RL78/G15 Group and RL78/G16 Group

Renesas Flash Sample Program Type 01 SC Version (Code Flash)

Introduction

This document explains Renesas Flash Sample Program Type 01 for the RL78/G15 group and RL78/G16 group in the case of using Smart Configurator(SC). It is a process which builds into a user program the functions for “the code flash memory program” included in RFSP Type 01, and it is a method for programming code flash memory using the attached sample program.

In this document, “Renesas Flash Sample Program Type 01” is described to be “RFSP Type 01” or “RFSP”.

This document includes the contents in comparison with conventional RFSP Type 01 not using SC. This document distinguishes and expresses it.

Simple version : Conventional RFSP Type 01 not using SC.
SC version : RFSP Type 01 using “SC” currently explained by this document.

Target Device

The target device group by which the operation for RFSP Type 01 was confirmed.

RL78/G15 group

RL78/G16 group

If this application note is applied to other microcomputers, it is necessary to modify in accordance with the specification of the microcomputer. And, be sure to evaluate enough.

Contents

1. Specification

The sample program included in RFSP Type 01 erases the block 3 (0x0C00) of a code flash area, and programs 64 bytes of data from the top of the block 3.

1.1 Operating Environment

- C Compiler Packages

Table 1-1 The Target C Compiler Packages

Package	Manufacturer	Version
CC-RL(for CS+ or e^2 studio)	Renesas Electronics	V1.10 or later
IAR (Embedded Workbench)	IAR Systems	V4.21 or later
LLVM(e^2 studio)	(Open source software)	V10.0.0.202309 or later

IAR Systems, IAR Embedded Workbench, C-SPY, IAR, and the logotype of IAR Systems are trademarks or registered trademarks owned by IAR Systems AB.

- Emulator

  Table 1-2 shows the emulator on which the operation of RFSP Type 01 was confirmed.

Table 1-2 Emulator on which RFSP Type 01 Operation was Confirmed

Emulator	Manufacturer
E2 emulator Lite	Renesas Electronics

- Target MCU

  RL78/G15

  RL78/G16

1.2 Structure of Sample Program Folders

Figure 1.1 shows the structure of sample program folders.

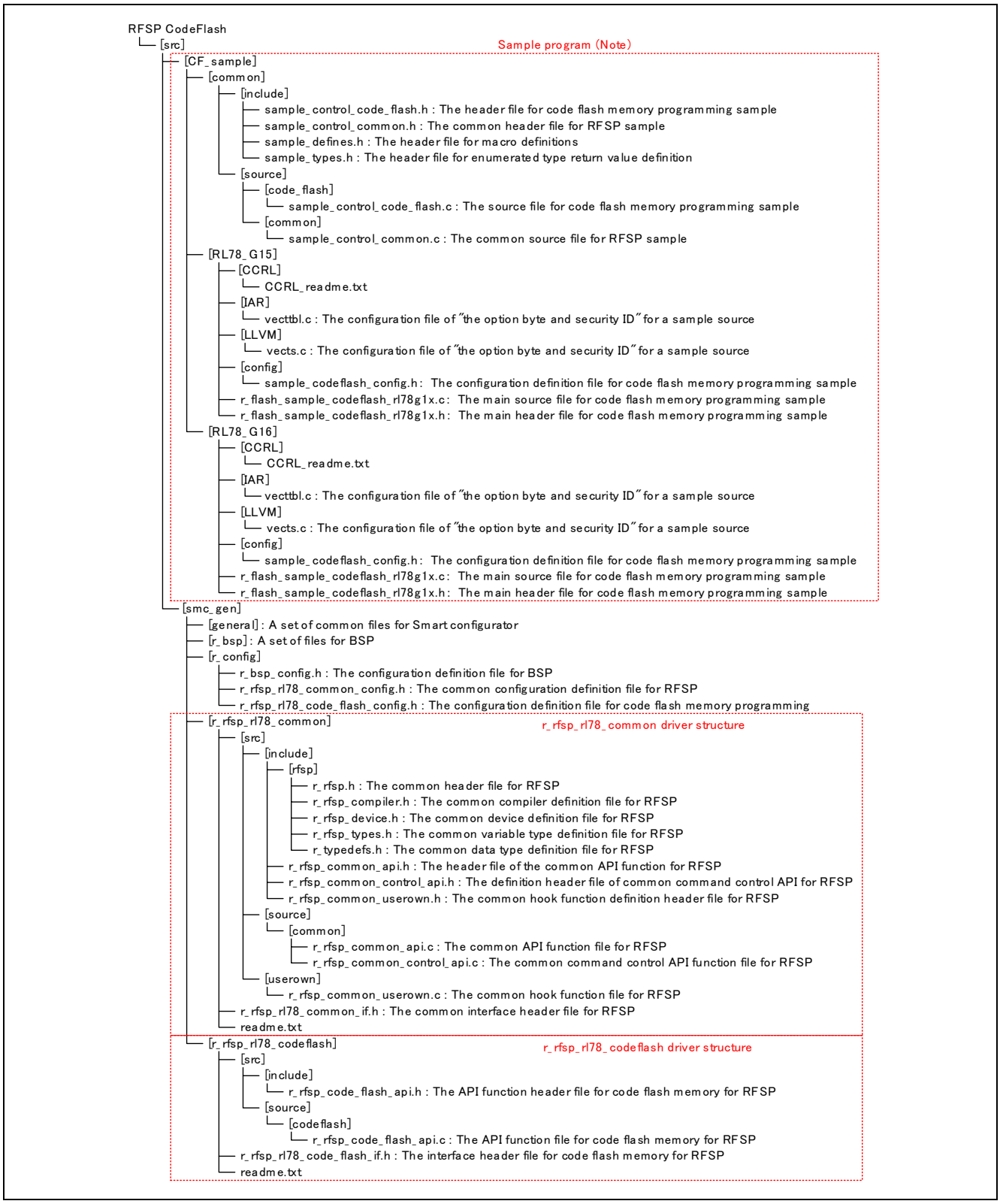


Figure 1.1 Structure of Sample Program Folders

Note: The sample program of a zip file format downloads by Smart configurator. Extract the compressed file (CF_sample.zip) outputted to the “demo” folder, and move the [CF_sample] folder under the [src] folder. Refer to “2.3 Project Registration of Sample Program” for the details of project registration.

1.3 File Structure of RFSP Program

1.3.1 File Structure of RFSP Common Program (r_rfsp_rl78_common)

The difference of SC version RFSP common program and Simple version common program is shown. Refer to the “Renesas Flash Sample Program Type01 Application Note (R20AN0652)” for the detail specification for the RFSP common program.

Table 1-3 Difference of the Files of the SC Version RFSP and the Simple Version RFSP (Common API : r_rfsp_rl78_common\src\source\common)

File name	Simple version	SC version
r_rfsp_common_api.c	No change	
r_rfsp_common_control_api.c	No change	

Table 1-4 Difference of the File of the SC Version RFSP and the Simple Version RFSP (Common API : r_rfsp_rl78_common\src\userown)

File name	Simple version	SC version
r_rfsp_common_userown.c	No change	

Table 1-5 Difference of the Files of the SC Version RFSP and the Simple Version RFSP (Common Header : r_rfsp_rl78_common\src\include)

File name	Simple version	SC version
r_rfsp_common_api.h	No change	
r_rfsp_common_control_api.h	No change	
r_rfsp_common_userown.h	No change	

Table 1-6 Difference of the Files of the SC Version RFSP and the Simple Version RFSP (Common Header : r_rfsp_rl78_common\src\include\rfsp)

File name	Simple version	SC version
r_rfsp.h	No change	
r_rfsp_compiler.h	No change	
r_rfsp_device.h	No change	
r_rfsp_types.h	No change	
r_typedefs.h	No change	

Table 1-7 Difference of the File of the SC Version RFSP and the Simple Version RFSP (Common Interface Header : r_rfsp_rl78_common)

File name	Simple version	SC version
r_rfsp_rl78_common_if.h	-	Newly created. Include the header file for common API.

1.3.2 File Structure for RFSP Code Flash Program (r_rfsp_rl78_codeflash)

The difference of SC version RFSP code flash program and Simple version code flash program is shown. Refer to the “Renesas Flash Sample Program Type01 Application Note (R20AN0652)” for the detail specification for the RFSP code flash program.

Table 1-8 Difference of the File of the SC Version RFSP and the Simple Version RFSP (Code Flash API : r_rfsp_rl78_codeflash\src\source\codeflash)

File name	Simple version	SC version
r_rfsp_code_flash_api.c	No change	

Table 1-9 Difference of the File of the SC Version RFSP and the Simple Version RFSP (Code Flash API Header : r_rfsp_rl78_codeflash\src\include)

File name	Simple version	SC version
r_rfsp_code_flash_api.h	No change	

Table 1-10 Difference of the File of the SC Version RFSP and the Simple Version RFSP (Code Flash Interface Header : r_rfsp_rl78_codeflash)

File name	Simple version	SC version
r_rfsp_rl78_code_flash_if.h	-	Newly created. Include the header file for code flash API.

1.4 Code Flash Programming Processing Using a Sample Program

Figure 1.2 shows the flow chart of the sample program. The sample_codeflash_main function executes processing which reprograms code flash.

Sample_CodeFlashControl function processing does not have change from Simple version. Refer to the item of “Sample_CodeFlashControl function” on Renesas Flash Sample Program Type01 Application Note (R20AN0652).

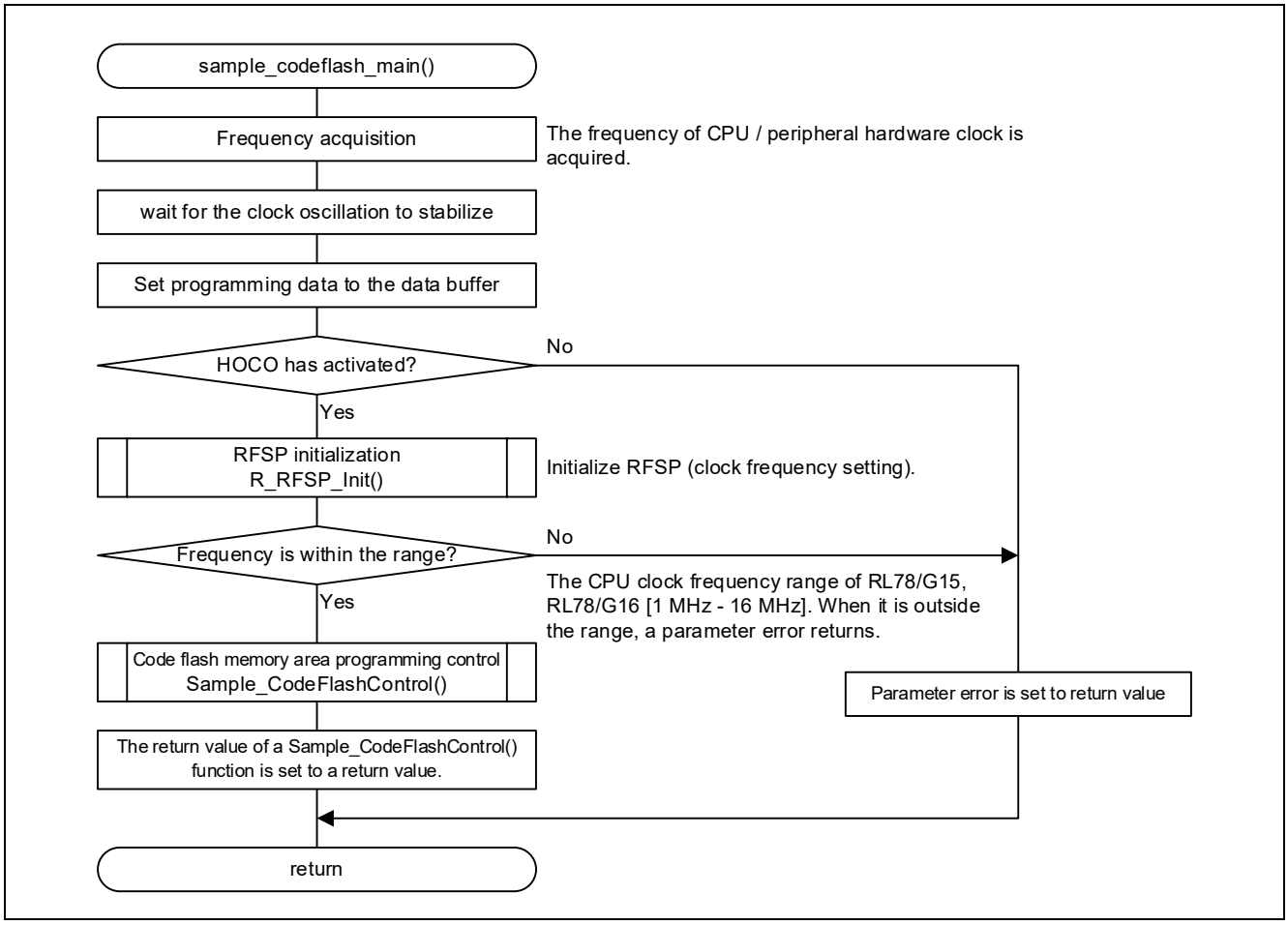


Figure 1.2 The Flow Chart of the Sample Program

Note: “Frequency acquisition” of CPU and peripheral hardware clock settings is using the function included in the “RL78 Family Board Support Package”.

2. Creating a Sample Project for Code Flash Reprogramming

2.1 Example of Creating a Sample Project

2.1.1 In Case of CS+

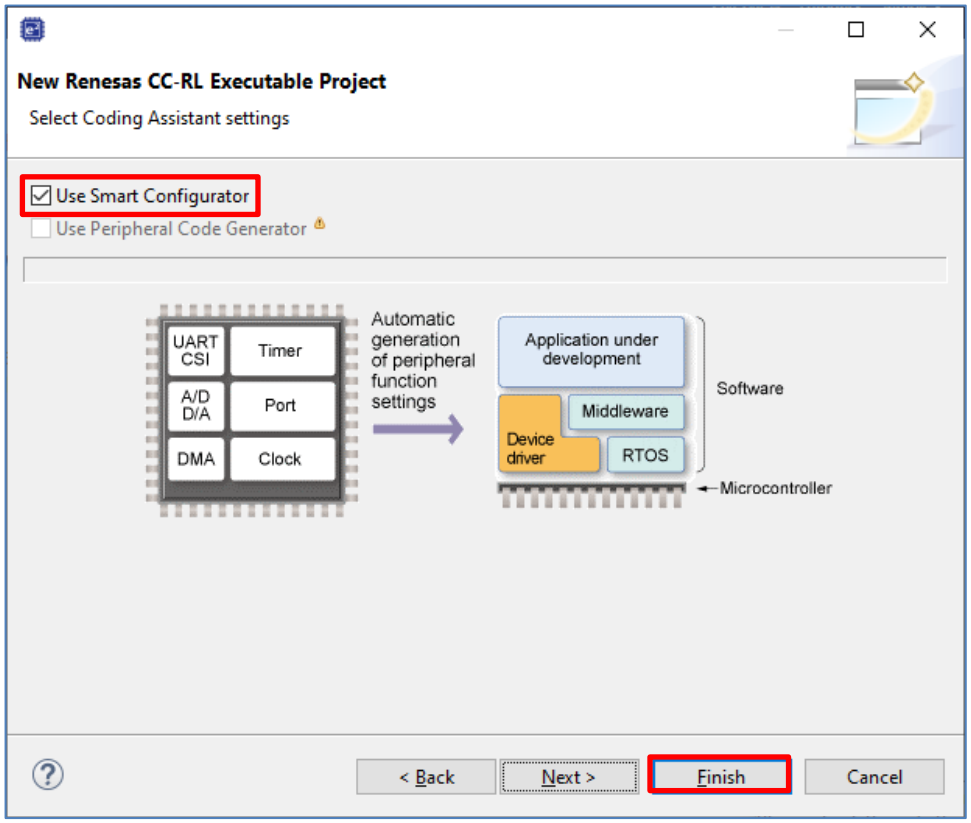
Refer to Renesas Flash Sample Program Type01 Application Note (R20AN0652) “Example of Creating a Sample Project” to create a project.

2.1.2 In Case of e² studio (CC-RL)

Refer to Renesas Flash Sample Program Type01 Application Note (R20AN0652) “Example of Creating a Sample Project” to create a project.

In this application note, because Smart Configurator is used, press a “Next” button after selecting a target device and a debugging tool. And perform the following processes.

Select “Use Smart Configurator” and press a “Finish” button.



2.1.3 In Case of IAR EW for Renesas RL78

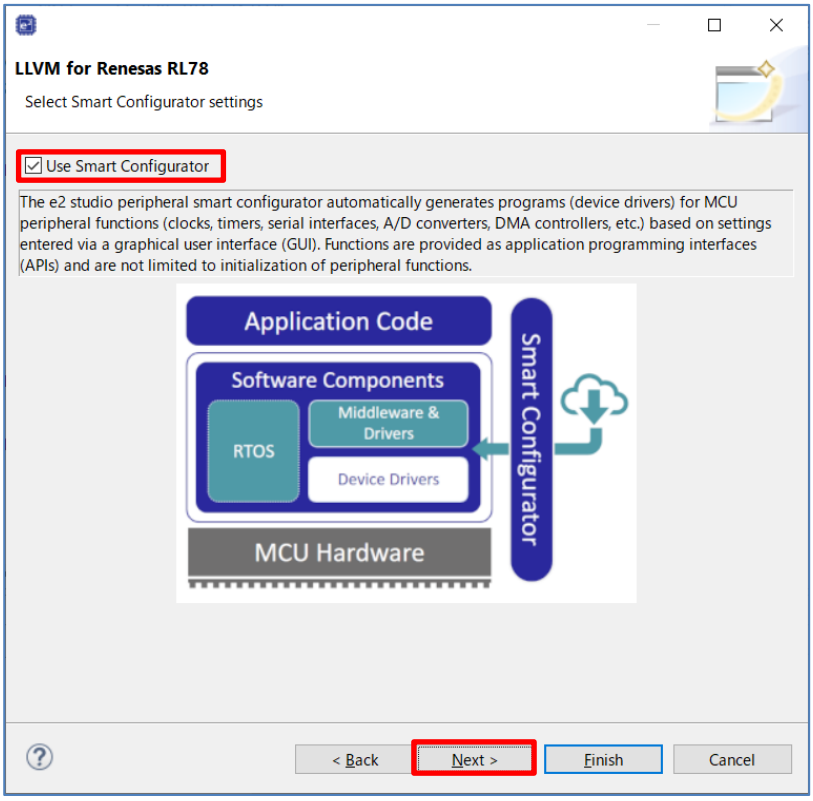
Refer to Renesas Flash Sample Program Type01 Application Note (R20AN0652) “Example of Creating a Sample Project” to create a project.

2.1.4 In Case of e² studio (LLVM)

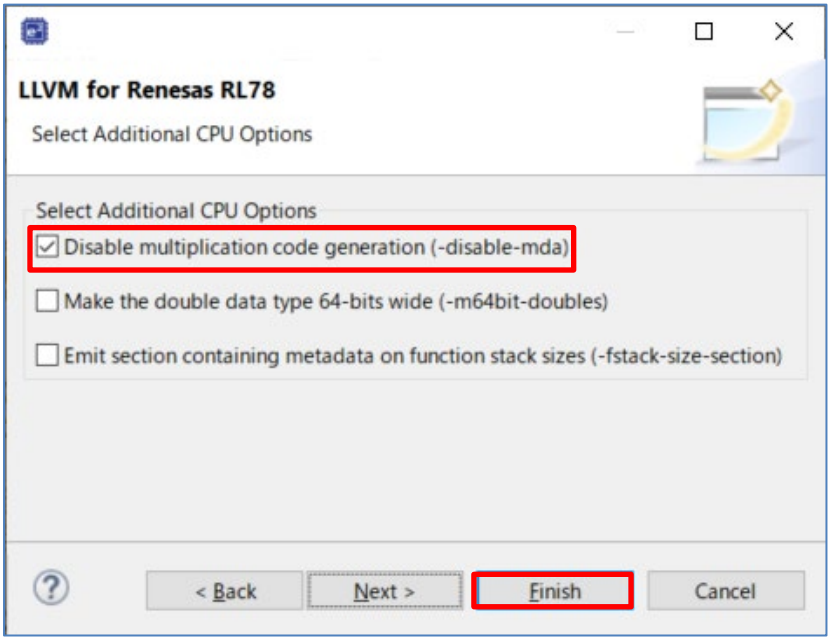
Refer to Renesas Flash Sample Program Type01 Application Note (R20AN0652) “Example of Creating a Sample Project” to create a project.

In this application note, because Smart Configurator is used, press a “Next” button after selecting a target device and a debugging tool. And perform the following processes.

Select “Use Smart Configurator” and press a “Next” button.



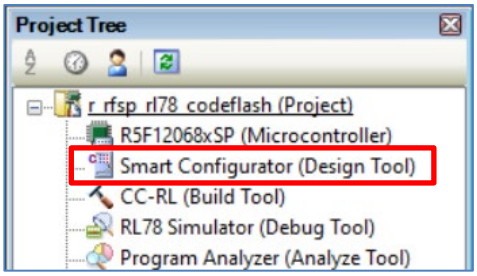
Select “Disable multiplication code generation (-disable-mda)” and press a “Finish” button.



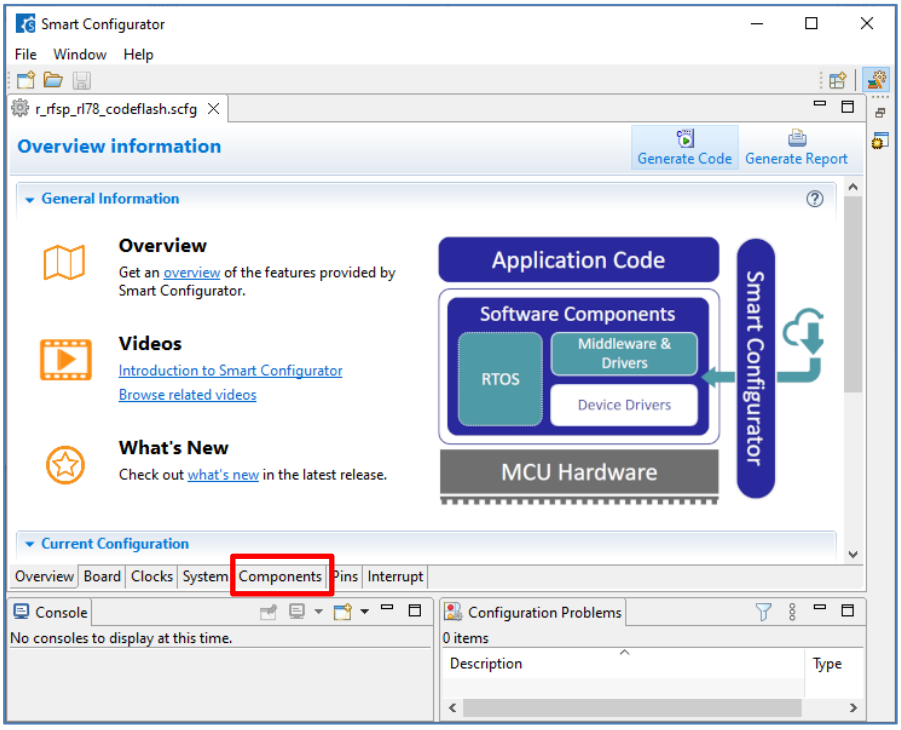
2.2 Example of Source Code Registration

2.2.1 In Case of CS+

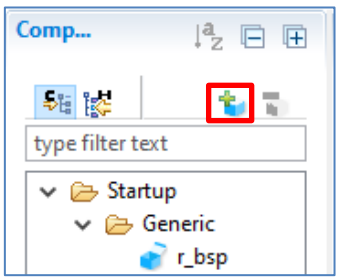
(1) Double-click “Smart Configurator” (design Tool) of “Project Tree”, and start Smart Configurator.



(2) Select a “Components” tab.

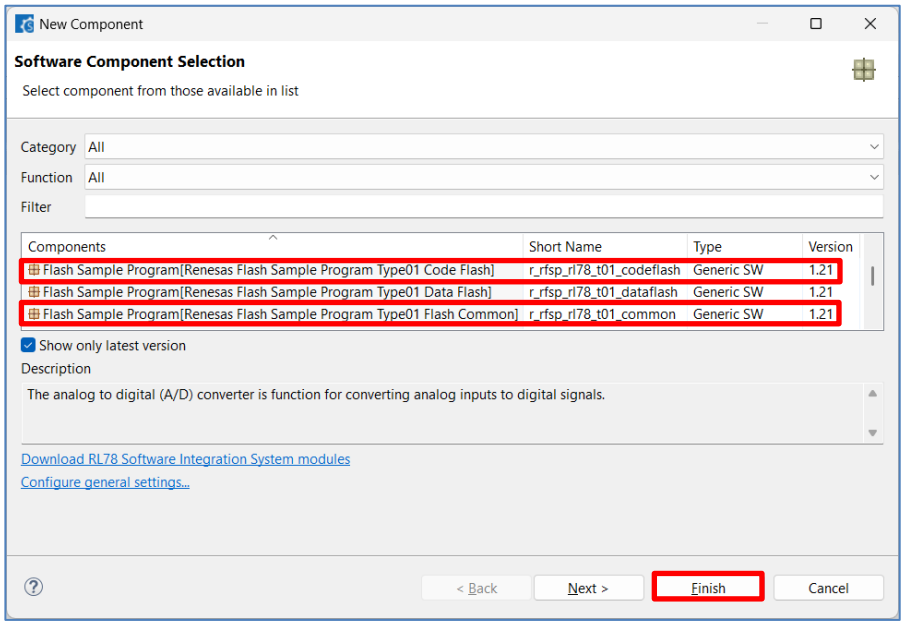


(3) Press the “Add component” button of “Components” and open the “New Component” dialog of “Components”.

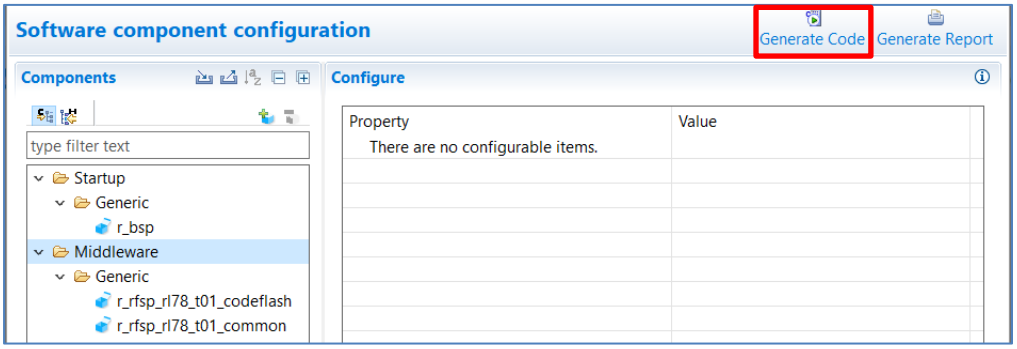


(4) Select the following components and press a “Finish” button.

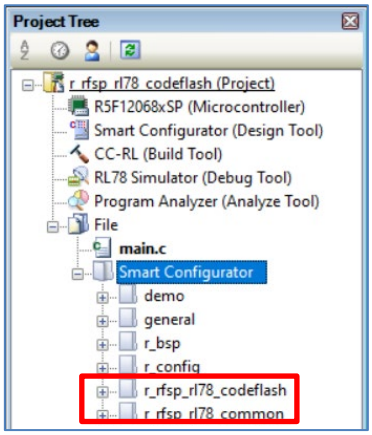
- Flash Sample Program[Renesas Flash Sample Program Type01 Code Flash]
  (r_rfsp _rl78_t01_codeflash)
- Flash Sample Program[Renesas Flash Sample Program Type01 Flash Common]
  (r_rfsp _rl78_t01_common)



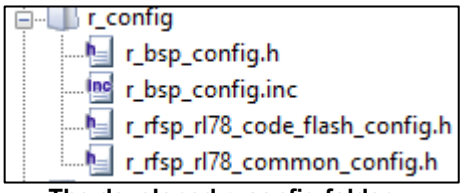
(5) Press a “Generate Code” button and close “Smart Configurator” after the completion of generation for the code.



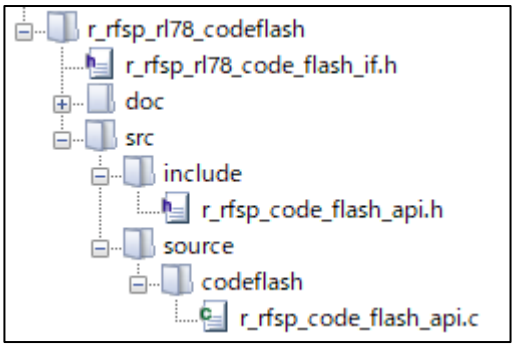
(6) The “r_rfsp_rl78_common” folder and the “r_rfsp_rl78_codeflash” folder is added to the project tree.



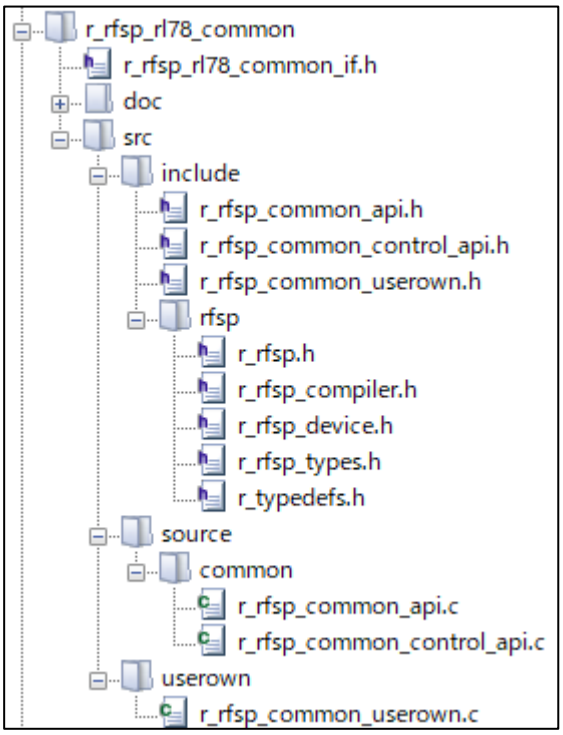
Each folder is developed as follows.



The developed r_config folder



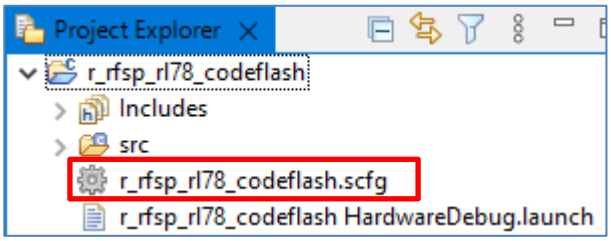
The developed r_rfsp_rl78_codeflash folder



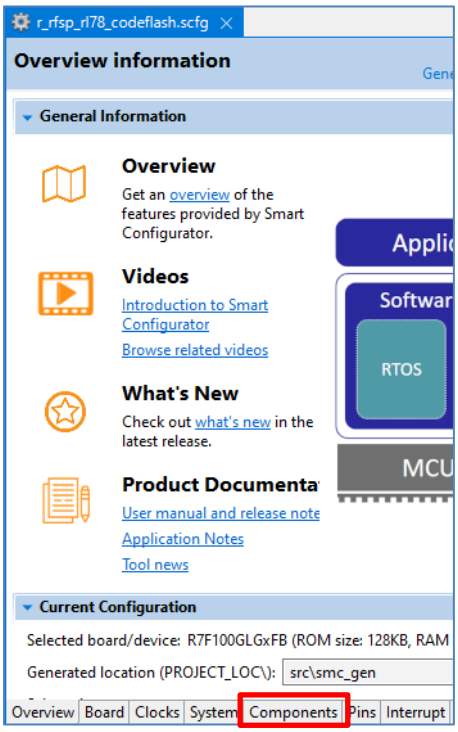
The developed r_rfsp_rl78_common folder

2.2.2 In Case of e² studio (CC-RL)

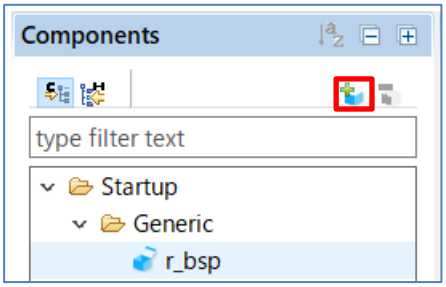
(1) Open the project file of “Smart Configurator” after starting e² studio.



(2) Select a “Components” tab.

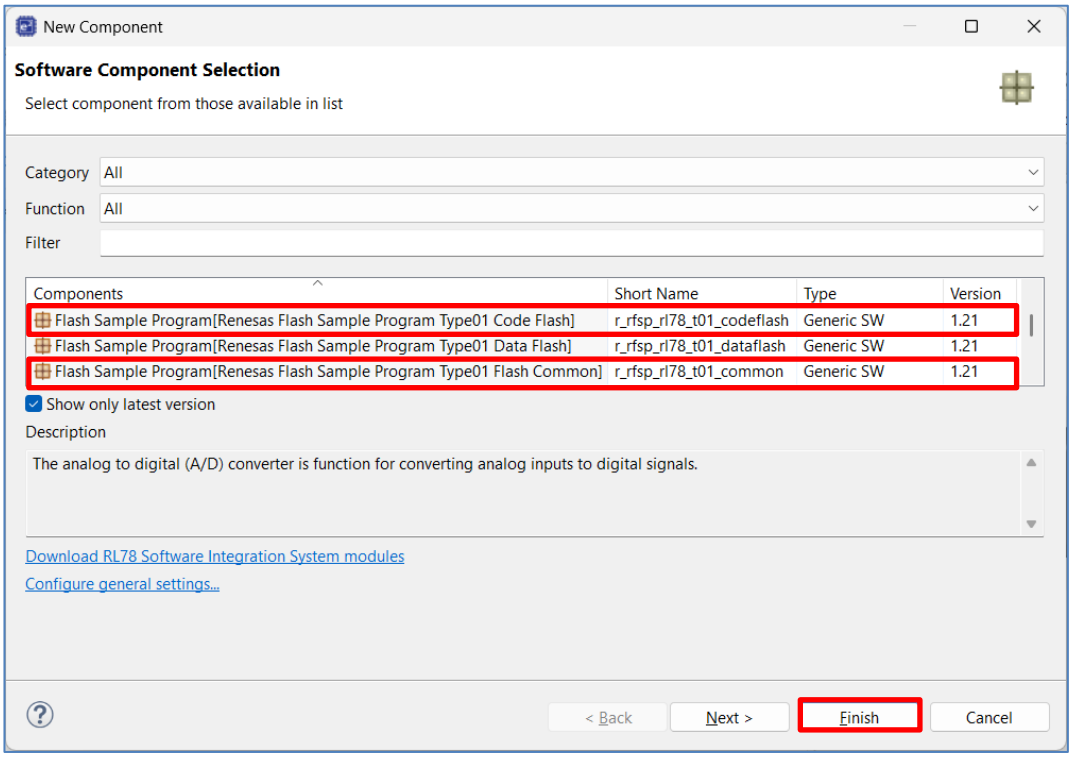


(3) Press the “Add component” button of “Components” and open the “New Component” dialog of “Components”.

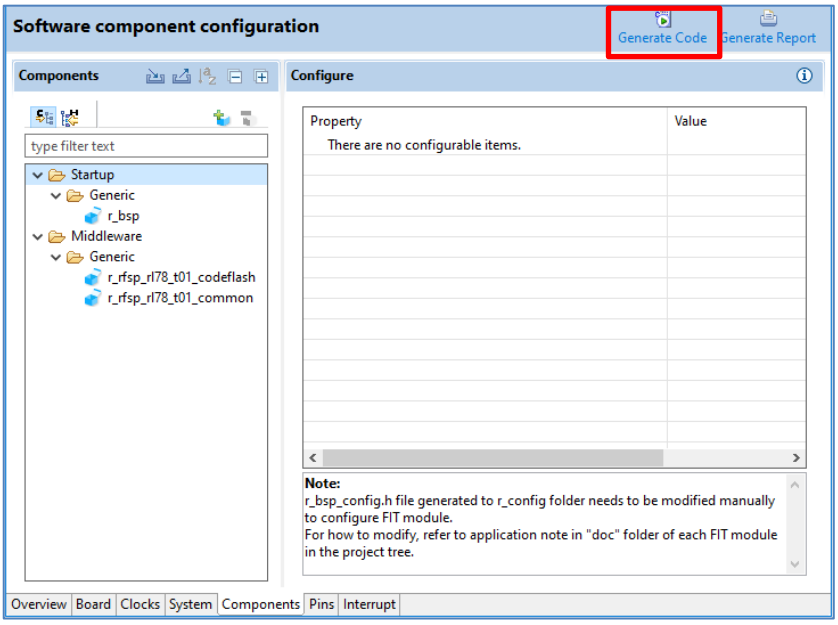


(4) Select the following components and press a “Finish” button.

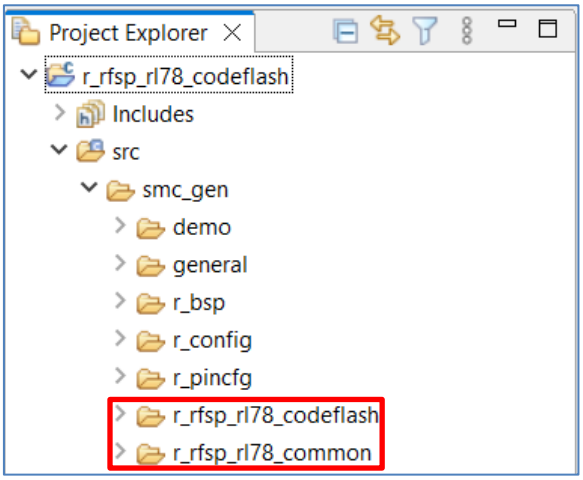
- Flash Sample Program[Renesas Flash Sample Program Type01 Code Flash]
  (r_rfsp _rl78_t01_codeflash)
- Flash Sample Program[Renesas Flash Sample Program Type01 Flash Common]
  (r_rfsp _rl78_t01_common)



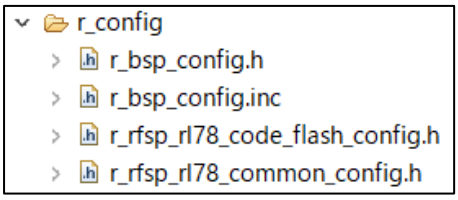
(5) Press a “Generate Code” button and generate the code.



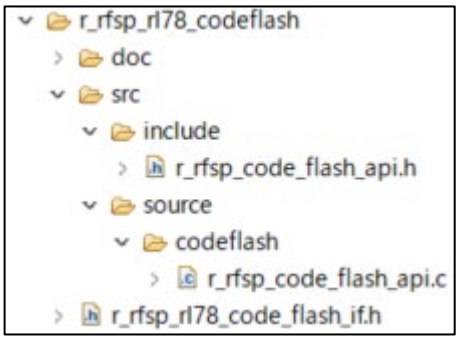
(6) The “r_rfsp_rl78_common” folder and the “r_rfsp_rl78_codeflash” folder is added to the project tree.



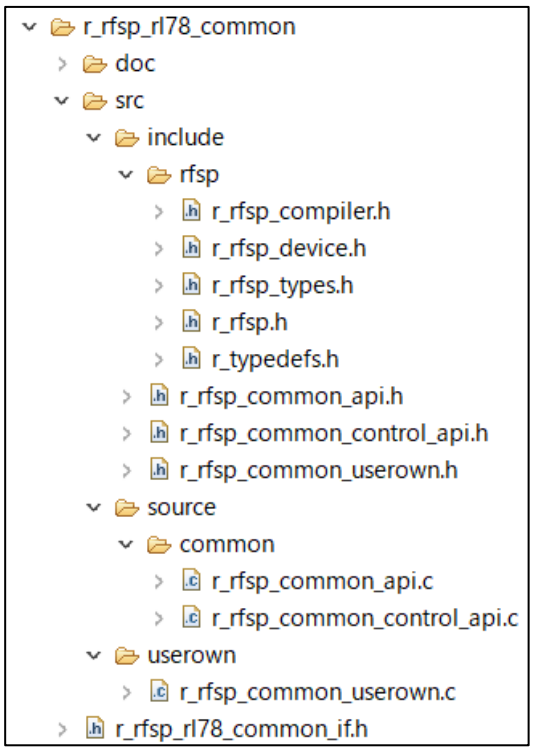
Each folder is developed as follows.



The developed r_config folder



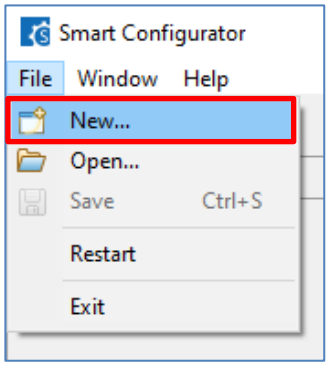
The developed r_rfsp_rl78_codeflash folder



The developed r_rfsp_rl78_common folder

2.2.3 In Case of IAR EW for Renesas RL78

(1) Select “File” [New...] after starting Smart Configurator for RL78.



(2) Select the item of “Platform” and “Toolchain”.

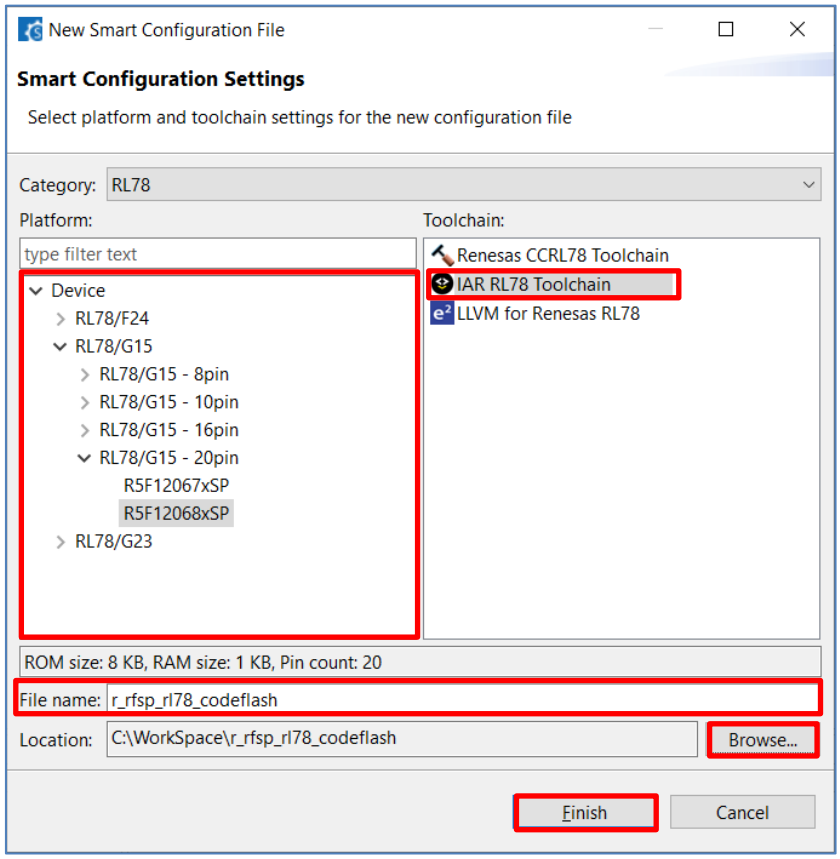
Select the same device as the device selected in “the project of IAR EW for Renesas RL78” by “Platform”.

Select “IAR RL78 Toolchain” as “Toolchain”.

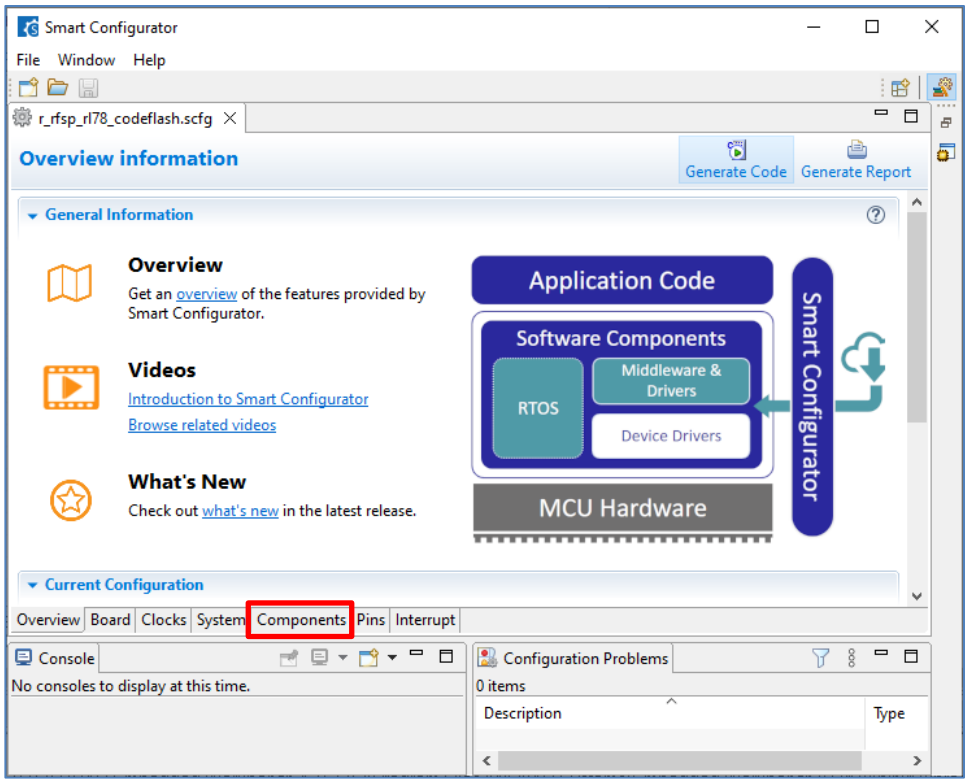
Input arbitrary names into “File name”.

Press the “Browse...” button and set the location of the project folder for IAR EW for Renesas RL78. And press “Finish” button.

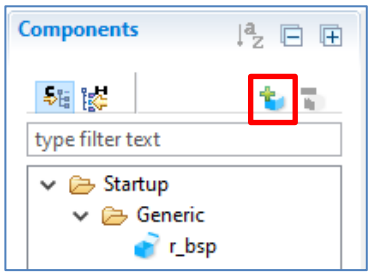
The “.setting” folder and the “<file name>.scfg” file are created to the set location.



(3) Select a “Components” tab.

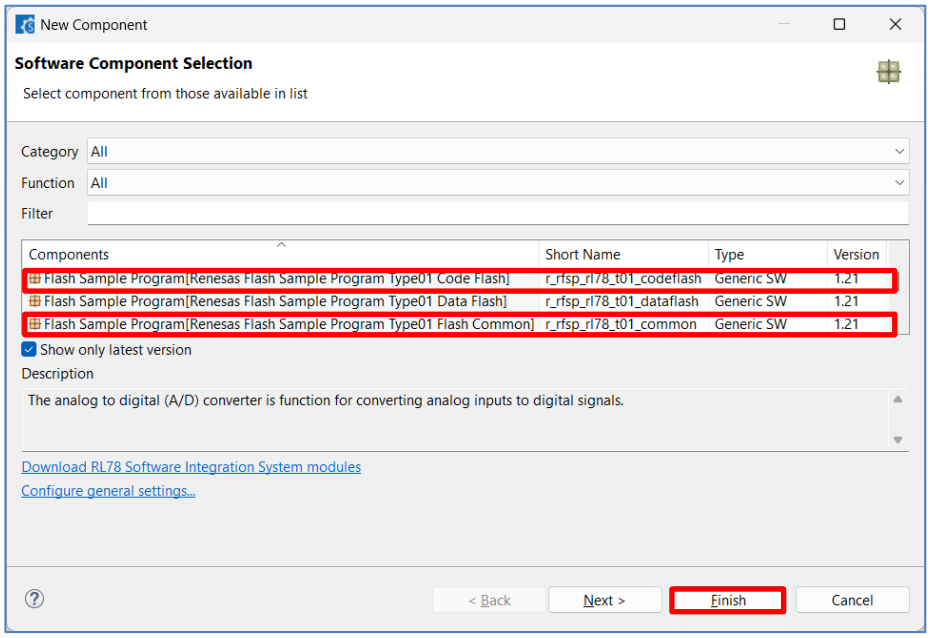


(4) Press the “Add component” button of “Components” and open the “New Component” dialog of “Components”.

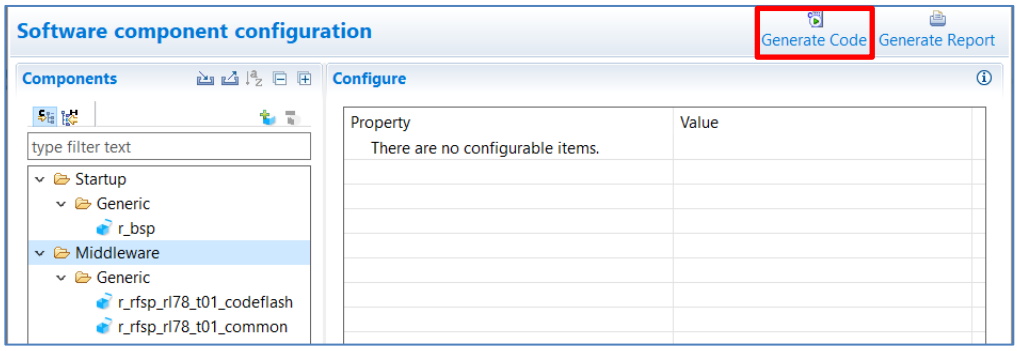


(5) Select the following components and press a “Finish” button.

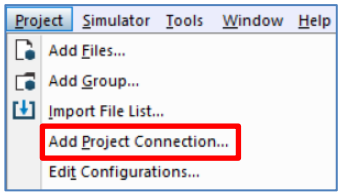
- Flash Sample Program[Renesas Flash Sample Program Type01 Code Flash]
  (r_rfsp _rl78_t01_codeflash)
- Flash Sample Program[Renesas Flash Sample Program Type01 Flash Common]
  (r_rfsp _rl78_t01_common)



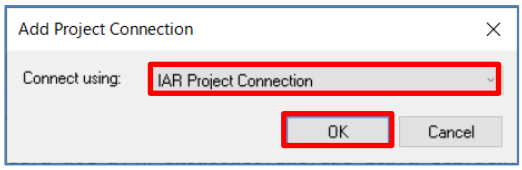
(6) Press a “Generate Code” button and close “Smart Configurator” after the completion of generation for the code.



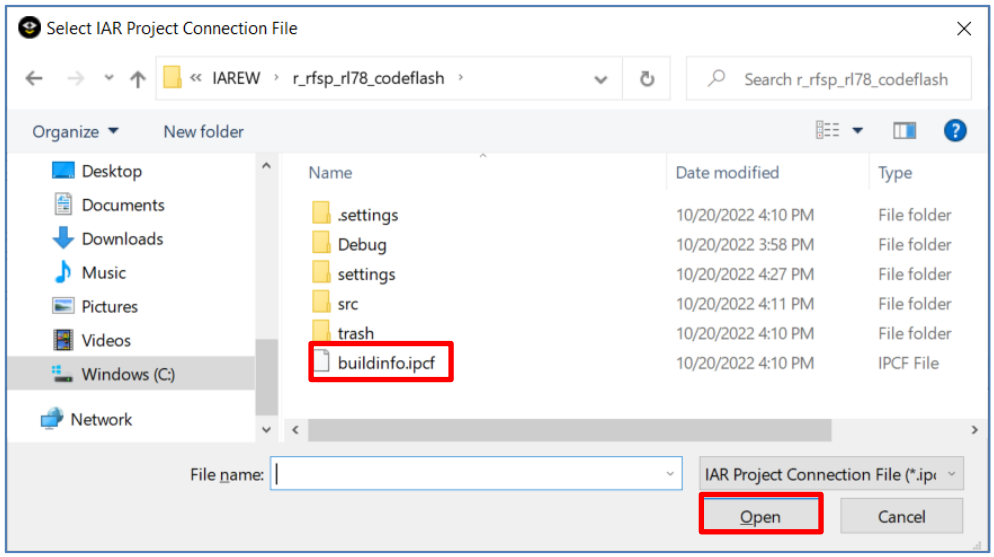
(7) Start IAR EW for Renesas RL78. And select “Project” menu [Add Project Connection], and open the additional dialog of Project Connection.



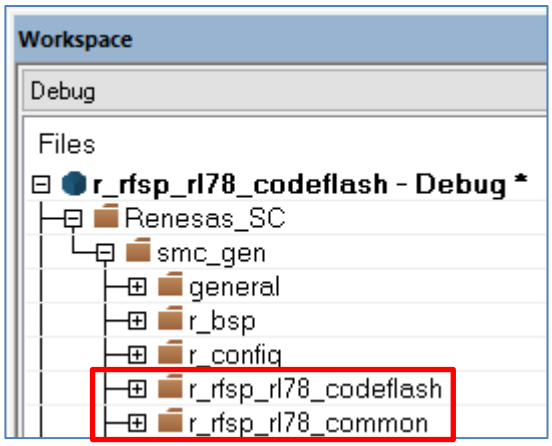
(8)Select “IAR Project Connection”, and press an “OK” button.



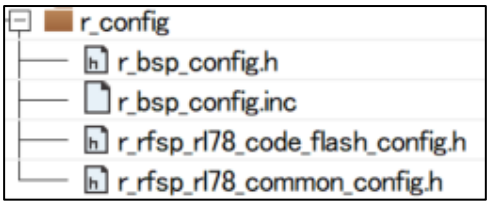
(9) Select the ipcf file created by Smart Configurator and press an “Open” button.



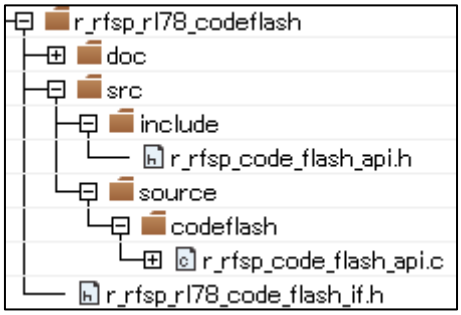
(10) “r_rfsp_rl78_common” and “r_rfsp_rl78_codeflash” are added to Workspace.



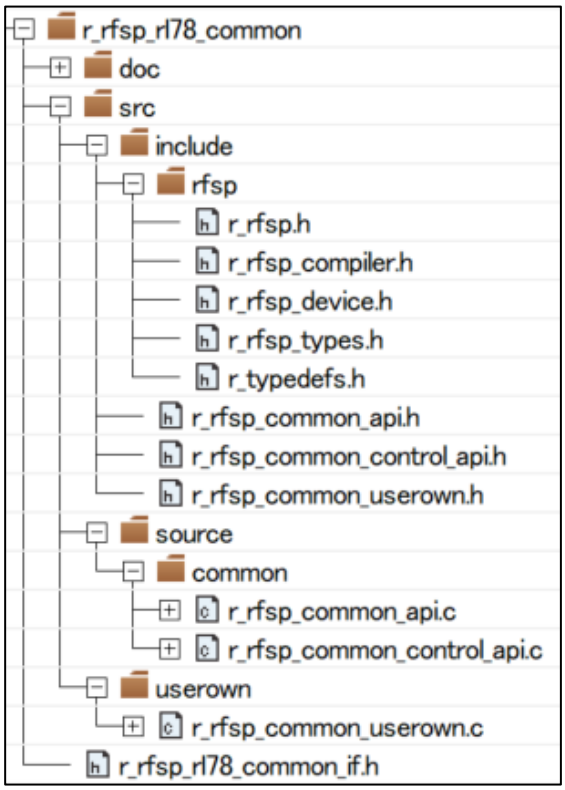
Each folder is developed as follows.



The developed r_config folder



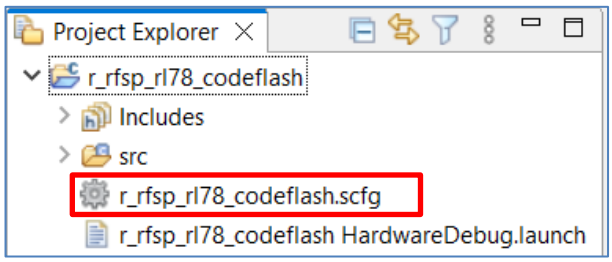
The developed r_rfsp_rl78_codeflash folder



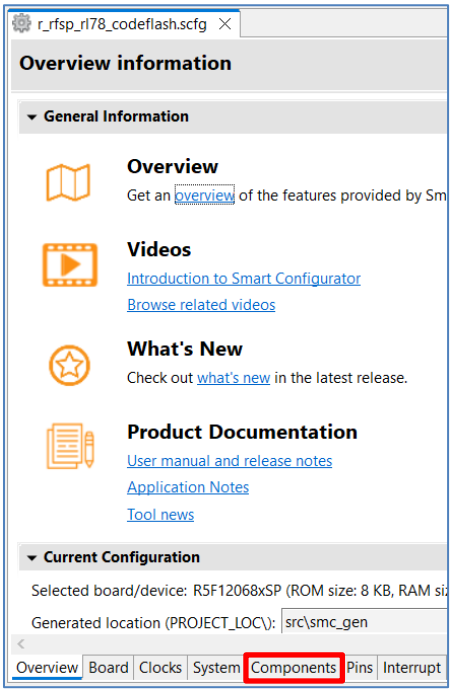
The developed r_rfsp_rl78_common folder

2.2.4 In Case of e² studio (LLVM)

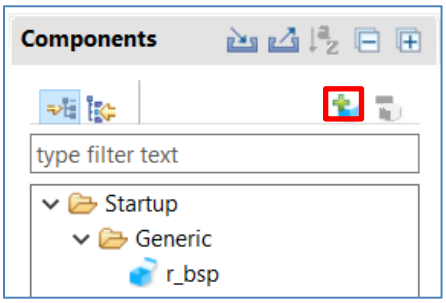
(1) Open the project file of “Smart Configurator” after starting e² studio.



(2) Select a “Components” tab.

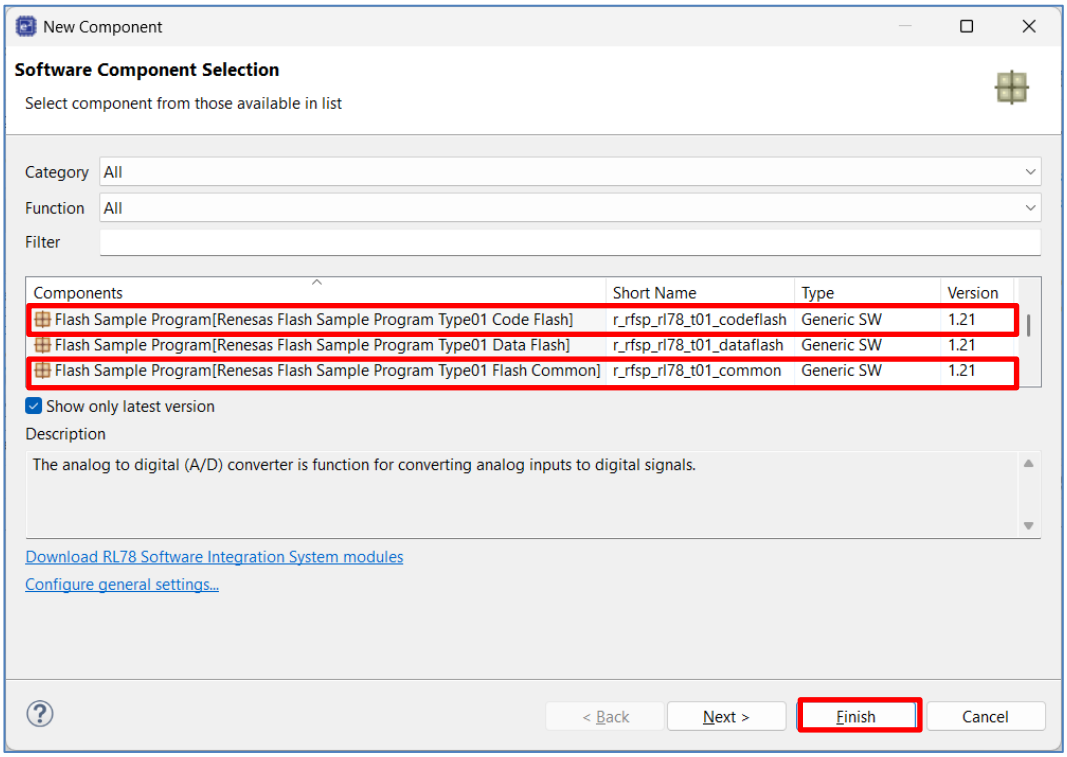


(3) Press the “Add component” button of “Components” and open the “New Component” dialog of “Components”.

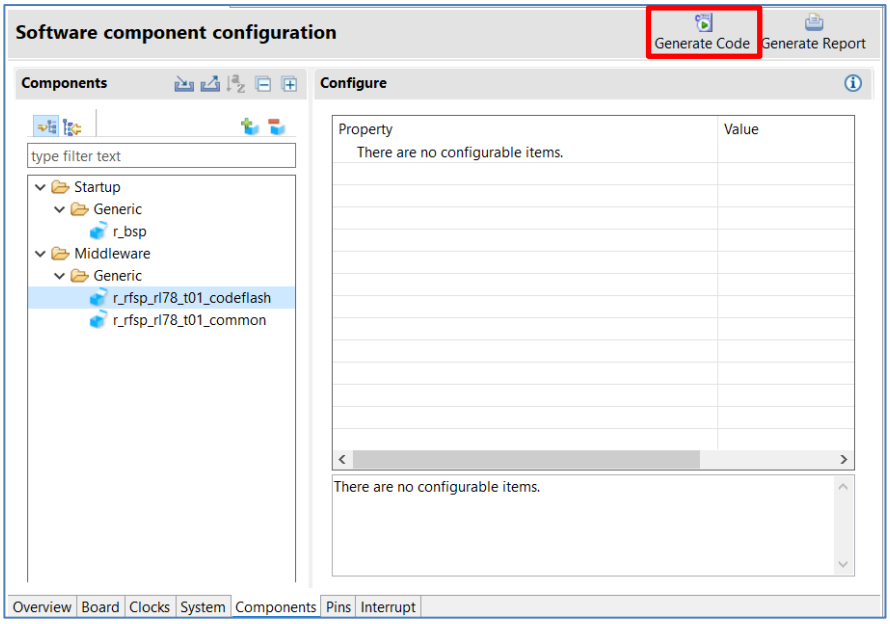


(4) Select the following components and press a “Finish” button.

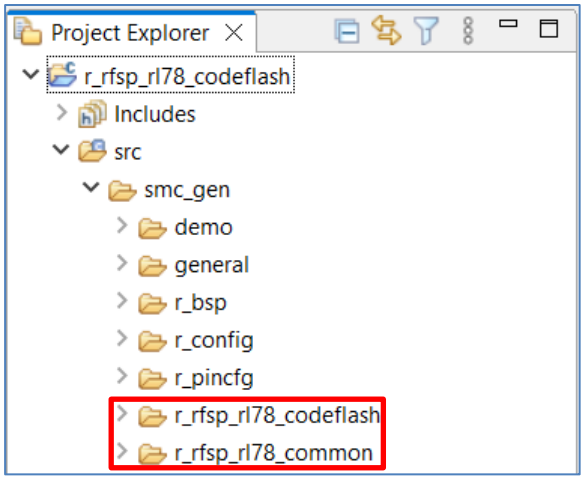
- Flash Sample Program[Renesas Flash Sample Program Type01 Code Flash]
  (r_rfsp _rl78_t01_codeflash)
- Flash Sample Program[Renesas Flash Sample Program Type01 Flash Common]
  (r_rfsp _rl78_t01_common)



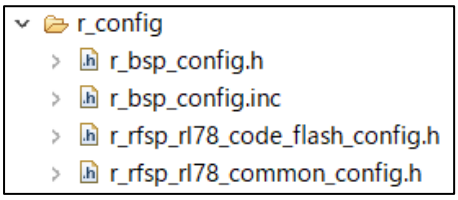
(5) Press a “Generate Code” button and generate the code.



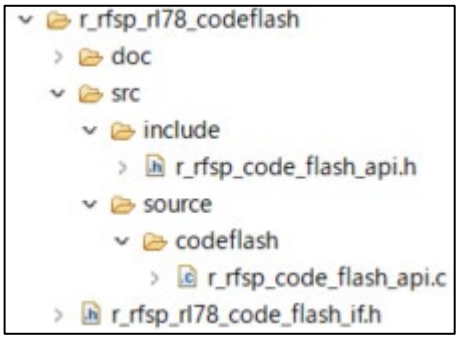
(6) The “r_rfsp_rl78_common” folder and the “r_rfsp_rl78_codeflash” folder is added to the project tree.



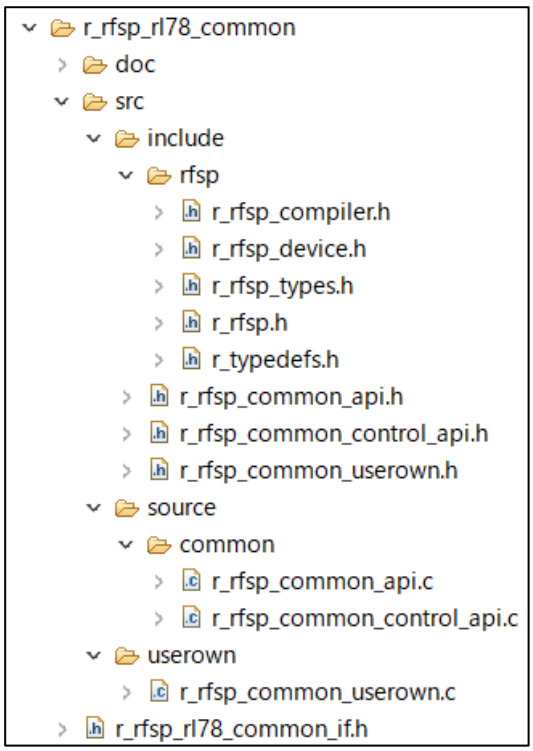
Each folder is developed as follows.



The developed r_config folder



The developed r_rfsp_rl78_codeflash folder



The developed r_rfsp_rl78_common folder

2.3 Project Registration of Sample Program

(1) Extract “CF_sample.zip.”

The folder of the device name generated after extracting leaves only the folder of a target device. Delete the folders of other device names. For example, when using the RL78/G16, leave only the “RL78_G16” folder, and delete the untargeted “RL78_G15” and other folders together.

The common file duplicates in the case which uses it at the same time with the sample program of a data flash area. Extract to overwrite both as the same folder name.

(2) Register the folder of the sample program into the project of CS+, e^2 studio, or IAR.

* Files included in the folder other than the compiler package used, do not need to be registered.

(3) Setting the section items

(3-1) Setting of the section items on CS+

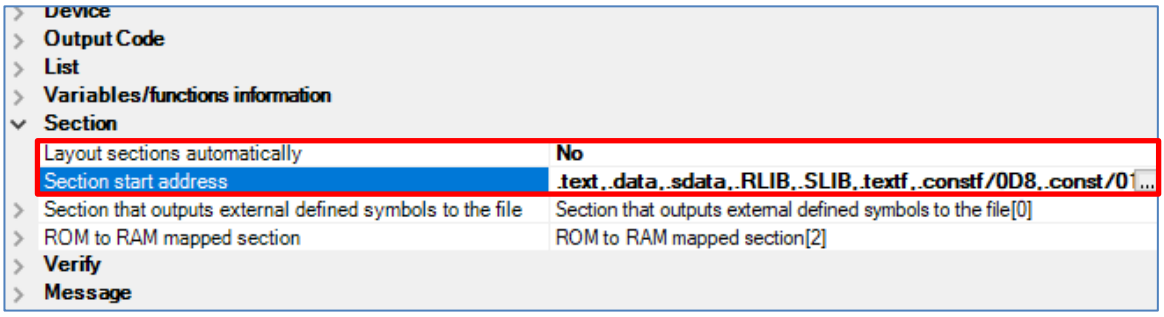
Setting of the section Items on CS+ inputs in the “Link Options” tab. (Common in each area)

- Setting the [Section] items

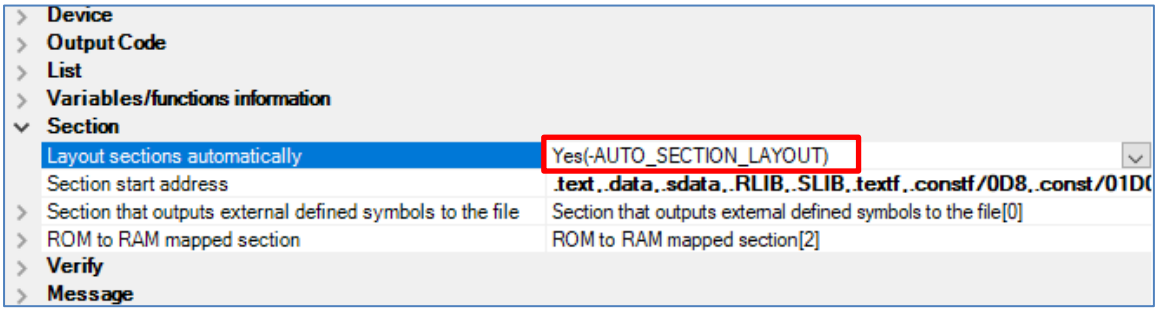
Set “No” to [Layout sections automatically], and the section items display on [Section start address]. And setting the following section items.

“.text,.data,.sdata,.RLIB,.SLIB,.textf,.constf/0D8,.const/01D00,.dataR,.bss/0FFB00,.sdataR,.sbss/0FFE20”

Note: The products of RL78/G16 group are available in two ROM sizes (16 KB or 32 KB). Replace the const section address from “01D00” to “03D00” or “07D00” according to the ROM size of the product to be used.



After the [Section start address] is set up, be sure to restore the set value of [Layout sections automatically] to "yes".



Note: Replace the folder name for the RL78/G15 sample (“RL78_G15”) with the folder name for the target product. Replace the folder name in the case of using RL78/G16 to “RL78_G16”.

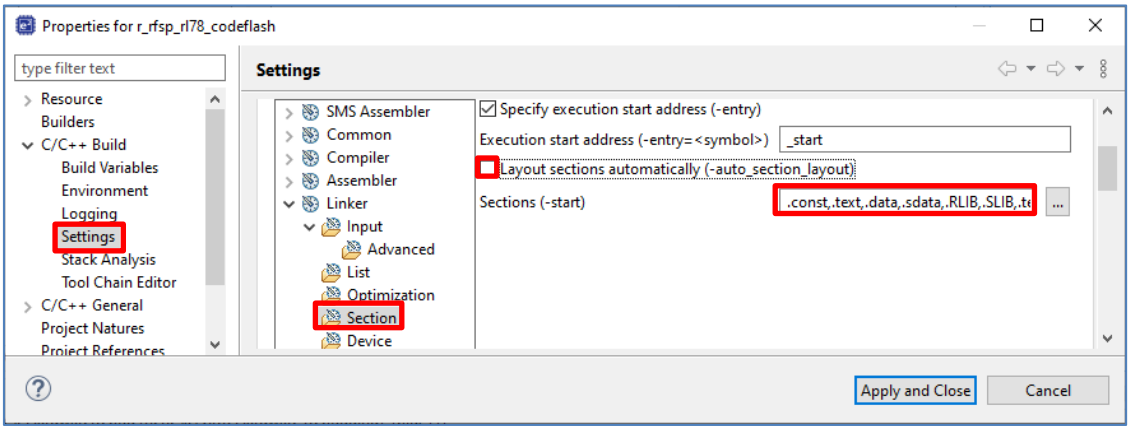
(3-2) Setting of the section items on e^2 studio (CC-RL)

Setting of the section Items on e^2 studio (CC-RL) inputs in the “Properties” window.

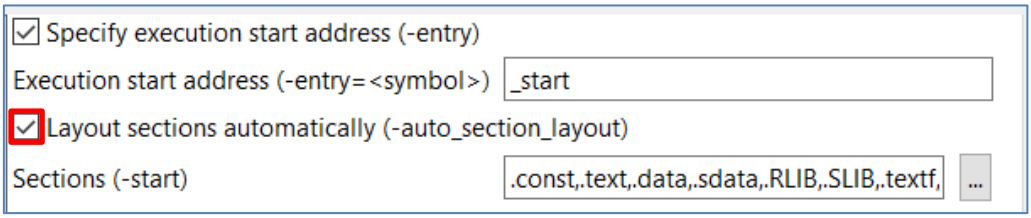
Select the [Property] in a project, and open a Properties window. Select “C/C++ Build” [Setting] - “Linker” [Section]. And set section items on the displayed screen. Remove a check mark to [Layout sections automatically(-auto_section_layout)]. And rewrite the contents of the text box of the [Section (-start)] to following section items.

“.text,.data,.sdata,.RLIB,.SLIB,.textf,.constf/0D8,.const/01D00,.dataR,.bss/0FFB00,.sdataR,.sbss/0FFE20”

Note: The products of RL78/G16 group are available in two ROM sizes (16 KB or 32 KB). Replace the const section address from “01D00” to “03D00” or “07D00” according to the ROM size of the product to be used.



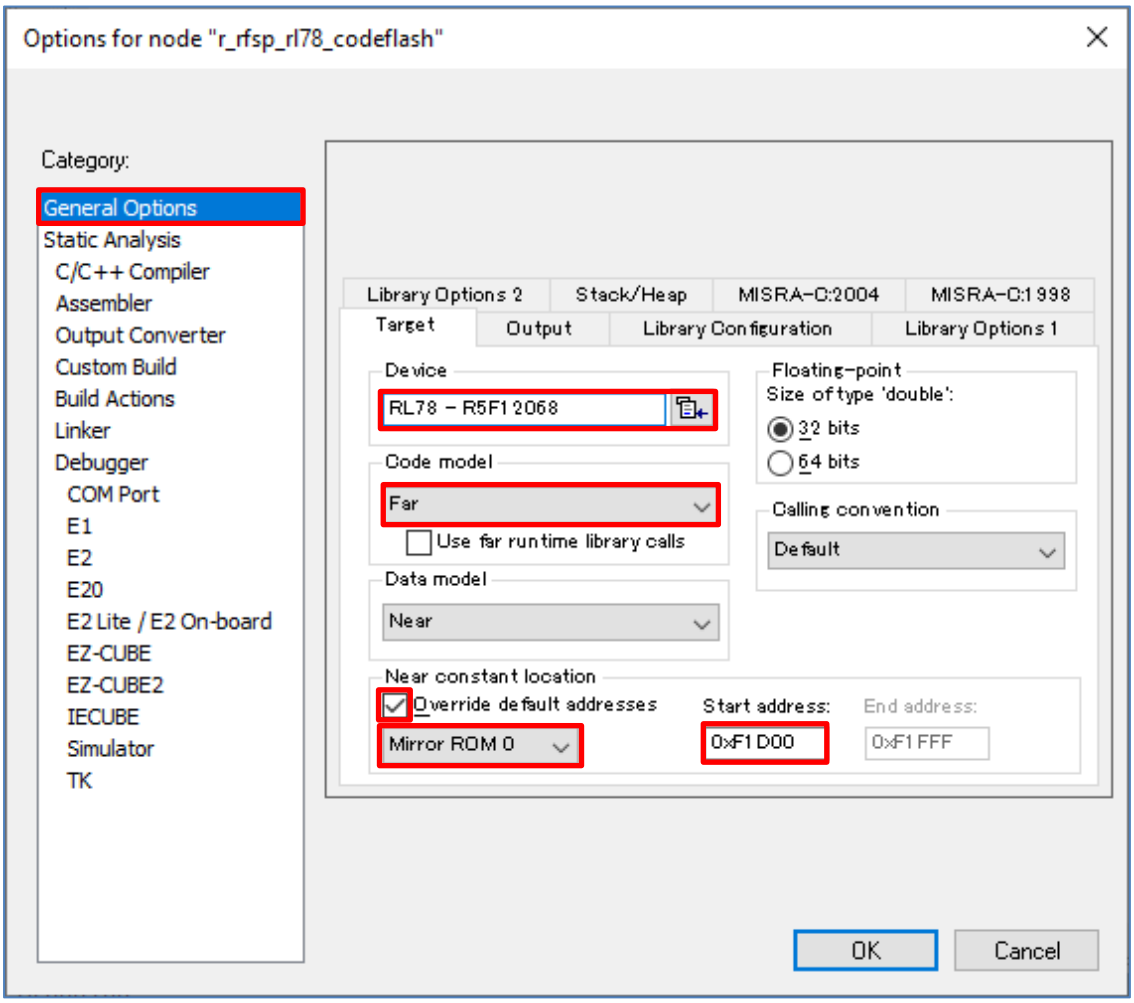
Be sure to put a check mark to [Layout sections automatically (-auto_section_layout)], then press the “OK” button.



(3-3) Setting of the section items on IAR EW for Renesas RL78

Set the items of [General Options] - [Target] tab in the “Options” screen. Select the target device for [Device] and “Far” for [Code model]. To specify the allocation address of the [Near constant location], set the check mark to [Override default address], select “Mirror ROM0”, and change [Start address:] to 0xF1D00.

Note: The products of RL78/G16 group are available in two ROM sizes (16 KB or 32 KB). Replace [Start address:] from “0xF1D00” to “0xF3D00” or “0xF7D00” according to the ROM size of the product to be used.



(3-4) Setting of the section items on e^2 studio (LLVM)

On e^2 studio (LLVM), Linker script file(*. ld) describes link setting executed by building.

Edit the “linker_script.ld” file in the [Project Name]/src folder to modify the start address of the “.rodata” section and the stack size.

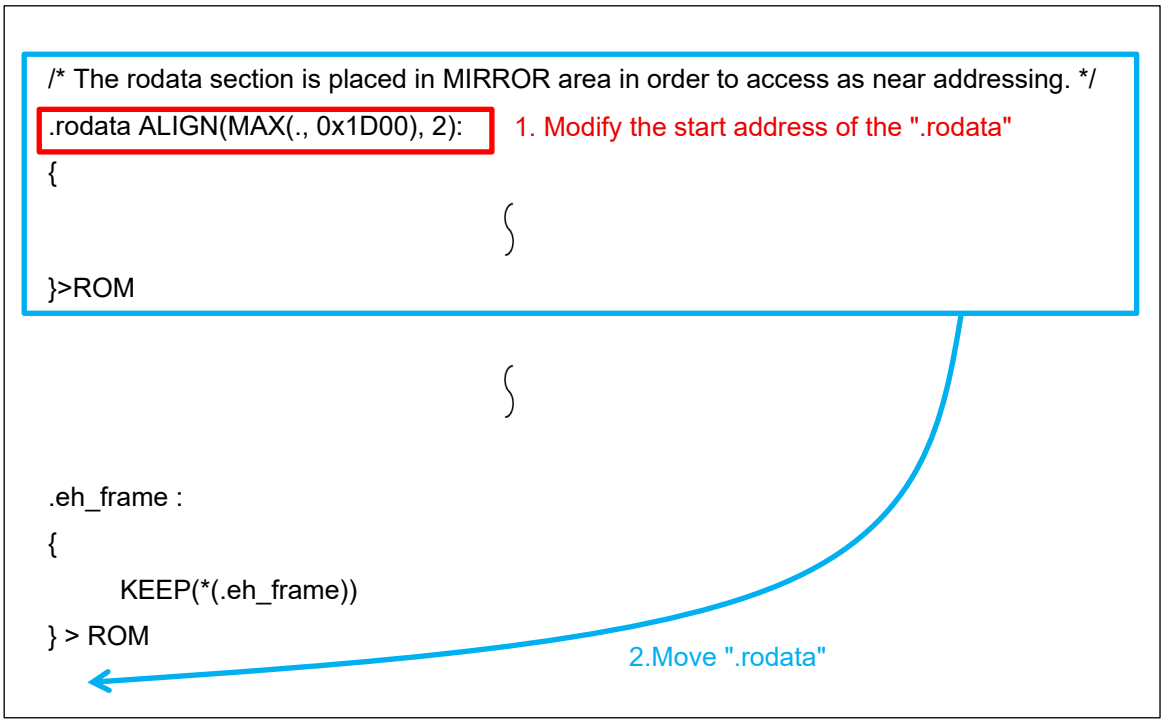
- Change the placement of “.rodata” section

1. Modify the start address of the “.rodata” section to “0x1D00”.

.rodata ALIGN(MAX(., (CONSTANT(MIRRORAREASTART)+0x800)), 2):

→.rodata ALIGN(MAX(., 0x1D00), 2):

2. Move “.rodata” section after “.eh_frame” section.



* The products of RL78/G16 group are available in two ROM sizes (16 KB or 32 KB). The start address of “.rodata” section for each ROM size is as shown below.

RL78/G16(ROM 16KB) : 0x3D00

RL78/G16(ROM 32KB) : 0x7D00

- Change the stack size

Change the stack size definition as follows.

PROVIDE(__stack_size = 0x64);

→PROVIDE(__stack_size = 0x80);

(4) Include Path Settings

The figure shows the case of RL78/G15. In this case, too, the folder name (“RL78_G15”) for the RL78/G15 sample should be read as the folder name of the target device. In the case of RL78/G16, the folder name should be change to “RL78_G16”.

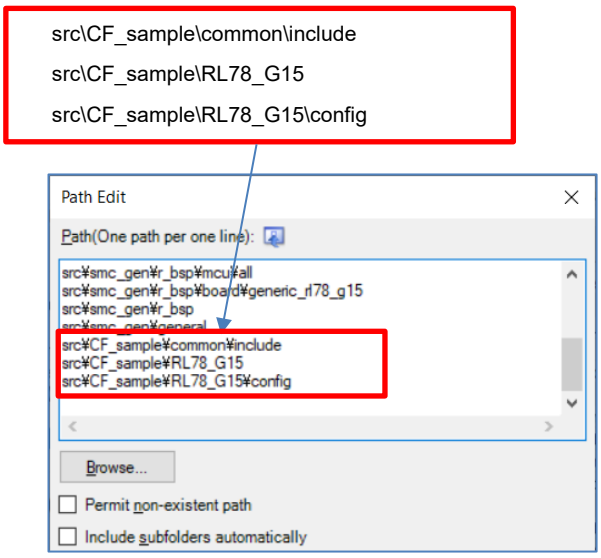
(4-1) Setting of the include path on CS+

Setting of the include path on CS+ inputs path in “Common Options” tab.

- Add the Include directory path in the “Path Edit” window displayed by selection of [Frequently Used Options(for Compile)] - [Additional include paths].

In the phase where code generation was performed by (5) of “2.2.1 In Case of CS+”, the include path of files other than the sample program is registered. For a reason, it needs to register the include path of the sample program.

Those include path to add is shown below.



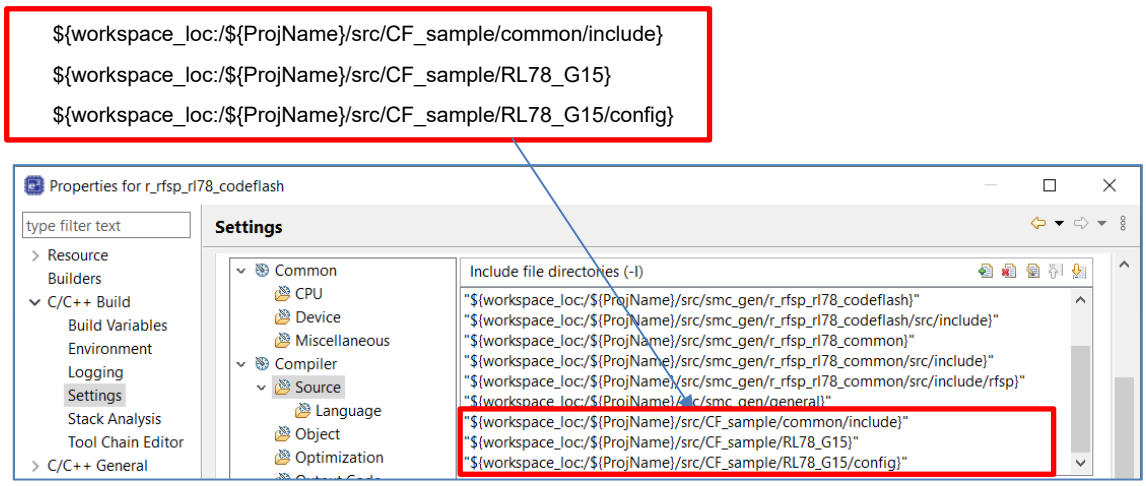
* It is an example of the include path in the case which extracted CF_sample.zip directly under the src folder.

(4-2) Setting of the include path on e^2 studio (CC-RL)

Setting of the include path on e^2 studio (CC-RL) inputs path in "Properties" window.

- Input the Include directory path in the window displayed by selection of “C/C++ Build” [Setting] - “Compiler” [Source].

Those include path to add is shown below.



* It is an example of the include path in the case which extracted CF_sample.zip directly under the src folder.

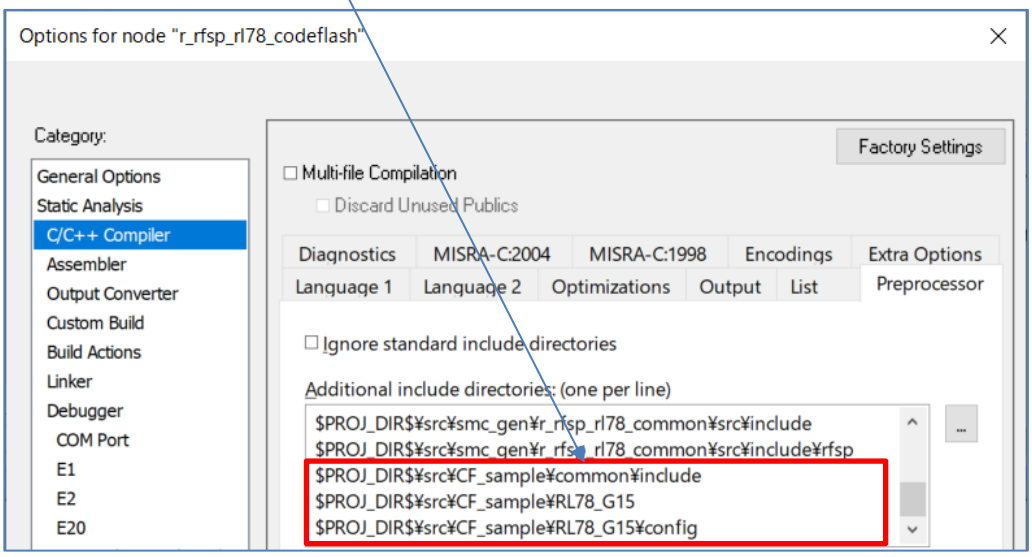
(4-3) Setting of the include path on IAR EW for Renesas RL78

Setting of the include path on IAR Embedded Workbench selects “C/C++ Compiler” of “Category”, and inputs path in “Preprocessor” tab.

- Input the Include directory path in the “Edit include Directories” window displayed by selection of [Additional include directories: (one per line)].

Those include path to add is shown below.

```
$PROJ_DIR$\src\CF_sample\common\include
$PROJ_DIR$\src\CF_sample\RL78_G15
$PROJ_DIR$\src\CF_sample\RL78_G15\config
```



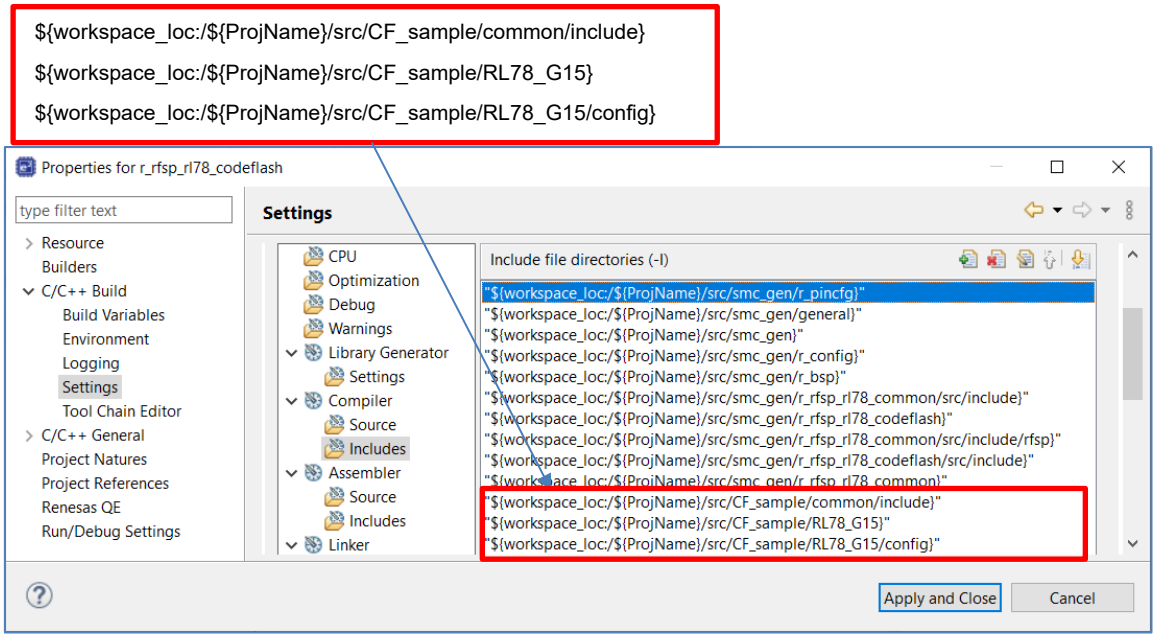
* It is an example of the include path in the case which extracted CF_sample.zip directly under the src folder.

(4-4) Setting of the include path on e^2 studio (LLVM)

Setting of the include path on e^2 studio (LLVM) inputs path in “Properties” window.

- Input the Include directory path in the window displayed by selection of “C/C++ Build” [Setting] - “Compiler” [Includes].

Those include path to add is shown below.



* It is an example of the include path in the case which extracted CF_sample.zip directly under the src folder.

(5) Device Item Settings

(5-1) Setting of the device Items on CS+

Refer to the chapter of “Device Item Settings” of Renesas Flash Sample Program Type01 Application Note (R20AN0652).

(5-2) Setting of the device Items on e^2 studio (CC-RL)

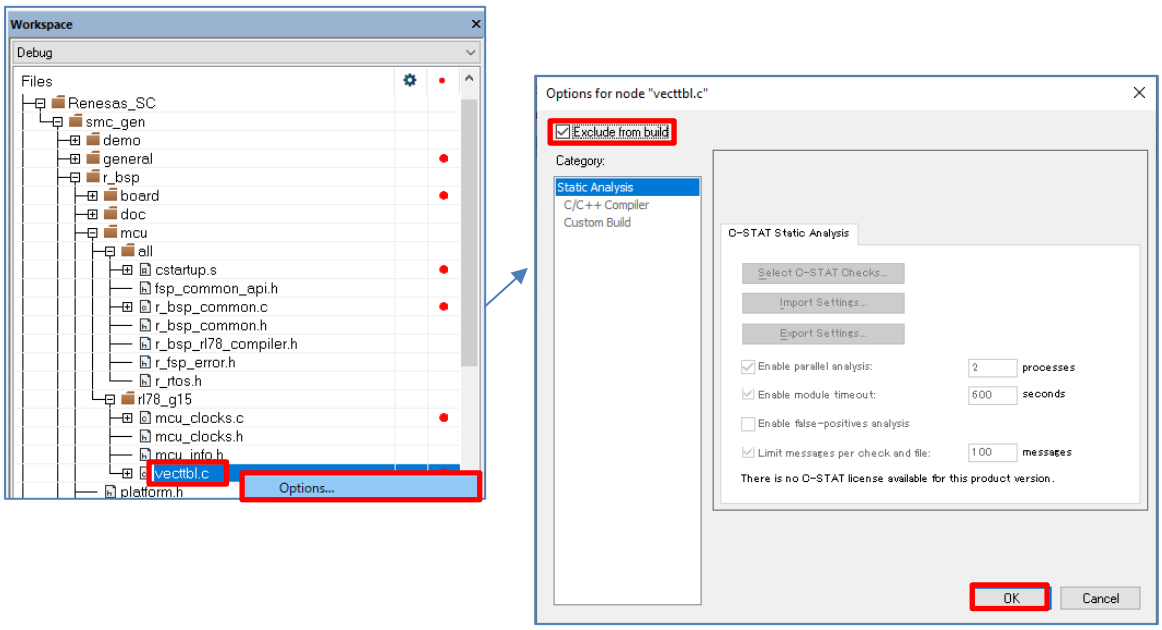
Refer to the chapter of “Device Item Settings” of Renesas Flash Sample Program Type01 Application Note (R20AN0652).

(5-3) Setting of the device Items on IAR EW for Renesas RL78

Build including vecttbl.c prepared as a sample program. A user option byte's value is set to 0xEEFFF9, and an on-chip debugging option byte's value is set to 0x85.

When the code is generated by Smart Configurator, vecttbl.c is generated to a “smc_gen\r_bsp\mcu\rl78_g15\” folder. And because vecttbl.c duplicates, it is necessary to repeal this file.

Right-click a mouse by “Renesas_SC\smc_gen\r_bsp\mcu\rl78_g15\vecttbl.c” in the [project] on a tree. And select an “option” and set a “check” to [Exclude from build] in the displayed screen.



(5-4) Setting of the device Items on e^2 studio (LLVM)

Build including vects.c prepared as a sample program. A user option byte's value is set to 0xEEFFF9, and an on-chip debugging option byte's value is set to 0x85.

When the code is generated by Smart Configurator, r_cg_vect_table.c is generated to a “src\general\” folder. And because r_cg_vect_table.c and vects.c duplicate the settings, it is necessary to repeal this file.

Right-click a mouse by “src\general\r_cg_vect_table.c” in the [project] on a Project Explorer. And select [Resource Configuration] - [Exclude from build...] and use it to exclude r_cg_vect_table.c from build target.

(6) Execute the sample program from a main function.

Describe the sample_codeflash_main function included in r_flash_sample_codeflash_rl78g1x.c like the “main function” for the project. And build, download and execute it.

* The header file which described the prototype declaration for sample_codeflash_main function. Include prepared “r_flash_sample_codeflash_rl78g1x.h.”

2.4 The Check of Operation for Sample Program

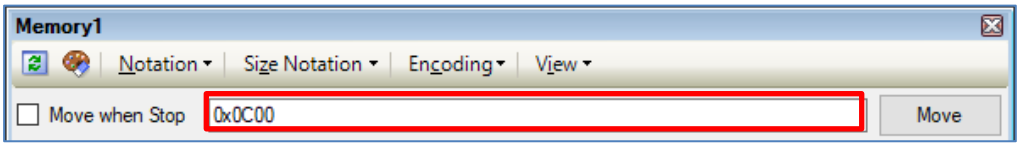
The block 3 (0x0C00) of a code flash area is erased. And 16 Words(64 bytes) of data is programmed from the top of the block 3. Confirm this operation by the following methods.

2.4.1 In Case of CS+

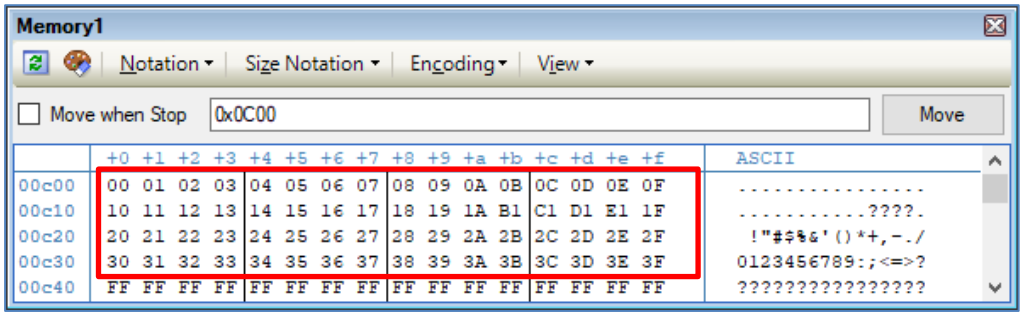
(1) Select [Debug]menu - [Download] and start debugging.

(2) After selecting [View]menu - [Memory], select “Memory1”, “Memory2”, “Memory3”, or “Memory4”, and display a memory window.

(3) Set “0x0C00” to the address of a memory window, and display the block 3 of a code flash area.



(4) Select [Debug]menu - [Go] and the program is executed. Check that the value of “0x0C00-0x0C3F” displayed on the memory window has changed after stopping the program.

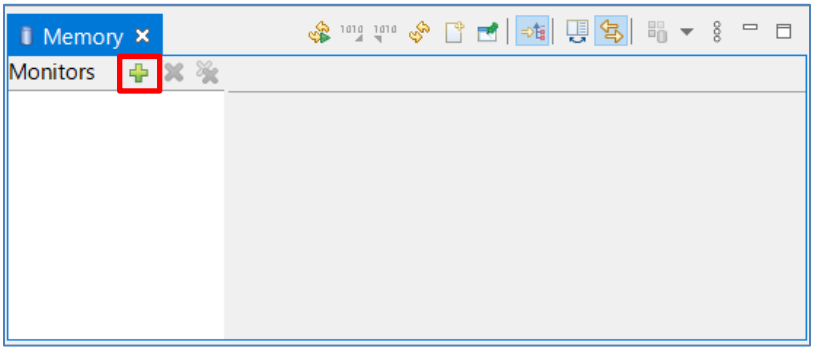


2.4.2 In Case of e² studio (CC-RL)

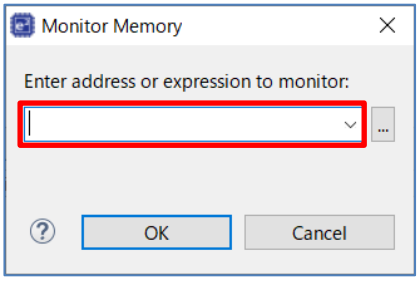
(1) Select [Run]menu - [Debug] and start debugging.

(2) Select [Window]menu - [Show View] - [Memory] and display Memory view.

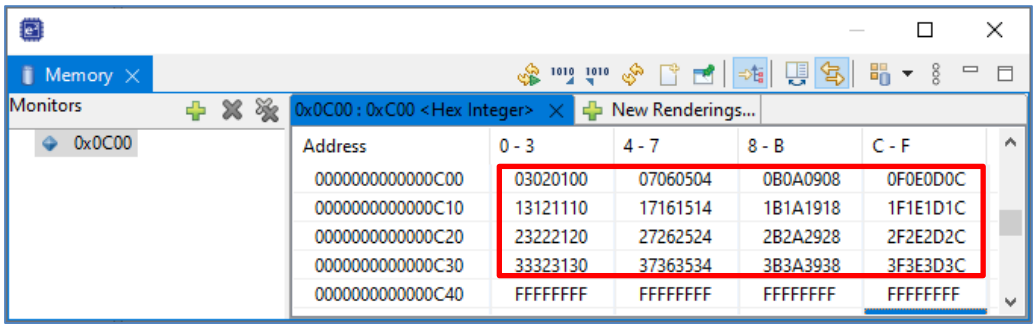
(3) Press " " button and display the Monitor Memory window.



(4) Input "0x0C00" into the address to monitor, and press the OK button. The block 3 of a code flash area is displayed on the memory view.



(5) After the program is executed, check that the value of "0x0C00-0x0C3F" displayed on the memory window is changing.



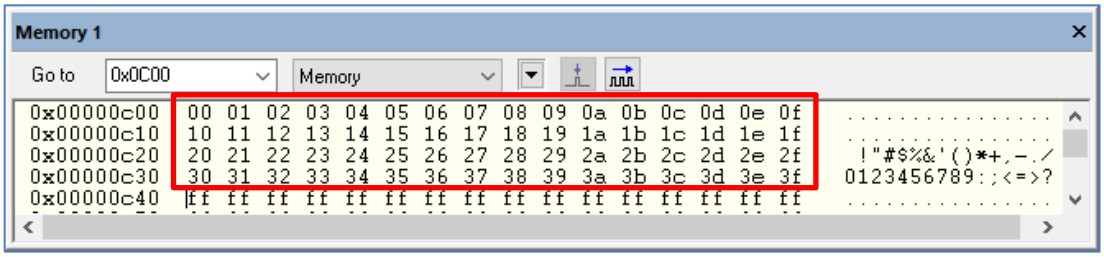
2.4.3 In Case of IAR EW for Renesas RL78

(1) Select [Project] menu - [Download and Debug] and start debugging.

(2) After select [View]menu - [Memory], select either from “Memory 1” to “Memory 4”, and the memory window is displayed.

(3) Set “0x0C00” to the address of the memory window. And the block 3 of code flash area is displayed.

(4) After the program is executed, check that the value of “0x0C00-0x0C3F” displayed on the memory window is changing.

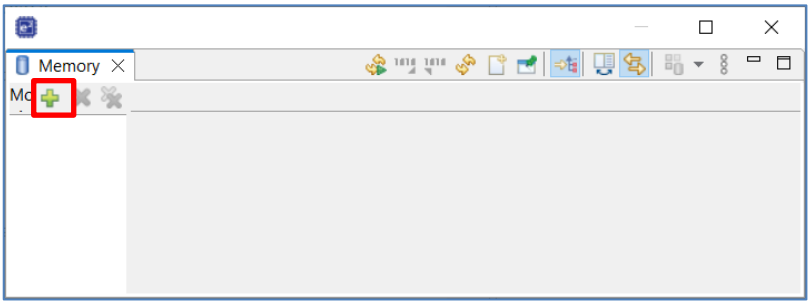


2.4.4 In Case of e² studio (LLVM)

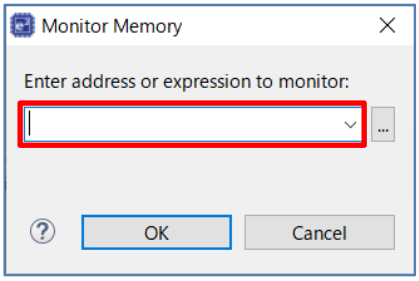
(1) Select [Run]menu - [Debug] and start debugging.

(2) Select [Window]menu - [Show View] - [Memory] and display Memory view.

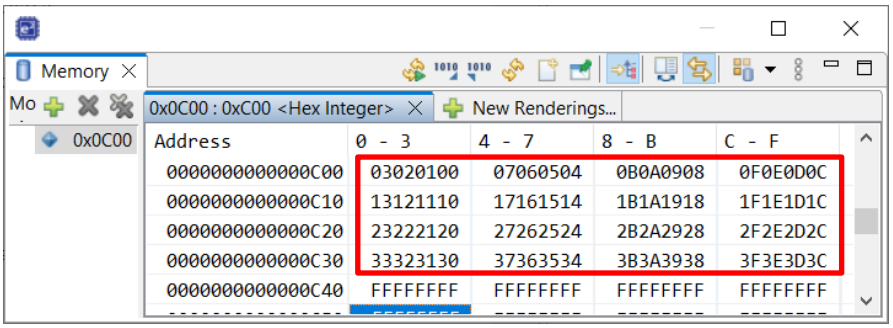
(3) Press " " button and display the Monitor Memory window.



(4) Input "0x0C00" into the address to monitor, and press the OK button. The block 3 of a code flash area is displayed on the memory view.



(5) After the program is executed, check that the value of "0x0C00-0x0C3F" displayed on the memory window is changing.



3. Precautions for Specifications

(1) Allocation of the user program for flash memory reprogramming operation

Allocate the user program for programming the code/data flash area to the code flash area. Self-programming by fetching from the RAM is prohibited. Additionally, reprogramming the boot area and the block for storing the user program for executing self-programming is prohibited.

(2) Prohibit the interrupts in self-programming mode

Prohibit an interrupt before setting the self-programming mode. To prohibit an interrupt, clear (0) the IE flag by the DI instruction in the same way as in the normal operation mode.

(3) Setting the CPU operating frequency for the flash memory sequencer

When using the flash memory sequencer to reprogram the code/data flash memory, set the value corresponding to the CPU operating frequency in the FSET4-0 bits of the FSSET register before proceeding. Note that if reprogramming is attempted while the value corresponding to the CPU operating frequency is not correct, operation is undefined and written data are not guaranteed. Even if the values in the flash memory are as expected immediately after reprogramming, retaining the values for any specified period is not guaranteed.

(4) Operation setting of high-speed on-chip oscillator

The high-speed on-chip oscillator should be kept operating before executing self-programming. If it is stopped, it should be made to operate again (HIOSTOP = 0), and the flash self-programming code should be re-executed after 30 us have elapsed.

(5) Restriction of execution of other operations during self-programming

Do not execute other settings or instructions which are not related to the self-programming procedure during the self-programming execution flow

(6) User program operation during flash memory reprogramming operation

The CPU is stopped during reprogramming through self-programming. The code flash or data flash memory cannot be accessed while it is being reprogrammed.

4. Reference Document

Please get the latest version of each document from the Renesas Electronics Corp. website (https://www.renesas.com).

No	Document Title	Document Number
1	RL78/G15 User's Manual Hardware	R01UH0959
2	RL78/G16 User's Manual Hardware	R01UH0980
3	RL78 Family Board Support Package Module	R01AN5522
4	Renesas Flash Sample Program Type01 Application Note	R20AN0652
5	E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RL78)	R20UT1994

5. Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct.31.22	—	Newly created.
1.10	Apr.28.23	—	RL78/G16 was added.
1.20	Nov.30.23	—	The LLVM compiler was supported.
1.21	Oct.31.25	—	The linker configuration files for RFSP for IAR compiler and the LLVM compiler are deleted. And there was modification which has the file linker configuration files which integrated environment outputs used.

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

2. Processing at power-on

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

3. Input of signal during power-off state

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

4. Handling of unused pins

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

5. Clock signals

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

6. Voltage application waveform at input pin

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

7. Prohibition of access to reserved addresses

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

8. Differences between products

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

Notice

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

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Contact information

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

Trademarks

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