User's Manual

32

RX231 Group

Renesas Starter Kit Tutorial Manual
For e^2 studio

RENESAS MCU
RX Family / RX200 Series

Renesas Electronics
www.renesas.com

Rev. 1.00 Aug 2015

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; anticrime 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

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 in 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 how to use the e^2 studio IDE to develop and debug software for the RSK platform. It is intended for users designing sample code on the RSK platform, using the many different incorporated peripheral devices.

The manual comprises of step-by-step instructions to load and debug a project in e^2 studio, but does not intend to be a complete guide to software development on the RSK platform. Further details regarding operating the RX231 microcontroller may be found in the RX231 Group Hardware Manual and within the provided sample code.

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 RX231 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.	RSKRX231 User's Manual	R20UT3027EG
Tutorial Manual	Provides a guide to setting up RSK environment, running sample code and debugging programs.	RSKRX231 Tutorial Manual	R20UT3032EG
Code Generator Tutorial	Provides a guide to code generation and importing into the e^2 studio IDE.	RSKRX231 Code Generator Tutorial Manual	R20UT3034EG
Quick Start Guide	Provides simple instructions to setup the RSK and run the first sample, on a single A4 sheet.	RSKRX231 Quick Start Guide	R20UT3033EG
Schematics	Full detail circuit schematics of the RSK.	RSKRX231 Schematics	R20UT3026EG
Hardware Manual	Provides technical details of the RX231 microcontroller.	RX231 Group Hardware Manual	R01UH0496EJ

2. List of Abbreviations and Acronyms

Abbreviation	Full Form
ADC	Analog-to-Digital Converter
API	Application Programming Interface
bps	Bits per second
CMT	Compare Match Timer
COM	COMmunications port referring to PC serial port
CPU	Central Processing Unit
DVD	Digital Versatile Disc
E1	Renesas On-chip Debugging Emulator
GUI	Graphical User Interface
IDE	Integrated Development Environment
IRQ	Interrupt Request
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LSB	Least Significant Bit
LVD	Low Voltage Detect
MCU	Micro-controller Unit
MSB	Most Significant Bit
PC	Personal Computer
Pmod™	This is a Digilent Pmod™ Compatible connector. Pmod™ is registered to Digilent Inc. Digilent-Pmod_Interface_Specification
PLL	Phase-locked Loop
RAM	Random Access Memory
ROM	Read Only Memory
RSK	Renesas Starter Kit
RTC	Realtime Clock
SAU	Serial Array Unit
SCI	Serial Communications Interface
SPI	Serial Peripheral Interface
TAU	Timer Array Unit
TFT	Thin Film Transistor
TPU	Timer Pulse Unit
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
WDT	Watchdog timer

Table of Contents

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 initialization 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, the compiler toolchains or the E1 emulator. Please refer to the relevant user manuals for more in-depth information.

2.1 Code Generator Plug in

The Code Generator plug in for the RX231 has been used to generate the sample code discussed in this document. Code Generator for e^2 studio is a plug in tool for generating template 'C' source code and project settings for the RX231. When using Code Generator, the engineer is able to configure various MCU features and operating parameters using intuitive GUI controls, thereby bypassing the need in most cases to refer to sections of the Hardware Manual.

Once the engineer has configured the project, the 'Generate Code' function is used to generate three code modules for each specific MCU feature selected. These code modules are name 'r_cg_xxx.h', 'r_cg_xxx.c', and 'r_cg_xxx_user.c', where 'xxx' is a three letter acronym for the relevant MCU feature, for example 'adc'. Within these code modules, the engineer is then free to add custom code to meet their specific requirement. Custom code should be added, whenever possible, in between the following comment delimiters:

```
/* Start user code for adding. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */
```

Code Generator will locate these comment delimiters, and preserve any custom code inside the delimiters on subsequent code generation operations. This is useful if, after adding custom code, the engineer needs to re-visit Code Generator to change any MCU operating parameters.

2.2 Note Regarding Source Code

Due to the project generator, it is possible that 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 differ from those in user code compiled from the same source. These differences are minor, and do not affect the functionality of the sample code nor the validity of this manual.

3.Tutorial 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 product on many of the Renesas microcontrollers.

3.2 Connecting the Debugger

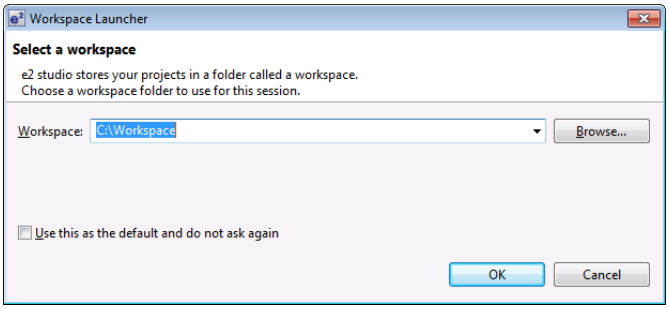
For this tutorial the E1 debugger will provide power to the RSK, no external power supply is required.

The Quick Start Guide provided with the Renesas Starter Kit board gives detailed instructions on how to connect the E1 to the host computer. The following assumes that the steps in the Quick Start Guide have been followed and the E1 drivers have been installed.

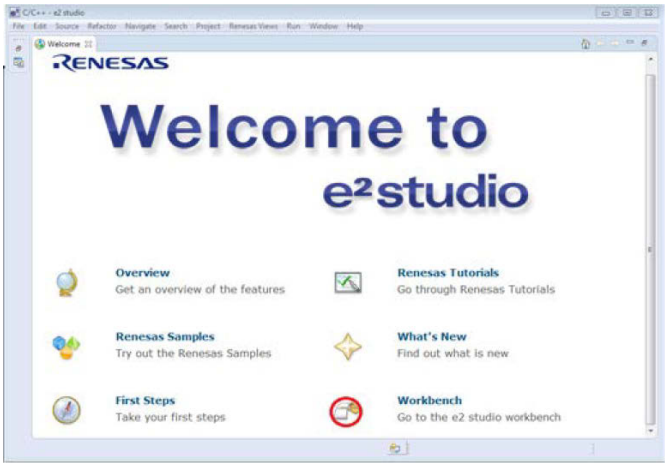
- Fit the PMOD LCD display to the board. Ensure all the pins of the connector are correctly inserted in the socket.
- Connect the E1 Debugger to a free USB port on your computer.
- Connect the E1 Debugger to the target hardware ensuring that it is plugged into the connector marked ‘E1’.

3.3 Starting e^2 studio and Importing Sample Code

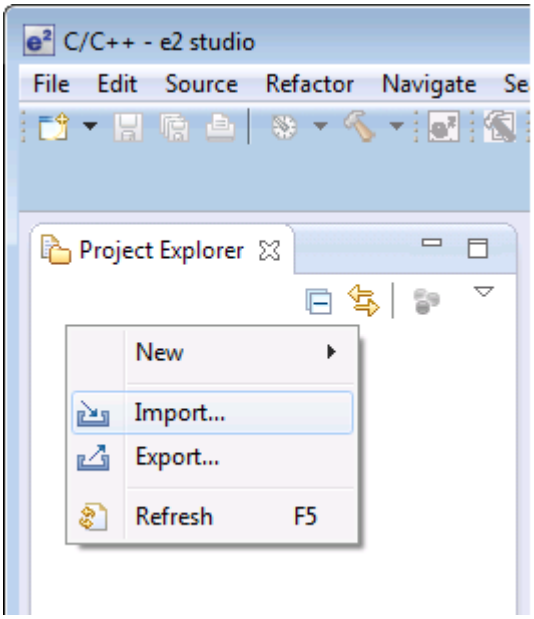
- Start e^2 studio by selecting it from the Windows™ Start Menu. The first dialog box to appear will be the Workspace Launcher.
- Click ‘Browse’ and select a suitable location to store your workspace, using the ‘Create New Folder’ option as necessary. Click ‘OK’.



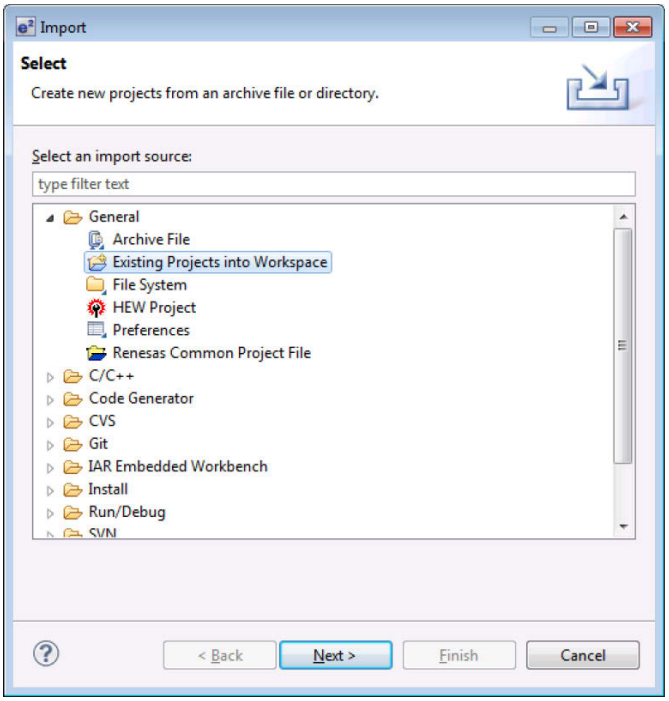
- The e^2 studio welcome splash screen will appear. Click the ‘Go to the workbench’ arrow button on the far right (circled in the screenshot opposite).



- Once the environment has initialized, right click in the ‘Project Explorer’ window and select ‘Import…’



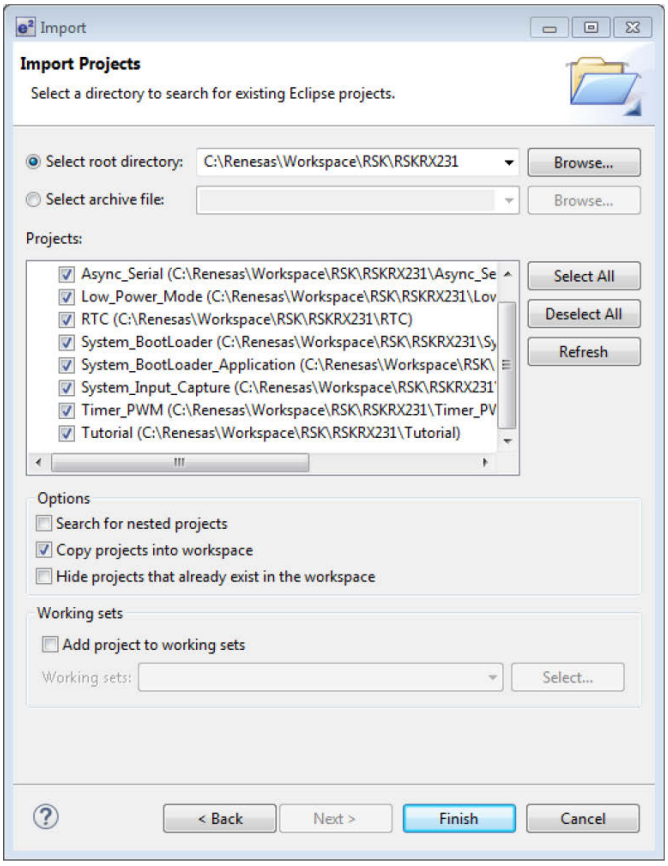
- The Import dialog box will now show. Expand the ‘General’ folder icon, and select ‘Existing Projects into Workspace’, then click ‘Next’.



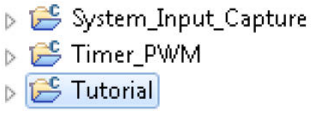
- The Import dialog box will allow you to specify a project to import. Click the 'Browse' button and locate the following directory:

  C:\Renesas\Workspace\RSK\RSKRX231

- 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.



3.4 Build Configurations and Debug Sessions

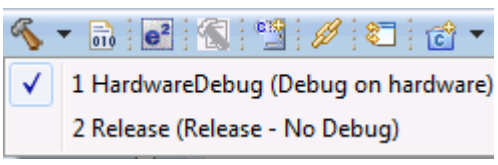
3.4.1 Build Configuration

The e^2 studio workspace will be created with two build configurations: 'HardwareDebug' and 'Release'.

Release
This build mode has optimisation turned on, and provides little debug information. The C code 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 whilst developing code as C code execution will be linear.

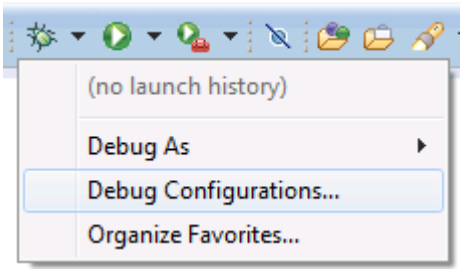
- Click the top level 'Tutorial' folder again, and then the arrow next to the build button (hammer icon), and select the 'HardwareDebug' option.



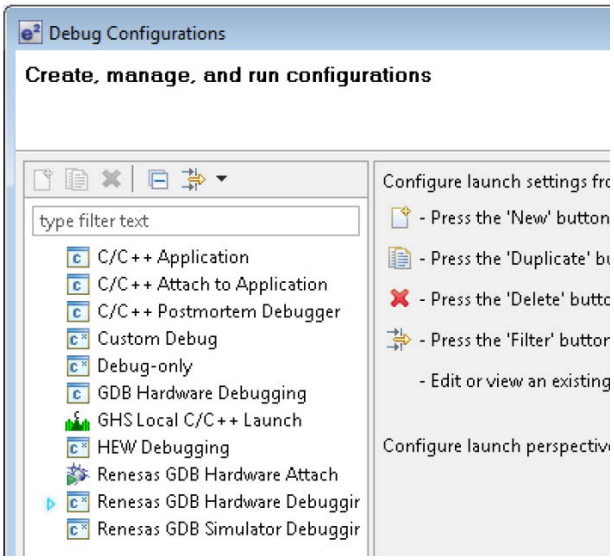
- e^2 studio will now build the code.

3.4.2 Debug Configuration

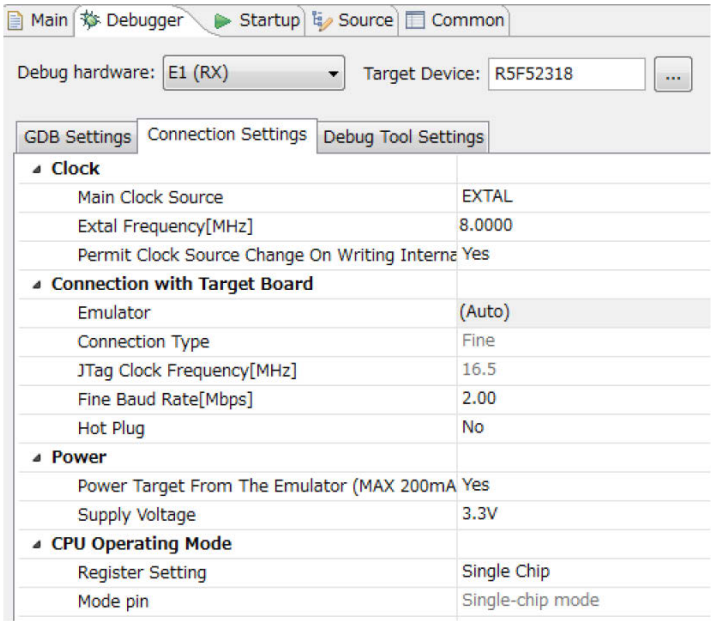
- Click the arrow next to the debug button (bug icon). Select 'Debug Configurations'.



- The 'Debug Configurations' dialog box will appear. Click the small arrow next to the 'Renesas GDB Hardware Debugging' option.
- The debug configurations for each project will appear. Select the entry for the 'Tutorial Hardware Debug'.



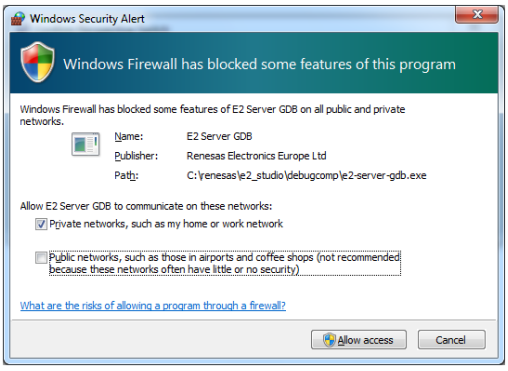
- The debug configurations control page will then show for the Tutorial project. Change the main tab to 'Debugger' and then select 'Connection Settings' on the secondary tab bar that appears.
- There is no need to change the debugger settings as they are preconfigured with the Tutorial project.
- Refer to the RSKRX231 User's Manual for details of power supply configuration.



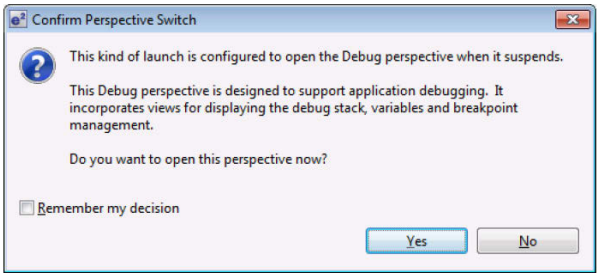
> **Note**: e² studio will display a warning if you attempt to connect with an incorrect power supply setting.

- Click the 'Debug' button to continue. e² studio will now connect to the debugger and download the code to the target.

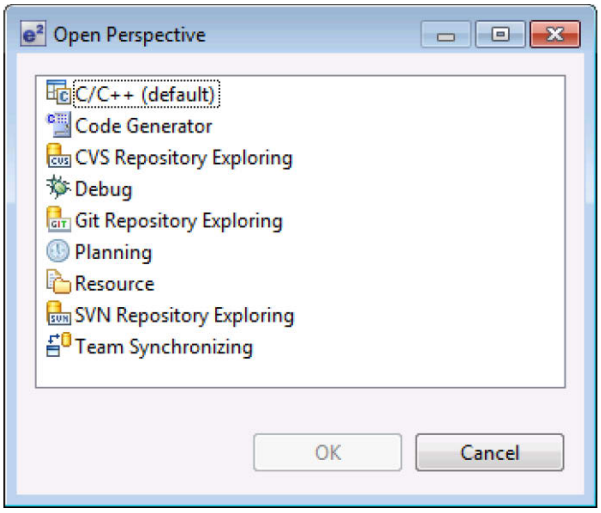
- A firewall warning may be displayed for ‘e2-server-gdb.exe’. Check the ‘Private networks, such as my home or work network’ box and click ‘Allow access’.
- A user account control dialog may be displayed. Enter the administrator password and click ‘Yes’.



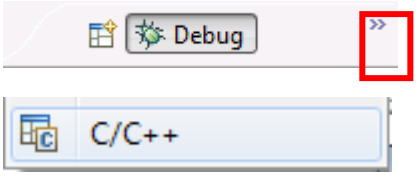
- After downloading the code a dialog box will appear asking if you would like to switch to the ‘Debug perspective’. Click ‘Remember my decision’ to prevent this dialog box from appearing in future, then click ‘Yes’
- e^2 studio will load the new perspective, which is optimised for debugging.



- 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 ‘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.



3.5 Running the Tutorial

- Refer to the description.txt file for instructions on how to configure the RSK and run the sample code.
- Once the code has been downloaded, click ‘Resume’ 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 in the ‘Debug’ perspective to run the rest of the code
- It is recommended that you run the entire tutorial demo first, before continuing to debug it.

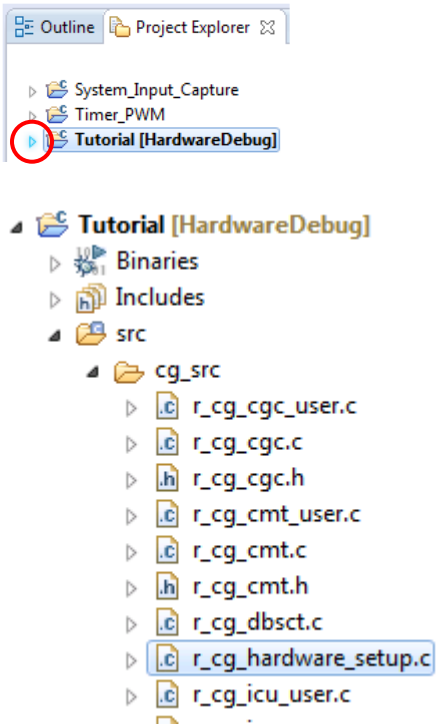
4.Reviewing the Tutorial Program

This section will look at each section of the tutorial code and basic debugging functionality in e^2 studio.

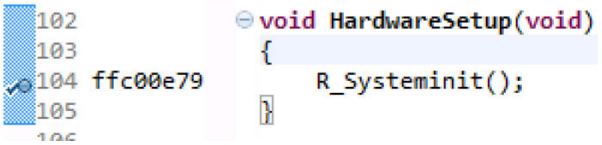
4.1 Program Initialization

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

- Build and download the code as shown in Section 3.3.
- On the Project Explorer tab expand the 'Tutorial' folder by clicking on the arrow next to the folder icon, as highlighted by the red circle.
- Click the arrow next to the 'src' folder to show the source files.
- Expand the 'cg_src' folder in the same way and double click on 'r_cg_hardware_setup.c' to open the file.



- Breakpoints can be set by double clicking at the left-hand edge of the source window. On the line with instruction R_Systeminit(), double click next to the vertical line to set a breakpoint.



Note: As an alternative breakpoints may be set in the C/C++ perspective by selecting a line and using Run > Toggle Breakpoint.

- Click the ‘Resume’ button in the Debug perspective (or press [F8]) to run the code up to this breakpoint.

```
102                 void HardwareSetup(void)
103                 {
104 ffc00e79           R_Systeminit();
105                 }
```

Note: The program counter is indicated by the blue arrow next to the breakpoint.

- Click the ‘Step Into’ button (or press [F5]), to step into the ‘R_Systeminit’ function.

- The ‘R_Systeminit’ function calls several initialization functions which configure the MCU for normal operation. This includes input/output ports, and system clocks.
- The user can step through all the initialization code by clicking the ‘Step Into’ icon and reading the code however for the purpose of this manual, it will be skipped.
- Click the ‘Resume’ button, to run the code up to the main function.

```
55                  /***********************************************************
56                  * Function Name: R_Systeminit
57                  * Description  : This function initializes every macro.
58                  * Arguments    : None
59                  * Return Value : None
60                  ***********************************************************
61 fff81084          void R_Systeminit(void)
62                  {
63                      /* Enable writing to registers related to operating modes,
64 fff81089            SYSTEM.PRCR.WORD = 0xA50FU;
65
66                      /* Enable writing to MPC pin function control registers */
67 fff81092            MPC.PWPR.BIT.B0WI = 0U;
68 fff81099            MPC.PWPR.BIT.PFSWE = 1U;
69
70                      /* Set peripheral settings */
71 fff8109b            R_CGC_Create();
72 fff8109f            R_ICU_Create();
73 fff810a3            R_PORT_Create();
74 fff810a7            R_CMT0_Create();
75 fff810ab            R_CMT1_Create();
76 fff810af            R_CMT2_Create();
77 fff810b3            R_SCI5_Create();
78 fff810b7            R_SCI8_Create();
79 fff810bb            R_S12AD_Create();
80
81                      /* Disable writing to MPC pin function control registers */
82 fff810c4            MPC.PWPR.BIT.PFSWE = 0U;
83 fff810cb            MPC.PWPR.BIT.B0WI = 1U;
84
85                      /* Enable protection */
86 fff810d2            SYSTEM.PRCR.WORD = 0xA500U;
87                  }
```

For further details regarding hardware configuration, please refer to the RSKRX231 User’s Manual and the RX231 Group Hardware Manual.

4.2 Main Functions

This section will look at the program code called from with the main() function, and how it works. It is necessary to connect the RSK G1CUSB0 to a PC USB port and open a terminal emulation program, such as HyperTerminal, with the settings 19200, 8, N, 1. For information on installation of the RSK virtual COM port driver, refer to the file 'description.txt' in the e^2 studio Tutorial project.

- Right click the 'R_SCI5_Serial_Receive()' function call and select 'Run to Line' to execute the program up to this line. The 'R_LCD_Init()' function call enables and configures the LCD panel, and 'R_LCD_Display()' will write "RSKRX231 Tutorial Press Any Switch" onto the LCD.

```
84                  void R_MAIN_UserInit(void);
86                  /* Function Name: main
91  fff811c6        void main(void)
92                  {
93  fff811c8            R_MAIN_UserInit();
94                      /* Start user code. Do not edit comment generated here */
95
96                      /* Initialize the switch module */
97  fff811cb            R_SWITCH_Init();
98
99                      /* Set the call back function when SW1 or SW2 is pressed */
100 fff811cf            R_SWITCH_SetPressCallback(cb_switch_press);
101
102                     /* Initialize the debug LCD */
103 fff811d9            R_LCD_Init();
104
105                     /* Displays the application name on the debug LCD */
106 fff811dd            R_LCD_Display(0, (uint8_t *)" RSKRX231 ");
107 fff811e9            R_LCD_Display(1, (uint8_t *)" Tutorial ");
108 fff811f5            R_LCD_Display(2, (uint8_t *)" Press Any Switch ");
109
110                     /* Start the A/D converter */
111 fff81201            R_S12AD_Start();
112
113                     /* Set up SCI5 receive buffer and callback function */
114 fff81207            R_SCI5_Serial_Receive((uint8_t *)&g_rx_char, 1);
115
116                     /* Enable SCI5 operations */
117 fff81211            R_SCI5_Start();
```

- Set a breakpoint on the 'R_SCI5_Start()' function call by double-clicking in the breakpoint column.
- Click the 'Step Into' button to step into the 'R_SCI5_Serial_Receive ()' function.

```
113                     /* Set up SCI5 receive buffer and callback function */
114 fff81207            R_SCI5_Serial_Receive((uint8_t *)&g_rx_char, 1);
115
116                     /* Enable SCI5 operations */
117 fff81211            R_SCI5_Start();
```

- The program counter should now move into the R_SCI5_Serial _Receive function definition. This function is an API function provided by the Code Generator. It sets up the UART interrupt handler code to receive a specified number of bytes into a receive buffer. Once the specified number of bytes has been received, the interrupt handler code calls a callback function as shown later on in this section.
- For full details on how to configure a project using Code Generator refer to the Code Generator Tutorial Manual.
- Click the 'Resume' button to resume program execution.

```
153 fff819a7    MD_STATUS R_SCI5_Serial_Receive(uint8_t * const rx_buf, uint16_t rx_num)
154             {
155 fff819ad          MD_STATUS status = MD_OK;
156
157 fff819b0          if (1U > rx_num)
158                   {
159 fff819b5              status = MD_ARGERROR;
160                   }
161                   else
162                   {
163 fff819ba              g_sci5_rx_count = 0U;
164 fff819c3              g_sci5_rx_length = rx_num;
165 fff819cc              gp_sci5_rx_address = rx_buf;
166 fff819da              SCI5.SCR.BIT.RIE = 1U;
167 fff819e1              SCI5.SCR.BIT.RE = 1U;
168               }
169
170 fff819e3          return (status);
171 fff819e9    }
```

- The program counter should come to a halt at the R_SCI5_Start function.
- Step over the function by clicking the 'Step Over' button. Alternatively, press [F6].

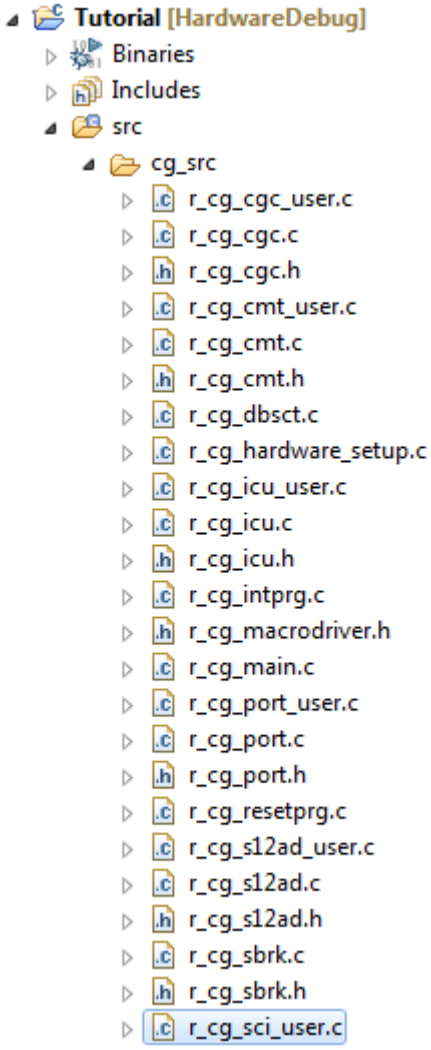
```
113                 /* Set up SCI5 receive buffer and callback function */
114 fff81207        R_SCI5_Serial_Receive((uint8_t *)&g_rx_char, 1);
115
116                 /* Enable SCI5 operations */
117 fff81211        R_SCI5_Start();
```

The R_SCI5_Start() function enables the UART interrupts. The program then proceeds to the main while() loop. The code inside the loop waits for user input from either the UART or RSK switches, and then performs an A/D conversion.

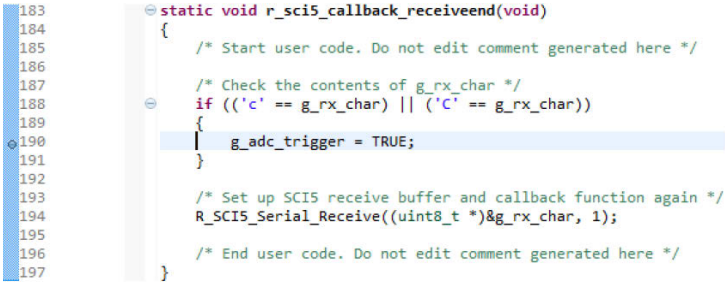
- Locate the function call to 'lcd_display_adc()' inside the while loop.
- Set a breakpoint on the 'lcd_display_adc()' function call by double-clicking in the breakpoint column.

```
120             while (1U)
121             {
122                 /* Variable to store the A/D conversion cou
123                 static uint8_t adc_count = 0;
124
125                 uint16_t       adc_result;
126
127                 /* Wait for user requested A/D conversion f
128 ffc00f86        if (TRUE == g_adc_trigger)
129                 {
130                     /* Call the function to perform an A/D
131 ffc00fa0            adc_result = get_adc();
132
133                     /* Display the result on the LCD */
134 ffc00fa5            lcd_display_adc(adc_result);
```

- In the Project Explorer pane, locate the file ‘r_cg_sci_user.c’ and double-click to open the source file. Scroll down to the function r_sci5_callback_receiveend().



- Set a breakpoint on the line of code inside the r_sci5_callback_receiveend function as shown opposite.
- Continue to execute the program by clicking the ‘Resume’ button.



```
183  static void r_sci5_callback_receiveend(void)
184  {
185      /* Start user code. Do not edit comment generated here */
186
187      /* Check the contents of g_rx_char */
188      if (('c' == g_rx_char) || ('C' == g_rx_char))
189      {
190          g_adc_trigger = TRUE;
191      }
192
193      /* Set up SCI5 receive buffer and callback function again */
194      R_SCI5_Serial_Receive((uint8_t *)&g_rx_char, 1);
195
196      /* End user code. Do not edit comment generated here */
197  }
```

- In the terminal emulation window, press the ‘c’ button on the keyboard.
- The program will halt at the breakpoint in the r_sci5_callback_receiveend function as shown opposite. Remove the breakpoint by double-clicking on the breakpoint column.
- Continue to execute the program by clicking the ‘Resume’ button.
- The program will halt at the breakpoint in the main while loop.
- Remove the breakpoint by double-clicking on the breakpoint column. Continue to execute the program by clicking the ‘Resume’ button.

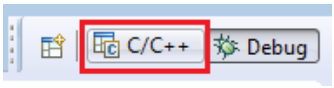
```
230                 /* Check the contents of g_
231 ffc015c1        if (('c' == g_rx_char) || (
232                 {
233 ffc015d3            g_adc_trigger = TRUE;
234                 }
```

```
133                 /* Display the result on the LCD */
134 ffc00fa5       lcd_display_adc(adc_result);
135
136                 /* Increment the adc_count and displ
137 ffc00fa8       if (16 == (++adc_count))
138                 {
139                     adc_count = 0;
140                 }
141 ffc00fb6        led_display_count(adc_count);
```

The program proceeds to display the result of the A/D conversion on the LCD and in the terminal window. In addition, the running count of A/D conversions performed is displayed in binary form using LEDs 0-3 on the RSK. Adjust the potentiometer and press SW1, SW2 or SW3 on the RSK and an additional A/D conversion will be performed.

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

- To change back to the default ‘C/C++’ perspective, from the menu bar select Window > Open Perspective > ‘C/C++’
- Alternatively, click on the ‘C/C++’ button in the top right corner of the screen, as shown opposite.



- This is the extent of the tutorial code.

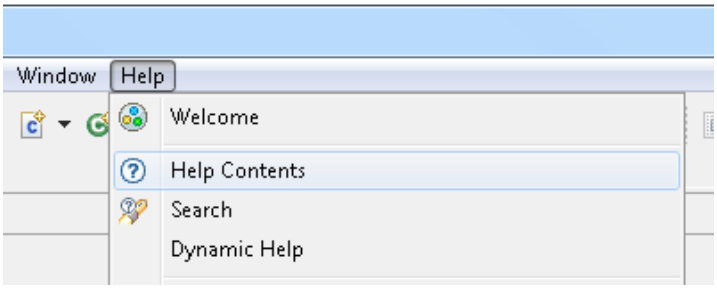
For further details regarding hardware configuration, please refer to the RX Series Software Manual and the RX231 Group Hardware Manual.

The E1 emulator features 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 E1 Emulator User’s Manual

5.Additional Information

Technical Support

For details on how to use e^2 studio, refer to the help file by opening e^2 studio, then selecting Help > Help Contents from the menu bar.



Parts of the sample code provided with the RSKRX231 can be reproduced using the Code Generator tool. Code Generator is included as a plug in with e^2 studio.
Source files and functions generated by Code Generator are prefixed with 'r_' and 'R_', respectively.

For information about the RX231 Group microcontrollers refer to the RX231 Group Hardware Manual.

For information about the RX assembly language, refer to the RX Family Software Manual.

Technical Contact Details

Please refer to the contact details listed in section 9 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	RSKRX231 Tutorial Manual

Rev.	Date	Description	
		Page	Summary
1.00	Aug 24, 2015	—	First Edition issued

Renesas Starter Kit Manual: Tutorial Manual

Publication Date: Rev. 1.00 Aug 24, 2015

Published by: Renesas Electronics Corporation

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.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

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.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +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 2-8175-9670

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 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.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0

RX231 Group

Renesas Electronics Corporation

R20UT3032EG0100