Contents

Chapter 1. Target Devices ........................................................................................................................ 2

Chapter 2. Keywords When Uninstalling the Product ................................................................................ 3

Chapter 3. Changes .................................................................................................................................. 4
  3.1 Problems fixed (Changes from V2.04.00 to V2.04.01) ..................................................................... 4
  3.2 Standard and Professional editions ................................................................................................. 4
  3.3 Checking of source code against MISRA-C:2004 rules [Professional] ............................................. 4
  3.4 Checking of source code against MISRA-C:2012 rules [Professional] ............................................ 4
  3.5 Detection of stack smashing [Professional] ...................................................................................... 4
  3.6 Extensions to CRC calculation .......................................................................................................... 5
  3.7 Enhanced support for UTF-8 ............................................................................................................. 6
  3.8 Other improvements .......................................................................................................................... 6

Chapter 4. Points for Caution .................................................................................................................... 7
  4.1 Note on a case of the W0523041 message [C/C++ compiler] .......................................................... 7
  4.2 Note on using MVT or POPC instructions [Assembler] .................................................................. 7
  4.3 Note on the delete option for linkage [Optimizing linkage editor] .................................................. 7

Chapter 5. Restrictions .............................................................................................................................. 8
  5.1 Options for compilers and other subsystems in the help system for the CS+ integrated environment ............................................................................................................................................... 8
  5.2 Restriction on usage of math.h functions (frexp, ldex, scalbn and remquo) in C++ language (including EC++) ........................................................................................................................................ 8
  5.3 Restriction of PIC/PID function (pic and pid options) ........................................................................ 10
  5.4 Eliminated options (for the C/C++ compiler) ................................................................................... 10
  5.5 Notes on C/C++ source-level debugging (for the C/C++ compiler) .................................................. 10
  5.6 Note on using sections that include the address 0xffffffff (in assembler) ....................................... 11
  5.7 Note on using -form and -output at the same time (in the linkage editor) ....................................... 11
  5.8 Note on using function names that begin with _builtin (for the C/C++ compiler) .......................... 11
  5.9 Note on using #pragma interrupt with functions for which save_acc is enabled and that have dummy arguments (for the C/C++ compiler) ....................................................................................... 11

Chapter 6. Standard Libraries Included ................................................................................................... 13
  6.1 Library files ...................................................................................................................................... 13
  6.2 Using the library files ........................................................................................................................ 14
Chapter 1. Target Devices

The target devices supported by the CC-RX are listed on the Website. Please see the URL below.
CS+ Product Page:

http://www.renesas.com/cs+
Chapter 2. Keywords When Uninstalling the Product

There are two ways to uninstall this product.

- Use the integrated uninstaller (uninstalls all CS+ components)
- Use the Windows uninstaller (only uninstalls this product)

To use the Windows uninstaller, select the following from the Control Panel:

- Add/Remove Programs (Windows XP)
- Programs and Features (Windows Vista, Windows 7)

Then select "CS+ CC-RX V2.04.01".
Chapter 3. Changes

This section describes changes on CC-RX from V2.03.00 to V2.04.01.

3.1 Problems fixed (Changes from V2.04.00 to V2.04.01)

The following problems, of which we informed you in RENESAS TOOL NEWS Document No. 151106/tn1, have been fixed:

"-stack_protector" and "-stack_protector_all" options (RXC#037)

http://tool-support.renesas.com/eng/toolnews/151106/tn1.htm

3.2 Standard and Professional editions

The CC-RX compiler has the following two editions.

- Standard edition
- Professional edition

The features of the latter can only be used if the compiler is registered under the professional license. They are indicated as [Professional] from here on.

3.3 Checking of source code against MISRA-C:2004 rules [Professional]

The following options can only be used if the compiler is registered under the professional license.

- -misra2004
- -ignore_files_misra
- -check_language_extension

3.4 Checking of source code against MISRA-C:2012 rules [Professional]

A -misra2012 option, which selects the checking of source code against the MISRA-C:2012 rules, has been added. An –ignore_files_misra option for the selection of files that are not to be checked, and a –check_language_extension option which enables the source-code checking, which are partially suppressed by language extensions, are specifiable at the same time as each other and -misra2012.

Note that the -misra2012 and -misra2004 options cannot be specified at the same time.

Specifying these options to statically check source code makes improving the quality of user programs in early stages possible.

3.5 Detection of stack smashing [Professional]

A feature for the detection of stack smashing has been added. This feature can be realized by the -stack_protector/-stack_protector_all compiler options, or the extended language specification #pragma stack_protector/#pragma no_stack_protector.
This feature generates the codes which detect stack smashing at the entry and end points of functions. Specifically, instructions to execute the following three processes are produced.

1. 4-byte immediately before (upper direction) the local variable area of the stack frame are allocated at the entrance to the function, and the values specified as arguments are stored there.

2. A check is run on whether the value which was stored in step 1 has not been changed at the end of the function.

3. The __stack_chk_fail function is called when the value has been changed, which would lead to smashing the stack.

The __stack_chk_fail function is defined by user and describes the processing to be executed when stack smashing is detected. For example, the __stack_chk_fail function will be executed when the following code, which smashes the stack area, is executed.

```c
void f1() {
    volatile char str[10];
    int i;
    for (i = 0; i <= 10; i++){
        str[i] = i;  // The stack is smashed when i=10
    }
    ...
}
```

Figure 3.1: Example of code that will lead to the __stack_chk_fail function being called

3.6 Extensions to CRC calculation

The -CRC option of the optimizing linkage editor has been extended as follows.

- The range for calculation can be specified by a section name.
- Various methods of calculation have been added. The methods added in this version are listed in table 3.1.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-CCITT-MBS</td>
<td>The result of calculation is obtained by applying CRC-16-CCITT to the input MSB first.</td>
</tr>
<tr>
<td>16-CCITT-MSB-LITTLE-4</td>
<td>The input is a 4-byte unit with little endian. The result of calculation is obtained by applying CRC-16-CCITT to the input MSB first.</td>
</tr>
<tr>
<td>16-CCITT-MSB-LITTLE-2</td>
<td>The input is a 2-byte unit with little endian. The result of calculation is obtained by applying CRC-16-CCITT to the input MSB first.</td>
</tr>
<tr>
<td>16-CCITT-LSB</td>
<td>The result of calculation is obtained by applying CRC-16-CCITT to the input LSB first.</td>
</tr>
</tbody>
</table>
### SENT-MSB
The input is 1 byte, with the higher- and lower-order 4 bit units in little endian. The result of calculation on the input MSB first is obtained with SENT compliance.

### 32-ETHERNET
The result of calculation is obtained by applying CRC-32-ETHERNET to the input. The initial value of the result is 0xFFFFFFFF, and is XOR inverted and the bit order is reversed.

#### 3.7 Enhanced support for UTF-8
The following options have been extended or added.
- Extended the `-utf8` option and `-outcode=utf8` option.
  Those options are now always selectable, not only when `-lang=c99` option is also selected.
- Added a new assembler option `--utf8`.
  String literals and comments in assembly source are handled as encoded in UTF-8.

#### 3.8 Other improvements
Other improvements are described below.

(a) Improved debugging information
   Unnecessary debugging information was deleted to reduce the sizes of object files (*.obj) and load module files (*.abs).

(b) Fixed incorrect display when stepping
   When stepping over an if statement, and it has a condition expression including any of the operators below, the debugger displayed an incorrect result. The problem has been fixed.
   ```
   &&  ||  !  ?:
   ```

(c) Internal errors
   Some of internal errors has been fixed.

(d) Optimization
   The performance and the size of generated code have been improved.

(e) Compiling time
   Compiling some programs with optimization took excessive time. This problem has been relieved by shortening the times.
Chapter 4. Points for Caution

This section describes points for caution regarding CC-RX.

4.1 Note on a case of the 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.

4.2 Note on using MVTC or POPC instructions [Assembler]

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

4.3 Note on 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 in your debugging. If you would like 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.
Chapter 5. Restrictions

This chapter describes restrictions on CC-RX.

5.1 Options for compilers and other subsystems in the help system for the CS+ integrated environment

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, the assembler (asrx), the optimizing linkage editor (rlink) and the library generator (lbgrx). The descriptions under “Build” are very similar to those under “Compiler”, but are for V2.02.00.

5.2 Restriction on usage of math.h functions (frexp, ldexp, scalbn and remquo) in C++ language (including EC++)

An object is generated which will be an infinit-loop at execution when the actual argument of some function (frexp, ldexp, scalbn or remquo) of math.h is int-type, at compiling C++ or EC++ program.

Conditions:
This problem occurs when both (1) and (2) are satisfied.
(1) This 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 infinity-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 ==> infinity-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 or int * into long or 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);
}

5.3 Restriction of 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.

5.4 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.

(b) -enable_register

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

5.5 Notes on C/C++ source-level debugging (for the C/C++ compiler)
(1) 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.

(2) 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).

5.6 Note on using sections that include the address 0xffffffff (in 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 0xffffffffeh
.byte 1
.byte 2 ; 0xffffffff
.section SS,ROMDATA
.org 0xffffffffh
.byte 3; ; 0xffffffff
.end

5.7 Note on 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.

5.8 Note on 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.

5.9 Note on using #pragma interrupt with functions for which save_acc is
enabled and that have dummy arguments (for the C/C++ compiler)

When `#pragma interrupt` is specified for a function and the `save_acc` flag is enabled (including where this is done by using the `-save_acc` compiler option), the compiler may not output code that reflects the correct values of dummy arguments which are passed via R4. Note: In general, we do not recommend defining arguments for functions with the `#pragma interrupt` specification.
Chapter 6. Standard Libraries Included

This chapter describes restrictions on standard libraries included in 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.

6.1 Library files

Table 6.1 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 6.1. Otherwise these library files are not usable, so specify your compiler options in the library generator to generate your own library file.

<table>
<thead>
<tr>
<th>Library File</th>
<th>Purposes</th>
<th>Optimize Options</th>
<th>Microcontroller Options</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx600lq.lib</td>
<td>For the RX600</td>
<td>-speed</td>
<td>-cpu</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Optimization type:Speed</td>
<td>-goptimize</td>
<td>-endian</td>
<td>-round=nearest</td>
</tr>
<tr>
<td></td>
<td>Little endian</td>
<td></td>
<td></td>
<td>-denormalize=off</td>
</tr>
<tr>
<td>rx600ls.lib</td>
<td>For the RX600</td>
<td>-size</td>
<td>-cpu</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Optimization type:Size</td>
<td>-goptimize</td>
<td>-endian=big</td>
<td>-dbl_size=4</td>
</tr>
<tr>
<td></td>
<td>Little endian</td>
<td></td>
<td></td>
<td>-unsigned_char</td>
</tr>
<tr>
<td>rx600bq.lib</td>
<td>For the RX600</td>
<td>-speed</td>
<td>-rtti</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Optimization type: Speed</td>
<td>-goptimize</td>
<td></td>
<td>-unsigned_bitfield</td>
</tr>
<tr>
<td></td>
<td>Big endian</td>
<td></td>
<td></td>
<td>-bit_order=right</td>
</tr>
<tr>
<td>rx600bs.lib</td>
<td>For the RX600</td>
<td>-size</td>
<td>-exception</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Optimization type: Size</td>
<td>-goptimize</td>
<td></td>
<td>-unpack</td>
</tr>
<tr>
<td></td>
<td>Big endian</td>
<td></td>
<td></td>
<td>-fint_register=0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-branch=24</td>
</tr>
</tbody>
</table>

Table 6.1 Library Files

*Notes:

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

*2 These option selections are same from the each default of them.
6.2 Using the library files
Copy the library file(s) included in the package from "lib" directory into a desired directory.
Then specify one of the copied library files for the Library option and start the linkage processing.

All trademarks and registered trademarks are the property of their respective owners.
SALES OFFICES

Renesas Electronics Corporation

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

Renesas Electronics America Inc.
2901 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6500, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8369 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dulles Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5PN, U.K.
Tel: +44-1628-585-100, Fax: +44-1628-585-900

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

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.97 Zhongshan, Haidian District, Beijing 100191, P.R.China
Tel: +86-10-6335-1155, Fax: +86-10-6335-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Lianyuan Road, Putuo District, Shanghai, P. R. China 200033
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1801-1811, 16/F, Tower 2, Grand Century Place, 193 Prince Edward Road West, Mong Kok, Kowloon, Hong Kong
Tel: +852-2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886-2-8175-9530

Renesas Electronics Singapore Pte. Ltd.
80 Bishan Road, Unit 406-02 Hubtex 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. 7203, 105 Field Road, HAL Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Taehapuro, Dorang-dong, Seul, 135-080, Korea
Tel: +82-2-556-3731, Fax: +82-2-556-5141

© 2015 Renesas Electronics Corporation. All rights reserved.
Colophon 5.0