

[Notes]

R20TS0827EJ0100

Rev.1.00

Apr. 01, 2022

## CS+ Integrated Development Environment (Note on Changing CC-RX Options)

### Outline

When using the CS+ integrated development environment, note the following point.

1. Note on building after changing built tool options (CC-RX family)

## 1. Note on Building after Changing Built Tool Options (CC-RX Family)

### 1.1 Applicable Products

CS+ for CC V8.07.00 and earlier

(The conditions vary depending on the product version. Refer to 1.4 Conditions.)

### 1.2 Applicable Devices

RX family MCUs that support the CS+ for CC

### 1.3 Details

If you change the options that affect operations of the CC-RX build tool's library generator and then build, the library generator may not be re-executed.

As a result, the changes in the options are not applied to the load module created by building.

### 1.4 Conditions

The problem occurs if [Generation mode of the standard library] property in Project tree > [CC-RX (Build Tool)] node > [Library Generate Options] tab > [Mode] category is "Build a library file (option changed)", any of the following is done, and then building is started.

- Using CS+ for CC V8.01.00 or later with CC-RX V3.01.00 or later

Change the setting for [Uses double-precision floating-point operation instructions] property in Project tree > [CC-RX (Build Tool)] node > [Common Options] tab > [CPU] category.

- Using CS+ for CC V2.02.00 or later with CC-RX V2.01.00 or later

Change the setting for [Uses single-precision floating-point operation instructions] property in Project tree > [CC-RX (Build Tool)] node > [Common Options] tab > [CPU] category.

- Using CS+ for CC V3.02.00 or later with CC-RX (any version)

Either of the following.

- Condition 1

Change [Use same optimization-related settings as Compile Options tab] property in Project tree > [CC-RX (Build Tool)] node > [Library Generate Options] tab > [Optimization] category to "Yes", and then change any of the following properties in [Compile Options] tab > [Optimization] category.

- ◇ [Optimization level]
- ◇ [Outputs additional information for inter-module optimization]
- ◇ [Optimization type]
- ◇ [Loop expansion]
- ◇ [Expansion maximum number]
- ◇ [Performs inline expansion automatically]

- ✧ [Maximum increasing rate of function size]
- ✧ [Files for inter-file inline expansion]
- ✧ [Expansion method of the switch statement]
- ✧ [Handles external variables as if they are volatile qualified]
- ✧ [Accesses to volatile qualified variables with the sizes of the variable types]
- ✧ [Performs the constant propagation of const qualified external variables]
- ✧ [Conversion method of the divisions and residues of integer constants]
- ✧ [Execution method of library function that can be expanded to RX instructions]
- ✧ [Divides the optimizing ranges into many sections before compilation]
- ✧ [Schedules the instruction taking into consideration pipeline processing]
- ✧ [Converts floating-point constant division into multiplication]
- ✧ [Allocates preferentially the variables with register storage class specification to registers]
- ✧ [Omits a check of the range for conversion between the floating type and unsigned integer type]
- ✧ [Optimizes modification of the operation order of a floating-point expression]
- Condition 2
  - Change [Use same object-related settings as Compile Options tab] property in Project tree > [CC-RX (Build Tool)] node > [Library Generate Options] tab > [Object] category to "Yes", and then change any of the following properties in [Compile Options] tab > [Object] category.
    - ✧ [Section name of program area]
    - ✧ [Section name of constant area]
    - ✧ [Section name of initialized data area]
    - ✧ [Section name of uninitialized data area]
    - ✧ [Section name of literal area]
    - ✧ [Section name of switch statement branch table area]
    - ✧ [Allocates uninitialized variables to 4-byte boundary alignment sections]
    - ✧ [Allocates initialized variables to 4-byte boundary alignment sections]
    - ✧ [Allocates const qualified variables to 4-byte boundary alignment sections]
    - ✧ [Allocates switch statement branch tables to 4-byte boundary alignment sections]
    - ✧ [Adjustment for instruction in branch]
    - ✧ [Align fetch address of string manipulation instructions]
    - ✧ [Generates divisions and residues with DIV, DIVU, and the FDIV instruction]

## 1.5 Workaround

If any of the above conditions are met, run [Clean Project] in [Build] before building.

## 1.6 Schedule for Fixing the Problem

This problem will be fixed in CS+ for CC V8.08.00. (Scheduled to be released in July 2022.)

**Revision History**

Rev.	Date	Description	
		Page	Summary
1.00	Apr.01.22	-	First edition issued

Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.

The past news contents have been based on information at the time of publication. Now changed or invalid information may be included.

The URLs in the Tool News also may be subject to change or become invalid without prior notice.

**Corporate Headquarters**

TOYOSU FORESIA, 3-2-24 Toyosu,  
Koto-ku, Tokyo 135-0061, Japan  
[www.renesas.com](http://www.renesas.com)

**Trademarks**

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

**Contact Information**

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