Compiler Package

Application Note: <STL V.1.00.00>

User's Manual

R20UT0074EJ0100
Rev.1.00
Jul. 30, 2010

This document explains the usage of STL for the Renesas C/C++ compiler (hereafter referred to as 'STL') V.1.00.00.

Contents

1. Introduction

This manual explains the procedures for adding functions of STL V.1.00.00 to the following C/C++ compiler manufactured by Renesas Electronics Corporation (hereafter referred to as 'Renesas Electronics') and the limitations on usage due to the compiler specifications.

- RX family C/C++ compiler V.1.00.00 or a later version (hereafter referred to as 'RXC')
- SuperH family C/C++ compiler V.9.04.00 or a later version (hereafter referred to as 'SHC')

1.1 Main Basis of the STL

The following version of the open source STLport provides the main basis for this STL.

- STLport-5.2.1

Usage of this product is based on the license policy for STLport, as described at URL [http://www.stlport.org].

1.2 Unsupported Functions

Of the functions indicated for the product in section 1.1, Main Basis of the STL, this STL does not support those in Table 1.

For the list of supported functions, refer to the application note, List of Functions Supported by the STL.

Table 1 Unsupported Functions

Unsupported Functions
fstream, iomanip, ios, iosfwd, iostream, istream, ostream, sstream, streambuf, strstream, hash_map, hash_set, pthread_alloc, rope, slist, type_traits, unordered_map, unordered_set, clocale, csignal, ctime, ciso646[*1], cwchar[*1], cwctype[*1], stream input/output of complex classes[*2], stream input/output of string classes[*3], and stream input/output of wstring classes[*4]

Notes: 1. These functions are only supported by RXC.
2. The char type is excepted. For details, refer to section 3.1, Limitations on Using Class complex.
3. The char type is excepted. For details, refer to section 6.3, Limitations on Using Stream Input/Output Handling.
4. For details, refer to section 6.3, Limitations on Using Stream Input/Output Handling.

1.3 Disclaimer

Renesas Electronics does not customer support this STL. Renesas Electronics assumes no liability for the use of this STL. You should fully test the product installing STL as your own responsibility.

2. Installing STL to Your Project

This section explains the procedure for installing STL functions to be used in RXC or SHC.

2.1 Location of the STL

Place the directory [stlport], which is included in this STL, anywhere you wish.

When you install the compiler package that includes this STL, the directories will be expanded as shown in figure 1. The following assumes that the directory of active High-performance Embedded Workshop (hereafter referred to as 'HEW') is 'C:\Program Files\Renesas\Hew', in which case directory [stlport] is expanded as below.

<stlport directory> = C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\stlport

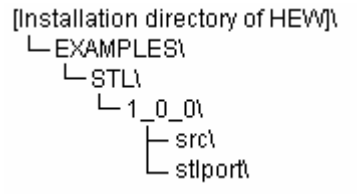

Figure 1 Placement of the STL when Installed as Part of a Compiler Package

2.2 Setting up the Environment

When the STL is used with RXC or SHC, the <stlport directory> as allocated in section 2.1 must be added to the directories of include files for use in building. There are two ways to add the directory: one is by using the [Option dialog box] for the toolchain of the HEW and the other is by using the command line. Each of the methods is explained in the following subsections.

2.2.1 Setting the Directory from the HEW (RXC or SHC)

This subsection explains how to specify a directory containing include files from the [Options] dialog box for the toolchain of the HEW. Follow the procedure below. Figures 2 to 5 show examples of the displays for RXC.

(1) Select the following item from the [Build] menu.
For RXC: [RX Standard Toolchain…] (figure 2)
(2) Open [Include file directories] from [Show entries for:] on the [C/C++] tabbed page in the dialog box (figure 3).
(3) Click on the [Add…] button.
(4) Select [HEW installation directory] as [Relative to:].
(5) Enter 'EXAMPLES\STL\1_0_0\stlport' in [Sub-Directory:] and click on the [OK] button (figure 4).
(6) Open the [Option dialog box] and check that '$(HEWDIR)\EXAMPLES\STL\1_0_0\stlport' has been added.
(7) Click on the [OK] button (figure 5).

The STL will now be available.

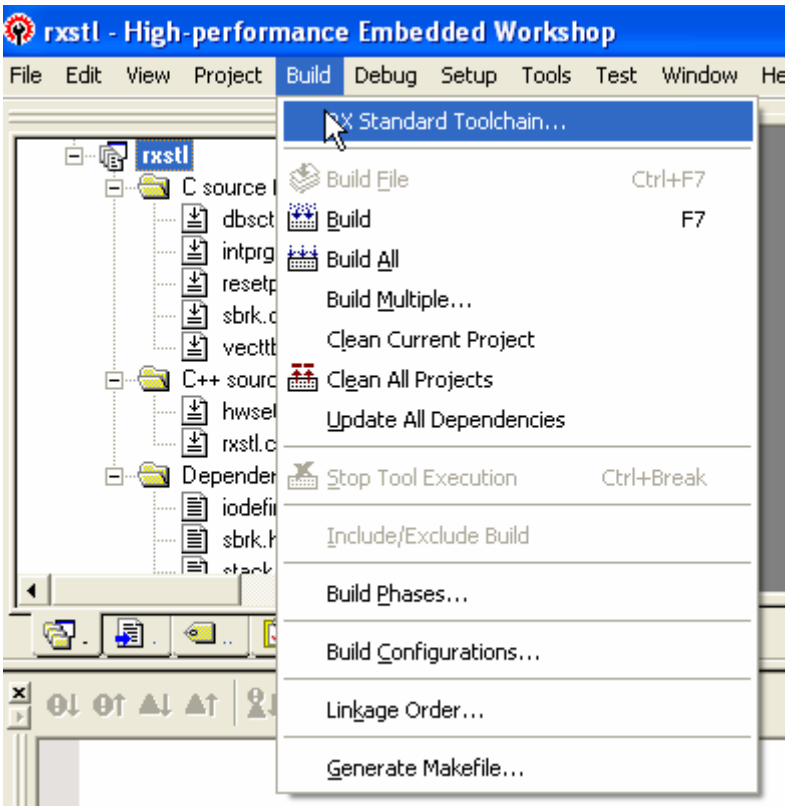

Figure 2 Position of [RX Standard Toolchain…]

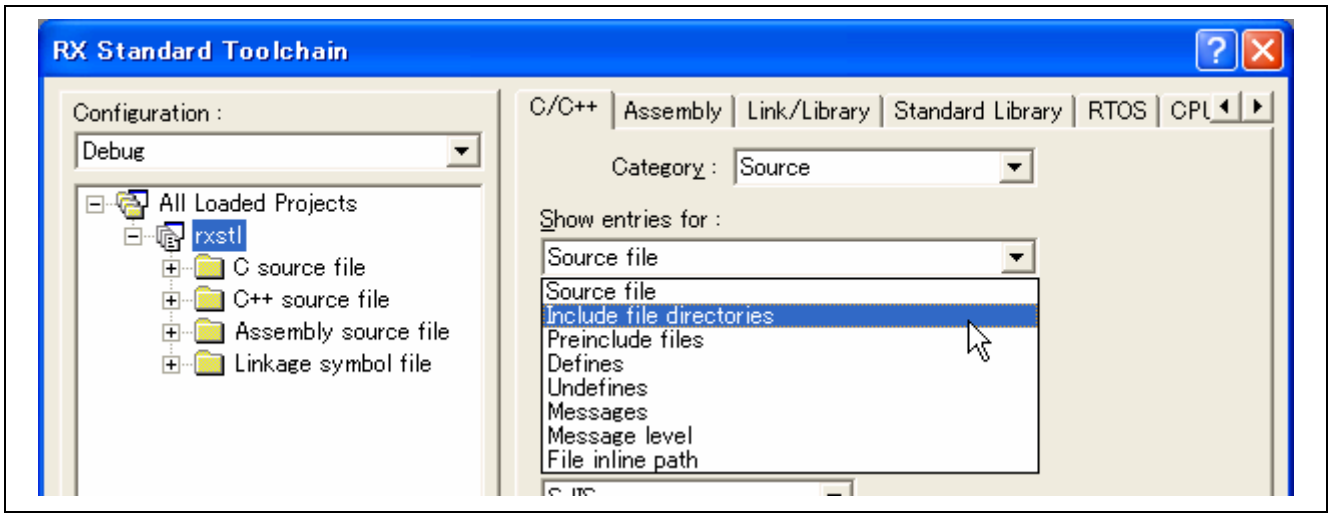

Figure 3 Position for Setting a Directory of Include Files

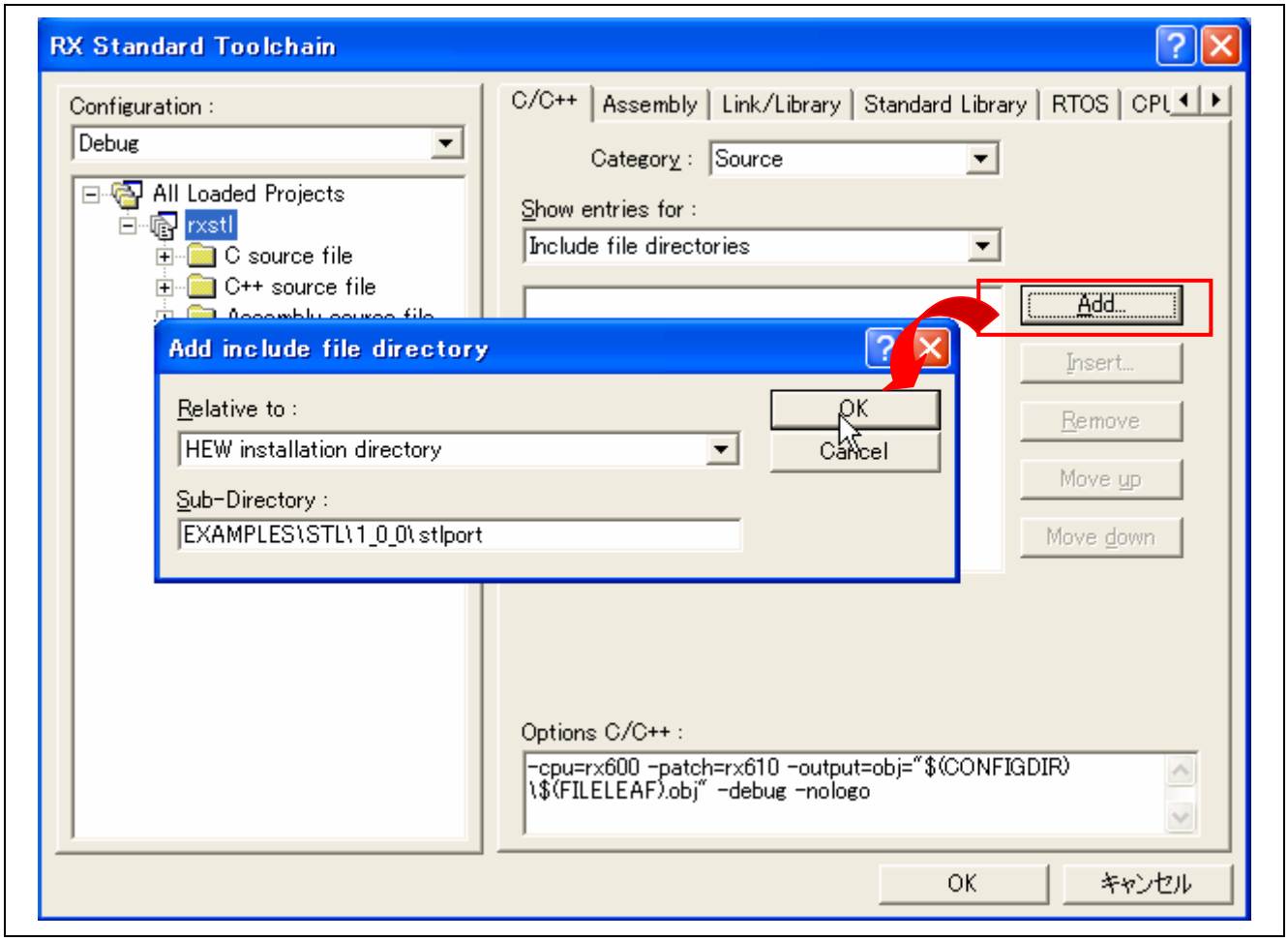

Figure 4 Example of Specifying the Directory

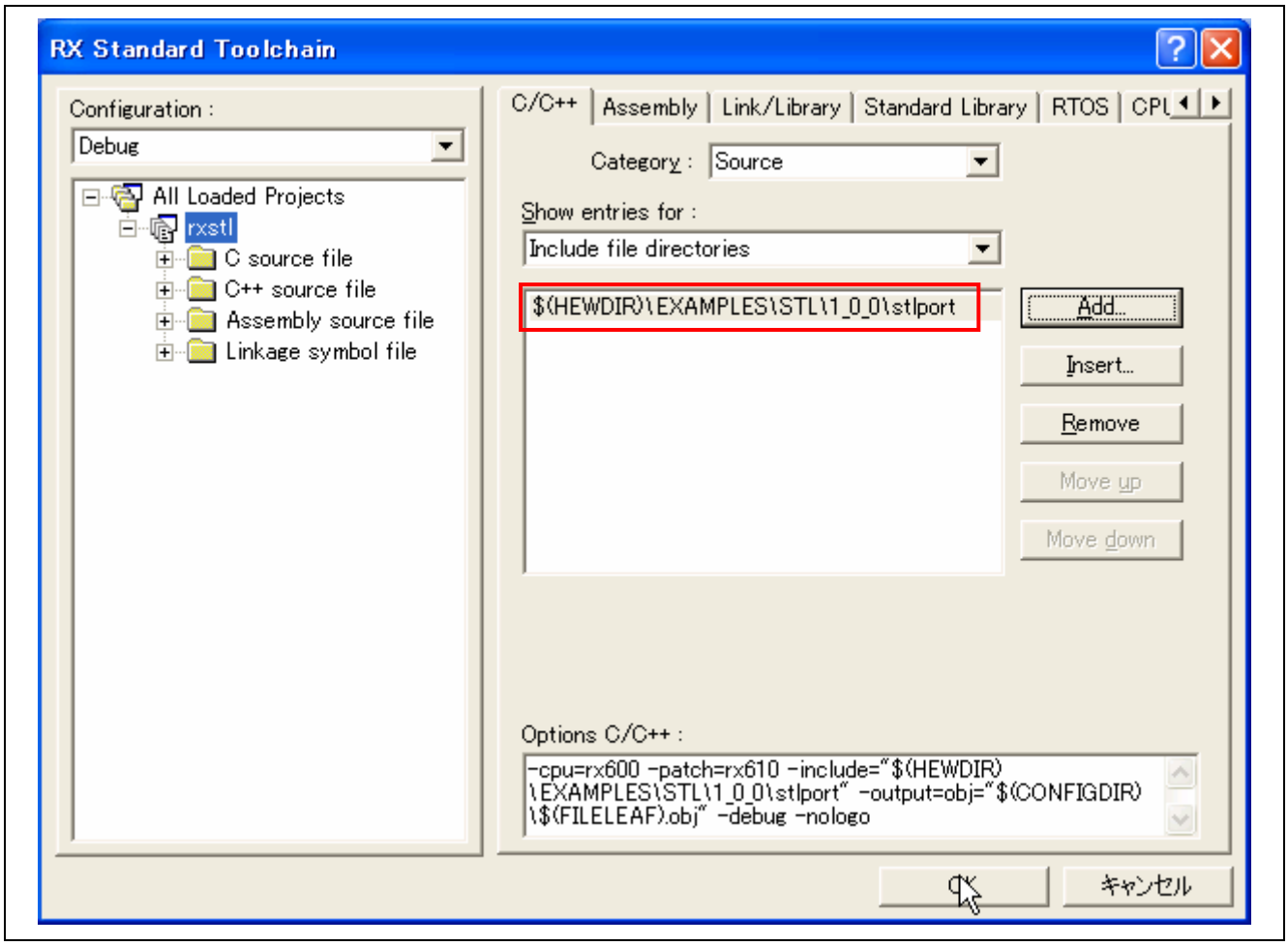

Figure 5 stlport Directory Added

2.2.2 Setting the Directory from the Command Line (RXC)

This subsection explains the procedure for adding a directory of include files by setting environment variables in RXC.

Specifically, the INC_RX environment variable is used to set <stlport directory> as a standard include directory by writing <stlport directory> to the left of the names of existing include directories.

The standard include directories are specified in sequence and separated by semicolons, as in INC_RX = <stlport directory>;<standard include directory>.

A standard example is shown below.

```
INC_RX = C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\stlport
         ;C:\Program Files\Renesas\Hew\Tools\Renesas\RX\1_0_0\Include
```

If you do not want to change the environment variable, the directory can be specified by the include option of the compiler.

Specify the directory by using the include option of the compiler as follows.

-include = <stlport directory>

As a concrete example, this might be part of the following command.

> ccrx tp.cpp –cpu = rx600 –include = "C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\stlport"

The STL will now be available.

2.2.3 Setting the Directory from the Command Line (SHC)

This subsection explains the procedure for adding a directory of include files by setting environment variables in SHC.

Specifically, the SHC_INC or SHC_LIB environment variable is used to set <stlport directory> as a standard include directory by writing <stlport directory> to the left of the names of existing include directories.

The standard include directories are specified in sequence and separated by semicolons, as in SHC_INC = <stlport directory>;<standard include directory>.

A standard example is shown below.

SHC_INC = C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\stlport
;C:\Program Files\Renesas\Hew\Tools\Renesas\SH\9_4_0\Include

If you do not want to change the environment variable, the directory can be specified by the include option of the compiler.

Specify the directory by using the include option of the compiler as follows.

-include = <stlport directory>

As a concrete example, this might be part of the following command.

> shc tp.cpp –cpu = sh2a –include = "C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\stlport"

The STL will now be available.

3. Procedure for Using Class complex

In RXC and SHC, the complex class is implemented in the EC++ library. However, the float_complex and double_complex classes are implemented as separate classes in that library, which differs from the standard template for the complex class of C++. Installing the STL makes the standard template class available. This section explains the usage and limitations on the complex class of the STL.

3.1 Limitations on Using Class complex

The following describes limitations on using the complex class in RXC or SHC.

(a) The complex class has a single type argument for the template, and this is used to specify the precision of the real and imaginary parts. Only the following three types are specifiable.

- complex<float>
- complex<double>
- complex<long double>

If a type other than those above is specified, the operation is unregulated in the language specification.

(b) In RXC or SHC, the Namespace for the template of the complex class and the mathematical functions is std, although that for the functions in the EC++ library is global.

```
// To declare a complex class, std:: is required.
std::complex<float> f1(4.0f, 3.0f);
//std:: is required for abs (complex), but must not be added to count or endl.
count << std::abs(f1) << endl;
// std:: must not be added for the standard library of RXC or SHC.
// Accordingly, std:: must not be added to abs (long).
count << abs(-10L) << endl;
```

(c) Stream input/output is only available for the char type.
The stream input/output library for RXC or SHC is used for stream input/output. This library is only usable for the char type. Stream input/output of wide characters (wchar_t) is not supported.

3.2 Including Source Files

In some of the operations for complex classes in this STL, the mathematical-function and stream-input/output libraries for RXC or SHC are used. As well as the mathematical libraries for RXC or SHC, other source files are required.

Table 2 Libraries and Additional Source Files for Class complex

Functions for Use	EC++ Library	Additional Source File
Arithmetic operations and comparison	None	None
Acquisition of real part and imaginary part	None	None
Mathematical functions	• Mathematical (C99) • Mathematical (C99) (float-type functions)	• complex.cpp • complex_trig.cpp
Stream input/output	• Stream input/output class	• complex_io.cpp

The additional source files are stored in the following directory.

C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\src

3.2.1 Arithmetic Operation and Comparison of complex

When complex classes are only used in arithmetic operations and comparison, no additional source files are required.

```
std::complex<float> f1(2.0f, 4.0f), f2(1.0f, 0.0f);
f += f2;
```

3.2.2 Acquisition of Real Part and Imaginary Part of a complex Class

When complex classes are only used to acquire their real and imaginary parts, no additional source files are required.

```
f1.real();
f1.imag();
```

As shown above, member functions real() and imag() are available for the complex class.

3.2.3 Passing complex Classes to Mathematical Functions

When complex classes are passed to mathematical functions for arithmetic operations, libraries for RXC or SHC and additional source files must be included.

- Libraries for RXC or SHC
  As shown in figure 6, tick on the checkboxes for ‘math.h’ and ‘mathf.h’. For RXC, the library structure must be ‘C99’.
  *If ‘C99’ is not set in RXC, an error ‘L2310 (E) Undefined external symbol “_hypot”’ will occur.

- Additional source files
  Add ‘complex.cpp’ and ‘complex_trig.cpp’ to the project in the way shown in figures 7, 8, and 9.

These settings make the mathematical functions available.

However, since the mathematical functions which handle the complex class belong within the std Namespace, a scope-resolution operator is required as shown below.

```
std::pow (f1, 5);
```

3.2.4 Passing complex Classes to Stream Input/Output

When complex classes are passed to stream input/output for the input or output of values, the libraries for RXC or SHC and additional source files must be included.

- Libraries for RXC or SHC
  As shown in figure 6, tick on the checkbox for 'ios(EC++)'.
- Additional source files
  As shown in figures 7, 8, and 9, add 'complex_io.cpp' to a project.

Those settings make stream input/output available.

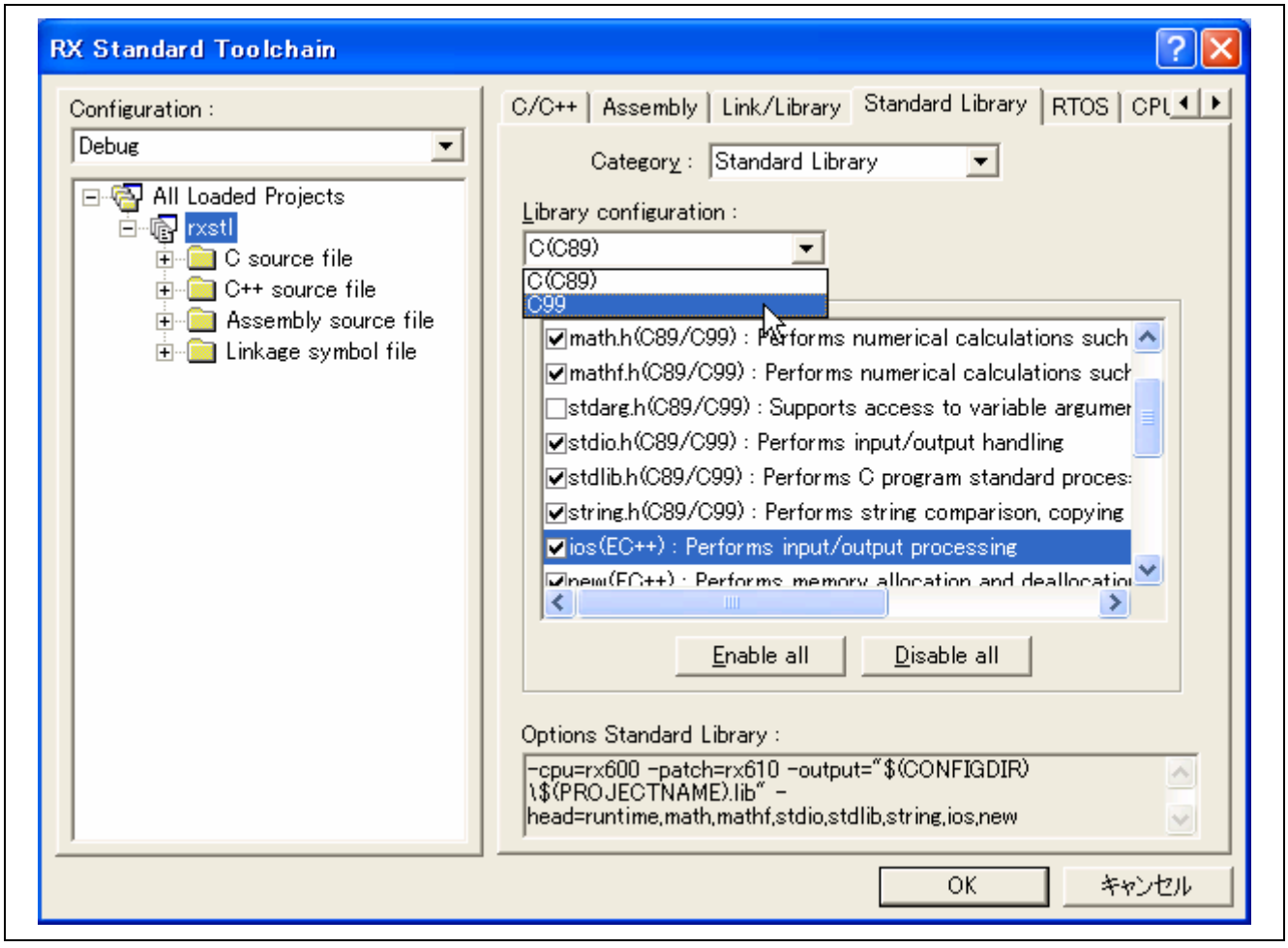

Figure 6 Libraries for RXC or SHC

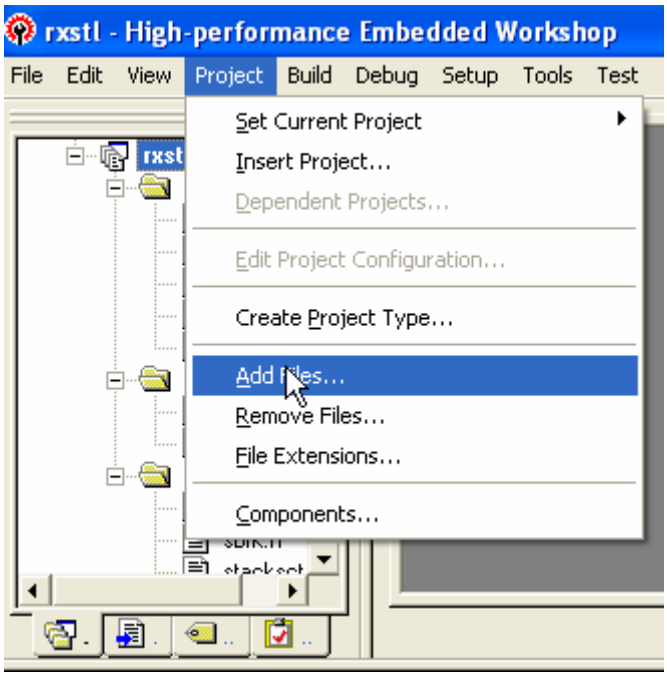

Figure 7 Adding a File to a Project

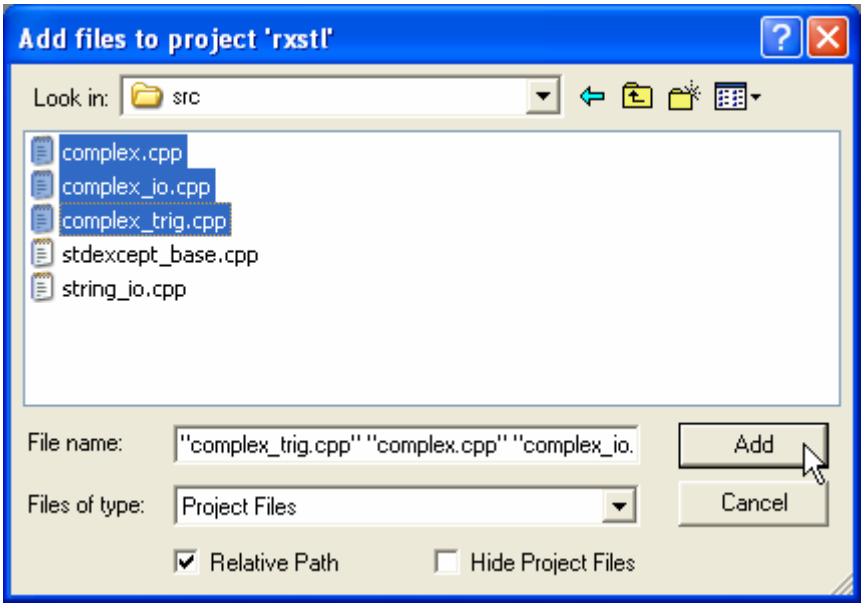

Figure 8 Selecting the Additional Source Files

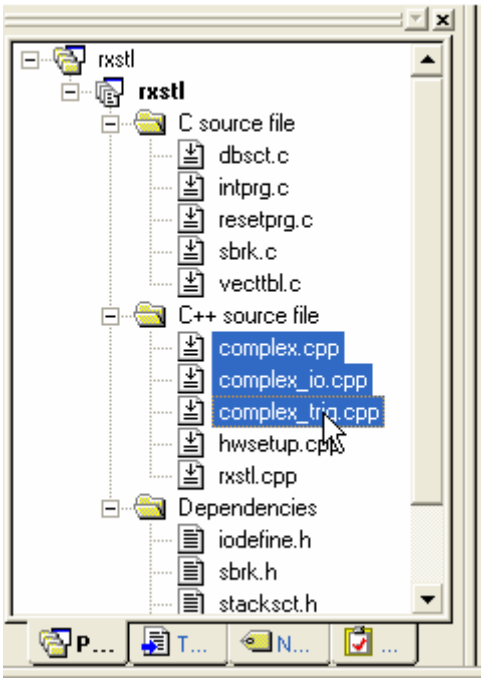

Figure 9 Project File after Adding the Source Files

4. Procedure for Using Wide Characters

When wide characters (wstring or wchar_t) are to be used, specify the library structure as ‘C99’ and tick on the checkbox for ‘wchar.h(C99): Performs wide character’.

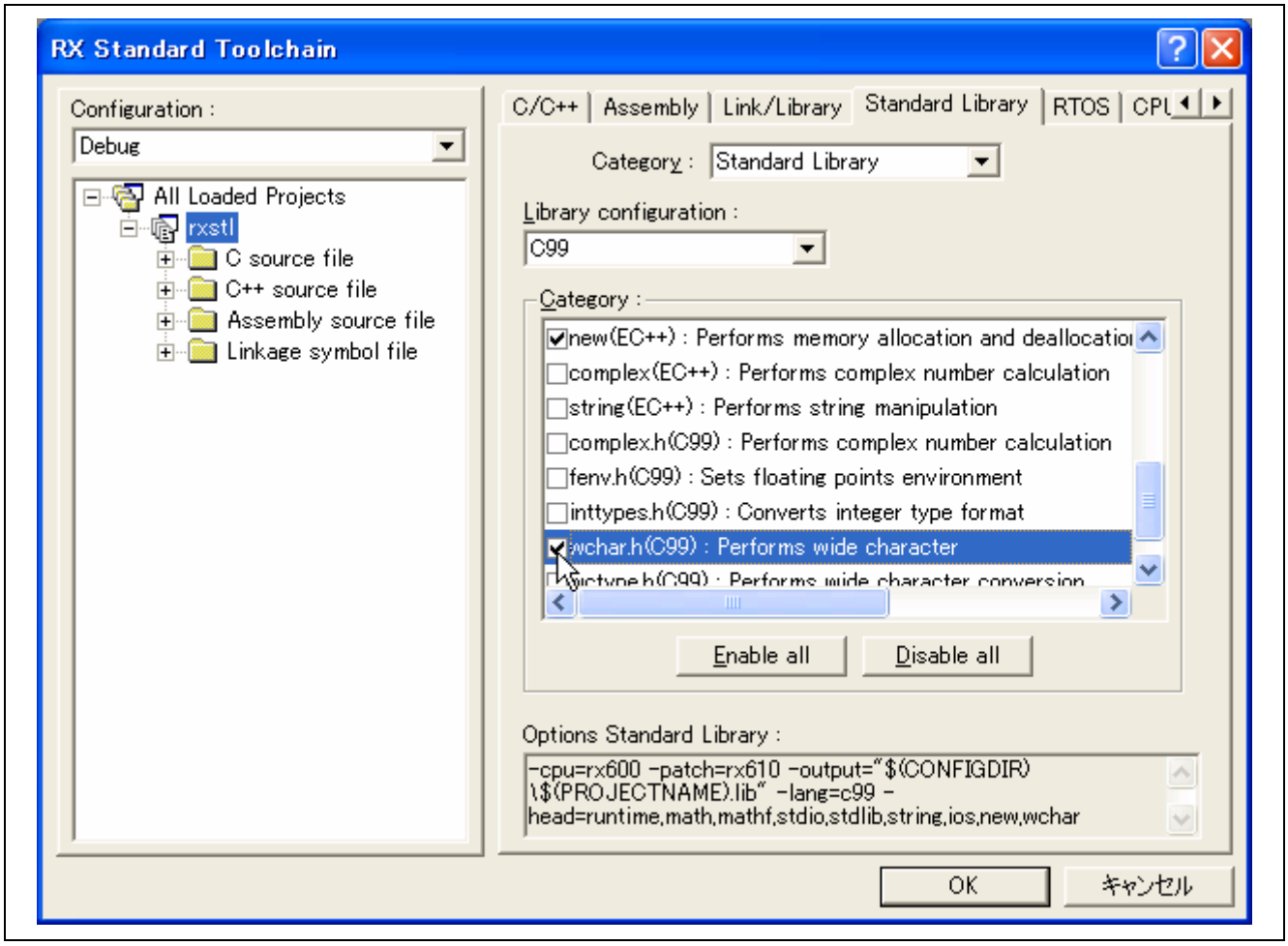

Figure 10 Specifying the Library When Using Wide Characters

5. Procedure for Using an Exception Class (exception or stdexcept)

When an exception class (exception or stdexcept) is used in RXC or SHC, all of the procedures in the remainder of this section must be executed.

5.1 Setting Compiler Options

For SHC, tick on the checkboxes for ‘Use try, throw and catch of C++’ and ‘Enable/disable runtime information’ as shown in figure 11.

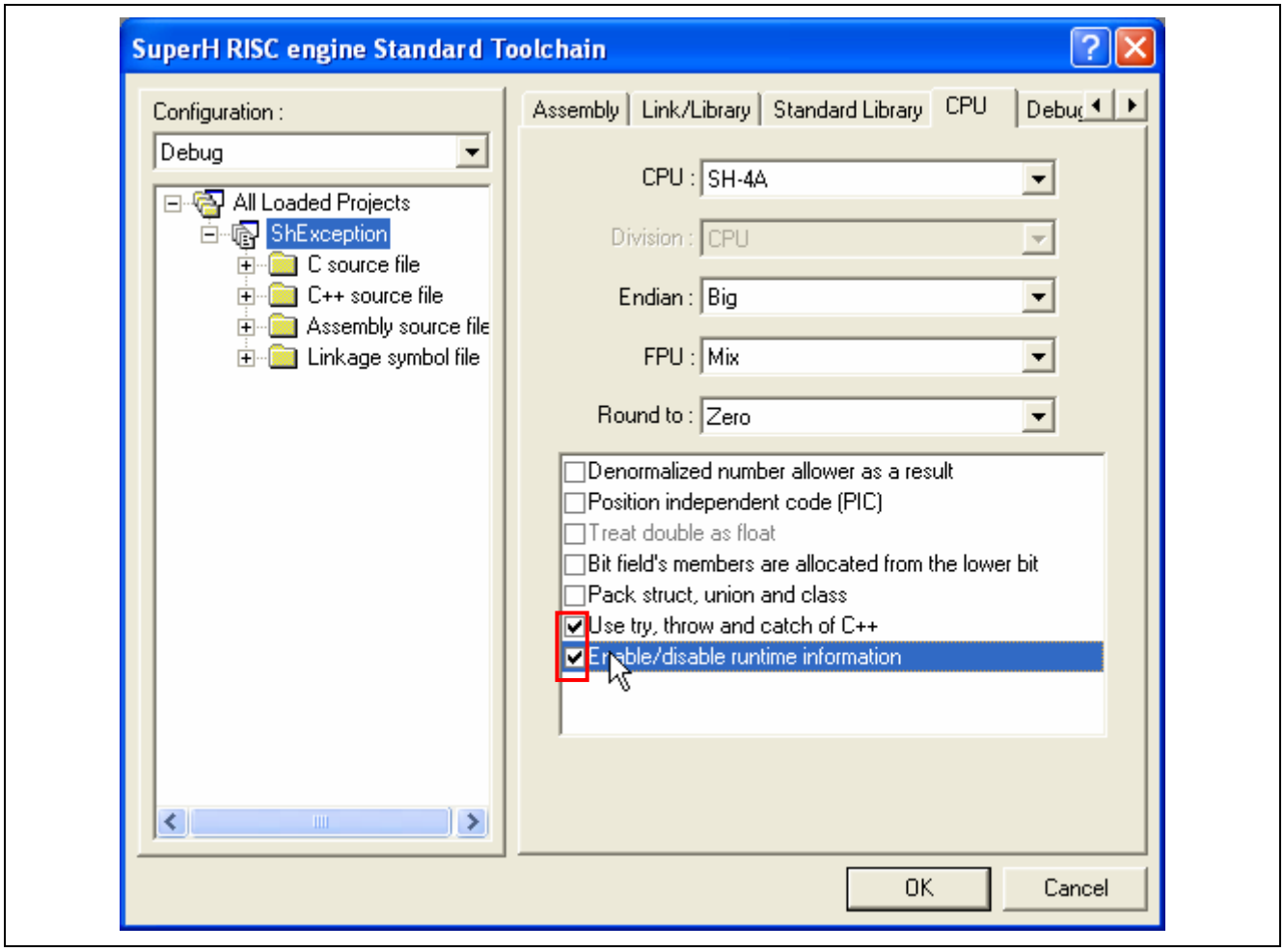

Figure 11 Setting Options When Using the Exception Class in SHC

For RXC, tick on the checkboxes for ‘Use try, throw and catch of C++’ and ‘Use dynamic_cast and typeid of C++’ as shown in figure 12.

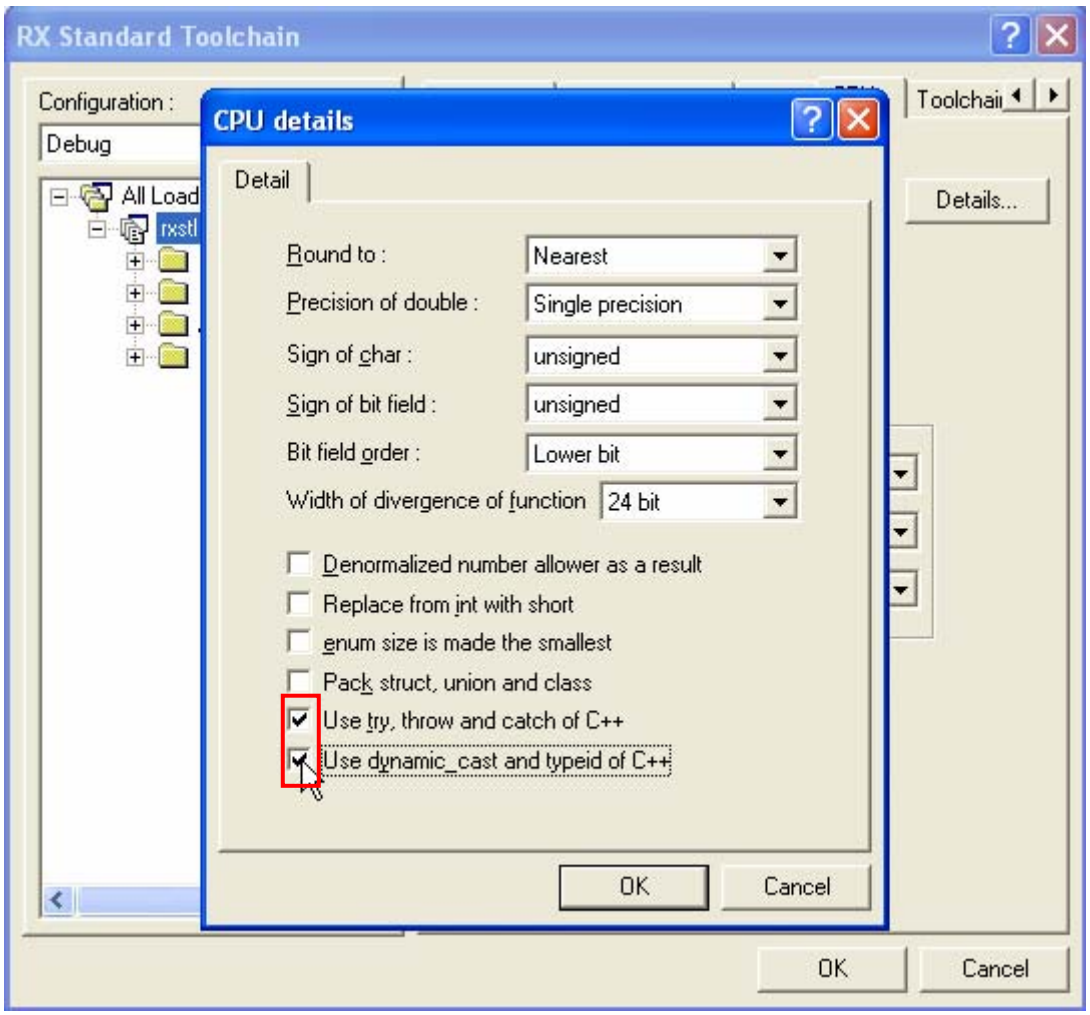

Figure 12 Setting Options When Using the Exception Class in RXC

5.2 Setting the Optimizing Linkage Editor Option

For both SHC and RXC, as shown in figure 13, select ‘Run prelinker’ for ‘Prelinker control’.

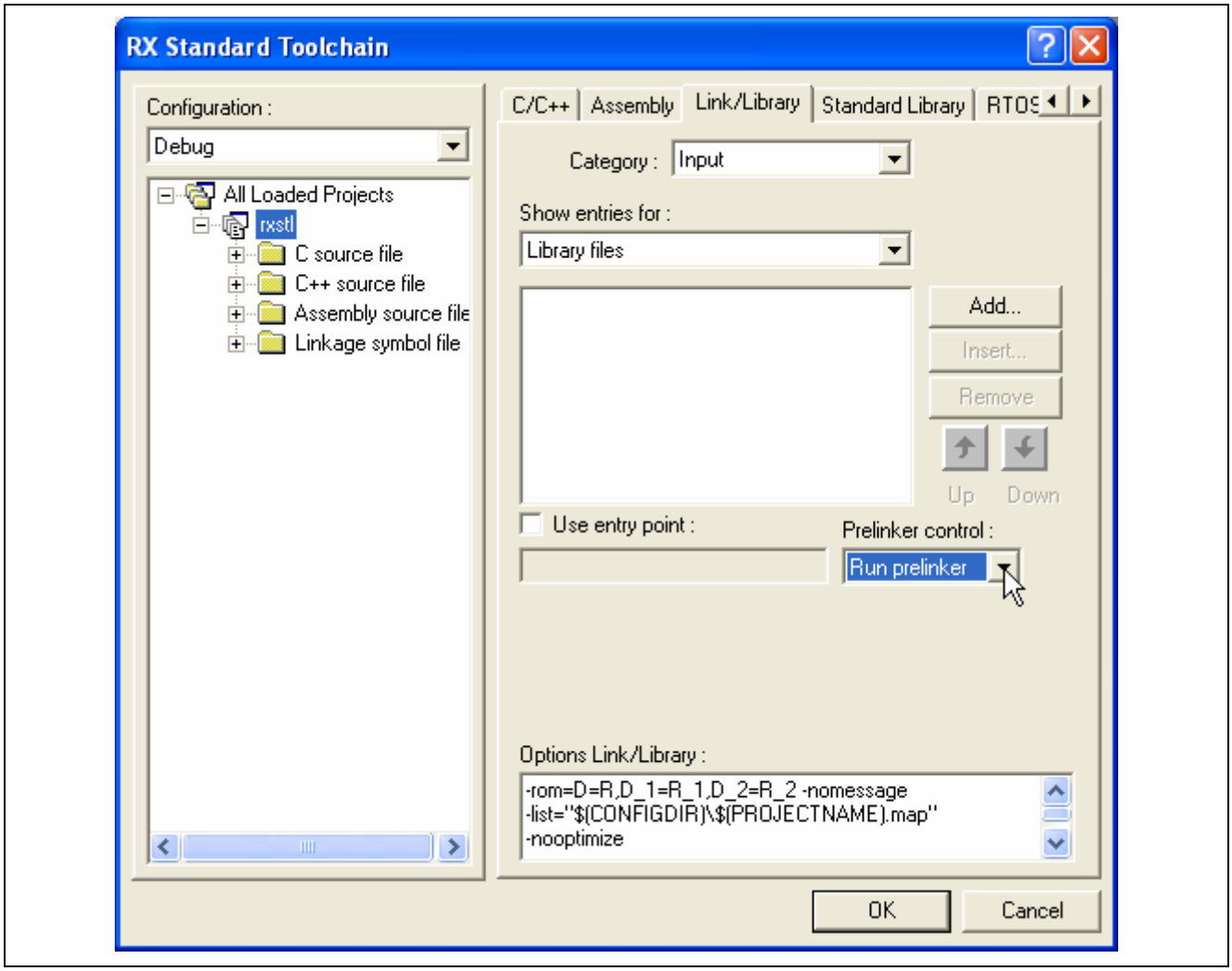

Figure 13 Setting the Optimizing Linkage Editor Option When Using the Exception Class

5.3 Setting the Standard Library

For SHC, as shown in figure 14, deselect ‘Check against EC++ language specification’ in [Miscellaneous options] under the [Other] item for [Category]. Since RXC does not have this option, the standard library setting is not required.

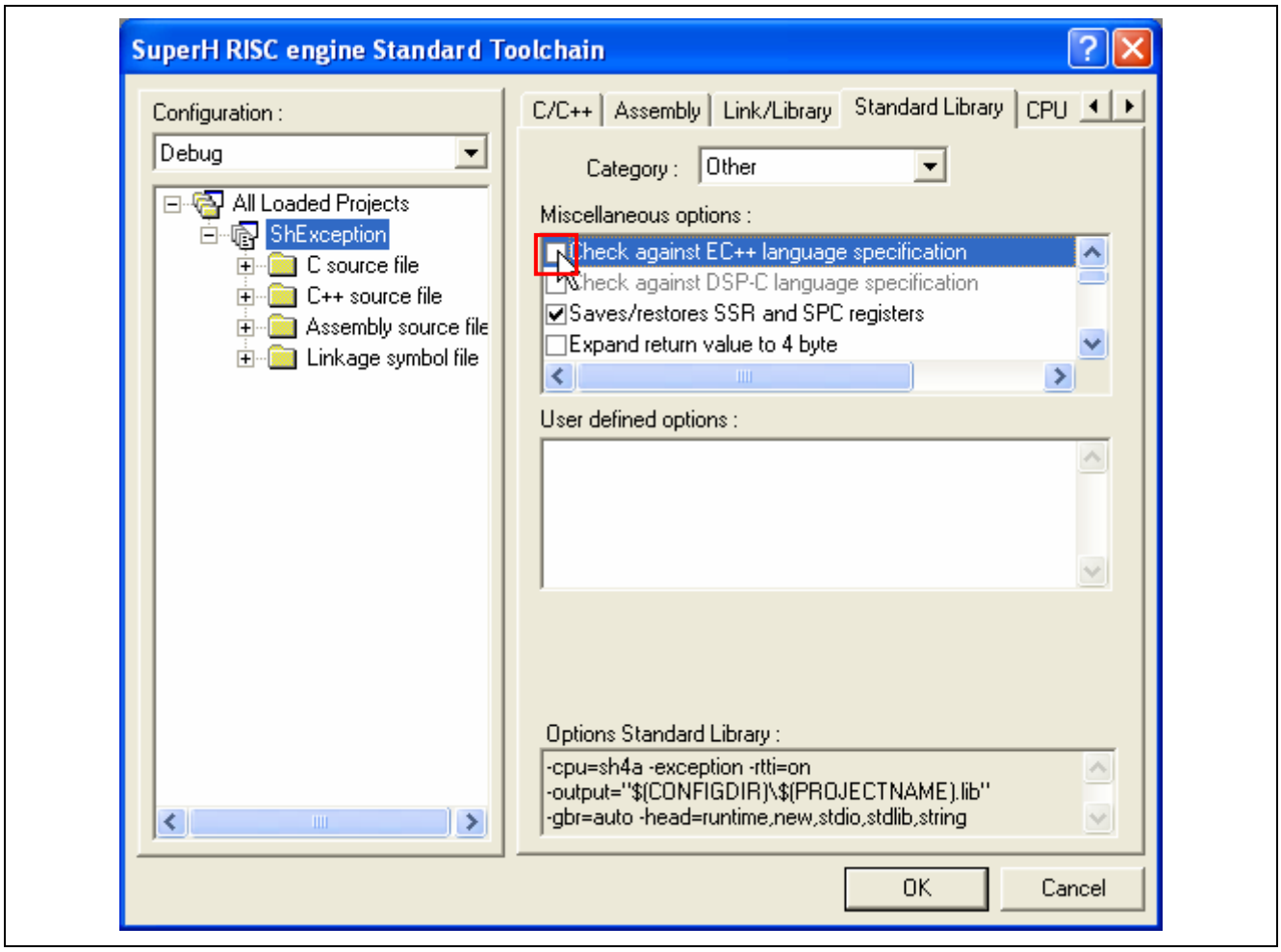

Figure 14 Setting Miscellaneous Option for the Standard Library When Using the Exception Class (SHC)

5.4 Including Source Files

For both SHC and RXC, as shown in figure 15, add the provided source file ‘stdexcept_base.cpp’ to the project.

The additional source files are stored in the following directory.

C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\src

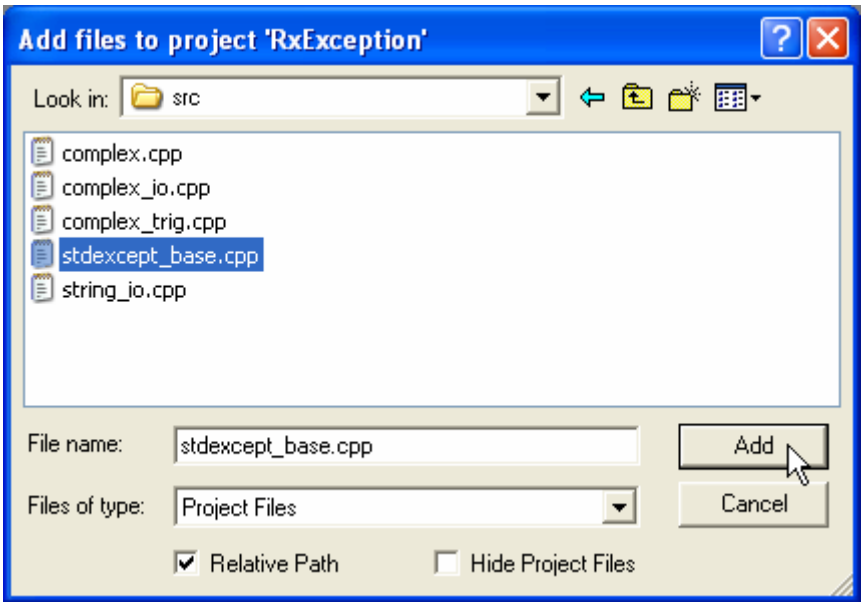

Figure 15 Adding a Provided Source File When Using the Exception Class

As shown in figure 16, check that 'stdexcept_base.cpp' has been added to the workspace.

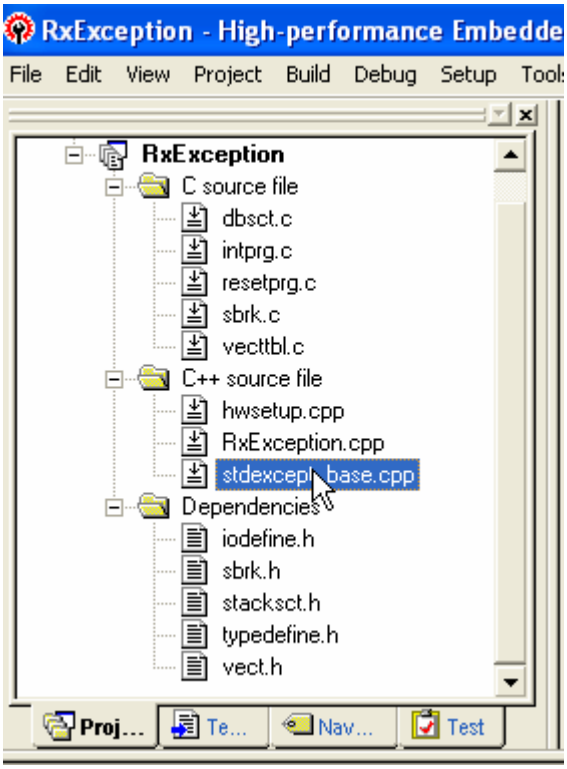

Figure 16 Workspace after Adding a Provided Source File When Using the Exception Class

6. Procedure for Using Class string

6.1 Essentials on Using Stream Input/Output Handling

When string classes are to be used with iostream of RXC or SHC, the following standard libraries provided with RXC or SHC must be included.

- ios(EC++): class library for stream input/output
- ctype.h: library for character operations

The following source file must also be included in the user project.

- string_io.cpp

6.2 Including Source Files

When stream input/output handling is to be used, add ‘string_io.cpp’ to the project in the way shown in figures 17, 18, and 19.

The additional source files are stored in the following directory:

C:\Program Files\Renesas\Hew\EXAMPLES\STL\1_0_0\src

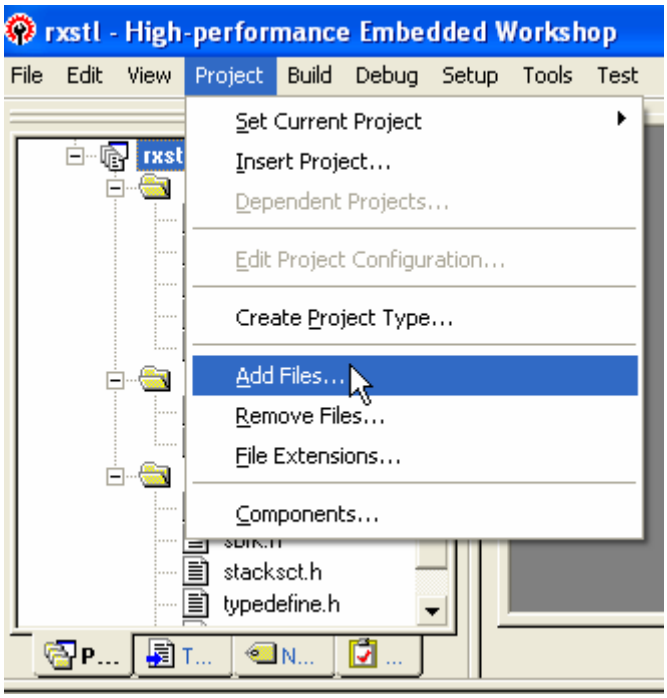

Figure 17 Adding a File to a Project

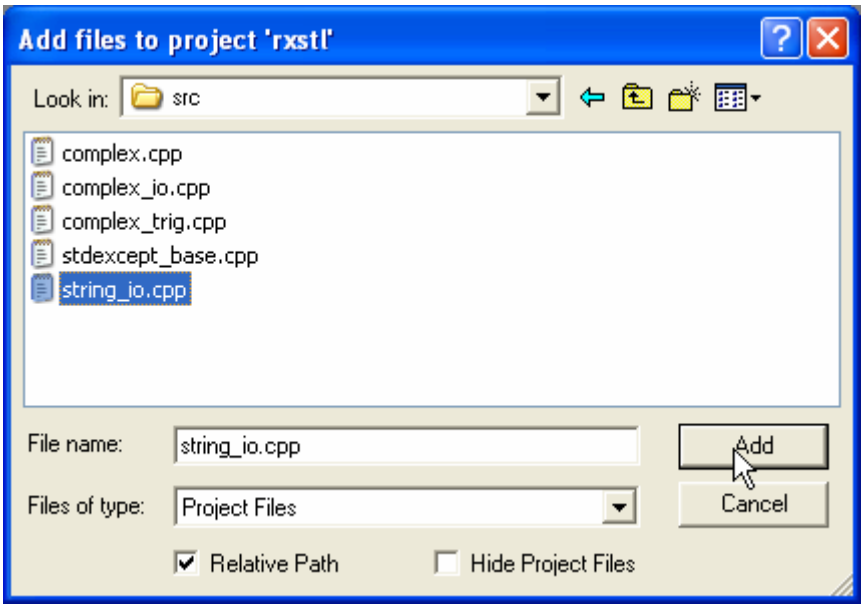

Figure 18 Selecting an Additional Source File

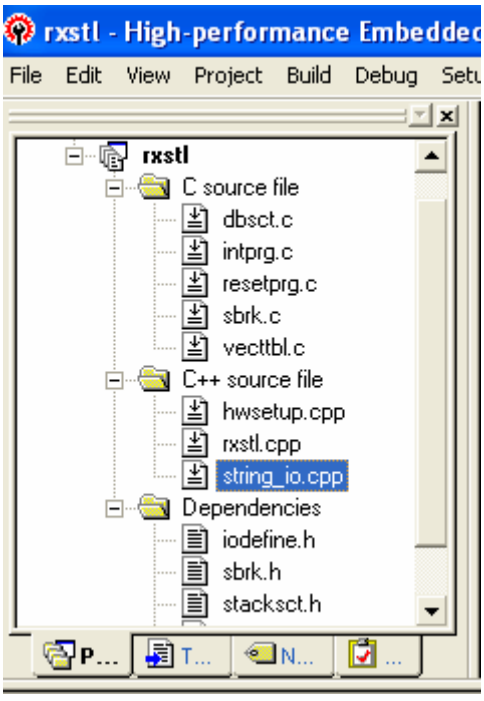

Figure 19 Project File after Adding the Source File

6.3 Limitations on Using Stream Input/Output Handling

In RXC or SHC, the standard stream inputs and outputs, cin and cout, are defined in the EC++ library. However, compared with the specifications in standard C++, the differences shown in Table 3 apply.

Table 3 Differences in Standard Stream Input/Output between Standard C++ and EC++

	Standard C++	EC++
Namespace	std	Global
Standard stream input/output	cin, cout, cerr, clog wcin, wcout, wcerr, wclog	cin, cout

This means, for example, that code which includes ‘std::cout << strl;’ will cause a compilation error.

In such a case, use ‘::cout << strl;’.

The EC++ library supports cin and cout, but does not support cerr and clog.

Wide characters are supported in the standard C library for RXC but not supported in the EC++ library. Accordingly, wcin and wcout, which are used for the input and output of wide characters, are not supported.

If a program includes ‘std::wcout << wstrl;’, consider the following alternatives.

- Switching the output by using wprintf()
- Using cout or cin after the character string type has been converted from wstring to string

7. Note

7.1 Initial Processing and Postprocessing for Global Class Objects (SHC/RXC)

When a global class object is used in the C++ language, Global Class Object Initial Processing (_CALL_INIT) and Global Class Object Postprocessing (_CALL_END) must be called before and after the main function.

This is because the declaration of a global class object is not executed even if a function is executed, so Global Class Object Initial Processing (_CALL_INIT), which explicitly calls a constructor for the target class, and Global Class Object Postprocessing (_CALL_END), which calls a destructor, must be called.

On generation of a project, when the generation of a start-up routine and of a main function in the C source file (figure 20) are specified in the High-performance Embedded Workshop, Global Class Object Initial Processing (_CALL_INIT) and Global Class Object Postprocessing (_CALL_END) are commented out in resetprg.c. Remove the comment marks if the function calls are required.

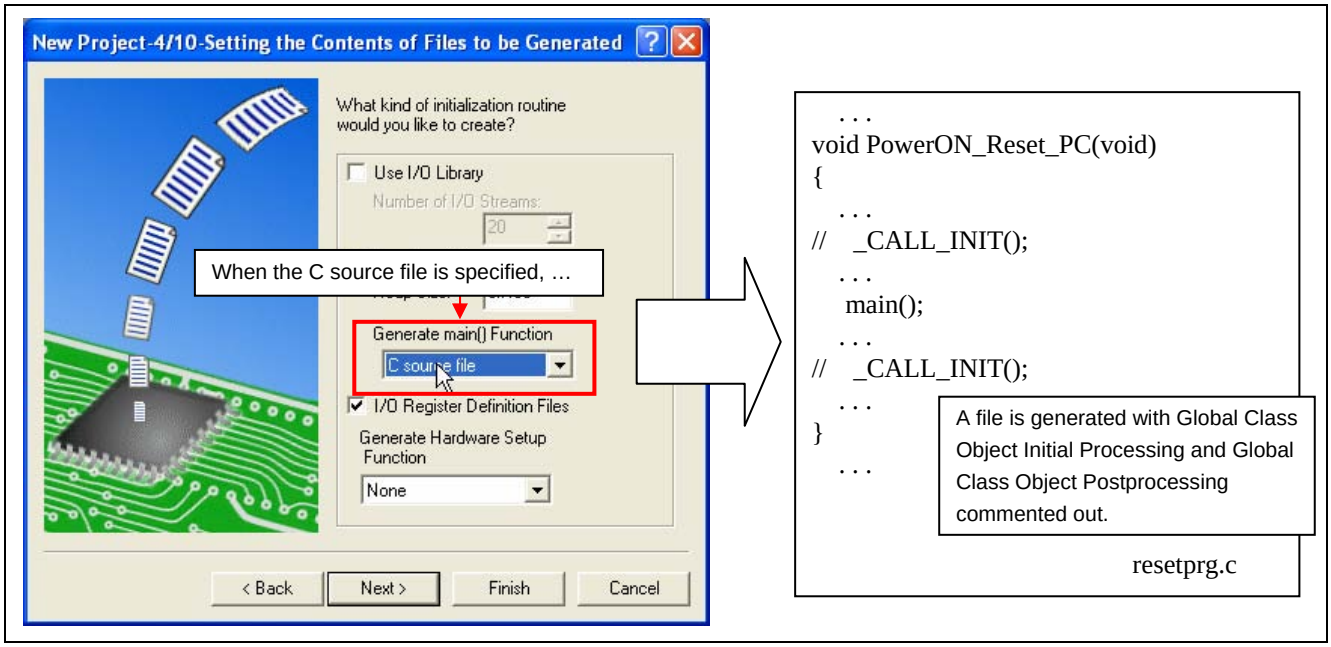

Figure 20 Note on Generating a Project

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Jul.30.10	—	First edition issued
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU 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. Handling of Unused Pins

   Handle unused pins in accord 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the 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. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of 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. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or system 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be 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, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

RENESAS

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

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

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

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

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan, R.O.C.
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, 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 Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0