Thank you for using the RL78 Family Flash Self-Programming Library Type01 Package Ver.3.00.

This document contains restrictions and notes regarding use of the Flash Self-Programming Library Type01 Package Ver.3.00. Please read this document before using the library.

Contents

Chapter 1 Target Product 2
Chapter 2 User's Manual 2
Chapter 3 Revisions 2
Chapter 4 Points for Caution 3
Chapter 5 Supported Tools 3
Chapter 6 Installation 4
6.1 Installation 4
6.2 Uninstallation 4
6.3 File Organization 5
Chapter 7 How to Build a Program 6
7.1 Software to be used 6
7.2 Building using CS+(former CubeSuite+) 6
7.2.1 Building a C program 6
7.2.2 Building an assembly language program 9
7.3 Notes at Build 12
7.3.1 When the CA78K0R Compiler is Used 12
7.3.2 When the CC-RL Compiler is Used 13
Chapter 8 How to Debug a Program 14
Chapter 9 Sample Program 15
9.1 Initial Settings of the Sample Program 15
9.2 Settings of Option byte and On-Chip Debugging 16
9.3 Compilation Switch for the C-Language Sample Program 17
9.4 Defining the Internal RAM Area 18
9.4.1 When the CA78K0R Compiler is Used 18
9.4.2 When the CC-RL Compiler is Used 19

Chapter 1 Target Product

The Flash Self-Programming Library Type01 Package Ver.3.00 was changed to install files using by an installer.

The following shows the target products for this release note.

Product Name	Ver.	Installer File Name	Ver.
Flash Self-Programming Library Type01 for CA78K0R Compiler for the RL78 Family	V2.20	RENESAS_RL78_FSL_T01_3V00.exe	V3.00
Flash Self-Programming Library Type01 for CC-RL Compiler for the RL78 Family	V2.21		

Chapter 2 User's Manual

The following user's manual covers this version of the library.

Title	Document Number
RL78 Family Flash Self-Programming Library Type01 User's Manual	R01US0050EJ0105

Chapter 3 Revisions

The following shows the items revised in this version.

No.	Package Ver.	Target	Contents
1	V3.00	Library V2.20 for CA78K0R Compiler	There are no changes in the library from the previous ZIP File (JP_R_FSL_RL78_T01_V2.21_B_E.zip).
		Library V2.21 for CC-RL Compiler	There are no changes in the library from the previous ZIP File (JP_R_FSL_RL78_T01_V2.21_B_E.zip).
		User's manual	Revised from Rev.1.04 to Rev.1.05. For details on the corrections to the user's manual in response to the revision, refer to the revision history of the user's manual.

Chapter 4 Points for Caution

For points for caution on using the Flash Self-Programming Library Type01, read this chapter and the user's manual described later.

No.	Description
1	• Debugging by a simulator The flash self-programming library cannot be debugged by a simulator. To perform debugging, either use the on-chip debugging function of the RL78 microcontroller or prepare the IECUBE.
2	• Restrictions regarding use of the flash self-programming library on RL78/G13 sample devices (not including mass-produced devices) Some RL78/G13 sample devices (not including mass-produced devices) have restrictions on the interrupt vector change processing through the flash self-programming library. Note that the following flash functions cannot be used on those devices. • FSL_ChangeInterruptTable • FSL_RestoreInterruptTable

Chapter 5 Supported Tools

Use the following tool version when using the Flash Self-Programming Library Type01.

Target library	Tool Name	Version
Library for CA78K0R Compiler	Integrated development environment CubeSuite+	V1.00.00 or later
	Integrated development environment CS+	V3.00.00 or later
Library for CC-RL Compiler	Integrated development environment CS+	V3.01.00 or later

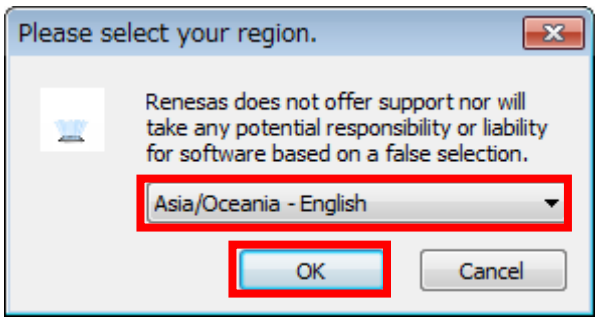
Chapter 6 Installation

This chapter describes how to install and uninstall the Flash Self-Programming Library Type01 Package Ver.3.00.

6.1 Installation

Install the Flash Self-Programming Library Type01 by using the following procedure:

(1) Start Windows.
(2) Decompress the file that contains the Flash Self-Programming Library Type01 Package and run the installer.
(3) Select "Asia/Oceania - English" from the drop-down list.
(4) Click on the "OK" button to proceed installation according to the instructions of the installer.



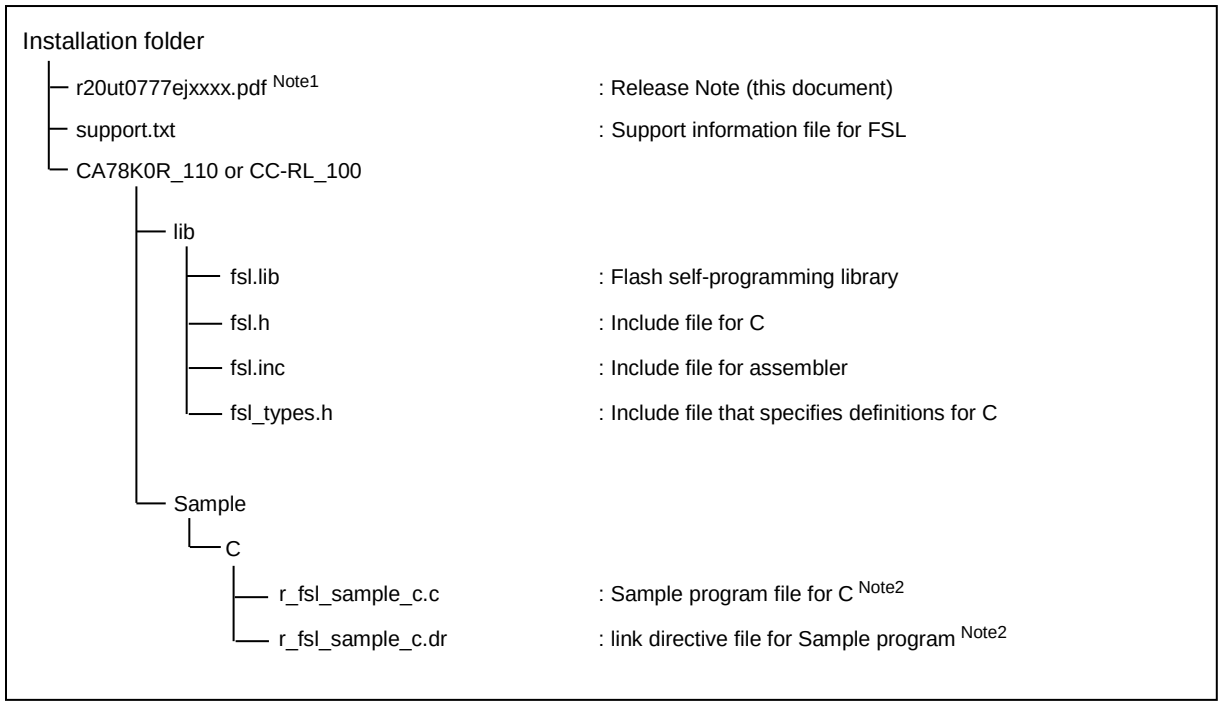
6.2 Uninstallation

Uninstall the Flash Self-Programming Library Type01 by using the following procedure:

(1) Start Windows.
(2) Delete the folder that contains the Flash Self-Programming Library Type01 files.

6.3 File Organization

The file organization after this library is installed is shown below.



Notes:
1. x indicates the omitted numerals in version or revision numbers.
2. To use the sample program, the program file (*.c) and link directive file (*.dr) should be embedded together.
   The link directive file (*.dr) that specifies the link information is not provided with the library for CC-RL. The link information for the sample program for CC-RL should be specified through the link setting window on CS+.

Chapter 7 How to Build a Program

This chapter describes how to build a program using the Flash Self-Programming Library Type01.

7.1 Software to be used

The following integrated development environment is necessary for building programs using the Flash Self-Programming Library Type01.

- Integrated development environment CS+ V3.00.00 or later for CA78K0R compiler
  /Integrated development environment CubeSuite+ V1.00.00 or later for CA78K0R compiler
- Integrated development environment CS+ V3.01.00 or later for CC-RL compiler

7.2 Building using CS+(former CubeSuite+)

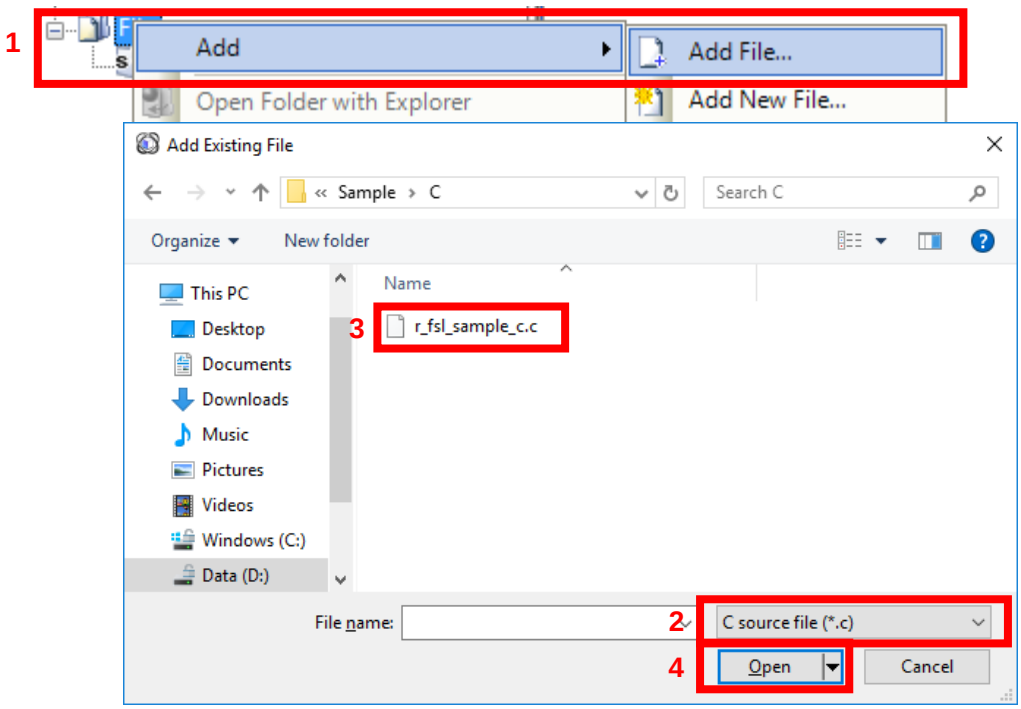
This section describes how to include the Flash Self-Programming Library Type01 in a user-created program and build the user program by using CS+.

7.2.1 Building a C program

(1) Creating a project and specifying the source file

Create a project by using CS+. In the Project Tree window displayed on the left, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-1).
Next, click the Files of type drop-down list to display a list of the file types. Select C source file (*.c), and then register the user-created program as the source file.

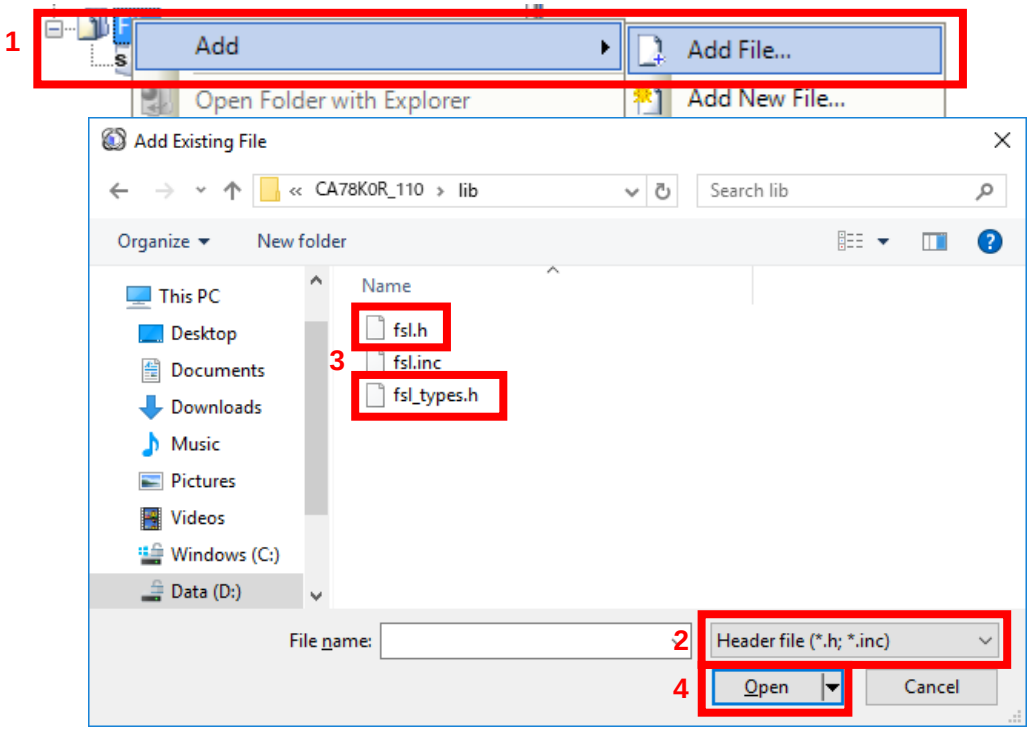
Figure 7-1. Registering the User Program File



(2) Specifying the include file

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File.
The Add Existing File dialog box is displayed (as shown in Figure 7-2).
Next, click the Files of type drop-down list to display a list of the file types. Select Header file (*.h;*.inc), and then register the header files (fsl.h, fsl_types.h) of the flash self-programming library.

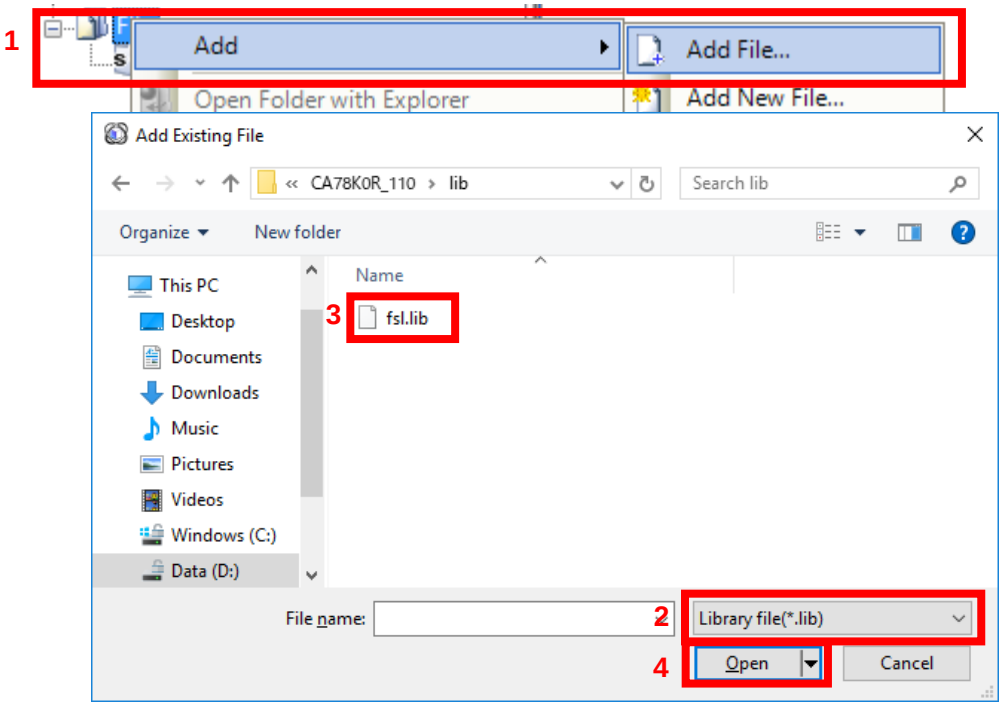
Figure 7-2. Registering the Include Files



(3) Specifying the library file

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-3).
Next, click the Files of type drop-down list to display a list of the file types. Select Library file (*.lib), and then register the flash self-programming library file (fsl.lib).

Figure 7-3. Registering the Library File

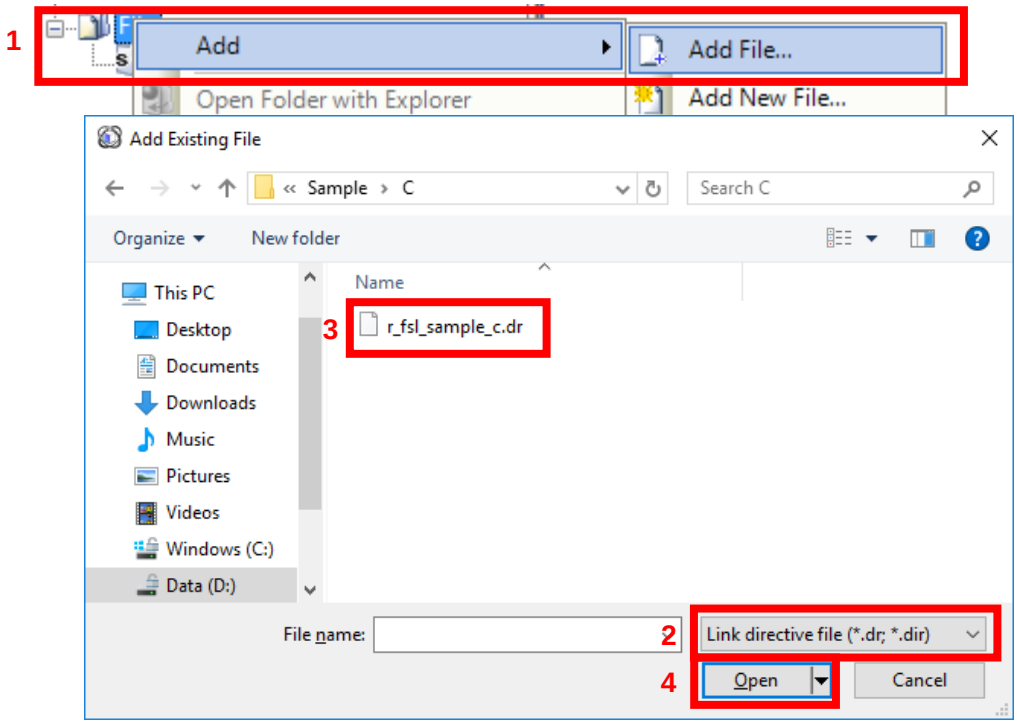


(4) Specifying the link directive file (only when the CA78K0R compiler is used)

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-4).

Next, click the Files of type drop-down list to display a list of the file types. Select Link Directive File (*.dr;*.dir), and then register the link directive file that has the same name as the user-created program.

Figure 7-4. Registering the Link Directive File



(5) Building

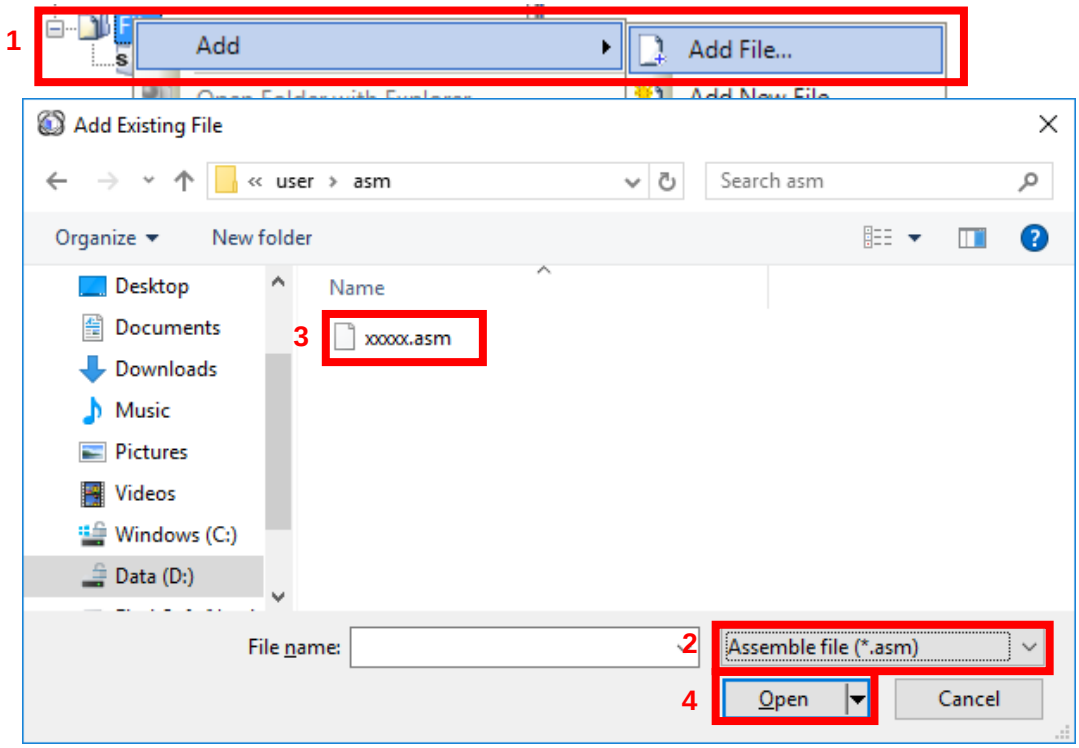
On the CS+ Build menu, click Build Project to build the project.

7.2.2 Building an assembly language program

(1) Creating a project and specifying the source file

Create a project by using CS+. In the Project Tree window displayed on the left, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-5).
Next, click the Files of type drop-down list to display a list of the file types. Select Assemble file (*.asm), and then register the user-created program as the source file.

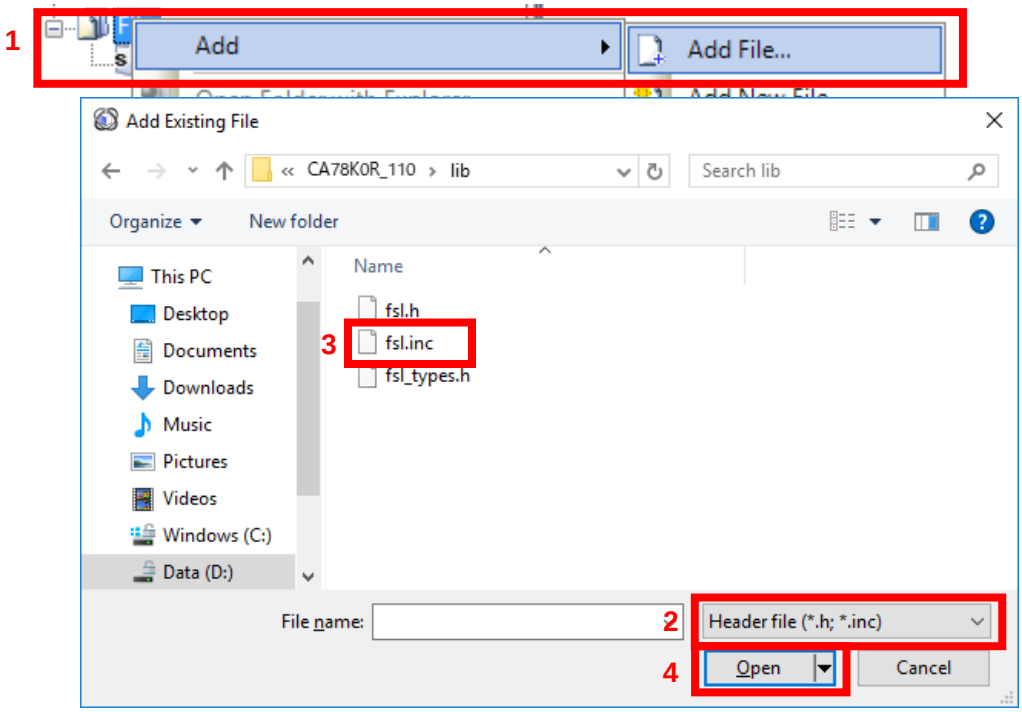
Figure 7-5. Registering the User Program File



(2) Specifying the include file

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File.
The Add Existing File dialog box is displayed (as shown in Figure 7-6).
Next, click the Files of type drop-down list to display a list of the file types. Select Header file (*.h;*.inc), and then register the header file (fsl.inc) of the flash self-programming library.

Figure 7-6. Registering the Include File

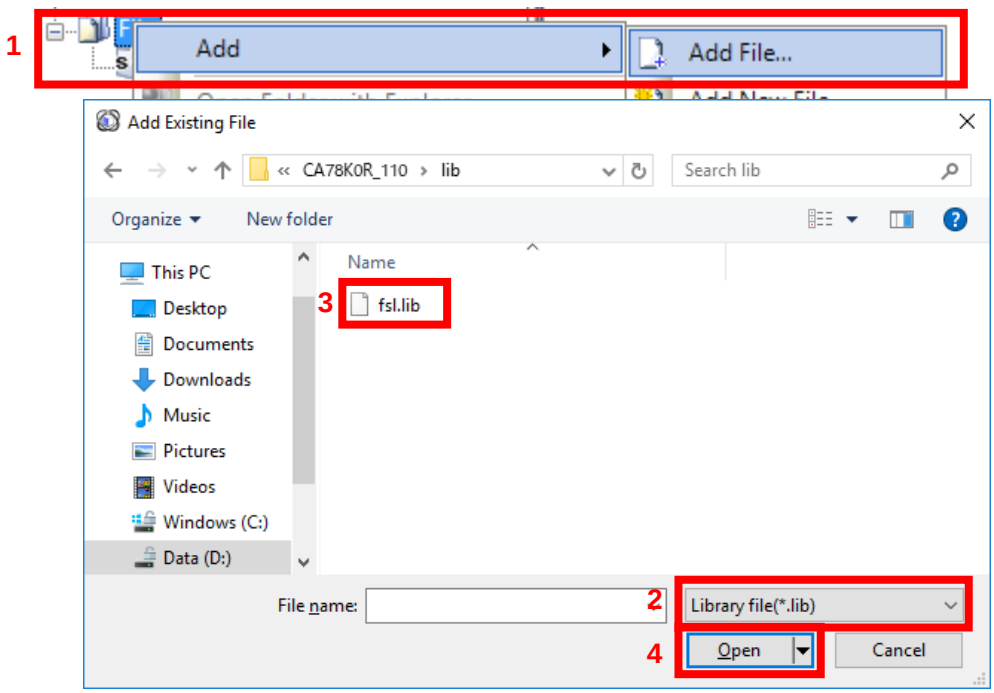


(3) Specifying the library file

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-7).

Next, click the Files of type drop-down list to display a list of the file types. Select Library file (*.lib), and then register the flash self-programming library file (fsl.lib).

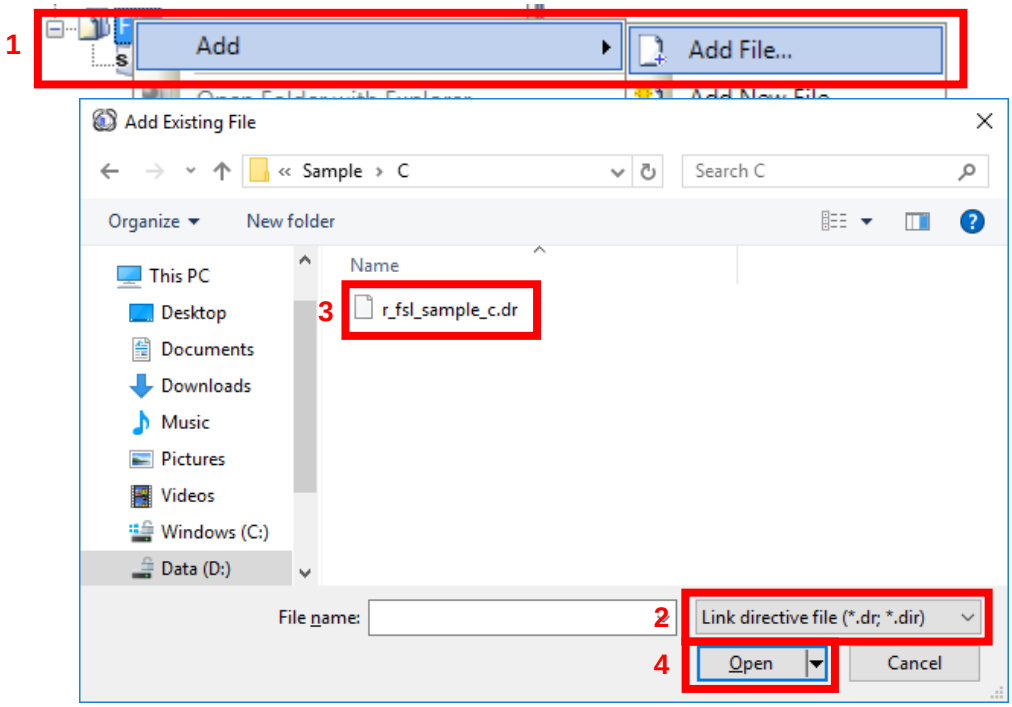
Figure 7-7. Registering the Library File



(4) Specifying the link directive file (only when the CA78K0R compiler is used)

In the CS+ Project Tree window, right-click the File node, click Add, and then click Add File. The Add Existing File dialog box is displayed (as shown in Figure 7-8).
Next, click the Files of type drop-down list to display a list of the file types. Select Link Directive File (*.dr;*.dir), and then register the link directive file that has the same name as the user-created program.

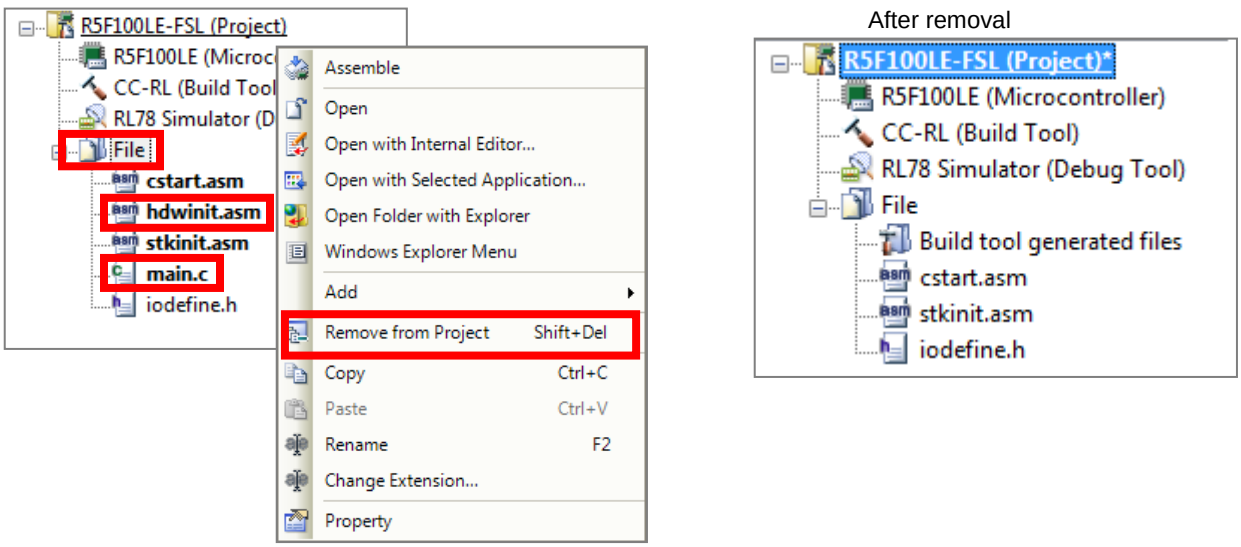
Figure 7-8. Registering the Link Directive File



(5) Removing the automatically generated files (only when the CC-RL compiler is used)

CS+ for the CC-RL compiler automatically generates some files under the File node in the Project Tree window. Among these, the processing of the "main.c" and "hdwinit.asm" files is included in the flash self-programming library. Therefore, remove these two files from the target of the build process.

Figure 7-9. Removing the Automatically Generated Files



(6) Building

On the CS+ Build menu, click Build Project to build the project.

7.3 Notes at Build

7.3.1 When the CA78K0R Compiler is Used

(1) When the on-chip debugging function is in use

After the on-chip debugging function is enabled in the CS+, building a program may generate the following type of error.

```
RA78K0R error E3212: Default segment can't allocate to memory - ignored
                     Segment '??OCDROM' at xxxxxH-200H
```

This error occurs when the segment for the monitor area (OCDROM) used by the on-chip debugging function cannot be allocated. Therefore, to avoid this error, add the following code to the link directive file (*.dr) embedded in the project and prepare a separate area for allocating the segment.

```
MEMORY OCD_ROM : ( 0xxxxxH, 00200H )
```

Remarks: 1. xxxxx indicates the start address of the location where the error occurred.

2. The area name OCD_ROM is an example of the notation.

(2) When the relink function is in use (on the flash area side)

The error shown below may occur when a program is built after a file including a declaration that specifies the section name is registered in the project on the flash area side with the use of the relink function of CS+.

```
CC78K0R error E0842: Unrecognized pragma SECTION '@@xxxxx'
```

This error occurs because the section name on the flash area side differs from the normal case when the relink function is used. To avoid the error, change the specified section name from "@@xxxxx" to "@Exxxxx" as shown below to conform to the rules for the section name of the flash area side.

```
#pragma section @Exxxxx CNST_DAT
```

Remarks: 1. xxxxx indicates the string of the desired section name.

2. The changed section name CNST_DAT is an example of the notation.

7.3.2 When the CC-RL Compiler is Used

(1) When the on-chip debugging function is in use

After the on-chip debugging function is enabled in the CS+, building a program may generate the following type of error.

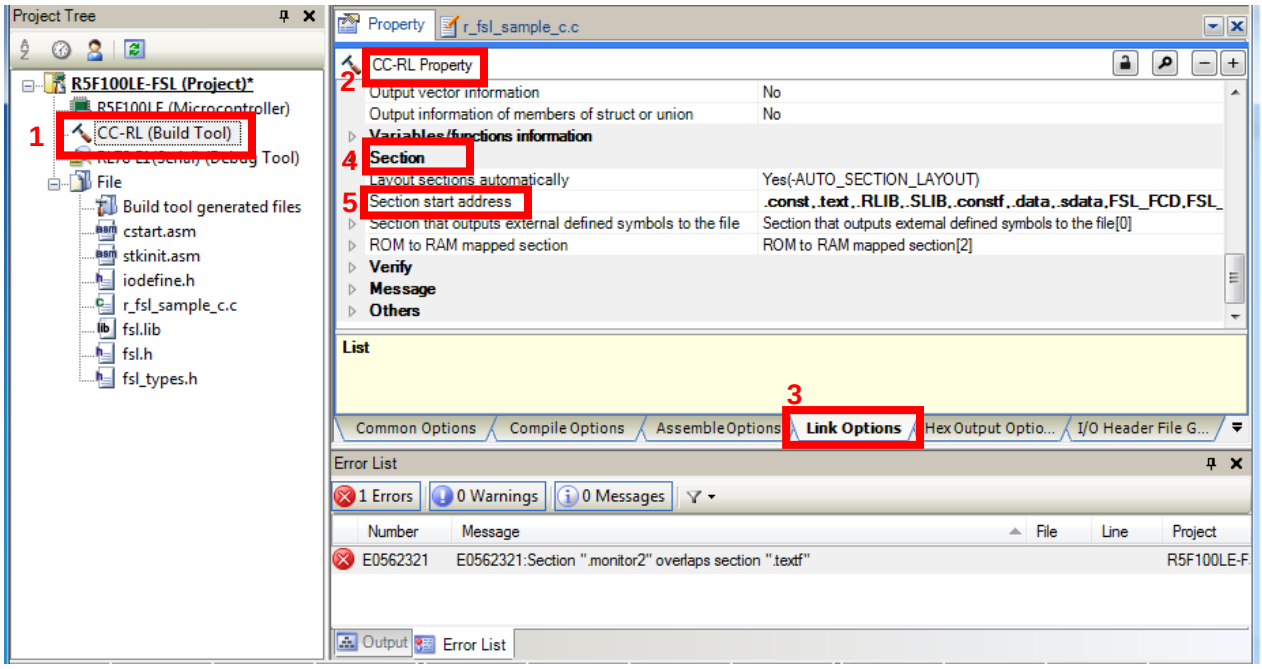
E0562321:Section ".monitor2" overlaps section "xxxxx"

This error occurs when the section for the monitor area (OCDROM) used by the on-chip debugging function cannot be allocated. Therefore, to avoid this error, right click the CC-RL (Build Tool) node (1) in the CS+ Project Tree window, select Property to open the CC-RL Property panel (2), and select the Link Options tab (3). In the Section category (4), modify the setting for Section start address (5) so that no other areas overlap the area where the section for the on-chip debugger monitor is allocated (monitor2: the initial address range is 0xFE00 to 0xFFFF in R5F100LE). (See Figure 7-10.)

For details of the section settings, refer to the CC-RL Compiler User's Manual.

Remark 1. xxxxx: Indicates the section name.

Figure 7-10. Modifying the Section Allocation



Chapter 8 How to Debug a Program

For details on how to perform debugging by using IECUBE or the on-chip debug emulator E1,E2,E2 emulator Lite or E20, see the following document:

Title
CubeSuite+ Integrated Development Environment User's Manual: RL78 Debug[CS+ for CA,CX] Note
CS+ Integrated Development Environment User's Manual: RL78 Debug Tool[CS+ for CC] Note

Note: You can download this document from the "CS+ Integrated Development Environment" page of the Renesas Electronics website.

Chapter 9 Sample Program

The attached sample program (r_fsl_sample_c.c) is provided to enable the usage method of the Flash Self-Programming Library Type01 to be easily confirmed on the QB-R5F100LE-TB boards with R5F100LEA (RL78/G13) as the target microcontrollers. The sample program is just a reference example and the user program does not have to be created to match the sample program. The sample program should be used as a simple program to confirm operation.

The link directive file (r_fsl_sample_c.dr) for the sample program for the CA78K0R compiler has a purpose to specify that a stack or data buffer used by the sample program is not allocated to an area where allocation is prohibited Note1. When using the sample program, this file should also be embedded with the sample program.Note2, 3

The sample program for the CC-RL compiler does not need the link directive file (r_fsl_sample_c.dr), but sections should be allocated appropriately in the Section category on the Link Options tabbed page in the CS+ window so that a stack or data buffer used by the sample program is not allocated to an area where allocation is prohibited Note1.

Notes: 1. For details, refer to chapter 2.2 "Software Environment" in the user's manual.

2. In the supplied link directive file, the RAM area size is set to 512 bytes. Even when the target microcontroller has 2 Kbytes or larger RAM, the sample program (r_fsl_sample_c.c) can be used for building without modifying the defined area setting.
3. The data in usage may be placed at an unintended area depending on how the environment in use or the program is changed. After an execution module is generated, the map file and allocation state of programs or data must be confirmed. For the definition method and allocation conditions of each code or data, refer to the user's manual of the CS+.

9.1 Initial Settings of the Sample Program

The sample program operates with the following initial settings. When these settings need to be changed, modify the sample program.

- CPU operating frequency: High-speed on-chip oscillator 32 MHz
- Voltage mode: High-speed mode

9.2 Settings of Option byte and On-Chip Debugging

When performing on-chip debug, set "Set enable/disable on-chip debug by link option" to "Yes" and specify "84" for "Option byte values for OCD". For the CC-RL compiler, set "Set debug monitor area" to "Yes".

The sample program normally operates by setting the high-speed on-chip oscillator at 32 MHz.

After setting "Set user option byte" to "Yes" on "Link Options" tabbed page, specify "xxxxE8" for "User option byte value" and set the high-speed on-chip oscillator at 32 MHz.

Figure 9-1 (a) Setting of Option byte (CA78K0R Compiler)

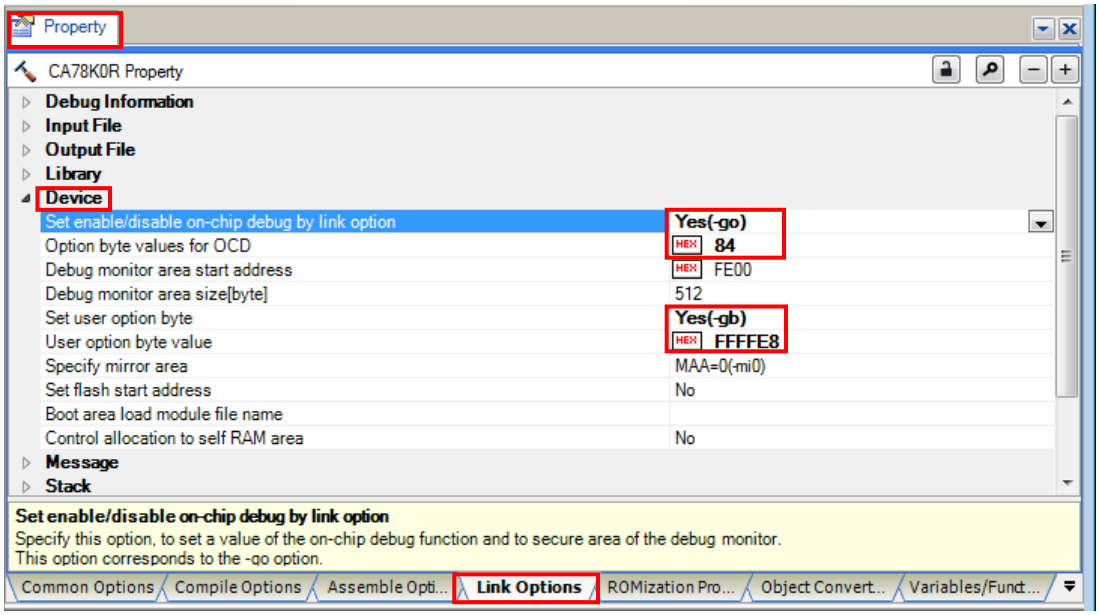
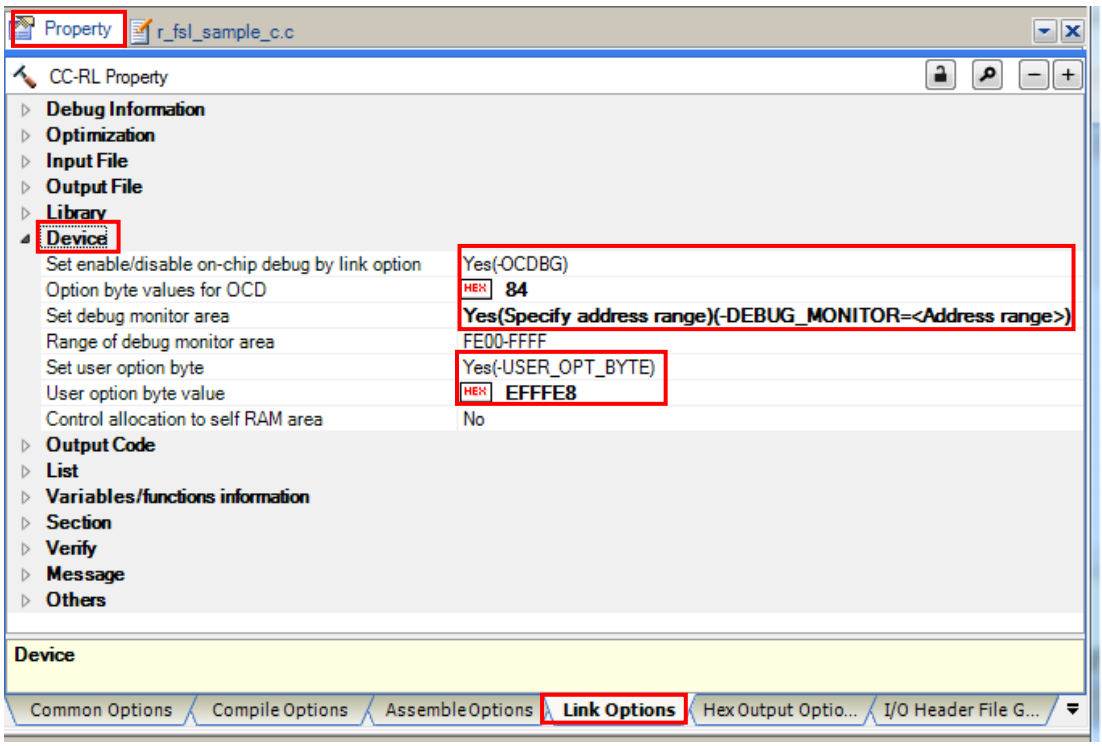


Figure 9-1 (b) Setting of Option byte (CC-RL Compiler)



9.3 Compilation Switch for the C-Language Sample Program

The sample program has a compilation switch as shown below. This compilation switch is used to turn on the LED to confirm operation on the QB-R5F100LE-TB board. To use this, modify "#if 0" to "#if 1" so that the #define declaration for the target CPU board becomes valid.

```
/************************************************************/
/* Sample Program – Program switch symbol */
/************************************************************/
/* Can be enabled when a single QB-R5F100LE-TB board is used */
#if 0
#define __QB_R5F100LE_TB__

/* Other boards */
#else
#define __NON_TARGET__
#endif
```

Can be modified to #if 1 when QB-R5F100LE-TB is used

9.4 Defining the Internal RAM Area

9.4.1 When the CA78K0R Compiler is Used

When the CA78K0R compiler is used, the entire internal RAM area is automatically defined as an area with the name "RAM" in the initial state. Unless otherwise stated in the link directive file, the stack and data buffers are to be allocated to this area Note. However, in this case, the stack and data buffers would be allocated by default to an area (FFE20H to FFEFFH in self-RAM) for which use by the flash self-programming library is prohibited, so the program may not run correctly.

In the attached link directive file for the sample program, as a solution, re-define the area with the name "RAM" so that it does not include the above area, ensuring that stack and so on are not allocated to the area for which usage is prohibited.

```
MEMORY RAM          :(0FF300H, 000B20H)
```

The above statement redefines the area with the name "RAM" to be the B20H bytes area starting from the address FF300H (FF300H to FFE1FH) Note. This prevents attempted use of the area which the flash self-programming library is prohibited to use by excluding the prohibited portion from the area with the name "RAM".

However, if this is the only change setting that is explicitly made, the area from FFE20H to FFEFFH is also unusable for any other purpose. Accordingly, separately add the following definition. No particular restrictions apply to the name of this area.

```
MEMORY SADDR_RAM:(0FFE20H, 0000E0H)
```

If there is a self-RAM area, automatic allocation of variables to this area can be restricted by defining its range as an area with the name "SELFRAM".

```
MEMORY SELFRAM    :(0FEF00H, 000400H)
```

An example of the settings for an RL78/G13 (the product with 4 Kbytes of RAM and 64 Kbytes of ROM) is given below.

```
; ----------------------------------------------------------
; Define new memory entry for Self-RAM
; ----------------------------------------------------------
MEMORY SELFRAM     : ( 0FEF00H, 000400H )   <-- Definition of the self-RAM area

; ----------------------------------------------------------
; Redefined default data segment RAM
; ----------------------------------------------------------
MEMORY RAM         : ( 0FF300H, 000B20H )   <-- Definition of the RAM area to be used normally

; ----------------------------------------------------------
; Define new memory entry for saddr area
; ----------------------------------------------------------
MEMORY RAM_SADDR : ( 0FFE20H, 0000E0H )     <-- Definition of the area from FFE20H to FFEFFH
```

Note: The CA78K0R linker allocates data with a non-specified destination for allocation (segment types DSEG and BSEG) to the internal RAM area according to the re-allocation attribute of the data. Accordingly, specific data may not be allocated to the area with the name "RAM" in some situations.
For details on the methods of defining and allocating the individual categories of data, refer to the user's manual for CS+.
Reference to the map file (*.map) generated at the time of building is required to confirm the state of allocation.

9.4.2 When the CC-RL Compiler is Used

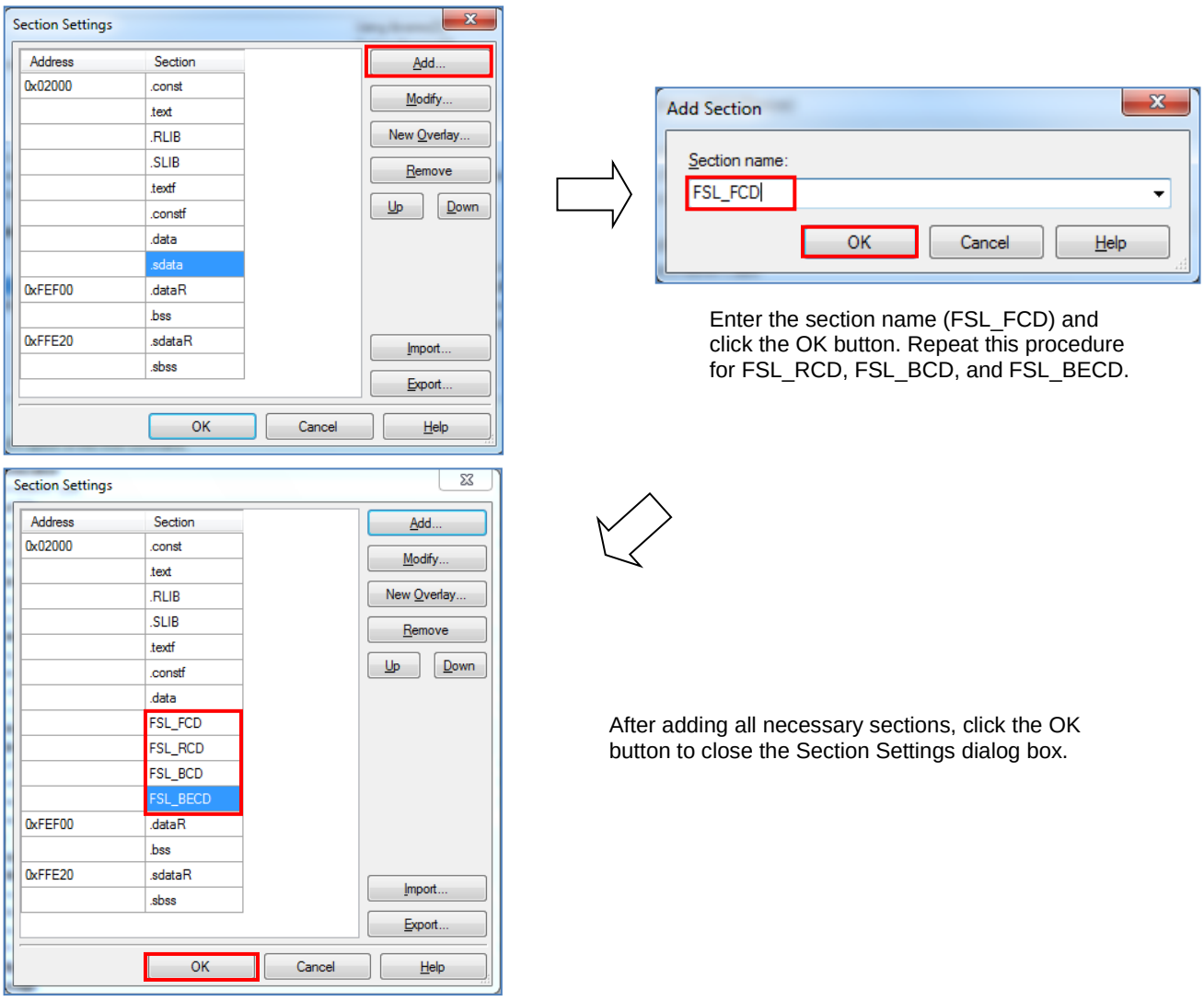
(1) Adding the include path

In CS+ for the CC-RL compiler, no include path is specified in the initial state; the include paths for the header files used by the flash self-programming library need to be added. The flash self-programming library uses header files "fsl.h", "fsl_types.h", and "iodefine.h" (this file is automatically generated by CS+). Add the include path for each header file in the Additional include paths in the Preprocess category on the Compile Options tabbed page.

(2) Defining sections

When CS+ for the CC-RL compiler is used, the sections used for the ROM and RAM areas need to be defined. Sections can be defined in the Section category on the Link Options tabbed page in the CS+ window. When the Layout sections automatically property is set to No, select the Section start address property to open the Section Settings dialog box and add the sections necessary for the flash self-programming library to the ROM area (Figure 9-2). (In this example, the FSL_FCD, FSL_RCD, FSL_BCD, and FSL_BECD sections that are necessary for operation of the sample program are added.)

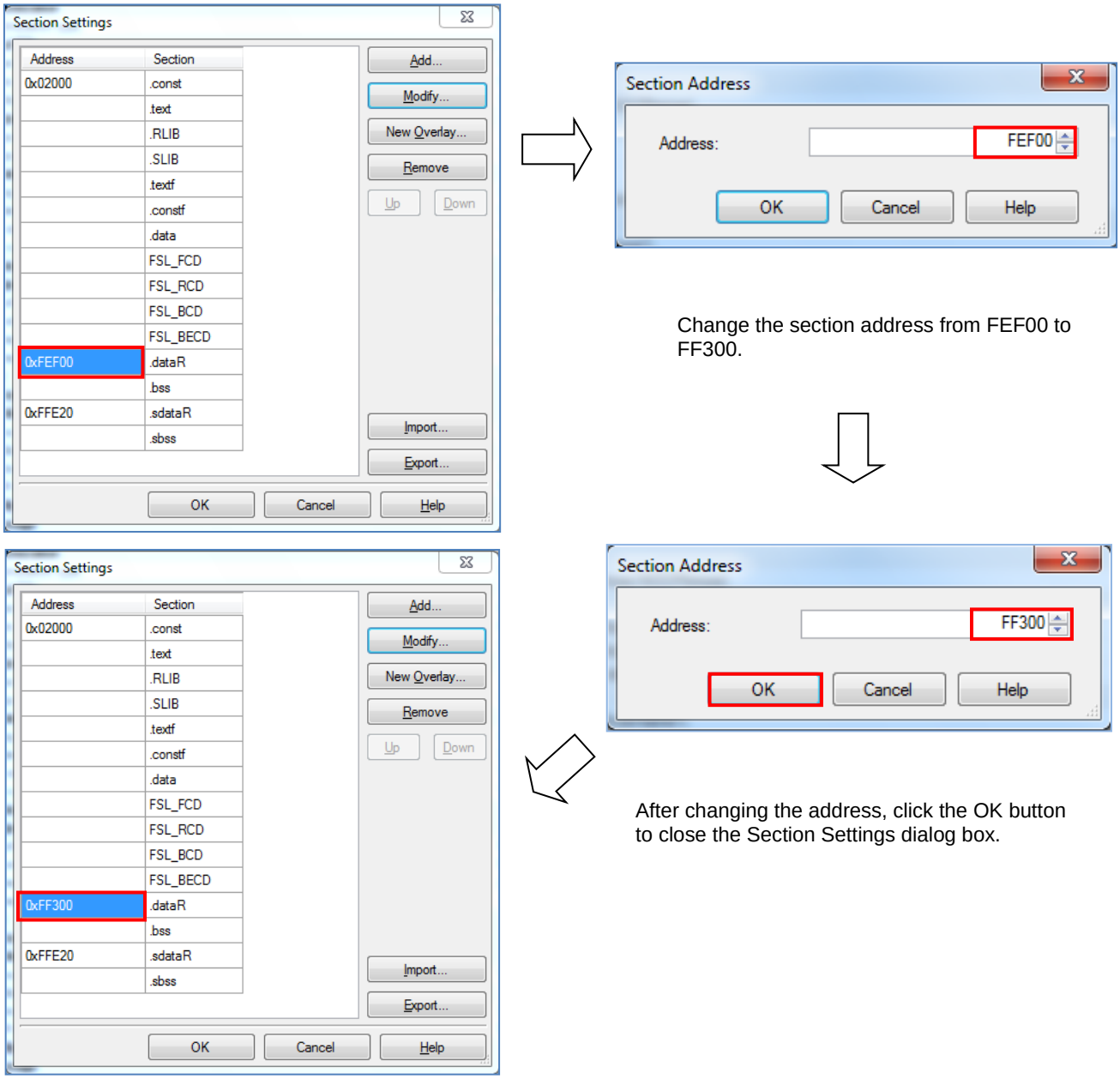
Figure 9-2. Example of Section Settings for the flash self-programming library (ROM Area)



(3) Allocating the Self-RAM Area

In the initial state of the section settings in CS+ for the CC-RL compiler, the user RAM area is allocated at the beginning of the internal RAM area (from address FEF00H for R5F100LEA, which is the target microcontroller of the sample program). However, in R5F100LEA, the flash self-programming library uses the address range from 0xFEF00 to 0xFF2FF as the self-RAM area. Therefore, the user RAM area must be allocated outside this area. In this example, the user data start address 0xFEF00 is changed to 0xFF300.

Figure 9-3. Example of Changing the User RAM Area Allocation (RAM Area)



Note: The sections including the user-specified sections are automatically re-allocated when the Layout sections automatically property is temporarily set to No, the user RAM allocation is changed, and then the property is again set to Yes. In this case, sections may be allocated to areas that are not specified by the user; that is, data may be placed in unintended areas. Be sure to refer to the map file to check if the software resources (especially RAM data) used by the flash self-programming library are placed in relocatable areas.

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.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.2