C/C++ Compiler Package for RX Family V3.03.00

R20UT4877EJ0101
Rev.1.01
Jan 16,2021

Release Notes

Thank you for using our product.

This document describes the restrictions and points for caution. Read this document before using the product.

Contents

Chapter 1. User’s Manuals

Please read the following user’s manuals along with this document.

Name	Document Number
CC-RX Compiler User's Manual	R20UT3248EJ0110
CS+ Integrated Development Environment User's Manual: CC-RX Build Tool Operation	R20UT3478EJ0108

Chapter 2. **Changes**

This section describes changes to the CC-RX compiler from V3.02.00 to V3.03.00.

2.1 Addition of the -branch_chaining and -nobranch_chaining options

The -branch_chaining option has been added for optimization to reduce the code size of branch instructions.

When this option is specified, a branch instruction may not directly branch to the final destination; but branch to another branch instruction with the same destination by using a smaller branch instruction. This slows down the execution speed, but reduces the code size.

This optimization is suppressed when the -nobranch_chaining option is specified.

2.2 Addition of the -VERBOSE option

The -VERBOSE option has been added to display detailed information at link time.

By specifying crc as a parameter, the results of CRC calculations and the output position addresses are displayed.

2.3 Generation of the MULHI, MACHI, MULLH, and MACLH instructions

Generation of the MULHI, MACH, MULLH, and MACLH instructions is now supported. Specify the following three options to enable this feature.

- -optimize=2 or max
- -speed
- -save_acc

In addition to the options above, also specify the following option to enable generation of the MULLH or MACLH instruction.

- -isa=rxv2 or -isa=rxv3

The following source code shows how this reduces the code size and improves the execution speed.

```
<Example of source code>
signed long mulhi(signed long lhs0, signed long rhs0,
                signed long lhs1, signed long rhs1,
                signed long lhs2, signed long rhs2) {
  lhs0 >>= 16;
  lhs1 >>= 16;
  lhs2 >>= 16;
  rhs0 >>= 16;
  rhs1 >>= 16;
  rhs2 >>= 16;
  return (lhs0 * rhs0 + lhs1 * rhs1 + lhs2 * rhs2);
}


signed long mac(signed long src, signed long lhs, signed long rhs) {
  signed short lhs0 = (signed short) lhs;
  signed short lhs1 = (signed short) (lhs >> 16);
  signed short rhs0 = (signed short) rhs;
  signed short rhs1 = (signed short) (rhs >> 16);
  src += lhs0 * rhs1;
  src += lhs1 * rhs0;
  src += lhs1 * rhs1;
  return (src);
}
```

<Code output by CC-RX V3.02.00 (-isa=rxv2 -speed -save_acc)>	<Code output by CC-RX V3.03.00 (-isa=rxv2 -speed -save_acc)>
_mulhi: .STACK _mulhi = 4 MOV.L 04H[R0], R5 SHAR #10H, R3 MOV.L 08H[R0], R15 SHAR #10H, R4 MULLO R4, R3 SHAR #10H, R1 SHAR #10H, R2 SHAR #10H, R5 MACLO R2, R1 SHAR #10H, R15 MACLO R15, R5 MVFACMI R1 RTS _mac: .STACK _mac = 4 SHAR #10H, R2, R14 MULLO R14, R3 SHAR #10H, R3 MACLO R3, R14 MACLO R3, R2 MVFACMI R14 ADD R14, R1 RTS	_mulhi: .STACK _mulhi = 4 MULHI R4, R3 MOV.L 04H[R0], R5 MOV.L 08H[R0], R3 MACHI R2, R1 MACHI R3, R5 MVFACMI R1 RTS _mac: .STACK _mac = 4 MULLH R3, R2, A0 MACHI R3, R2 MACLH R2, R3, A0 MVFACMI R14 ADD R14, R1 RTS

2.4 Improved precision of alias analysis

The precision of alias analysis has been improved to make it easier to apply optimizations such as moving memory access instructions across intrinsic function calls or aggregate copies.

The following source code shows how this reduces the code size and improves the execution speed.

```
<Example of source code>

unsigned short ShortArray[2];

signed long LongArray[2];

void test(void) {

  ShortArray[0] = 0;

  __xchg(&LongArray[0], &LongArray[1]);

  ShortArray[1] = 0;

}
```

```
< Code output by CC-RX V3.02.00 (-isa=rxv1)>
_test:
    .STACK __test = 4
    MOV.L #_ShortArray, R1
    MOV.W #0000H, [R1]
    MOV.L #_LongArray, R14
    MOV.L [R14], R15
    ADD #04H, R14, R5
    XCHG [R5].L, R15
    MOV.L R15, [R14]
    MOV.W #0000H, 02H[R1]
    RTS
```

```
< Code output by CC-RX V3.03.00 (-isa=rxv1)>
_test:
    .STACK _test = 4
    MOV.L #_ShortArray, R14
    MOV.L #00000000H, [R14]
    MOV.L #_LongArray, R14
    MOV.L [R14], R15
    ADD #04H, R14, R5
    XCHG [R5].L, R15
    MOV.L R15, [R14]
    RTS
```

2.5 Rectified points for caution

The following points for caution no longer apply. For details, refer to Tool News.

- Using rmpab, rmpaw, rmpal or memchr intrinsic functions (No.55)
- Performing the tail call optimization (No.56)
- Using the -ip_optimize option (No.57)
- Using the multidimensional array (No.58)

Chapter 3. Points for Caution

This section describes points for caution regarding the CC-RX compiler.

3.1 W0523041 message [C/C++ compiler]

When the **-int_to_short** option is specified and a file including a C standard header is compiled as C++ or EC++, the compiler may show the W0523041 message. In this case, simply ignore the message because there are no problems.

[NOTE]

In compilation of C++ or EC++, the **-int_to_short** option will be invalid.

Data that are shared between C and C++ (EC++) program must be declared as the long or short type rather than as the int type.

3.2 Using MVTC or POPC instructions [Assembler]

In the assembly language, the program counter (PC) cannot be specified for MVTC or POPC instructions.

3.3 Using the -delete option for linkage [Optimizing linkage editor]

When a function symbol is removed by the **-delete** option, its following function in the source program is not allowed to have a breakpoint at its function name on the editor while debugging. If you intend to set a breakpoint via the [Label] window at the function entrance, set the breakpoint via the [Label] window or at the program code of the function.

3.4 Path names

Absolute paths that include drive letters or relative paths can be used as the path names for specifying input/output files or folders. Each path name should consist of no more than 259 characters.

Chapter 4. Restrictions

This chapter describes restrictions on the CC-RX compiler.

4.1 Reference to command-line options in the CS+ help system

In the CS+ help system, refer to “Compiler” for the RX (the CC-RX environment) rather than “Build” for details of the command-line options for the CC-RX (ccrx) C/C++ compiler, assembler (asrx), optimizing linkage editor (rlink), and library generator (lbgrx). The descriptions under “Build” are very similar to those under “Compiler”, but are for V2.02.00.

4.2 Usage of math.h functions (frexp, ldex, scalbn and remquo) in C++ language (including EC++)

When certain arguments of the **frexp**, **ldexp**, **scalbn**, and **remquo** functions in **math.h** are of the int type, compiling the C++ or EC++ program generates object code that will enter an endless loop.

Conditions:

This problem occurs when both (1) and (2) are satisfied.

(1) The program is in C++ or the **-lang=cpp** option is effective.

(2) **math.h** is included and any of the following functions is called.

(a) frexp(double, long*) with 'int *' type second argument (except when the first argument is float-type and the **-dbl_size=8** option is effective).

(b) ldexp(double, long) with int type second argument (except when the first argument is float-type and the **-dbl_size=8** option is effective).

(c) scalbn(double, long) with int type second argument (except when the first argument is float-type and the **-dbl_size=8** option is effective).

(d) remquo(double, double, long*) with 'int *' type third argument (except when the both the first and second arguments are float-type and the **-dbl_size=8** option is effective).

Examples:

file.cpp:

```
// Example of compiling C++ source that generates an endless loop
#include <math.h>
double d1,d2;
int i;
void func(void)
{
   d2 = frexp(d1, &i);
}
```

Command Line:

```
ccrx -cpu=rx600 -output=src file.cpp
```

file.src: Example of the generated assembly program

```
_func:
   ; ...(Omitted)
    ; Calling substitute function of frexp
   BSR __$frexp__tm__2_f__FZ1ZPi_Q2_21_Real_type__tm__4_Z1Z5_Type
   ; ...(Omitted)

__$frexp__tm__2_f__FZ1ZPi_Q2_21_Real_type__tm__4_Z1Z5_Type:
L11:
   BRA L11 ; Calls itself ==> endless loop
```

Countermeasures:

Select one of the following ways to avoid the problem.

(1) Compile the program with the **-lang**=c or **-lang=c99** option.

(2) Change int and int * into long and long *.

(3) Append the following declarations to each function that is being used.

```
/* For the frexp function */
static inline double frexp(double x, int *y)
{ long v = *y; double d = frexp(x,&v); *y = v; return (d); }
/* For the ldexp function */
static inline double ldexp(double x, int y)
{ long v = y; double d = ldexp(x,v); return (d); }
/* For the scalbn function */
static inline double scalbn(double x, int y)
{ long v = y; double d = scalbn(x,v); return (d); }
/* For the remquo function */
static inline double remquo(double x, double y, int *z)
{ long v = *z; double d = remquo(x,y,&v); *z = v; return (d); }
```

Example of (2):

Change in file.cpp:

```
#include <math.h>
double d1,d2;
int i;
void func(void)
```

```
{
  long x = i; /* Accept as long type temporary */
  d2 = frexp(d1, &x); /* Call with long type argument */
  i = x; /* Set the result for variable 'i' */
}
```

Example of (3):

Change in file.cpp:

```
#include <math.h>
/* Append declaration */
static inline double frexp(double x, int *y)
{ long v = *y; double d = frexp(x,&v); *y = v; return (d); }
double d1,d2;
int i;
void func(void)
{
  d2 = frexp(d1, &i);
}
```

4.3 PIC/PID function (-pic and -pid options)

When a standard library is created by the library generator (lbgrx) with the **-pic** or **-pid** option specified, the following warning may appear once or more.

```
W0591301:"-pic" option ignored (When the -pic option has been specified)
W0591301:"-pid" option ignored (When the -pid option has been specified)
```

Despite the warning, the created standard library has no problems.

4.4 Eliminated options (for the C/C++ compiler)

(a) -file_inline, -file_inline_path

Specifying these options has no effect and the compiler will output a warning. Instead of **-file_inline** or **-file_inline_path**, write **#include** in the source code. In case of C and C99, **-merge_files** can be used instead.

(b) -enable_register

This option is simply ignored and does not affect the generated code.

4.5 C/C++ source-level debugging (for the C/C++ compiler)

(a) Even when **-debug** is specified, you may not be able to set a breakpoint or stop stepped execution on lines that contain a dynamic initialization expression for a global variable (in C++), are the first lines of functions that begin with a loop statement (e.g. **do** or **while**) and do not have an **auto** variable or of functions for which **#pragma inline_asm** has been specified, or contain the control section and body of a loop statement (e.g. **for**, **while**, or **do**) written as a single line.

(b) The values of members of union type and of dummy variables that are to be passed via registers may be displayed incorrectly (e.g. in the [Watch] window).

4.6 Using sections that include address 0xffffffff (in the assembler)

If two or more **.section** directives in the assembly source code contain **.org** directives, the sections have the same name, and the sections overlap at 0xffffffff, the assembler outputs an internal error message (C0554098).

Example)

```
.section SS,ROMDATA
.org 0fffffffeh
.byte 1
.byte 2 ; 0xffffffff
.section SS,ROMDATA
.org 0ffffffffh
.byte 3; ; 0xffffffff
.end
```

4.7 Using -form and -output at the same time (in the linkage editor)

When **-form=rel** and **-output=<filename>** are specified for the linkage editor (**rlink**) at the same time, the filename extension given as **<filename>** is ignored and replaced with **.rel**.

Example)

```
rlink -form=relocate -output=DefaultBuild\lib_test.lib
```

The filename specified for output, **test.lib**, is changed to **test.rel**.

4.8 Using function names that begin with _builtin (for the C/C++ compiler)

Declaration of a function with a name that begins with **_builtin** and for which the definition is in **machine.h** in the **include** directory may lead to an internal error. In general, do not use any names that begin with an underscore (_) in your source code, since such names are reserved.

4.9 -merge_files

Under certain conditions, compilation with **-merge_files** or **-whole_program** specified as the translation unit of code that includes union-type variables will produce error code F0530800 or warning code W0530811.

[Conditions]

If all of the following conditions are satisfied, error code F0530800 or warning code W0530811 will be produced.

(1) **-merge_files** or **-whole_program** is specified.

(2) A union-type external variable having two or more members has been initialized outside any function, and, other than the members that have been initialized, a member has an alignment and size larger than the other member or members.

(3) The variable described in (2) above is declared as **extern** for reference by either of the following.

(3-1) Source files other than the one in which the definition of external variable described in (2) exists.

(3-2) Header files included directly or indirectly by the source files other than the one in which the definition of external variable described in (2) exists.

[Workarounds]

Take any of the following steps.

(1) Specify neither of the options in condition (1).

(2) Initialize the union-type external variable described in condition (2) within a function.

(3) Refer to the variables corresponding to condition (2) only in the source file that includes the definition of the external variable.

4.10 -cfi_ignore_module

When C/C++ source files are compiled with **-output=abs**, the generated object files are not specifiable for **-cfi_ignore_module**.

Only object files generated by using **-output=obj** are specifiable for **-cfi_ignore_module**.

4.11 Using fenv.h when -dpfpu is specified

For the following standard library functions provided by **fenv.h**, even if **-dpfpu** is specified when compilation proceeds, these functions only specify and refer to the relevant values of the FPSW register; and not to the values of the DPSW register.

* feclearexcept
* fegetexceptflag
* feraiseexcept
* fesetexceptflag
* fetestexcept
* fegetround
* fesetround
* fegetenv
* feholdexcept
* fesetenv
* feupdateenv

To specify and refer to the values of the DPSW register, use the **__set_dpsw** and **__get_dpsw** intrinsic functions.

Chapter 5. Standard Libraries

This chapter describes restrictions on standard libraries included in the RX Family C/C++ Compiler.

This compiler package includes four library files (*.lib) for the RX600. You can use any of the library files if they correspond to the options that you wish to specify. Using these files shortens the time required for building.

5.1 Library files

Table 5.1 shows the standard library files and compiler options.

[NOTE]
The compiler options you specify should be the same as the microcontroller options defined for each of the library files listed in Table 5.1. Otherwise these library files are not usable, so specify your compiler options in the library generator to generate your own library file.

Table 5.1 Library Files

Library File	**Purposes**	**Optimize*2 Options**	**Microcontroller Options *1 *2**		
			-endian	**-cpu** **-rtti** **-exception** **-noexception**	**Others *3**
rx600lq.lib	For use with RX600 MCUs Priority in optimization: Speed Little endian	-speed -goptimize	-endian=little	-cpu=rx600 -rtti=on -exception	-round=nearest -denormalize=off -dbl_size=4 -unsigned_char -unsigned_bitfield -bit_order=right -unpack -fint_register=0 -branch=24
rx600ls.lib	For use with RX600 MCUs Priority in optimization: Size Little endian	-size -goptimize			
rx600bq.lib	For use with RX600 MCUs Priority in optimization: Speed Big endian	-speed -goptimize	-endian=big		
rx600bs.lib	For use with RX600 MCUs Priority in optimization: Size Big endian	-size -goptimize			

*Notes:

*1 For details on microcontroller options, see the “Microcontroller Options” columns of the “(1) Compile Options” of section A.1.3, “Options” in the CS+ Integrated Development Environment User’s Manual: RX Build.

*2 The listed option settings produce the same behavior as the default settings.

5.2 Using the library files

Copy the library file(s) included in the package from the "lib" directory into a desired directory.

Then specify one of the copied library files for the **-library** option and start the linkage processing.

Revision History

Rev.	Date	Description	
		Page	Summary
Rev1.00	Dec 01, 2020		First Edition issued
Rev1.01	Jan 16, 2021	3	The description of “-nobranch_chaining option” is added.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

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