User's Manual

32

RX210 Group

Renesas Starter Kit Tutorial Manual

RENESAS MCU
RX Family / RX200 Series

Renesas Electronics
www.renesas.com

Rev.1.00 Jul 2011

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 tutorial 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 RX210 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.	RSK RX210 User's Manual	R20UT0302EG
Software Manual	Describes the functionality of the sample code, and its interaction with the Renesas Peripheral Driver Library (RPDL)	RSK RX210 Software Manual	R20UT0305EG
Tutorial	Provides a guide to setting up RSK environment, running sample code and debugging programs.	RSK RX210 Tutorial Manual	R20UT0303EG
Quick Start Guide	Provides simple instructions to setup the RSK and run the first sample, on a single A4 sheet.	RSK RX210 Quick Start Guide	R20UT0304EG
Schematics	Full detail circuit schematics of the RSK.	RSK RX210 Schematics	R20UT0301EG
Hardware Manual	Provides technical details of the RX210 microcontroller.	RX210 Hardware Manual	R01UH0037EJ

2. List of Abbreviations and Acronyms

Abbreviation	Full Form
ADC	Analogue to Digital Converter
API	Application Programming Interface
CD	Compact Disk
CPU	Central Processing Unit
E1	E1 Emulator
E20	E20 Emulator
HEW	High-performance Embedded Workshop
LCD	Liquid Crystal Display
LED	Light Emitting Diode
ROM	Read-Only Memory
RPDL	Renesas Peripheral Driver Library
RSK	Renesas Starter Kit
USB	Universal Serial Bus

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 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?

The project generator will create a tutorial project with two selectable build configurations.

- 'Debug' is a project built with the debugger support include.
- 'Release' is a project with optimised compile options, producing code suitable for release in a product.

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 the High-performance Embedded Workshop (HEW) debugger, 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 generator, it is possible that the line numbers for source code illustrated in this document does 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 effect the functionality of the sample code or the validity of this accompanying manual.

3. Tutorial Project Workspace

The workspace includes all of the files for two build configurations, 'Build' and 'Release'. The tutorial code is common to both build configurations; and is designed to show how code can be written, debugged and then downloaded without the debug monitor in a 'Release' situation.

The build configuration menu in High-performance Embedded Workshop (HEW) allows the project to be configured such that certain files may be excluded from each of the build configurations. This allows the inclusion of the debug monitor within the Debug build, and its exclusion in the Release build. Contents of common C files are controlled with defines set up in the build configuration options and `#ifdef` statements within the source files. Maintaining only one set of project files means that projects are more controllable.

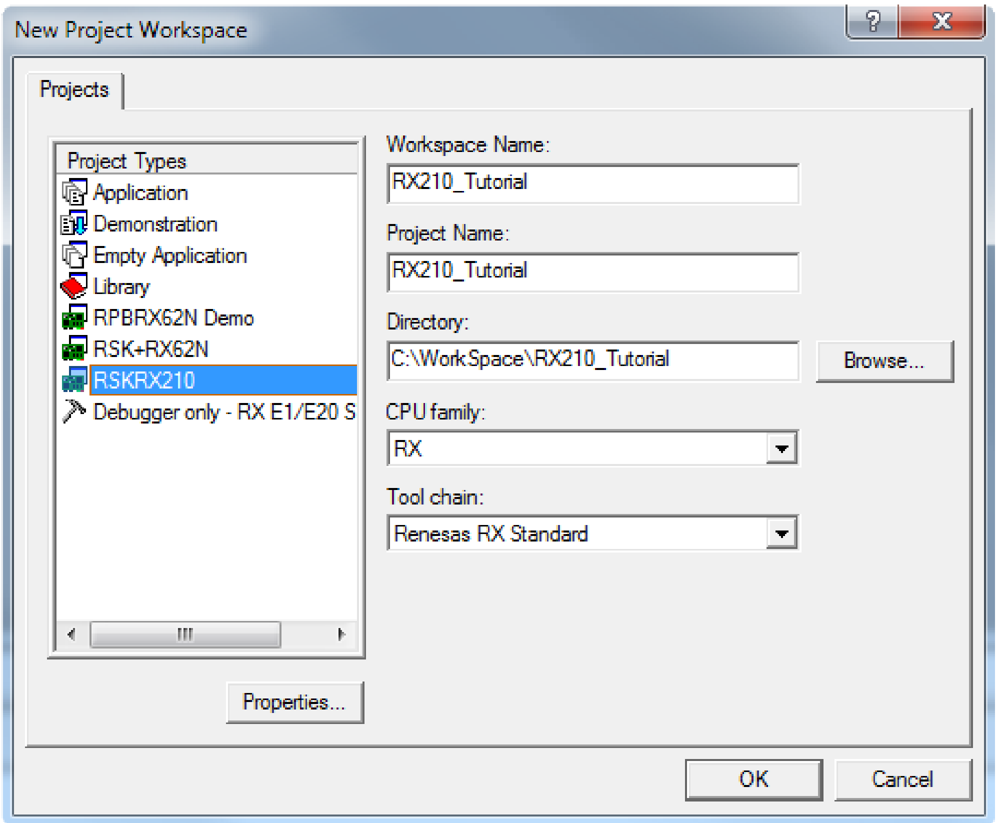
4. Project Workspace

4.1 Introduction

High-performance Embedded Workshop is an integrated development tool that allows the user to write, compile, program and debug a software project on any of the Renesas Microcontrollers. High-performance Embedded Workshop will have been installed during the installation of the software support for the Renesas Starter Kit product. This manual will describe the stages required to create and debug the supplied tutorial code.

4.2 Starting HEW and Connecting the E1 Debugger

To look at the program, start High-performance Embedded Workshop from the Windows Start Menu. Open a new tutorial workspace from the [File > New Workspace…] menu or select 'Create a new project workspace' when presented with the 'Welcome!' dialog.



The example above shows the 'New Project Workspace' dialog with the RSKRX210 selected.

- Select the RX CPU family and 'Renesas RX Standard' toolchain.
- Select the 'RSKRX210' project type from the left-hand projects list.
- Enter a name for the workspace – all your files will be stored under a directory with this name.
- The project name field will be pre-filled to match the workspace name above, but this name may be changed manually.
- Note: High-performance Embedded Workshop allows you to add multiple projects to a workspace. You may add the sample code projects later so you may wish to choose a suitable name for the tutorial project now.
- Click [OK] to start the Renesas Starter Kit Project Generator wizard.

The next dialog presents the three types of example project available:

- Tutorial: this is the one of interest at this time – the code is explained later in this manual.
- Sample Code: This provides examples for using various peripherals. If you select this and click <Next> it will open a new dialog, allowing the selection of many code examples for the peripheral modules of the device.
- Application: where the debugger is configured but there is no program code. This project is suitable for the user to add code without having to configure the debugger.

The project generator wizard will display a confirmation dialog. Press [OK] to create the project and insert the necessary files. A tree showing all the files in this project will appear in High-performance Embedded Workshop.

To view the file 'main.c', double click on the file in the Workspace window. A new window will open showing the code.

4.3 Build Configurations and Debug Sessions

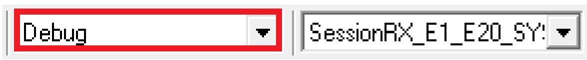
The workspace that has been created contains two build configurations and two debug sessions. The Build Configuration allows the same project to be built but with different compiler options. The options available to the user are described fully in the High-performance Embedded Workshop Manual.

4.3.1 Build Configuration

The build configurations are selected from the left hand drop down list on the tool bar. The options available are Debug and Release. The debug build is configured for use with the debugger. The Release build is configured for final ROM-programmable code.

A common difference between the two builds may be the optimisation settings. With optimisation turned on, the Debugger may seem to execute code in an unexpected order. To assist in debugging it is often helpful to turn optimisation off on the code being debugged.

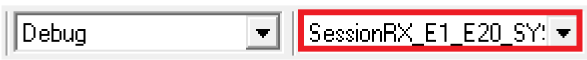
- Select the 'Debug' build configuration



4.3.2 Debug Session

The debug sessions are selected from the right hand drop down list on the tool bar. The options vary between Renesas Starter Kit types however one will always start Debug and include the type of debug interface. The alternate selection will be 'SessionRX_E1_E20_SYSTEM'. The purpose of the debug sessions is to allow the use of different debugger tools or different debugger settings on the same project.

- Select the session:
  "SessionRX_E1_E20_SYSTEM"

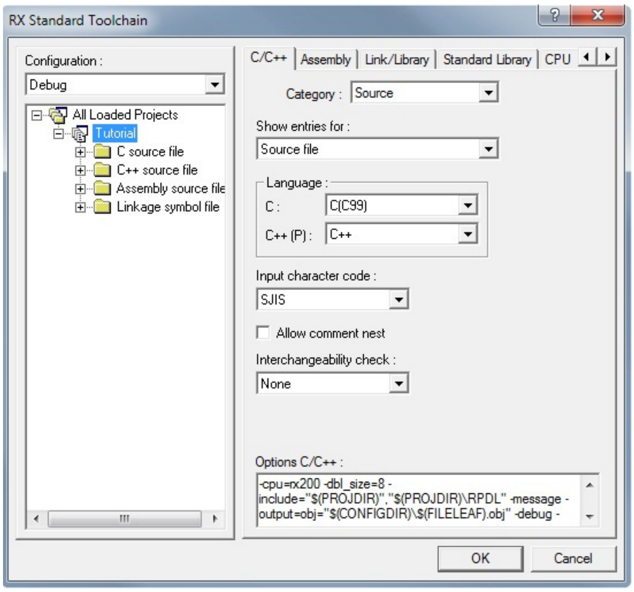


5. Building the Tutorial Program

The tutorial project build settings have been pre-configured in the toolchain options. To view the toolchain options select the 'Build' menu item and the relevant toolchain. This should be the first option on the drop down menu. The dialog that is displayed will be specific to the toolchain selected.

The Configuration pane on the left hand side will exist on all the toolchain options. It is important when changing any setting to be aware of the current configuration that is being modified. If you wish to modify multiple or all build configurations this is possible by selecting 'All' or 'Multiple' from the 'Configuration' drop down list.

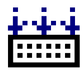
- Review the options on each of the tabs and 'Category' drop down lists to be aware of the options available. For the purposes of the tutorial, leave all options at default.
- When complete close the dialog box by clicking [OK]



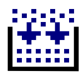
5.1 Building Code

There is a choice of three shortcuts available for building the project.

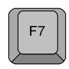
- Selecting the 'Build All' tool bar button. This will build everything in the project that has not been excluded from the build. The standard library is built only once.



- Selecting the 'Build' tool bar button. This will build all files that have changed since the last build. The standard library will not be built unless an option has been changed.



- Pressing [F7]. This is equivalent to pressing the 'Build' button described above.



Build the project now by pressing [F7] or pressing one of the build icons as shown above. During the build each stage will be reported in the Output Window. The build will complete with an indication of any errors and warnings encountered during the build.

5.2 Connecting the Debugger

For this tutorial it is not necessary to provide an external power supply to the board. The power will be obtained from the USB port. Please be aware that if you have too many devices connected to your USB port it may be shut down by Windows. If this happens remove some devices and try again. Alternatively provide an external power source taking care to ensure the correct polarity and voltage.

> Other sample code supplied with this RSK will require a variable power supply; in which case an external 5V variable power supply should be used. Refer to the RSKRX210 User Manual for further details.

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.

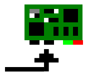
- Fit the LCD module to LCD connector on the board, via the header marked 'LCD'. Ensure all the pins of the connector are correctly inserted in the socket.
- Connect the E1 Debugger to a free USB port on your computer.
- Connect the E1 Debugger to the target hardware ensuring that it is plugged into the connector marked 'E1'.
- If supplying external power to the board, it can be turned on now.

5.3 Connecting to the Target with the E1 Debugger

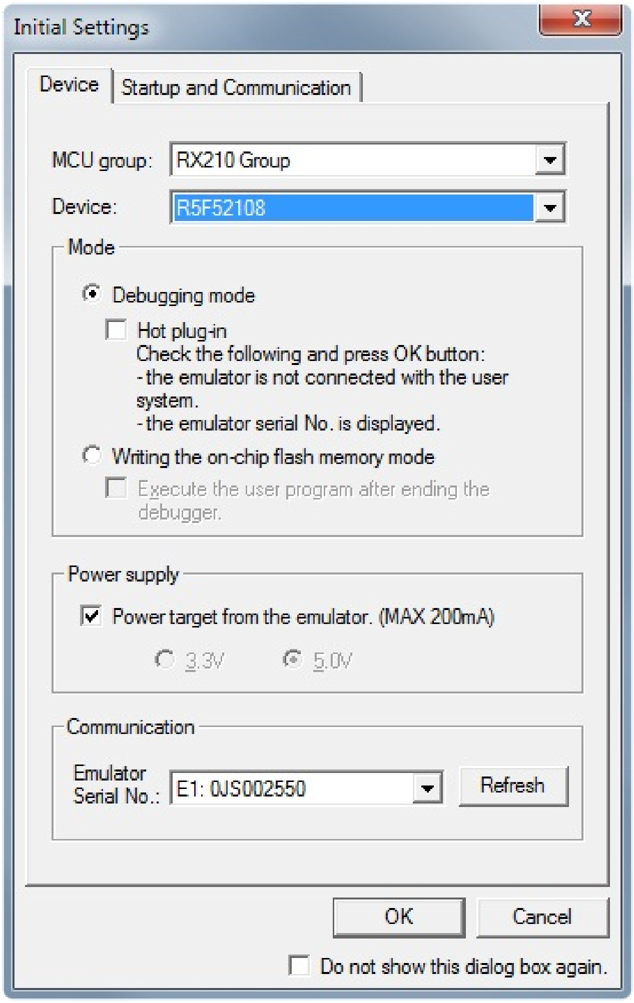
This section will take you through the process of connecting to the device, programming the Flash and executing the code.

> Please note that the "Emulator Mode" wizard shown here will only appear the FIRST time you connect to the target within a project. On subsequent connections the "Emulator Setting" dialog will appear please choose the same options to connect.

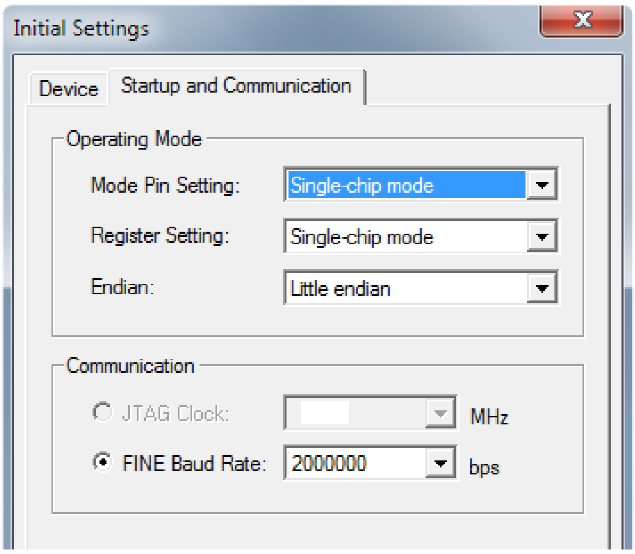
- Select the ‘SessionRX_E1_E20_SYSTEM’ debug session.
- Click the [Connect] button on the debug toolbar.



- The ‘Initial Settings’ configuration dialog will appear. Ensure the follow configurations are set:
  - MCU group: RX210 Group
  - Device: R5F52108
  - Mode: Debugging mode
- If the E1 is to provide power to the CPU board, select ‘Power Target from Emulator’ and choose the “5.0V” option. Otherwise connect a suitable power supply (refer to the RSKRX210 User Manual for details).

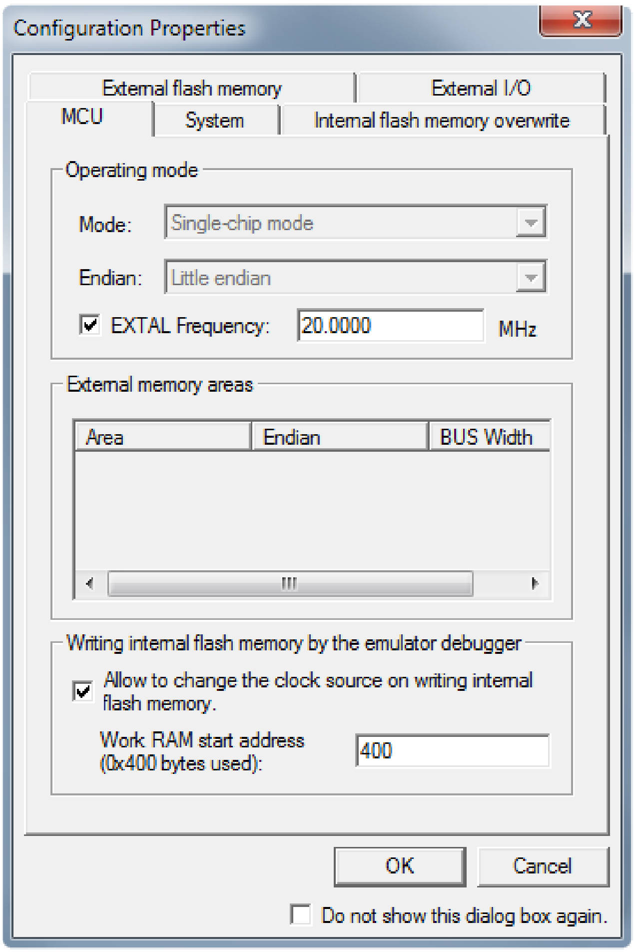


- Click the ‘Startup and Communication’ tab and ensure the Mode Pin, Register & Endian settings match the screenshot opposite. Ensure the FINE Buad Rate is set to 2000000bps. Once these settings have been confirmed, click the [OK] button to continue.
- The Flash Memory write program will be downloaded to the target.



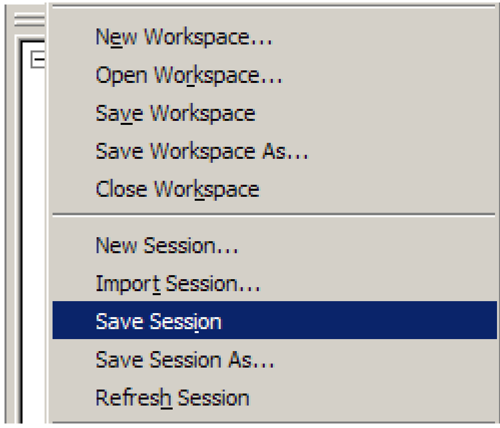
- A connecting dialog will appear, show the status of the connection process. Under default settings, this dialog box will disappear once the connection is complete.

- Once the debugger has connected, the configuration properties dialog will appear.
- Ensure the following configurations are set:
  - — Mode: Single-chip mode
  - — Endian: Little Endian
  - — Input Clock (EXTAL): 20.0000 MHz
  - — Work RAM Start Address: 400
- Once the settings have been reviewed, click [OK] to proceed. The output window in High-performance Workshop will show 'Connected'.
- The connection to the target will activate the debugger buttons on the HEW toolbar. The function of these buttons will be explained in subsequent sections of this tutorial.



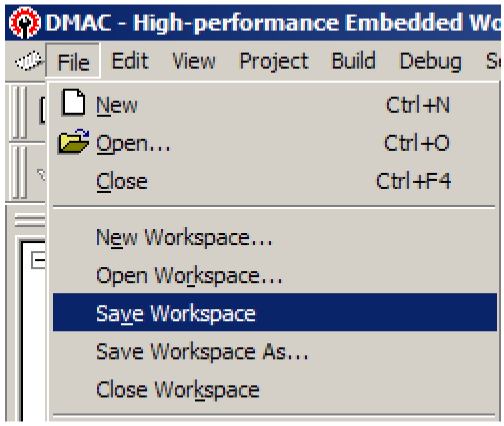
Now is a good time to save the High-performance Embedded Workshop session.

- Select 'File' | 'Save Session'.



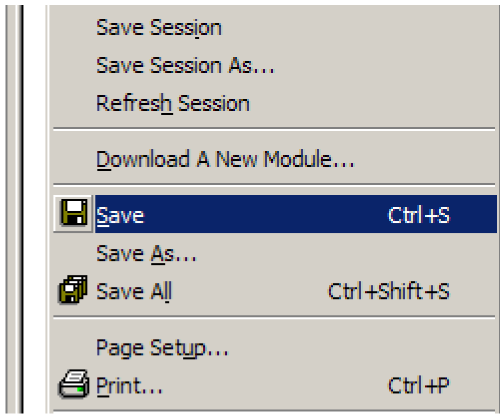
If you have changed any workspace settings now is a good time to save the workspace.

- Select 'File' | 'Save Workspace'.

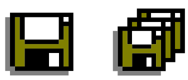


If you make any changes to files in HEW and want to preserve these change, you can save them by:

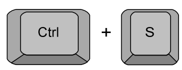
- Select 'File' | 'Save'.



You can also save files by clicking the 'Save' or 'Save All' buttons from the HEW toolbar.



You can also save files using the following keyboard shortcut:



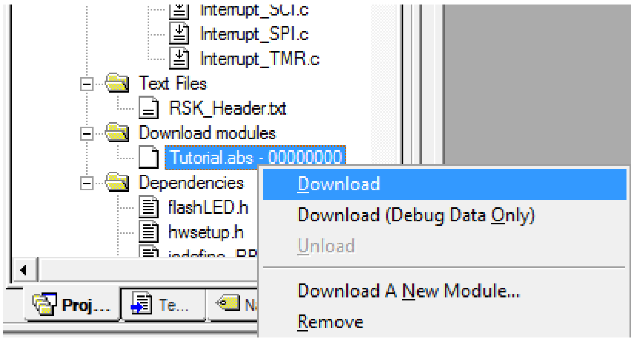
6. Downloading and Running the Tutorial

6.1 Downloading the Program Code

Now the code has been built in HEW it needs to be downloaded to the RSK.

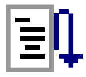
Now that you are connected to the target you should see an additional category in the workspace view called 'Download Modules'

- Right click on the download module listed and select 'Download'
- On completion the debugger and code are ready to be executed



6.2 Running the Tutorial

Once the program has been downloaded onto the RSK device, the program can be executed. Click the 'Reset Go' button to begin the program. It is recommended that you run through the program once first, and then continue to the review section.



7. Reviewing the Tutorial Program

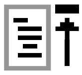
This section will look at each section of the tutorial code, how it works, and how it could be altered to be implemented into more complex code.

> It is recommended that a copy of the RX210 API Manual is made available, as the tutorial program uses RPDL and it is outside the scope of this manual to fully document the API system.

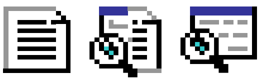
7.1 Program Initialisation

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.

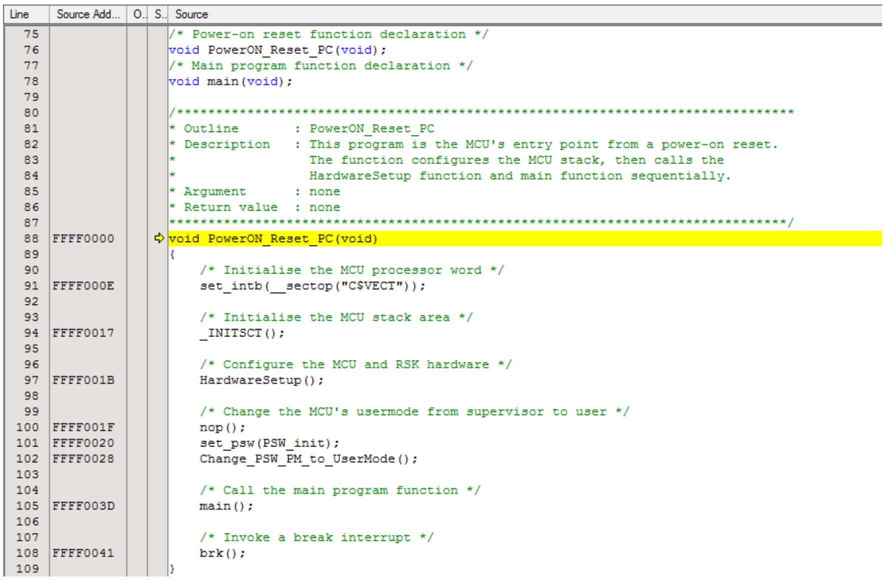
Ensuring the tutorial program has been downloaded onto the RX210, press the 'Reset CPU' button on the Debug Tool Bar.



- The File window will open the Tutorial code at the entry point. An arrow and a yellow highlight marks the current position of the program counter.
- Use these buttons to switch between 'source, disassembly and mixed modes'.

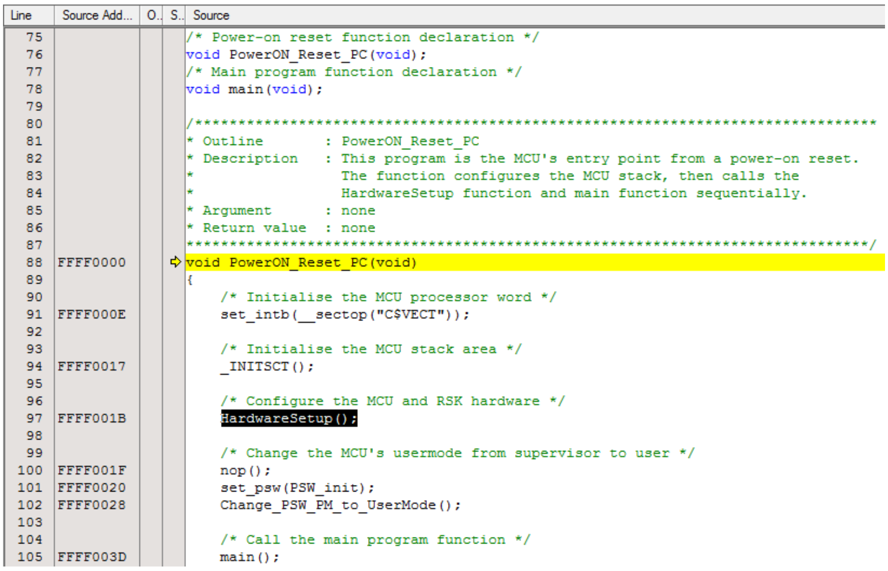


> Ensure the view is switched to 'source' before continuing.

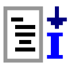


```
Line  Source Add...  O.. S.. Source
 75                          /* Power-on reset function declaration */
 76                          void PowerON_Reset_PC(void);
 77                          /* Main program function declaration */
 78                          void main(void);
 79
 80                          /*******************************************************************************
 81                          * Outline       : PowerON_Reset_PC
 82                          * Description   : This program is the MCU's entry point from a power-on reset.
 83                          *                 The function configures the MCU stack, then calls the
 84                          *                 HardwareSetup function and main function sequentially.
 85                          * Argument      : none
 86                          * Return value  : none
 87                          *******************************************************************************/
 88   FFFF0000            => void PowerON_Reset_PC(void)
 89                          {
 90                              /* Initialise the MCU processor word */
 91   FFFF000E                   set_intb(__sectop("C$VECT"));
 92
 93                              /* Initialise the MCU stack area */
 94   FFFF0017                   _INITSCT();
 95
 96                              /* Configure the MCU and RSK hardware */
 97   FFFF001B                   HardwareSetup();
 98
 99                              /* Change the MCU's usermode from supervisor to user */
100   FFFF001F                   nop();
101   FFFF0020                   set_psw(PSW_init);
102   FFFF0028                   Change_PSW_PM_to_UserMode();
103
104                              /* Call the main program function */
105   FFFF003D                   main();
106
107                              /* Invoke a break interrupt */
108   FFFF0041                   brk();
109                          }
```

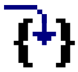
- Highlight the ‘Hardware-Setup()’ function call by left clicking to the right of the text, and holding the left mouse button and dragging over to the left of it and re-leasing the left mouse button.



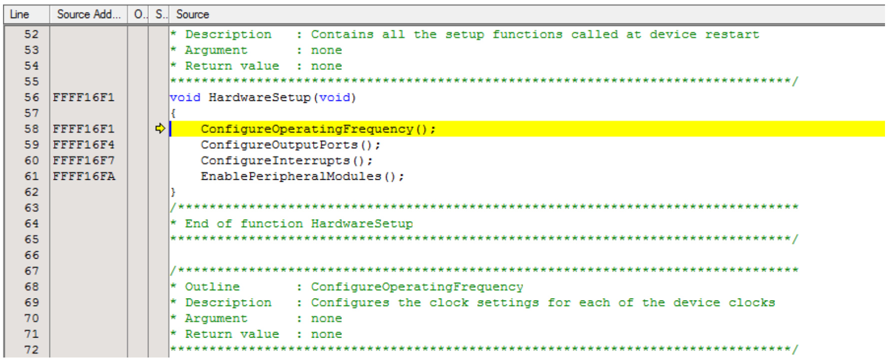
- Click the ‘Go to Cursor’ but-ton to run the program up to this point.



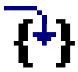
- Click ‘Step In’ to enter the HardwareSetup function.



- The program counter should now move to the Hardware-Setup 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.



- Click ‘Step In’ again to enter the ConfigureOperatingFre-quency function.



- The ConfigureOperatingFrequency function is used to set the speed of the system clocks.
- Several RPDL APIs are called within this function - their exact function can be found by referring to the RPDL manual.
- We will now skip past the hardware setup functions to look at the tutorial’s main program code.

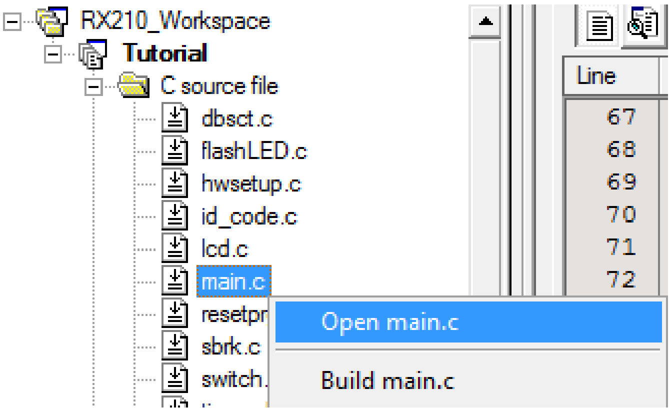
```
/******************************************************************************
* Outline       : ConfigureOperatingFrequency
* Description   : Configures the clock settings for each of the device clocks
* Argument      : none
* Return value  : none
******************************************************************************/
void ConfigureOperatingFrequency(void)
{
    /* Declare error flag */
    bool err = true;

    /* Configure the main clock interface */
    err &=  R_CGC_Set
            (
                PDL_CGC_CLK_MAIN,
                PDL_CGC_BCLK_DIV_1 | PDL_CGC_MAIN_EXTERNAL,
                20E6,
                20E6,
                20E6,
                20E6,
                20E6,
                PDL_NO_DATA
            );

    /* Configure the clocks as follows -
    Clock Description              Frequency
    ----------------------------------------
    System Clock Frequency.............50MHz
    Peripheral Module Clock B..........25MHz
    Peripheral Module Clock D..........50MHz
    FlashIF Clock......................25MHz
    External Bus Clock.................25MHz */

    err &=  R_CGC_Set
            (
                PDL_CGC_CLK_PLL,
                PDL_CGC_BCLK_DIV_2,
                100E6,
                50E6,
                50E6,
                25E6,
                25E6,
                25E6
            );
```

For further details regarding hardware configuration, please refer to the RSKRX210 User’s Manual and the RX210 Hardware Manual.

7.2 Main Functions

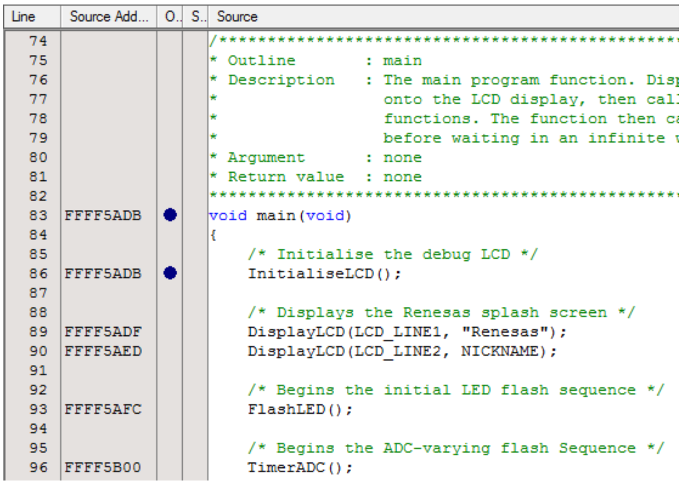
This section will look at the program code called from with the main() function, and how it works.

- Find the main.c file from the file tree on the left hand side, then right click it and select 'Open main.c'



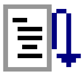
- Place an event at the call to main(); by double clicking in the On-Chip Breakpoint column next to the line to stop at.

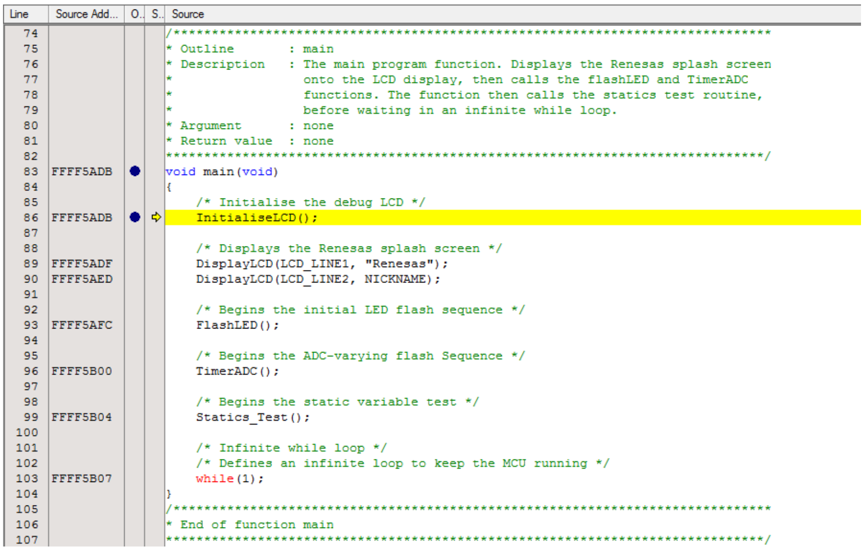
Note that two event points will appear because they share the same source address.



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 RX Family E1/E20 Emulator User's Manual

- Press ‘Reset Go’ on the Debug Tool Bar.

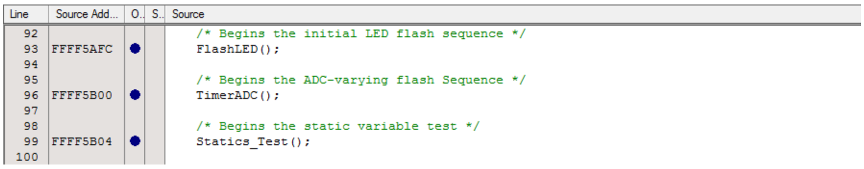




```
/***********************************************************************
* Outline       : main
* Description   : The main program function. Displays the Renesas splash screen
*                 onto the LCD display, then calls the flashLED and TimerADC
*                 functions. The function then calls the statics test routine,
*                 before waiting in an infinite while loop.
* Argument      : none
* Return value  : none
***********************************************************************/
void main(void)
{
    /* Initialise the debug LCD */
    InitialiseLCD();

    /* Displays the Renesas splash screen */
    DisplayLCD(LCD_LINE1, "Renesas");
    DisplayLCD(LCD_LINE2, NICKNAME);

    /* Begins the initial LED flash sequence */
    FlashLED();

    /* Begins the ADC-varying flash Sequence */
    TimerADC();

    /* Begins the static variable test */
    Statics_Test();

    /* Infinite while loop */
    /* Defines an infinite loop to keep the MCU running */
    while(1);
}
/***********************************************************************
* End of function main
***********************************************************************/
```

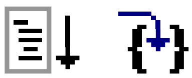
- The code will execute to the event point. At this point all the device initialisation will have been completed. The code window will open ‘main.c’ and show the new position of the program counter.
- Support for the LCD display is included in the tutorial code. We do not need to be concerned about the details of the LCD interface – except that the interface is write-only and so is not affected if the LCD display is attached or not.

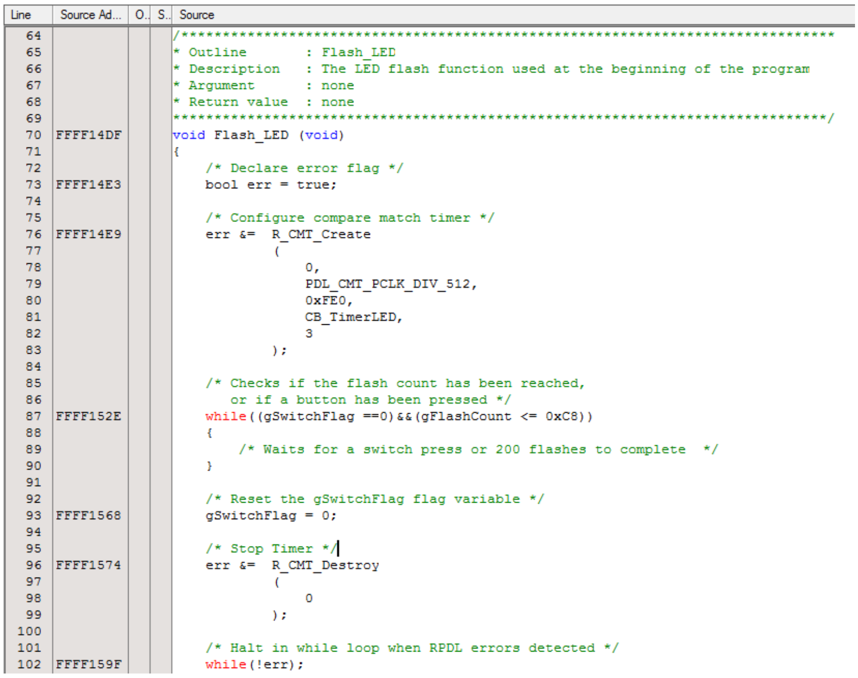
- Insert event points on the FlashLED, TimerADC and Statics_Test function calls.



```
    /* Begins the initial LED flash sequence */
    FlashLED();

    /* Begins the ADC-varying flash Sequence */
    TimerADC();

    /* Begins the static variable test */
    Statics_Test();
```

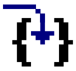
- Press ‘Go’ to run the program up to the FlashLED event point, then press ‘Step In’, to move the program counter to the beginning of the FlashLED function definition.



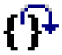


```
/*************************************************************************
* Outline       : Flash_LED
* Description   : The LED flash function used at the beginning of the program
* Argument      : none
* Return value  : none
*************************************************************************/
void Flash_LED (void)
{
    /* Declare error flag */
    bool err = true;

    /* Configure compare match timer */
    err &=  R_CMT_Create
            (
                0,
                PDL_CMT_PCLK_DIV_512,
                0xFE0,
                CB_TimerLED,
                3
            );

    /* Checks if the flash count has been reached,
       or if a button has been pressed */
    while((gSwitchFlag ==0)&&(gFlashCount <= 0xC8))
    {
        /* Waits for a switch press or 200 flashes to complete  */
    }

    /* Reset the gSwitchFlag flag variable */
    gSwitchFlag = 0;

    /* Stop Timer */
    err &=  R_CMT_Destroy
            (
                0
            );

    /* Halt in while loop when RPDL errors detected */
    while(!err);
```

- The FlashLED function uses RPDL functions to create a periodic CMT callback, which toggles the LEDs at regular intervals.
- The ‘while’ statement checks the value of the gFlashCount variable, which counts down with every LED flash. Once this reaches zero the function destroys the CMT timer and exits the infinite ‘while(1)’ loop.

- Click 'Go' to resume the program, then push any switch to proceed. The program should halt at the event point set on the TimerADC function call.
- Press 'Step In' twice to step into the StartTimer function. 
- The StartTimer function configures the timer used to periodically flash the LEDs.

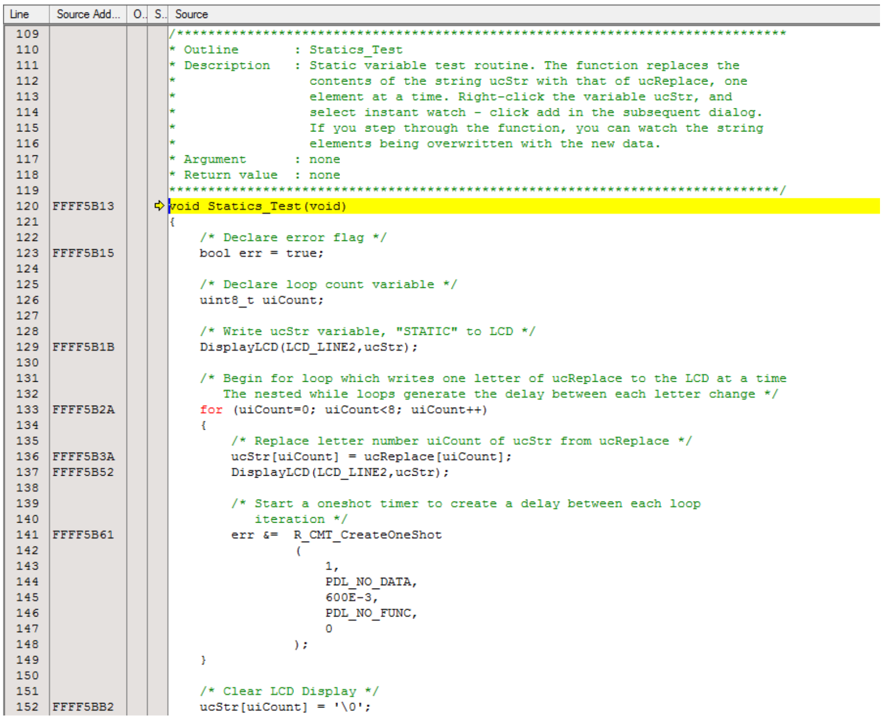
```
/*******************************************************************************
* Outline       : StartTimer
* Description   : Configures CMT channel 0 to call the CB_TimerADC callback
*                 function which starts the AD conversion
* Argument      : none
*******************************************************************************/
void StartTimer(void)
{
    /* Declare error flag */
    bool err = true;

    /* Configure compare match timer */
    err &=  R_CMT_Create
            (
                0,
                PDL_CMT_PCLK_DIV_512,
                0x03FF,
                CB_TimerADC,
                3
            );

    /* Halt in while loop when RPDL errors detected */
    while(!err);
}
```

- Press 'Step Out' button to exit the StartTimer function, then press 'Step In'. The program should now reach the StartADC function 
- The StartADC function configures the ADC unit to make repeat conversions of the voltage from the potentiometer RV1.

```
/*******************************************************************************
* Outline       : StartADC
* Description   : Initialises the ADC's channel 0 for oneshot operations and set
*                 the callback function.
* Argument      : none
* Return value  : none
*******************************************************************************/
void StartADC(void)
{
    /* Declare error flag */
    bool err = true;

    /* Set analog channel AN000 */
    err &=  R_ADC_12_Set
            (
                PDL_ADC_12_PIN_AN000_P40
            );

    /* Configure the ADC in single mode, software triggered */
    err &=  R_ADC_12_CreateUnit
            (
                0,
                PDL_ADC_12_SCAN_SINGLE |\
                PDL_ADC_12_ADSSTR_CALCULATE | PDL_ADC_12_ADSHCR_CALCULATE,
                PDL_NO_DATA,
                PDL_NO_DATA,
                5E-6,
                3E-6,
                0,
                CB_ADConversion,
                6,
                PDL_NO_FUNC,
                PDL_NO_DATA
            );

    /* Configure the ADC channel AN000 */
    err &=  R_ADC_12_CreateChannel
            (
                0,
                0,
                PDL_ADC_12_CH_GROUP_A|\
                PDL_ADC_12_CH_SAMPLE_AND_HOLD_ENABLE,
                5E-6
            );
```

- Press F5 to resume the code, where it will then halt at the break point on the Statics_Test function call.

- Press F11 to step into the function.



- The Statics_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 'Go' or press F5 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 'RX210'.

- This is the extent of the tutorial code. For further information on the RPDL function calls used in the tutorial sample, please refer to *Renesas Peripheral Driver Library User's Manual*.

8. Additional Information

Technical Support
For details on how to use High-performance Embedded Workshop (HEW), refer to the HEW manual available on the CD or from the web site.

For information about the RX210 series microcontrollers refer to the RX210 Group hardware manual.

For information about the RX210 assembly language, refer to the RX200 Series Software Manual.

Online technical support and information is available at: http://www.renesas.com/rskrx210

Technical Contact Details
America: techsupport.america@renesas.com
Europe: tools.support.eu@renesas.com
Japan: csc@renesas.com

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	RSKRX210 Tutorial Manual

Rev.	Date	Description	
		Page	**Summary**
1.00	Jul 30, 2011	—	First Edition issued

Renesas Starter Kit Tutorial Manual

Publication Date: Rev. 1.00 Jul 30, 2011

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.
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: +49-211-65030, 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.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
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

RX210 Group

Renesas Electronics Corporation

R20UT0303EG0100