User's Manual

32

RX63T144 Group

Renesas Starter Kit Tutorial Manual
For e²studio

RENESAS MCU
RX Family / RX600 Series

Renesas Electronics
www.renesas.com

Rev.1.00 Dec 2013

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 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended 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; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 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. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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.

(2012.4)

Disclaimer

By using this Renesas Starter Kit (RSK), the user accepts the following terms:

The RSK is not guaranteed to be error free, and the entire risk as to the results and performance of the RSK is assumed by the User. The RSK is provided by Renesas on an “as is” basis without warranty of any kind whether express or implied, including but not limited to the implied warranties of satisfactory quality, fitness for a particular purpose, title and non-infringement of intellectual property rights with regard to the RSK. Renesas expressly disclaims all such warranties. Renesas or its affiliates shall in no event be liable for any loss of profit, loss of data, loss of contract, loss of business, damage to reputation or goodwill, any economic loss, any reprogramming or recall costs (whether the foregoing losses are direct or indirect) nor shall Renesas or its affiliates be liable for any other direct or indirect special, incidental or consequential damages arising out of or in relation to the use of this RSK, even if Renesas or its affiliates have been advised of the possibility of such damages.

Precautions

The following precautions should be observed when operating any RSK product:

This Renesas Starter Kit is only intended for use in a laboratory environment under ambient temperature and humidity conditions. A safe separation distance should be used between this and any sensitive equipment. Its use outside the laboratory, classroom, study area or similar such area invalidates conformity with the protection requirements of the Electromagnetic Compatibility Directive and could lead to prosecution.

The product generates, uses, and can radiate radio frequency energy and may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment causes harmful interference to radio or television reception, which can be determined by turning the equipment off or on, you are encouraged to try to correct the interference by one or more of the following measures;

- ensure attached cables do not lie across the equipment
- reorient the receiving antenna
- increase the distance between the equipment and the receiver
- connect the equipment into an outlet on a circuit different from that which the receiver is connected
- power down the equipment when not is use
- consult the dealer or an experienced radio/TV technician for help NOTE: It is recommended that wherever possible shielded interface cables are used.

The product is potentially susceptible to certain EMC phenomena. To mitigate against them it is recommended that the following measures be undertaken;

- The user is advised that mobile phones should not be used within 10m of the product when in use.
- The user is advised to take ESD precautions when handling the equipment.

The Renesas Starter Kit does not represent an ideal reference design for an end product and does not fulfil the regulatory standards for an end product.

How to Use This Manual

1. Purpose and Target Readers

This manual is designed to provide the user with an understanding of the RSK hardware functionality, and electrical characteristics. It is intended for users designing sample code on the RSK platform, using the many different incorporated peripheral devices.

The manual comprises of an overview of the capabilities of the RSK product, but does not intend to be a guide to embedded programming or hardware design. Further details regarding setting up the RSK and development environment can found in the User's manual.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

The following documents apply to the RX63T Group. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document Title	Document No.
User's Manual	Describes the technical details of the RSK hardware.	RSKRX63T144 User's Manual for e^{2}studio	R20UT2121EG
Tutorial	Provides a guide to setting up RSK environment, running sample code and debugging programs.	RSKRX63T144 Tutorial Manual for e^{2}studio	R20UT2122EG
Quick Start Guide	Provides simple instructions to setup the RSK and run the first sample, on a single A4 sheet.	RSKRX63T144 Quick Start Guide for e^{2}studio	R20UT2123EG
Schematics	Full detail circuit schematics of the RSK.	RSKRX63T144 Schematics	R20UT2116EG
Hardware Manual	Provides technical details of the RX63T microcontroller.	RSKRX63T Hardware Manual	R01UH0331EJ

2. List of Abbreviations and Acronyms

Abbreviation	Full Form
ADC	Analog to Digital Converter
CD	Compact Disk
E1	E1 Emulator
E20	E20 Emulator
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LVD	Low Voltage Detect
ROM	Read-Only Memory
RSK	Renesas Starter Kit
USB	Universal Serial Bus

Table of Contents

1. Overview 7
1.1 Purpose 7
1.2 Features 7

2. Introduction 8
2.1 Note Regarding Source Code 8

3. Project Workspace 9
3.1 Introduction 9
3.2 Starting e^{2}studio and Importing Sample Code 9
3.3 Build Configurations and Debug Sessions 13
3.3.1 Build Configuration 13
3.3.2 Debug Configuration 13
3.4 Running the Tutorial 15

4. Basic Debugging the Tutorial Program 16
4.1 Program Initialisation 16
4.2 Main Function 19

5. Additional Information 23

1. Overview

1.1 Purpose

This RSK is an evaluation tool for Renesas microcontrollers. This manual describes how to get the RSK tutorial started, and basic debugging operations.

1.2 Features

This RSK provides an evaluation of the following features:

- Renesas microcontroller programming
- User code debugging
- User circuitry such as switches, LEDs and a potentiometer
- Sample application
- Sample peripheral device initialisation code

The RSK board contains all the circuitry required for microcontroller operation.

2. Introduction

This manual is designed to answer, in tutorial form, the most common questions asked about using a Renesas Starter Kit (RSK). The tutorials help explain the following:

- How do I compile, link, download and run a simple program on the RSK?
- How do I build an embedded application?
- How do I use Renesas' tools?

Files referred to in this manual are installed using the project generator as you work through the tutorials. The tutorial examples in this manual assume that installation procedures described in the RSK Quick Start Guide have been completed. Please refer to the quick start guide for details of preparing the configuration.

> These tutorials are designed to show you how to use the RSK and are not intended as a comprehensive introduction to e^{2}studio, compiler toolchains or the E1 emulator. Please refer to the relevant user manuals for more in-depth information.

2.1 Note Regarding Source Code

During the project generation, it is possible that the line numbers for source code illustrated in this document do not match exactly with that in the actual source files. It is also possible that the source address of instructions illustrated in this manual differs from a user's code compiled from the same source. These differences are minor, and do not affect the functionality of the sample code or the validity of this accompanying manual.

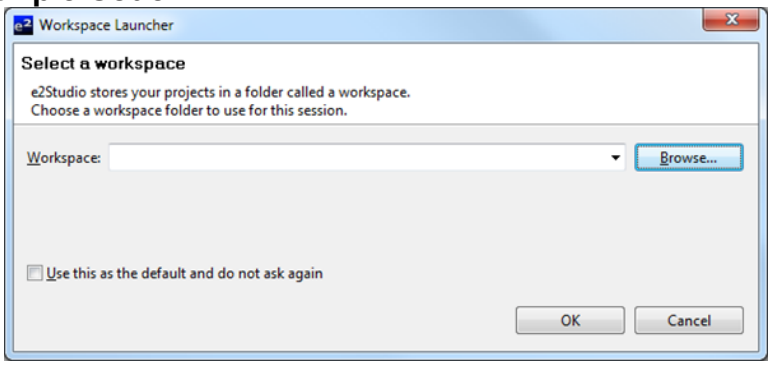
3. Project Workspace

3.1 Introduction

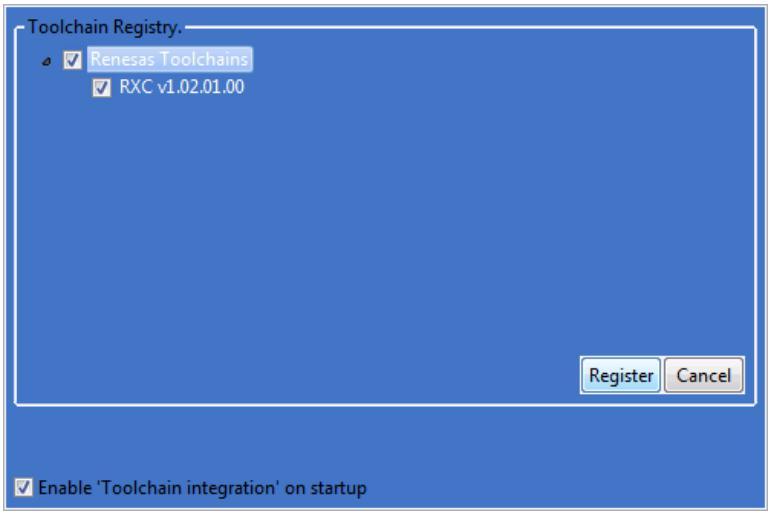
e^{2}studio is an open source integrated development tool that allows the user to write, compile, program and debug a software project on many of the Renesas microcontrollers.

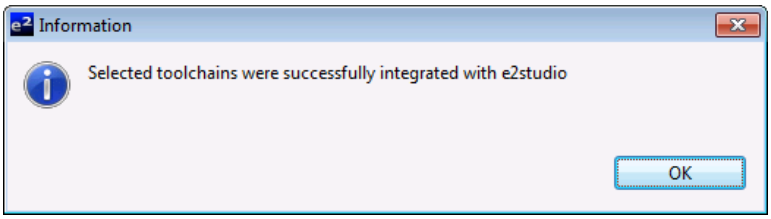
3.2 Starting e^{2}studio and Importing Sample Code

- Start e^{2}studio by selecting it from Start Menu. The first dialog box to appear will be the Workspace Launcher.
- Click ‘Browse’ and select a suitable location to store your workspace, using the ‘Create New Folder’ option as necessary. Click ‘OK’.

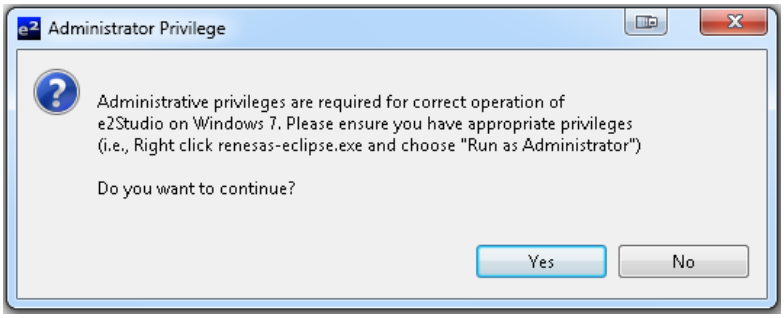


- In the Toolchain registry dialog. Select Renesas Toolchains. RXC v1.02.01.00. Click on Register. A dialog will appear "Selected Toolchains were successfully integrated with e2studio". Click OK.



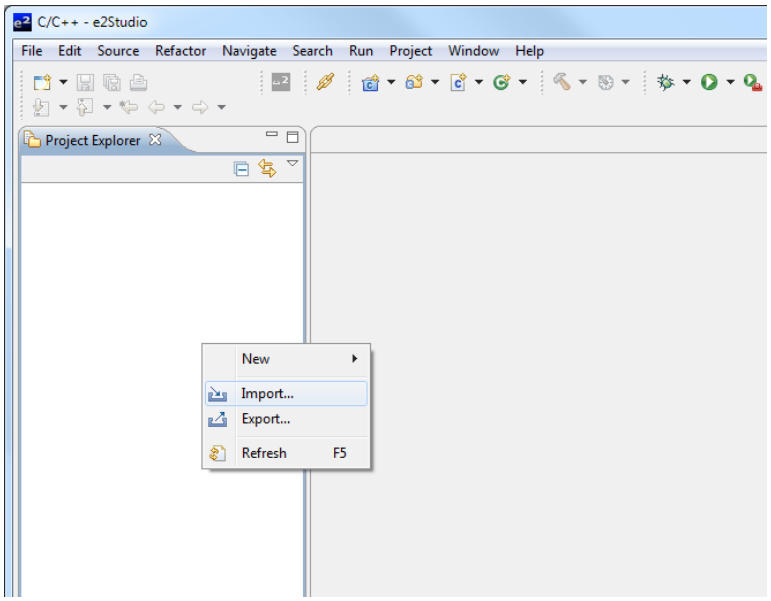


- Click ‘Yes’ when presented with the ‘Administrator Privilege’ dialog box.

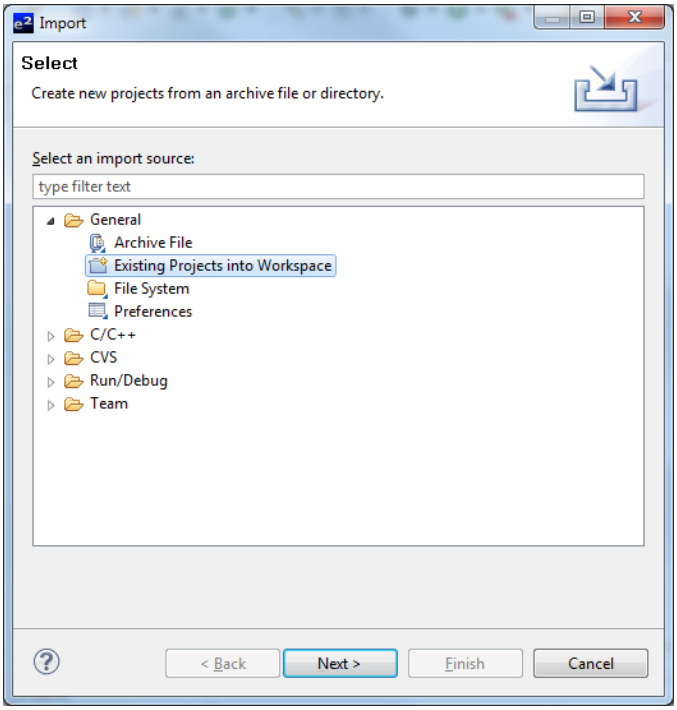


- The e^{2}studio welcome splash screen will appear. Click the ‘Go to the workbench’ arrow button on the far right.

- Once the e^{2}studio environment has initialised, right click in the Project Explorer window and click ‘Import…’



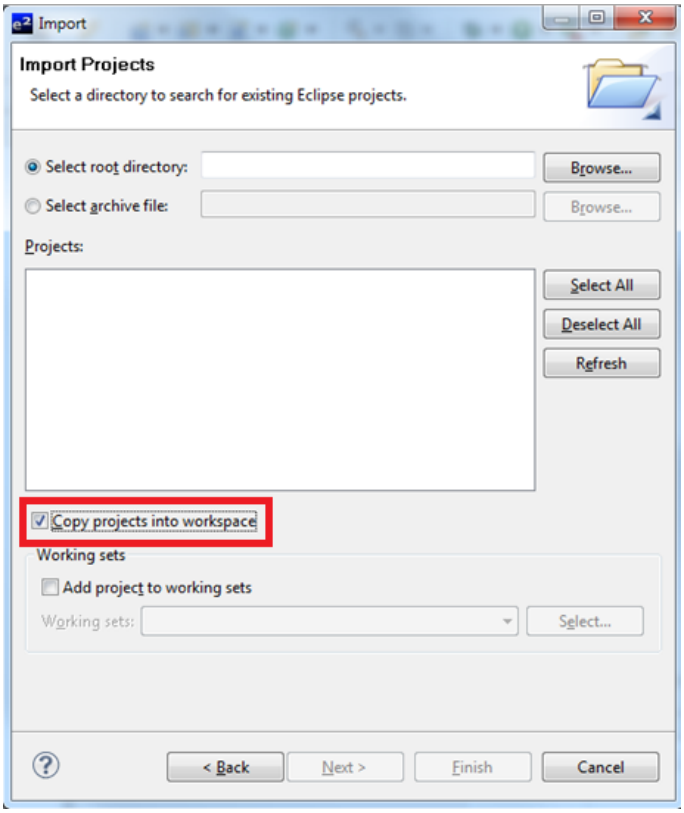
- The Import dialog box will now show. Expand the ‘General’ folder icon, and select “Existing Projects into Workspace”, then click ‘Next’.



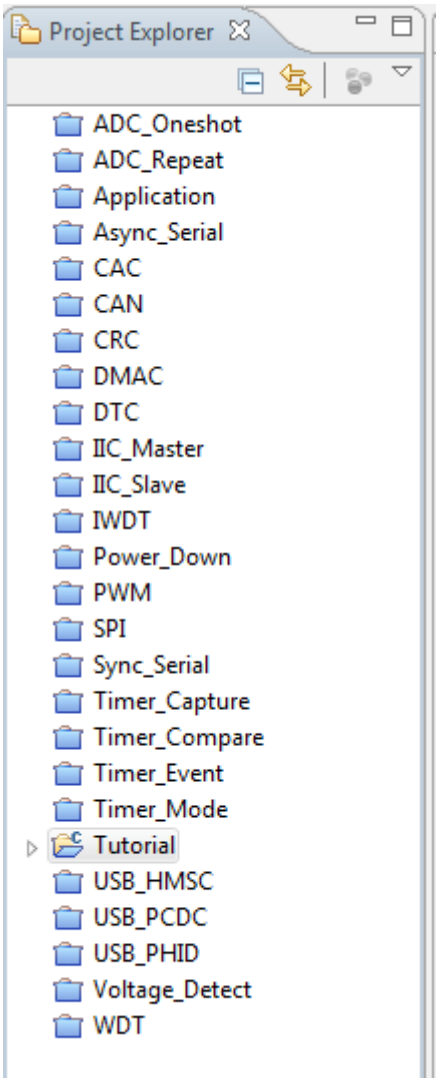
- The Import dialog box will now allow you to specify the project to import. Click the ‘Browse’ button, and locate the following directory:

C:\Renesas\Workspace\RSK\RSKRX63T144

- Ensure that the ‘Copy projects into workspace’ option is ticked, and then click ‘Finish’.



- Click on Tutorial from the list of projects in the ‘Project Explorer’ on the left hand side.



3.3 Build Configurations and Debug Sessions

3.3.1 Build Configuration

The e^{2}studio workspace will be created with several build configurations – the two we will address in this manual are ‘HardwareDebug’ and ‘Release’.

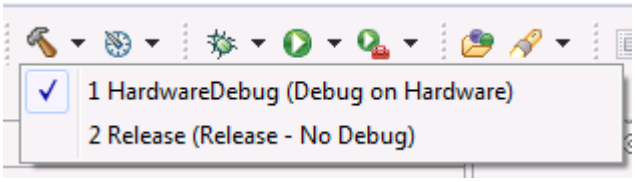
Release

This build mode has optimisation turned on, and provides little debug information. The C code instruction execution may appear to be out of order, due to the way compiler optimises the code. This build configuration is intended for final ROM-programmable code.

HardwareDebug

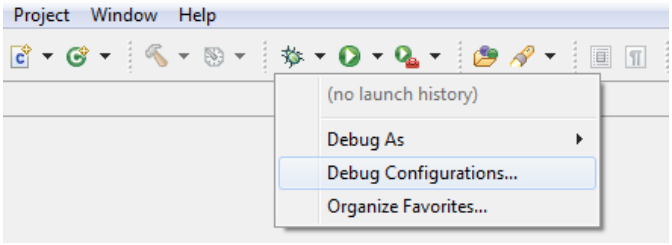
This build mode has all optimisation turned off, and provides full debug information. This is the best configuration to use while developing code. C code instruction execution will be linear.

- Click the top level tutorial project folder again, and then the arrow next to the build button (hammer icon), and select the ‘HardwareDebug’ option.
- e^{2}studio will now build the code.

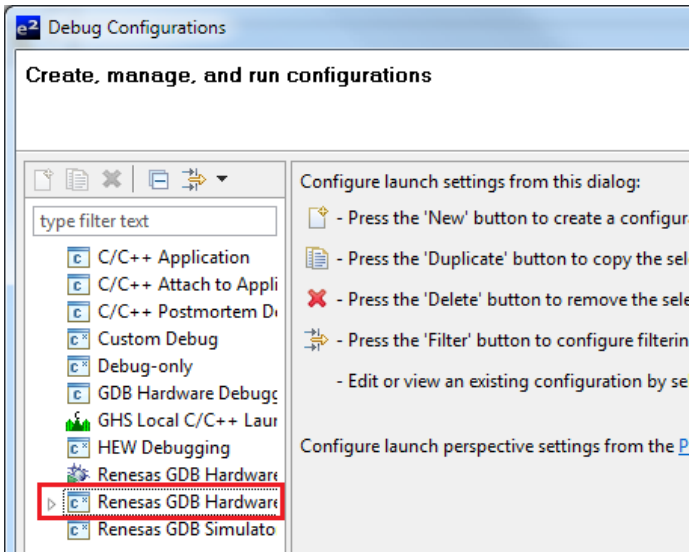


3.3.2 Debug Configuration

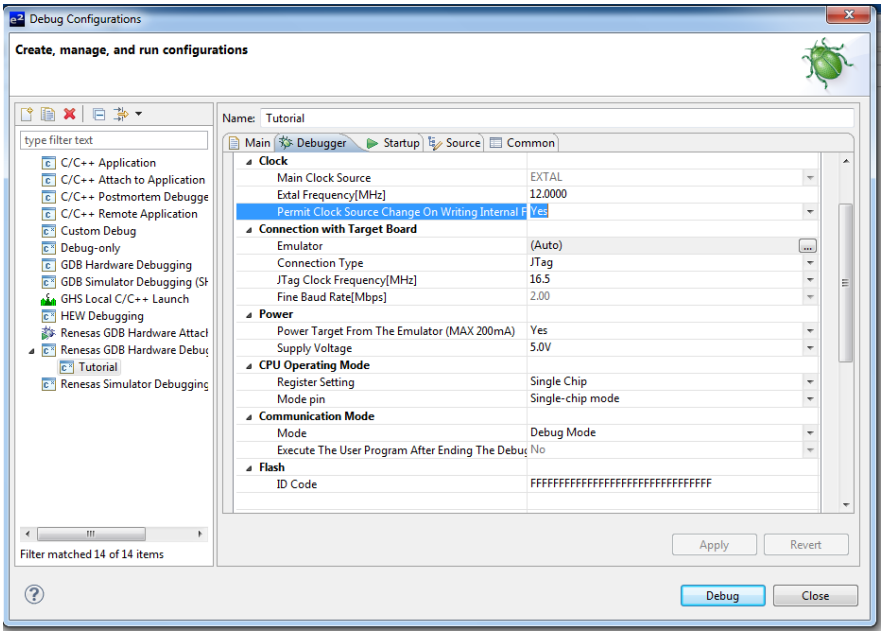
- Click the arrow next to the debug button (bug icon). Select ‘Debug Configurations…’.



- The ‘Debug Configuration’ dialog box will appear. Click the small arrow next to ‘Renesas GDB Hardware Debugging’ option.
- The build configurations for each project will appear. Select the entry for the Tutorial project.

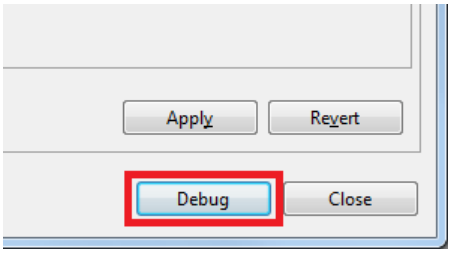


- The Debug Configurations control page will then show for the Tutorial project. Change the main tab to ‘Debugger’, and the secondary tab to ‘Connection Settings’. Check through the debugger settings. If you intend to use an external power supply, set the ‘Power Target From The Emulator’ option to No (drop down menu).
- Refer to the RSK’s User Manual for details of power supply configurations.

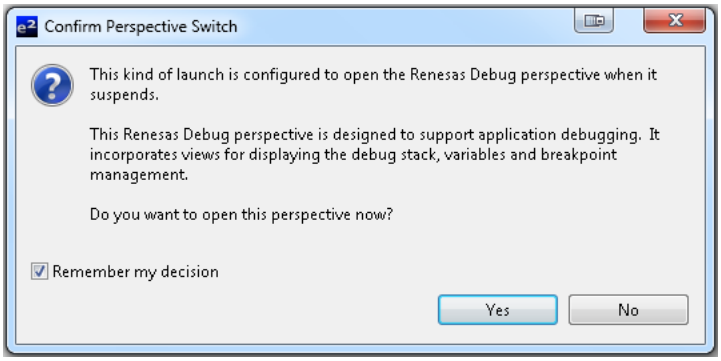


Note: e^{2}studio will display a warning dialog box if you attempted to connect with an incorrect power supply setting.

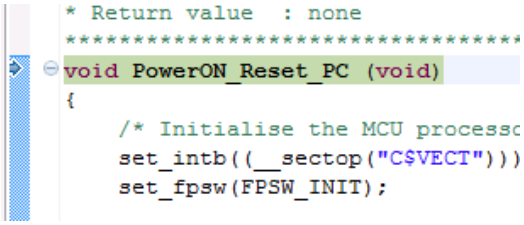
- Click the ‘Apply’ button to save the changes.
- Click the ‘Debug’ button to continue. e^{2}studio will now connect to the debugger, and download the code to the target.



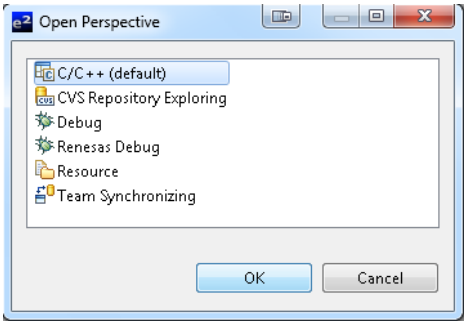
- e^{2}studio may display a dialog box, asking if you would like to switch to the ‘Renesas Debug perspective’. Click ‘Remember my decision’ to prevent this dialog box from appearing in future. Click ‘Yes’
- The new e^{2}studio perspective layout is optimised for debugging.

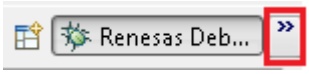


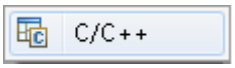
- After downloading the code, the File window will open the 'PowerON_Reset_PC()'.



- To change back to the default 'C/C++' perspective, from the menu bar select:
- Window > Open Perspective > Other
- The 'Open Perspective' dialog box will appear. Click on the desired perspective to select it then click 'OK'.
- Alternatively, click on the button with the double arrow in the top right corner of the screen, as shown opposite, and select the 'C/C++' option that appears.

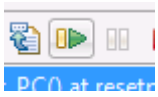






3.4 Running the Tutorial

- Refer to the main.c file for instructions on how to configure the RSK and run the sample code.
- Once the code has been downloaded, click the 'Resume' toolbar button (see image) to run the code to the main function. The main function is set as the program entry point by default. The program counter will stop on the first instruction in the main function.
- Click the 'Resume' button to run the rest of the code
- It is recommended that you run the entire tutorial demo first, before continuing to debug it.



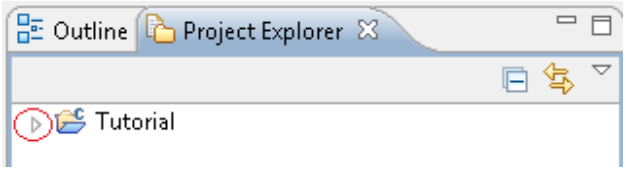
4. Basic Debugging the Tutorial Program

This section will look at basic debugging functionality in e^{2}studio.

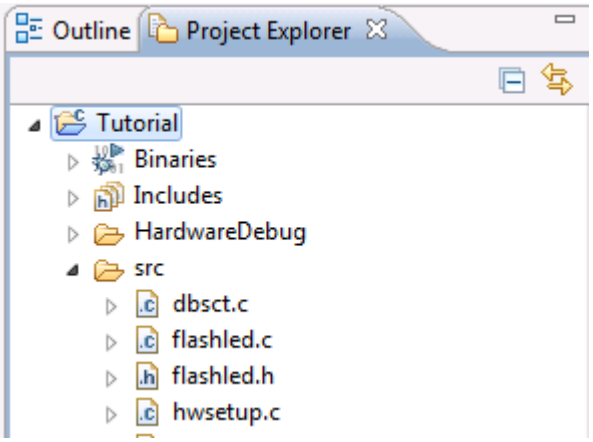
4.1 Program Initialisation

Before the main program can run, the microcontroller must be configured. The following parts of the tutorial program are used exclusively for initialising the RSK device so that the main function can execute correctly. The initialisation code is run every time the device is reset via the reset switch or from a power reboot.

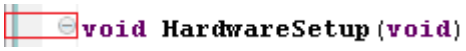
- After downloading the code, navigate to the Project Explorer window on the right hand side.



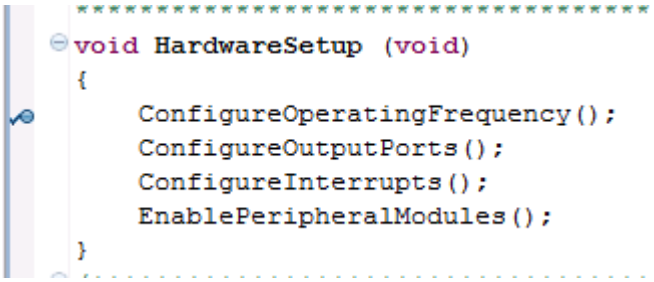
- Click the arrow next to the Tutorial project to expand the folder contents, and then click the arrow next to the 'src' folder to show the source files.



- Double click on 'hwsetup.c' to open the file.



- Breakpoints can be set by double clicking within the column width space indicated by the red box. The column width is hereafter referred to as the breakpoint column.
- On the line with instruction ConfigureOperatingFrequency(), double click next to the vertical line to set a breakpoint.



Note:

The alternative to the above method requires reverting back to the default 'C/C++' perspective.

If in the C/C++ perspective, set the mouse cursor on the instruction, then from the menu bar select Run > Toggle Breakpoint.

- Press ‘Resume’ on the Debug toolbar.

- The debugger should now hit the breakpoint inside the HardwareSetup function definition. This function groups together several key functions that are used to ensure the device is setup correctly before the main program is executed.

```
/***********************************************************************
* Function Name : HardwareSetup
* Description   : Contains all the setup functions called at device restart
* Argument      : none
* Return value  : none
***********************************************************************/
void HardwareSetup (void)
{
    ConfigureOperatingFrequency();
    ConfigureOutputPorts();
    ConfigureInterrupts();
    EnablePeripheralModules();
}
/***********************************************************************
* End of function HardwareSetup
***********************************************************************/
```

- Click ‘Step Into’ to enter the ConfigureOperatingFrequency function.

- The ConfigureOperatingFrequency function is used to set the speed of the system clocks.

```
80          /***********************************************************
81          * Function Name : ConfigureOperatingFrequency
82          * Description   : Configures the clock settings for each of
83          * Argument      : none
84          * Return value  : none
85          ***********************************************************
86 ffff86c8 void ConfigureOperatingFrequency (void)
87          {
88              /* Declare and initialise a loop count variable */
89 ffff86ca     uint16_t i = 0;
90
91              /* Protection off */
92 ffff86d0     SYSTEM.PRCR.WORD = 0xA503;
93
94              /* Specify a clock stabilisation time, greater than 10ms
95                 (21.845 msec) */
96              /* XTAL = 12MHz, Period = 83.3 ns, ( {MOSCWTCR = 0x0E} 2
97                 = 21.845 ms */
98 ffff86de     SYSTEM.MOSCWTCR.BYTE = 0x0E;
99
100             /* 4194304 state (default)*/
101             /* wait over 12ms  @PLL=192MHz(12MHz*16) */
102 ffff86e8    SYSTEM.PLLWTCR.BYTE = 0x0F;
103
104             /* x16 @PLL */
```

- The list of initialised system clocks is shown on the opposite screenshot.

- We will now skip past the hardware setup functions to look at the tutorial's main program code.

resetprg.c | hwsetup.c | main.c

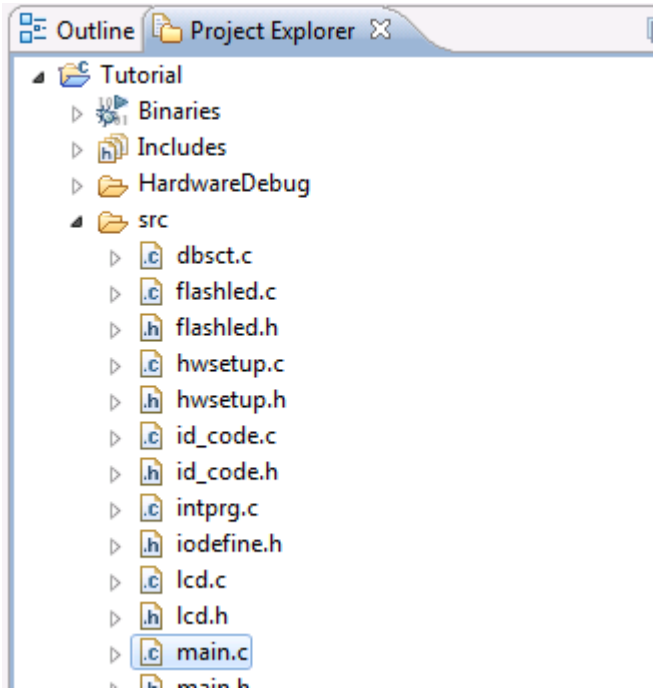
```
/* Configure the clocks as follows -
Clock Description              Frequency
----------------------------------------
PLL Clock frequency.............192MHz
System Clock Frequency..........96MHz
Peripheral Module Clock B.......48MHz
FlashIF Clock...................48MHz
External Bus Clock..............48MHz */
SYSTEM.SCKCR.LONG = 0x21821211;
```

For further details regarding hardware configuration, please refer to the RSKRX63T User's Manual and the RX63T Hardware Manual.

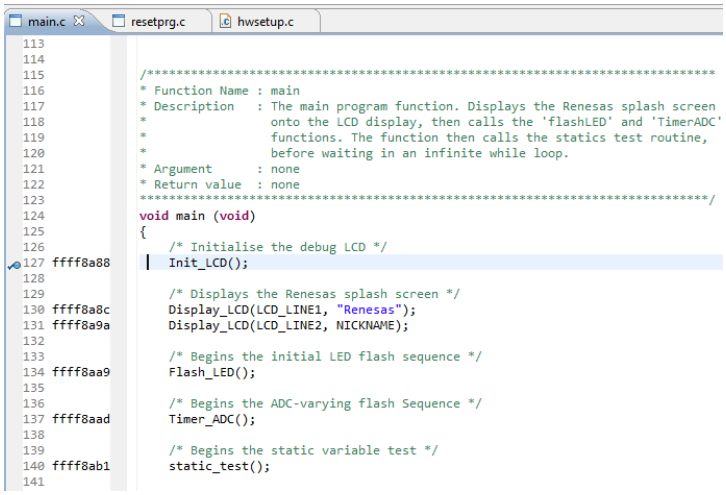
4.2 Main Function

This section will examine the program code called from within the main() function.

- Find the main.c file from the Project Explorer, and then double-click it to open the file in the text editor.



- Set a breakpoint at the call to the main() function by double clicking the column next to void main (void).



The E1 emulator features an advanced logic-based event point trigger system, and full instruction on its use is outside the scope of this tutorial. For further details, please refer to the RX Family E1/E20 Emulator User's Manual

- Press ‘Resume’ on the Debug toolbar.

- The code will execute to the breakpoint. At this point all the device initialisation will have been completed. The code window will open ‘main.c’ and show the new position of the program counter.

```
115                 /*****************************************************
116                 * Function Name : main
117                 * Description   : The main program function. Display
118                 *                 onto the LCD display, then calls t
119                 *                 functions. The function then calls
120                 *                 before waiting in an infinite whil
121                 * Argument      : none
122                 * Return value  : none
123                 *****************************************************
124                 void main (void)
125                 {
126                     /* Initialise the debug LCD */
127 ffff8a88            Init_LCD();
128
129                     /* Displays the Renesas splash screen */
130 ffff8a8c            Display_LCD(LCD_LINE1, "Renesas");
131 ffff8a9a            Display_LCD(LCD_LINE2, NICKNAME);
132
133                     /* Begins the initial LED flash sequence */
134 ffff8aa9            Flash_LED();
135
136                     /* Begins the ADC-varying flash Sequence */
137 ffff8aad            Timer_ADC();
138
139                     /* Begins the static variable test */
140 ffff8ab1            static_test();
141
142                     /* Infinite while loop */
143 ffff8ab4            while (1)
144                     {
```

- Right click the ‘Flash_LED()’ function call and select ‘Run to Line’ to execute the program up to this line. The ‘Init_LCD()’ function call enables and configures the LCD panel, and ‘Display_LCD()’ will write “Renesas” on the top line and “RX63T-L” onto the bottom line
- Set a breakpoint on the ‘timer_adc()’ function call by double-clicking in the breakpoint column

```
133                     /* Begins the initial LED flash sequence */
134 ffff8aa9            Flash_LED();
135
136                     /* Begins the ADC-varying flash Sequence */
137 ffff8aad            Timer_ADC();
138
139                     /* Begins the static variable test */
140 ffff8ab1            static_test();
141
142                     /* Infinite while loop */
143 ffff8ab4            while (1)
```

- Click the ‘Step Into’ button to step into the ‘Flash_LED()’ function.

- The program counter should now move into the Flash_LED function definition. This function periodically polls the user switches and flashes all the LEDs 200 times or until a user switch has been pressed.

```
65                  *****************************************************
66 ffff85a4         void Flash_LED (void)
67                  {
68
69                      /* Variable used to count down the number of
70                      static uint16_t flash_count = 0xC8;
71
72                      /* Declare a delay count variable */
73 ffff85a6            uint32_t        ulLed_Delay = 0;
74
75                      /* Flash the LEDs for 200 times or until a us
76 ffff85ac            while ((0 == g_switch_flag ) && (--flash_coun
77                      {
78 ffff85df                for (ulLed_Delay = 0; ulLed_Delay < 40000
79                          {
80                              /* delay */
81                          }
82
83                          /* Toggles the LEDs after a specific dela
84 ffff8601                Toggle_LED();
85                      }
86
87                      /* Reset the g_switch_flag flag variable */
88 ffff8637            g_switch_flag = 0;
89                  }
```

.

- Click ‘Resume’ to continue the program, and then push any switch to proceed. The program should halt at the event point set on the Timer_ADC function call.
- Press ‘Step Into’ twice to step into the start_timer function.

- The start_timer function configures the timer CMT1 to periodically flash the LEDs. And timer CMT2 to provide a simple delay function.
- Press ‘Step Return’ button to exit the start_timer function, then press ‘Step Into’. The program should now reach the start_adc function

- Press resume to continue with program execution. The start_adc function configures the ADC unit to make repeat conversions of the voltage from the potentiometer RV1.

- Set a breakpoint at ‘static_test’.

```
102                 /*****************************************************
103                 * Function Name : start_timer
104                 * Description   : Configures CMT channel 1 to call th
105                 *                 function which starts the AD conver
106                 * Argument      : none
107                 * Return value  : none
108                 *****************************************************
109                 static void start_timer (void)
110                 {
111                     /* Protection off */
112 ffff9469            SYSTEM.PRCR.WORD = 0xA503;
113
114                     /* Cancel the CMT1 module clock stop mode */
115 ffff9477            MSTP_CMT1 = 0;
116
117                     /* Cancel the CMT2 module clock stop mode */
118 ffff9493            MSTP_CMT2 = 0;
119
120                     /* Protection on */
121 ffff94ae            SYSTEM.PRCR.WORD = 0xA500;
122
123                     /* Set CMT1 interrupt priority level to 10 */
```

```
164                 static void start_adc (void)
165                 {
166                     /* Protection off */
167 ffff959a            SYSTEM.PRCR.WORD = 0xA503;
168
169                     /* Cancel the S12AD module clock stop mode */
170 ffff95a8            MSTP_S12AD = 0;
171
172                     /* Protection on */
173 ffff95c4            SYSTEM.PRCR.WORD = 0xA500;
174
175                     /* Clear the S12AD interrupt flag */
176 ffff95d2            IR(S12AD, S12ADI) = 0x0;
177
178                     /* Set the S12AD interrupt level to 5 */
179 ffff95e1            IPR(S12AD, S12ADI) = 0x5;
```

```
136                     /* Begins the ADC-varying flash Sequence */
137 ffff8aad            Timer_ADC();
138
139                     /* Begins the static variable test */
140 ffff8ab1            static_test();
141
142                     /* Infinite while loop */
143 ffff8ab4            while (1)
144                     {
145                         /* Do nothing */
146                     }
```

- Press F8 to resume the code, where it will then halt at the break point on the static_test function call.

- Press F5 to step into the function.

- The static_test function initialises a character string with the contents of a static variable; then gradually replaces it, letter by letter, with another static string.
- Click ‘Resume’ or press F8 to resume the program code. You should observe the word ‘STATIC’ appear on the second LCD line, to be gradually replaced with the string ‘TESTTEST’. The program then reverts the LCD back to the original message of ‘RX63T144’.

```
152
153             /***********************************************
154             * Function Name : static_test
155             * Description   : Static variable test routine. The fun
156             *                 contents of the string 'cStr' with th
157             *                 element at a time. Right-click the va
158             *                 select 'instant watch' - click add in
159             *                 If you step through the function, you
160             *                 elements being overwritten with the n
161             * Argument      : none
162             * Return value  : none
163             ***********************************************
164 ffff8ac0   static void static_test (void)
165             {
166                 /* Declare loop count variable */
167 ffff8ac2        uint8_t uicount = 0;
168
169                 /* Write cStr variable, "STATIC" to LCD */
170 ffff8ac8        Display_LCD(LCD_LINE2, cStr);
171
172                 /* Begin for loop which writes one letter of ucRepl
173                    The nested while loops generate the delay betwee
```

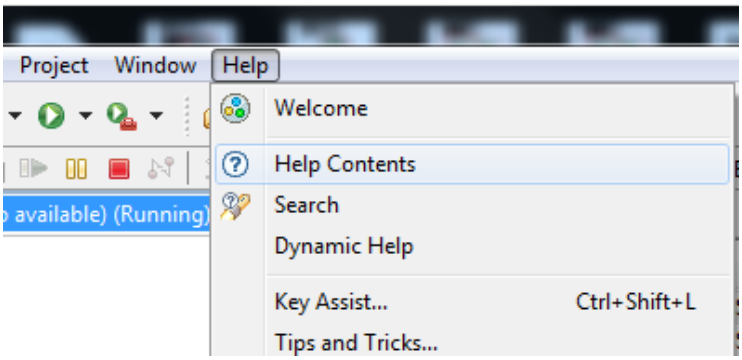
- This Press the ‘Suspend’ button to halt program execution.

This is the extent of the tutorial code.

5. Additional Information

Technical Support

For details on how to use e^{2}studio, refer to the help file by opening e^{2}studio and clicking 'Help' and selecting 'Help Contents'.



Parts of the sample code provided with the RSKRX63T144 can be reproduced using the 'Applilet3 for RX63T' code generator tool. Applilet can be downloaded from the Renesas website.
Source files and functions generated by Applilet are prefixed with 'r_' and 'R_', respectively.

For information about the RX63T series microcontrollers refer to the RX63T Group Hardware Manual.

For information about the RX63T assembly language, refer to the RX600 Series Software Manual.

Online technical support and information is available at: http://www.renesas.com/rskrx63t

Technical Contact Details

Please refer to the contact details listed in section 7 of the "Quick Start Guide".

General information on Renesas microcontrollers can be found on the Renesas website at:
http://www.renesas.com/

Trademarks

All brand or product names used in this manual are trademarks or registered trademarks of their respective companies or organisations.

Copyright

REVISION HISTORY	RSKRX63T Tutorial Manual

Rev.	Date	Description	
		Page	Summary
1.00	Dec 06, 2013	¾	First Edition issued

Renesas Starter Kit Tutorial Manual

Publication Date: Rev. 1.00 Dec 06, 2013

Published by: Renesas Electronics Corporation

Renesas Electronics Corporation

SALES OFFICES 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: +44-1628-651-700, Fax: +44-1628-651-804
Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, 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.
13F, No. 363, Fu Shing North Road, Taipei, 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 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 1.3

RX63T144 Group

Renesas Electronics Corporation

R20UT2122EG0100