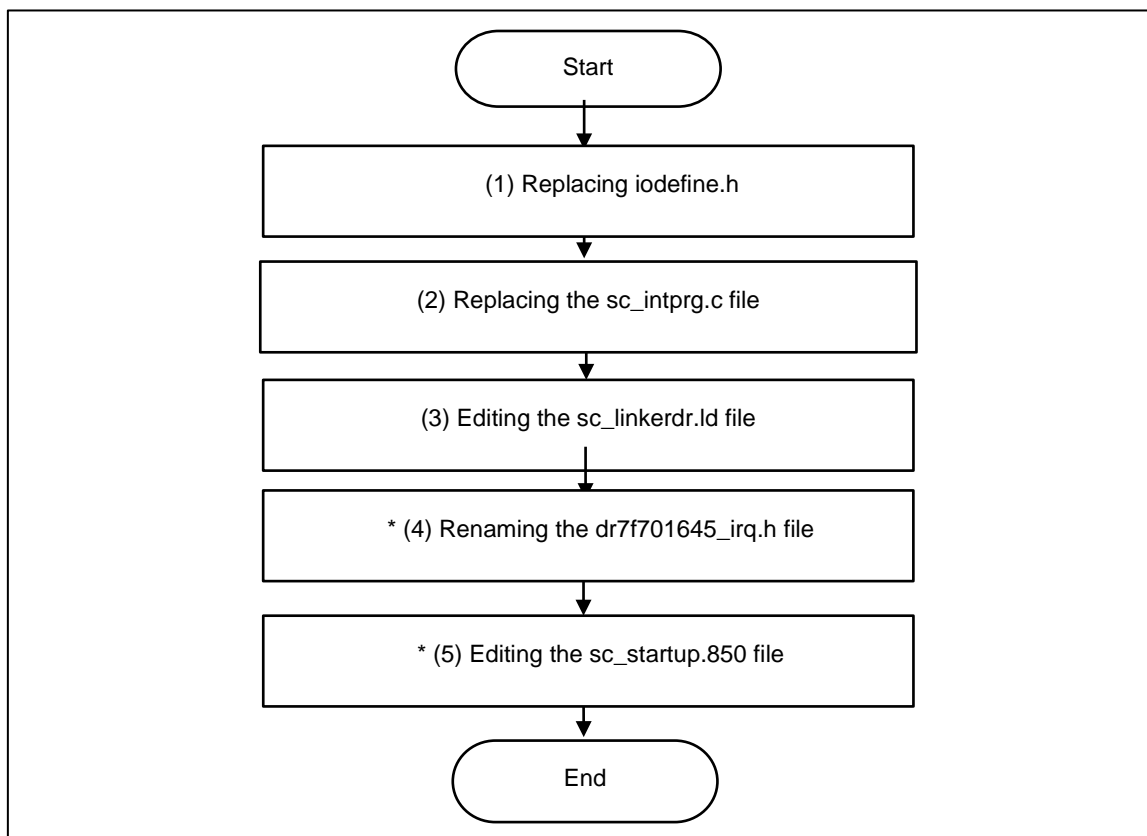
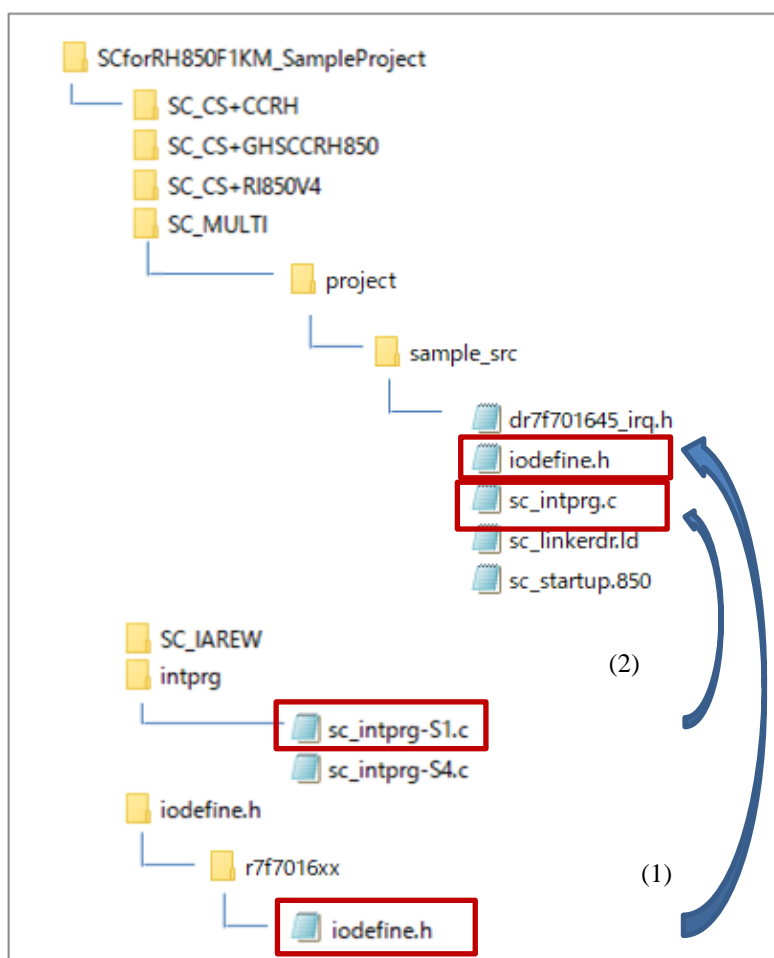


Figure 7.2 Changing the Target Device of the Sample Project (MULTI Project)

*: Steps (4) and (5) are optional.

The procedure as described hereafter is based on the sample project "SC_MULTI".



(Files to which the description does not apply are omitted from this figure.)

(1) Replacing iodefine.h

In Windows Explorer, replace this with iodefine.h for the device to be used. Copy "iodefine.h" for the device to be used from the iodefine.h folder to the sample project.

(2) Replacing the sc_intprg.c file

This step is only necessary when a device of the RH850/F1KM-S4 Group was changed to a device of the RH850/F1KM-S1 Group.

(2)-1. Since the sc_intprg.c file included in the sample project is for RH850/F1KM-S4 devices, it should be replaced with a file for RH850/F1KM-S1 devices. In Windows Explorer, copy sc_intprg-S1.c from the intprg folder to the sample project.

(2)-2. Delete "sc_intprg.c" in the sample project.

(2)-3. Rename "sc_intprg-S1.c" in the sample project to "sc_intprg.c".

(3) Editing the sc_linkerdr.ld file

(3)-1. Open the “sc_linkerdr.ld” in any editor to edit it.

(3)-2. Change the start address and size of memories to match the specifications of the microcontroller in use.
(Renaming microcontroller name in the file header is optional.)

```

44 /*
45 /*=====
46 /* Environment:
47 /*      Device:      R7F701645
48 /*      IDE:         GHS Multi for V800 V6.xx or later
49 /*=====
50
51 CONSTANTS
52 {
53     zero_start = 0xffff8000
54 }
55
56 MEMORY
57 {
58     iROM_0      : ORIGIN = 0x00000000, LENGTH = 4096k
59     iROM_1      : ORIGIN = 0x01000000, LENGTH = 32k
60     iRAM_0      : ORIGIN = 0xFEBC0000, LENGTH = 256k
61     iRAM_R      : ORIGIN = 0xFEFD0000, LENGTH = 64k
62 }
63
64 SECTIONS

```

(3)-3. Change the “EIINTTBL_end” that is the end address of EIINTTBL section to match the specifications of the microcontroller in use.

```

64 SECTIONS
65 {
66
67     /* Start of internal ROM area (iROM_0) */
68
69     /* original */
70     /* .intvect          :>iROM_0 /* start of inter
71     /* .intvect_end 0x07E4      :>. /* end of inter
72     /* */
73
74     /* Changed for sample project */
75     .intvect          :>iROM_0 /* start of interr
76     .intvect_end 0x01FF      :>. /* end of interr
77     EIINTTBL_align (4)      :>iROM_0 /* start of EI mask
78     EIINTTBL_end 0x07E4      :>. /* end of EI mask
79     /* */

```

Reference value:

RH850/F1KM-S1: 0x0798

RH850/F1KM-S4: 0x07E4

The following steps (4) and (5) are optional. Build of the sample project is possible without steps (4) and (5).

(4) Renaming the dr7f701645_irq.h file

(4)-1. In Windows Explorer, rename from “dr7f01645_irq.h” to “dr7f0xxxx_irq.h” to match the microcontroller name in use.

(5) Editing the sc_startup.850 file

(5)-1. Open the “sc_startup.850” in any editor to edit it.

- (5)-2. Change the include file name to the filename changed in step (4) and save the sc_startup.850 file. (Renaming microcontroller name in the file header is optional.)

```
45 -----
46 -- Environment:
47 --      Device: R7F701645
48 --      IDE:    GHS Multi for
49 -----
50
51 -----
52 ----- Selection of external interrupt
53 ----- User modifiable section
54 ----- Please uncomment the required
55 -----
56 #include "dr7f701645_IRQ.h"
57
```

7.4 Settings in the Sample Project for MULTI and GHS CCRH850

The sample project has an added build target setting for sample files and an added include path for “iodefine.h” to the MULTI project.

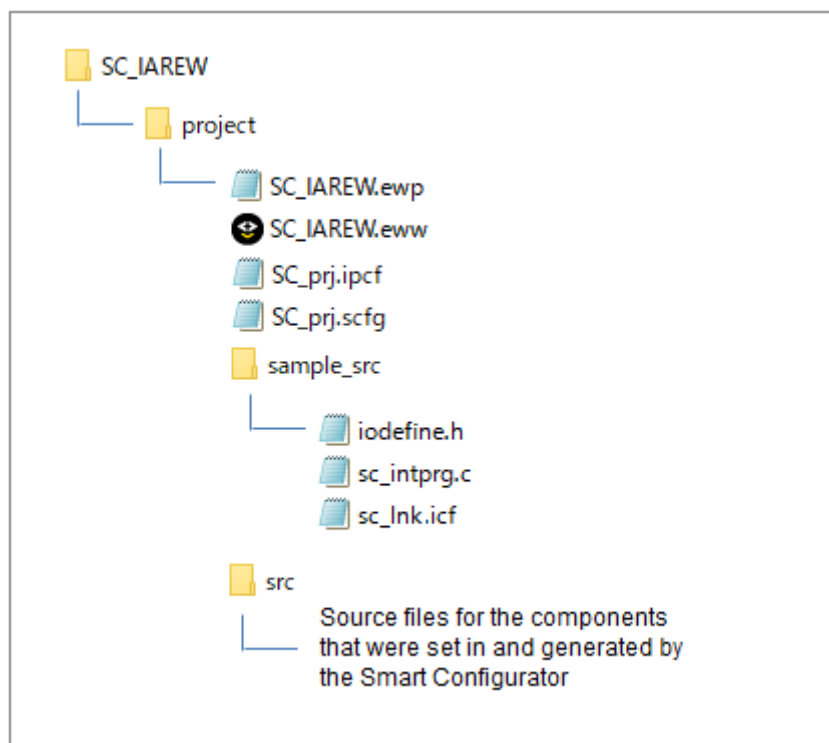
Remarks: A build target setting for sample files and an added include path for “iodefine.h” are removed by step (5) in section 7.2.

8. Description of the Sample Project for IAREW and IAR ICC

8.1 Configuration of the Sample Project for IAREW and IAR ICC

The following shows the configuration of the sample project.

Table 8-1 File Configuration of the Sample Project for IAREW and IAR ICC



File Name	Outline of File
SC_IAREW.eww*	Workspace for IAREW
SC_IAREW.ewp*	Project file for IAREW
SC_prj.ipcf	Connection file for the IAREW project
SC_prj.scfg	Project file for the Smart Configurator
sc_Ink.icf*	Linker configuration file
sc_intprg.c*	Definition of the EI-level maskable interrupt vector table
iodefine.h*	Definitions of registers

*: The Smart Configurator does not output these files.

8.2 Basic Operating Procedure

Figure 8.1 shows the operating procedure when using the Smart Configurator based on a sample project.

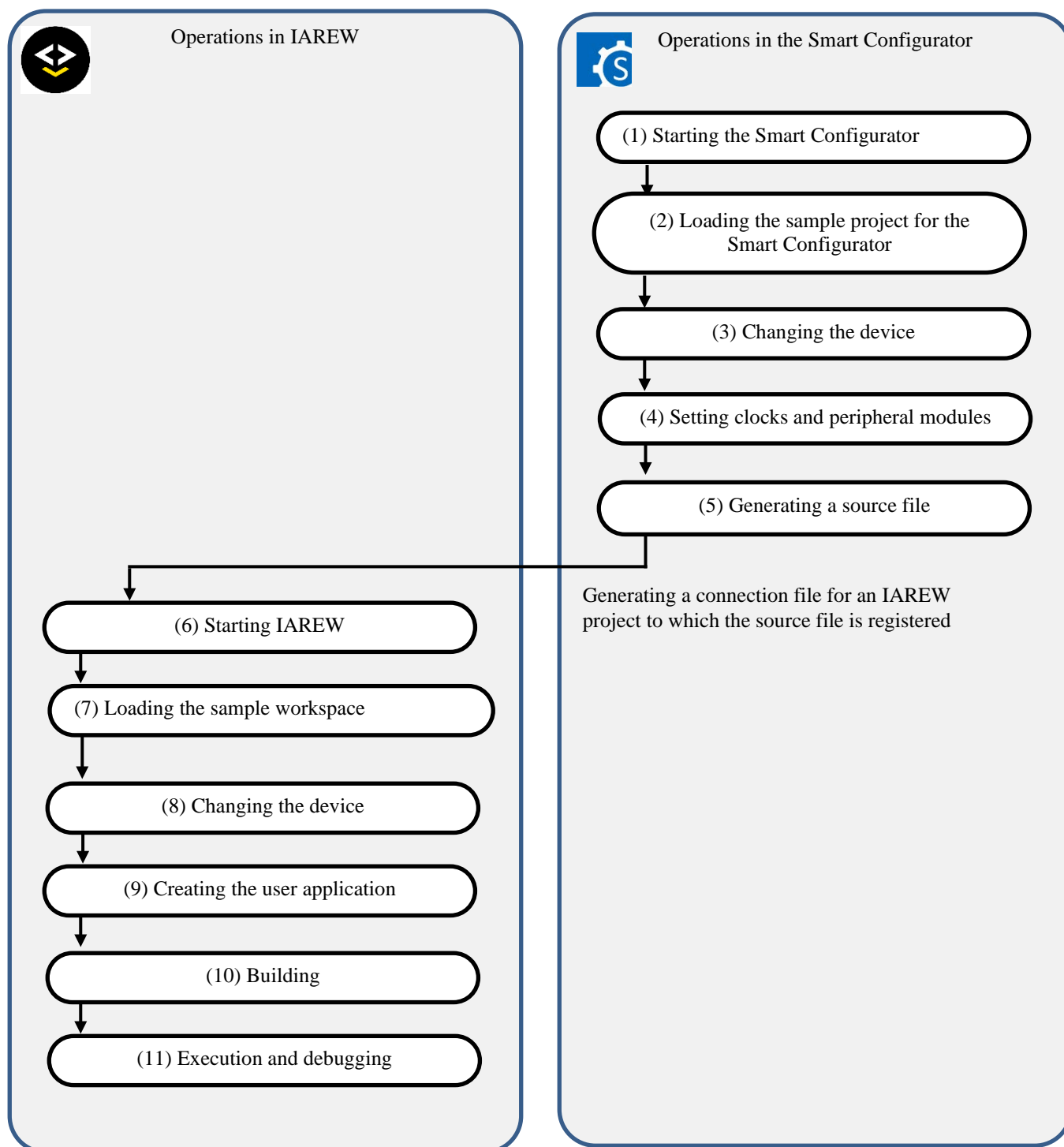



Figure 8.1 Operating Procedure for the Sample Project for IAREW and IAR ICC

(1) Starting the Smart Configurator

In the [Start] menu of Windows, select [Renesas Electronics Smart Configurator] → [Smart Configurator for RH850].

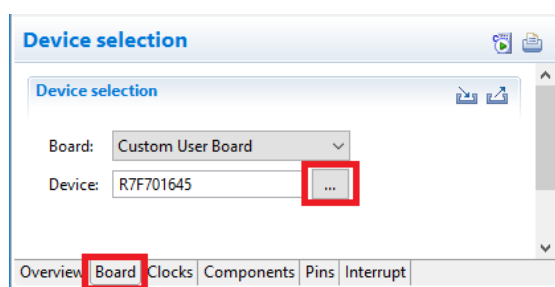
(2) Loading the sample project for the Smart Configurator

From the [Open...] item of the [File] menu or with the [Open Existing Configuration File] toolbar button  of the Smart Configurator, select "SC_proj.scfg".

(3) Changing the device

The R7F701645 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the target device in the Smart Configurator. If the device does not require changing, proceed to step (4).

- Changing the target device in the Smart Configurator
Change the device in [Device selection] on the [Board] page.



For steps (4) and (5), the procedure for setting clocks and components and generating a driver in the Smart Configurator, see section 9, "Operations in the Smart Configurator".

(6) Starting IAREW

In the [Start] menu of Windows, select [IAR Systems] → [IAR Embedded Workbench].

(7) Loading the sample workspace

Select [File] → [Open] → [Workspace...] in IAREW. Then select "SC_IAREW.eww" in the [Open Workspace] dialog box.

(8) Changing the device

The R7F701645 (RH850/F1KM-S4) is selected as the target device in the sample project. If you are using another device, change the file to be used with reference to section 8.3, "Procedure for Changing the Device". If the device does not need to be changed, proceed to step (9).

(9) Creating the user application

For the procedure for creating the user application, see section 9, "Operations in the Smart Configurator".

(10) Building

Select [Project] → [Rebuild All] in IAREW.

(11) Execution and debugging

For program execution and debugging in the emulator, refer to the user's manual for IAREW. Also, set the value of the INTBP register to 0x00000200 before executing the main function.

8.3 Procedure for Changing the Device

When the target device of the sample project in Table 2-1, "Sample Projects" differs from the device that is actually to be used, the target device or file to be used must be changed according to the following procedure.

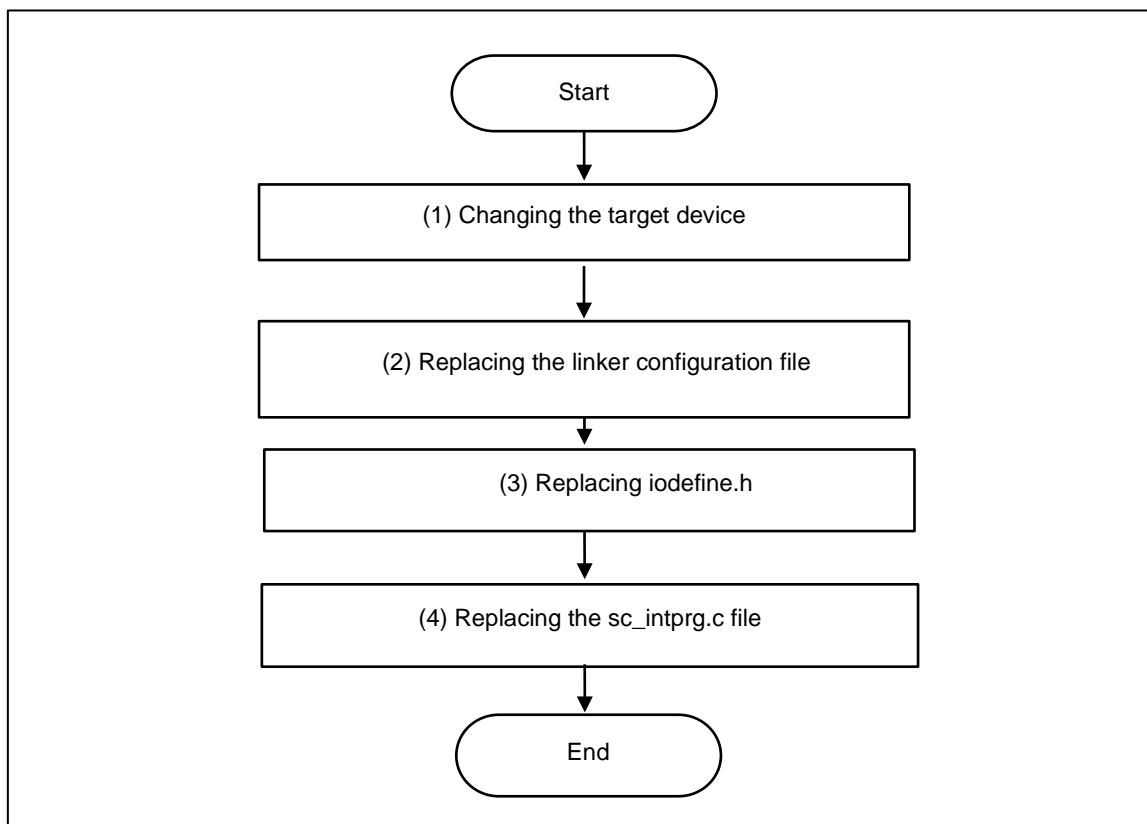
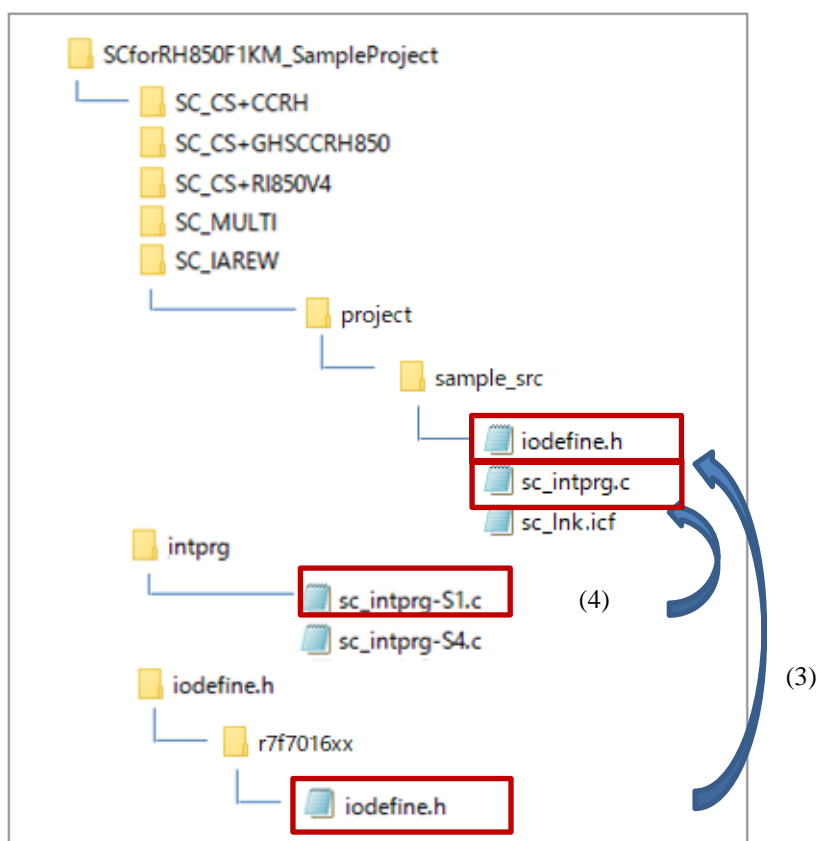


Figure 8.2 Changing the Target Device of the Sample Project (IAREW Project)

The procedure as described hereafter is based on the sample project "SC_IAREW".

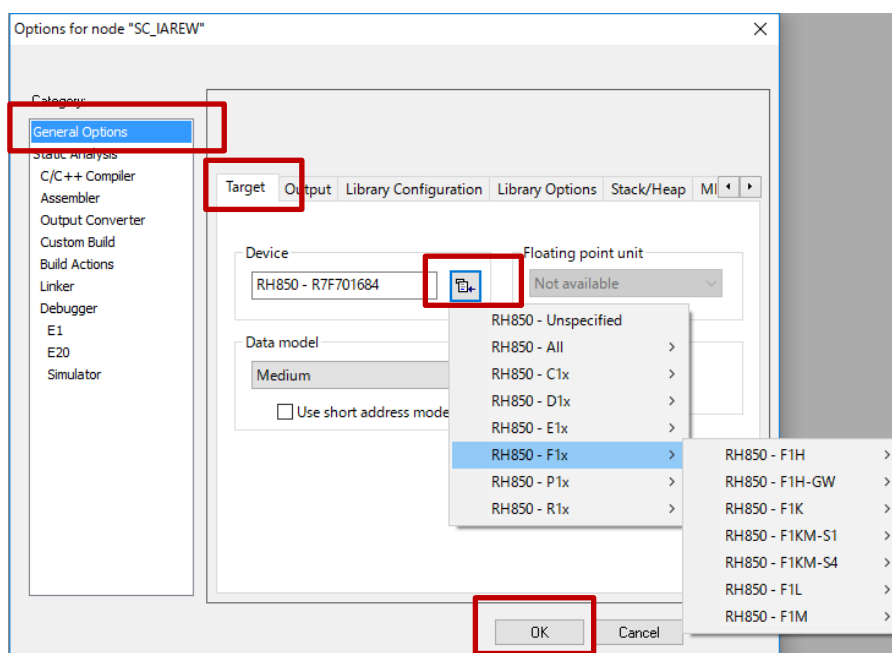


(Files to which the description does not apply are omitted from this figure.)

(1) Changing the target device

(1)-1. Select [Options...] menu in [Project] of IAREW.

(1)-2. Select [General Options] → [Target] in the [Options for node “SC_IAREW”] dialog box. Select the device to be used from the drop-down list under [Device].



(2) Replacing the linker configuration file

(2)-1. In Windows Explorer, copy the "lnkr7f70xxxx.icf" file for the device you will be using from the installation folder for IAREW (C:\Program Files (x86)\IAR Systems\Embedded Workbench x.x\rh850\config) to the sample project.

(2)-2. Edit the copied "lnkr7f70xxxx.icf" file. Open it in any editor.

(2)-3. Add a section to be used in the sample project. Change the code as shown below so that the EIINTTBL section is located from address 0x200 and then save the file.

[Before the change]

```

69 //Uncomment and adapt the following section if you want to place the interrupt ta
70 // define block .interrupt_core1 with alignment = 8 {
71 //   ro section .table.interrupt_core_1 };
72 //
73 // "INTERRUPT":place at address mem:0x200
74 // {
75 //   block .interrupt_core1
76 // };
77 //
78 include "layout.icf";
79 //
80 //-----
81 //      End of file
82 //-----
[EOF]
```

[After the change]

```

69 //Uncomment and adapt the following section if you want to place the interrupt t
70 define block .interrupt_core1 with alignment = 8 {
71   ro section .table.interrupt_core_1 };
72 //
73 // "INTERRUPT":place at address mem:0x200
74 // {
75 //   block .interrupt_core1
76 // };
77 //
78 "EIINTTBL":place at address mem:0x200
79 {
80   block .interrupt_core1,
81   section EIINTTBL
82 };
83 //
84 //
85 include "layout.icf";
86 //
87 //-----
88 //      End of file
89 //-----
[EOF]
```

(The line number may differ according to which file you are using.)

(3) Replacing iodefne.h

In Windows Explorer, replace this with iodefne.h for the device to be used. Copy "iodefne.h" for the device to be used from the iodefne.h folder to the sample project.

(4) Replacing the sc_intprg.c file

This step is only necessary when a device of the RH850/F1KM-S4 Group was changed to a device of the RH850/F1KM-S1 Group.

- (4)-1. Since the sc_intprg.c file included in the sample project is for RH850/F1KM-S4 devices, it should be replaced with a file for RH850/F1KM-S1 devices. In Windows Explorer, copy sc_intprg-S1.c from the intprg folder to the sample project.

- (4)-2. Delete "sc_intprg.c" in the sample project.

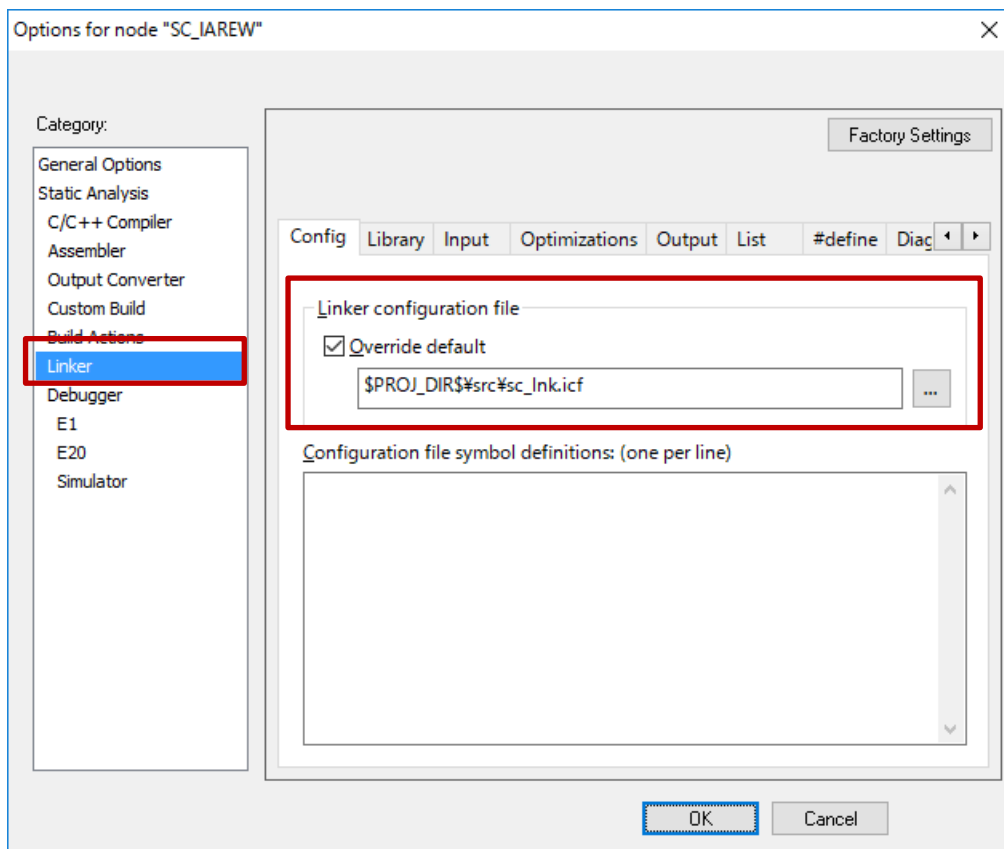
- (4)-3. Rename "sc_intprg-S1.c" in the sample project to "sc_intprg.c".

8.4 Settings in the Sample Project for IAREW and IAR ICC

The sample project is created in IAREW. An include path is added and settings of the following options are changed.

- (a) [Project] of IAREW → [Options...] menu → [Options for node "SC_IAREW"] dialog box → [Linker] → [Linker configuration file]

[Settings in the sample project]



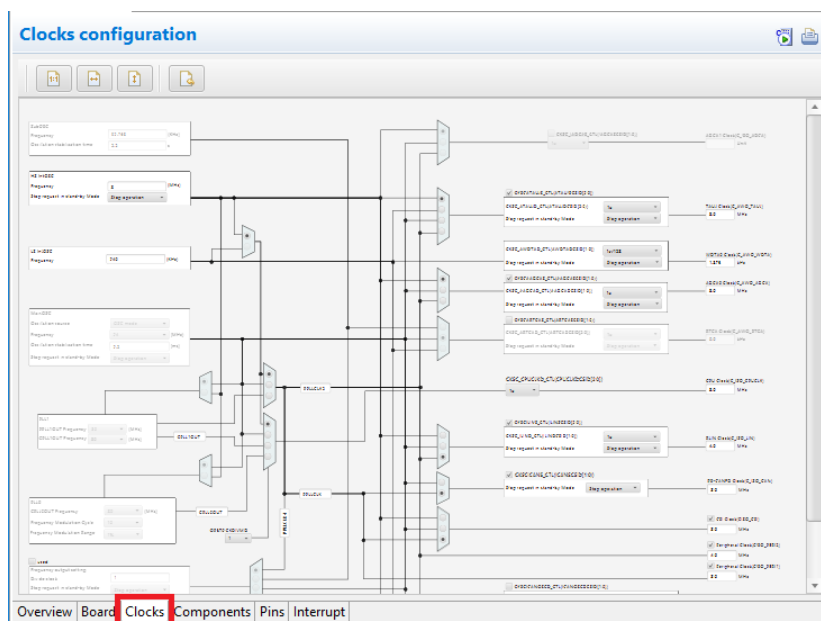
9. Operations in the Smart Configurator

This section gives an overview of setting the drivers of peripheral modules of the device and handling of the Smart Configurator for the generation of code.

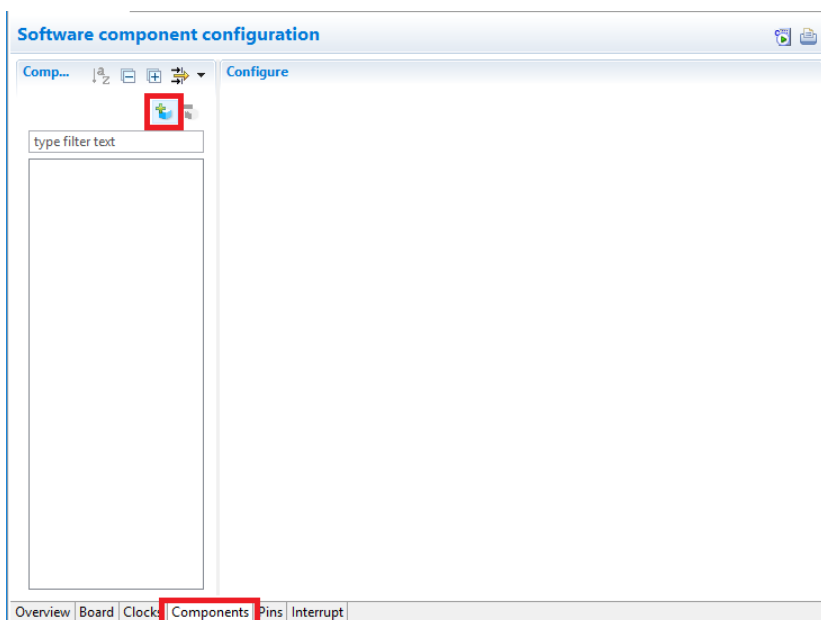
For details, refer to *Smart Configurator User's Guide: CS+* (R20AN0470).

9.1 Setting the Peripheral Modules (Software Components)

- (1) Configure the clocks of the device on the [Clocks] tabbed page.

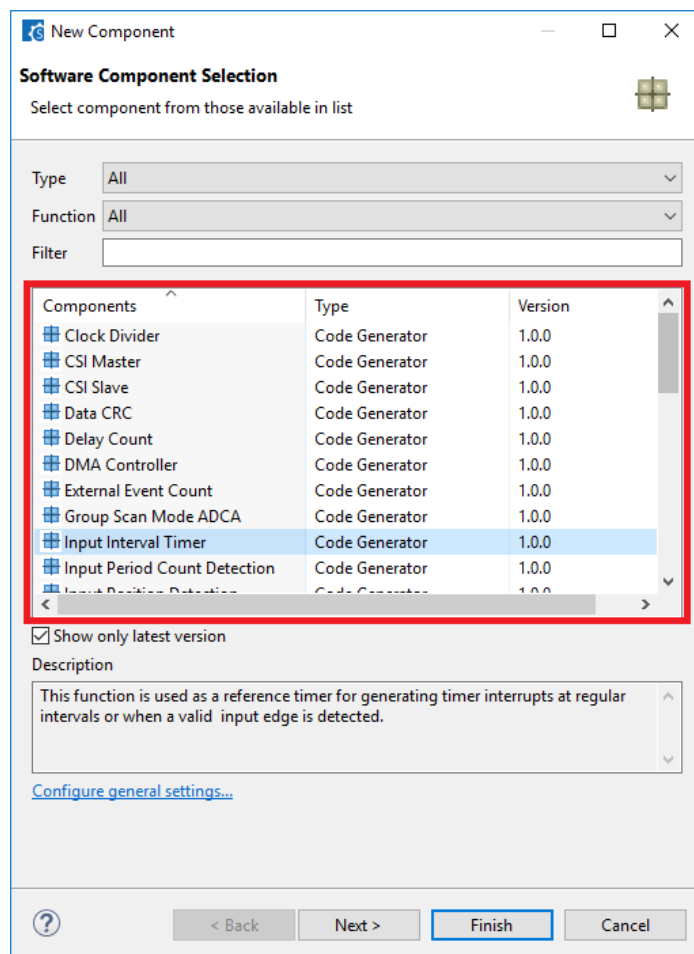


- (2) Add or set the peripheral modules of the device on the [Components] tabbed page. The peripheral modules are set as software components. Click on the [Adding components] icon .

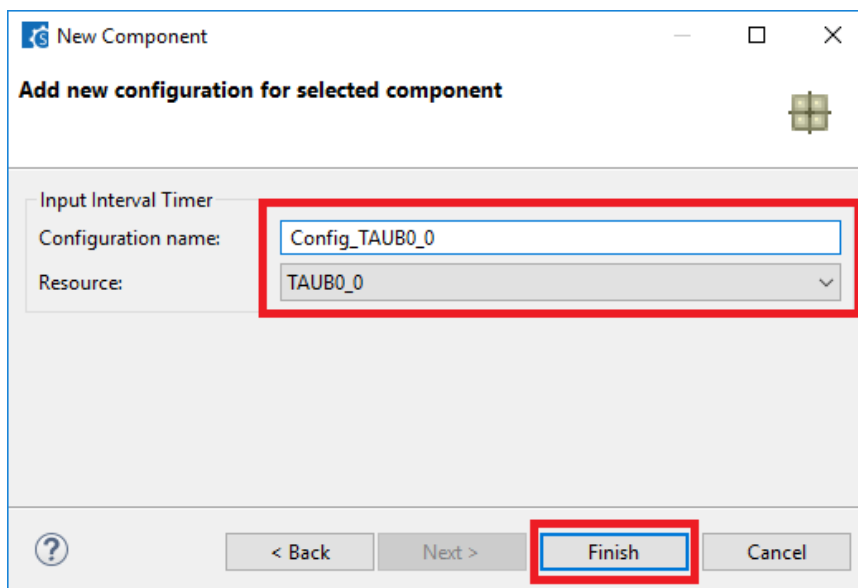


Remarks: Importing middleware in the form of FIT (Firmware Integration Technology) modules is not supported for the RH850 Family.

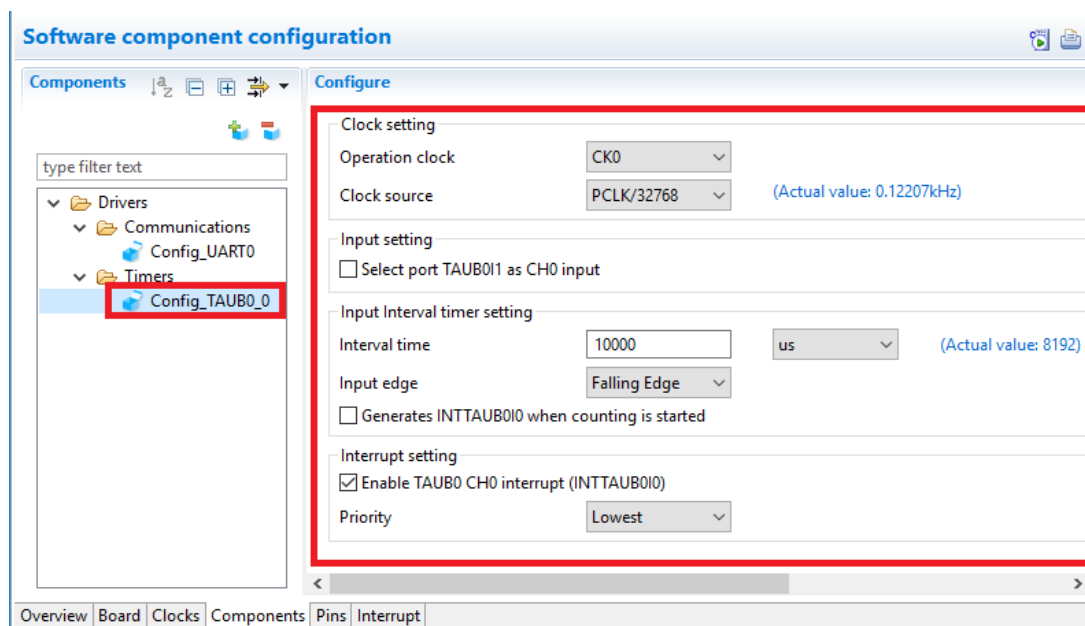
- (3) Select components on the [Software Component Selection] page of the [New Component] dialog box. Select each component to be used from the list and click on the [Next] button.



- (4) Select the configuration name and resource of the selected component. On the [Add new configuration for selected component] page of the [New Component] dialog box, enter an appropriate configuration name or use the default name. Select the resource or use the default resource. After you have made the selections, click on the [Finish] button.




- (5) Set the configuration of the component. Click on the configuration icon in the Component Tree panel and make detailed settings in the right-hand panel.

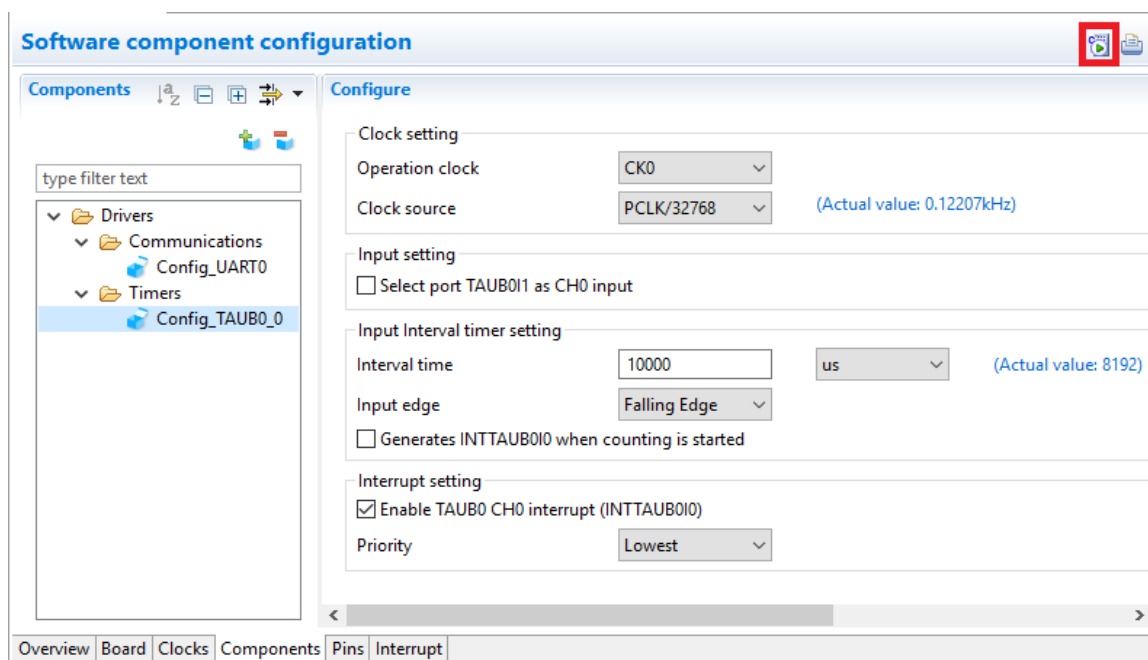


- (6) Repeat steps (2) to (5) for each component you intend to use.

9.2 Generating Drivers

- (1) Click on the [Code Generator] button . The source files generated by the Smart Configurator are stored in the <ProjectDir>\src\smc_gen folder.

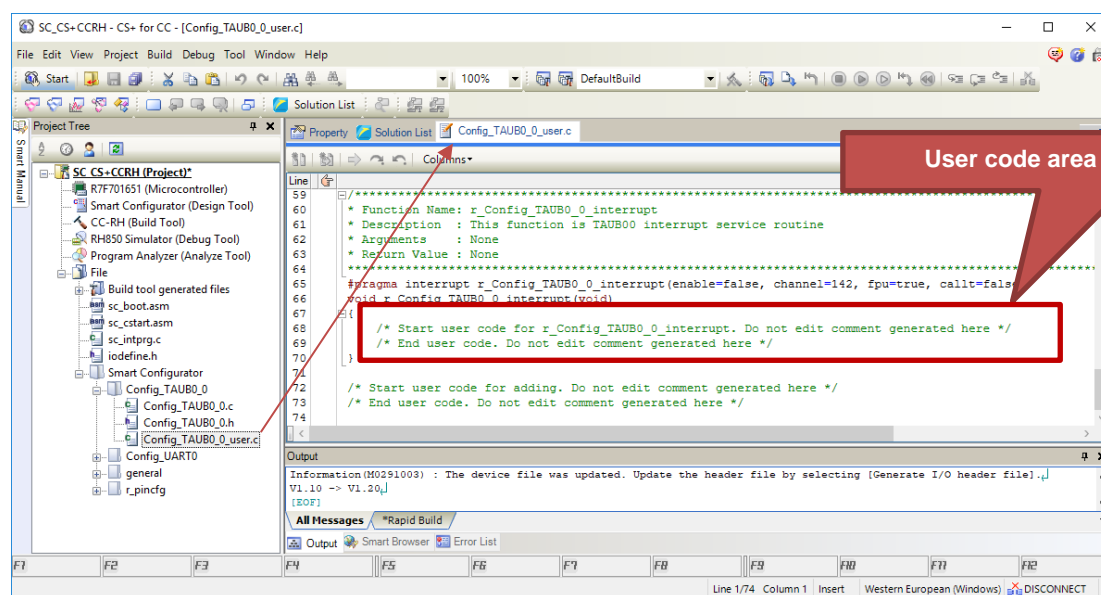
<ProjectDir> is the folder containing the project files (scfg) for the Smart Configurator.



9.3 Adding the Application Code to the User Code Area

Some generated source files have a user code area for the writing of user code. Open such files in an editor from the integrated development environment you are using and add the application code (e.g. code for interrupt processing) to the user code areas as necessary.

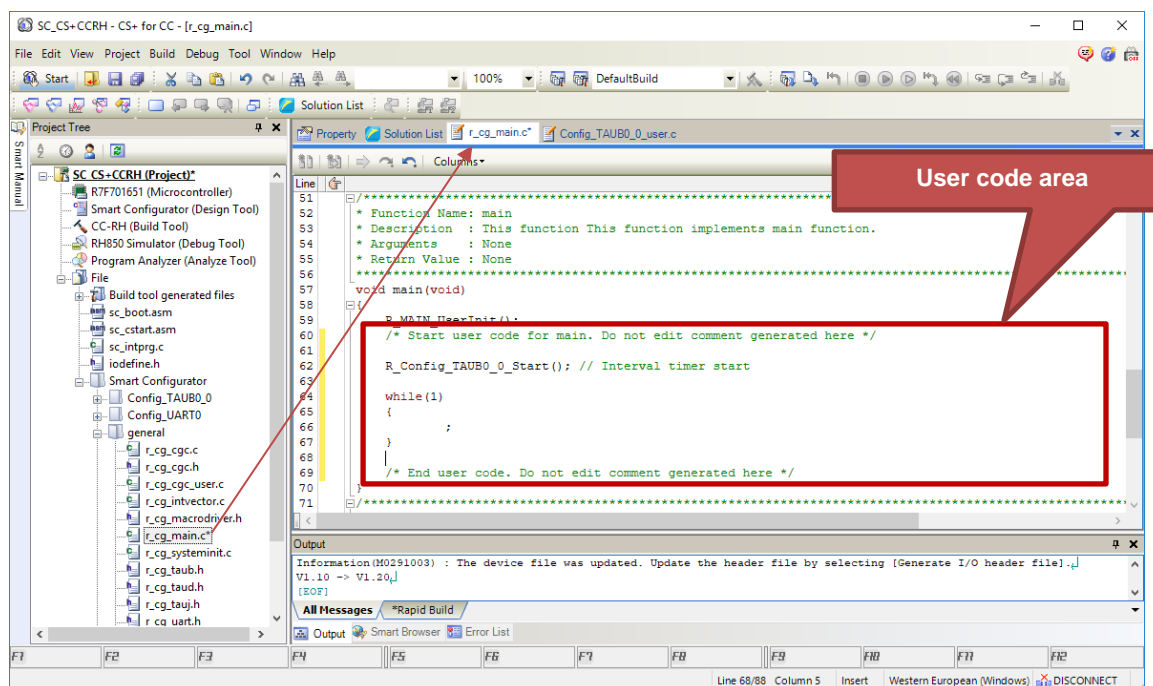
Example: File generated for the interval timer component in the CS+ environment



9.4 Adding the Application Code to main()

The main function is in "<ProjectDir>\src\smc_gen\general\r_cg_main.c". Open the file in an editor from the integrated development environment you are using and add the application code to the user code area.

Example: CS+ environment



Revision History

Rev.	Date	Description	
		Page	Summary
1.10	January 2019	—	New release
1.20	February 2020	18	Changed step (3) in Figure 5.1
		21	Added Remark
		23	Changed steps (3) to (5) in Figure 5.2
		25-26	Changed steps (3) to (5)
		38	Added steps (9) and (10) in Figure 7.1
		39-40	Added steps (9) and (10)
		41	Changed steps (3) to (5) in Figure 7.2
		43-44	Changed steps (3) to (5)
		45	Changed section 7.4

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 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.
5. 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.

6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. 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.4.0-1 November 2017)

Corporate Headquarters

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

Contact Information

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

Trademarks

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