User's Manual

32

RZ/A1H Group

Renesas Starter Kit+ FreeRTOS Integration Manual

RENESAS MCU
RZ Family / RZ/A1H Series

Renesas Electronics
www.renesas.com

Rev. 1.00 Jun 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.

(2012.4)

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins
   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.
   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

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 evaluate the FreeRTOS sample application provided. It is intended for users designing sample code on the RSK platform, using the many different incorporated peripheral devices.

The manual comprises instructions to load and debug the enhanced FreeRTOS project in e^2 studio, but does not intend to be a complete guide on the FreeRTOS software, or software development on the RSK platform. Further details regarding operating the RZA1H microcontroller may be found in suite of Renesas documentation.
Further details on the FreeRTOS software and documentation available online.

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 form part of the suite of Renesas documentation and apply specifically to the RZA1H 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.
Quick Start Guide	Provides simple instructions to setup the RSK+ and run the first sample.	RSK+RZA1H Quick Start Guide	R20UT3006EG
FreeRTOS Quick Start Guide	Provides simple instructions on integrating one of Renesas Samples into the pre-configured FreeRTOS project.	RSK+RZA1H FreeRTOS Quick Start Guide	RXXUTXXXXEG
User's Manual	Describes the technical details of the RSK+ hardware.	RSK+RZA1H User's Manual	R20UT3007EG
Tutorial	Provides a guide to setting up RSK+ environment, running sample code and debugging programs.	RSK+RZA1H Tutorial Manual	R20UT3008EG
Schematics	Full detail circuit schematics of the RSK+.	RSK+RZA1H Schematics	R20UT2586EG
Hardware Manual	Provides technical details of the RZA1H microcontroller.	RZA1H Group Hardware Manual	R01UH0403EJ

2. List of Abbreviations and Acronyms

Abbreviation	Full Form
ADC	Analog-to-Digital Converter
e^2 studio	Renesas *Eclipse Embedded Studio* Integrated Debugging Environment
EMC	Electromagnetic Compatibility
ESD	Electrostatic Discharge
J-LINK	On-chip Debugger
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MCU	Micro-controller Unit
QSPI	Quad Serial Peripheral Interface
RSK	Renesas Starter Kit
RSK+	Renesas Starter Kit + (denotes extra functionality over standard RSK)
RTOS	Real Time Operating System

Table of Contents

1. Overview

1.1 Purpose

The FreeRTOS project was conceived to provide an embedded, free, RTOS solution that was easy to use, build and deploy. The solution can be cross compiled on a windows PC deployed onto target microcontroller, and has been provided by Real Time Engineers Ltd.

This document describes the FreeRTOS demonstration application RZ_RSK_FreeRTOS (known as CommonCore) that is used as the host for the Renesas software sample modules. Renesas has chosen to adopt this approach to allow them to deploy discrete sample code that can be mixed and matched as the solution demands. On top of the standard FreeRTOS

The bare bones FreeRTOS project, officially supported is available on the website. This version does not provide the support required to run the renesas samples.

1.2 Features of CommonCore project

This FreeRTOS project provides the following features :

- RSK+RZA1H board support.
- LED driver layer.
- PMOD display driver (PMOD connected to CN25/CN26)
- Serial console (connected via USB mini-port CN18)
- I2C driver layer (use to activate the Port Expanders on the RSK board).
- Real-time Clock driver layer.

2.Introduction

This manual is designed to provide guide to the Renesas CommonCore FreeRTOS project (referred to as CommonCore project). The document is arranged into the following sections

- How to build the CommonCore project
- How to test my project
- How do the samples interact
- How does the CommonCore project fit with FreeRTOS

Files referred to in this manual are installed using the import wizard 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.

This manual is not intended as a comprehensive introduction to the e^2 studio environment, compiler toolchains, RSK or the J-Link LITE debugger. Please refer to the relevant user manuals for more in-depth information.

3. CommonCore Project Workspace

3.1 Introduction

e^2 studio is an integrated development tool that allows the user to write, compile, program and debug a software project on the RZ family of Renesas microcontrollers. e^2 studio will have been installed during the installation of the software support for the Renesas Starter Kit product. e^2 studio has been used to provide the build environment for the FreeRTOS CommonCore project.
This manual will describe the stages required to create an instance of the CommonCore project. It is strongly advised that customers never modify the original instance of the CommonCore project and always use a copy.

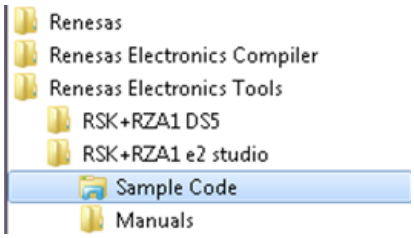
3.2 Jumper and Switch Configuration

Ensure that the RSK is in its default state, please refer to the schematic if you have changed any of the links to return the board to its default state. The jumpers and switches settings are in the following positions:

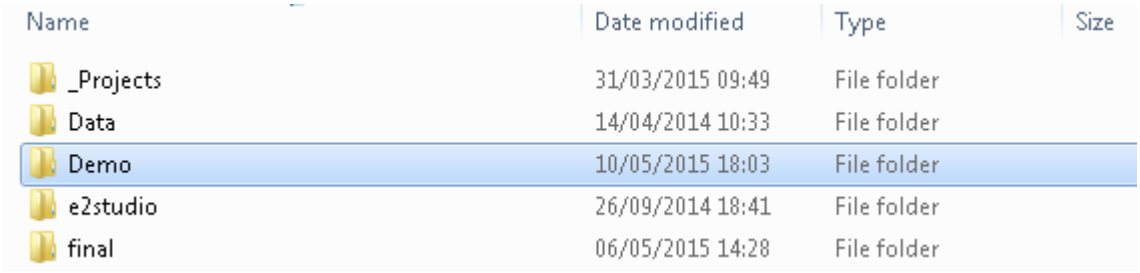
Jumper	Pins	Switch	Position	Switch	Position
JP11	1 - 2	SW4-1	OFF	SW6-1	OFF
JP12	1 - 2	SW4-2	OFF	SW6-2	ON
JP18	1 - 2	SW4-3	OFF	SW6-3	OFF
JP21	1 - 2	SW4-4	OFF	SW6-4	ON
PWR_SEL	2 - 3	SW4-5	OFF	SW6-5	ON
--	--	SW4-6	OFF	SW6-6	ON
--	--	SW4-7	OFF	--	--
--	--	SW4-8	OFF	--	--

3.3 Copying the CommonCore FreeRTOS project

- Open the Renesas sample code by selecting the folder from the Start Menu -> All Programs -> Renesas Electronics Tools -> Renesas e^2 studio -> Sample Code.

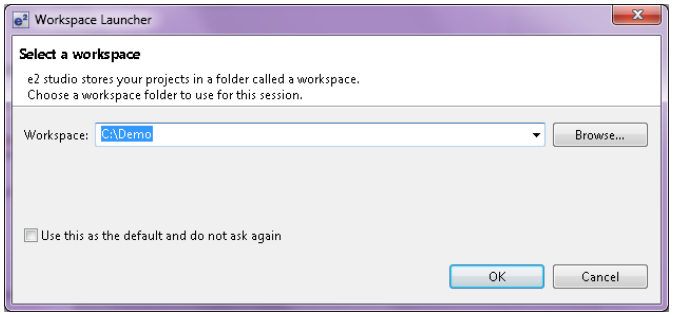


- Open the FreeRTOS folder and copy the zipfile to your desired workspace location.
- Extract the Zip file.
- Click ‘Browse’ and select a suitable location to store your workspace, using the ‘Create New Folder’ option as necessary. Click ‘OK’.

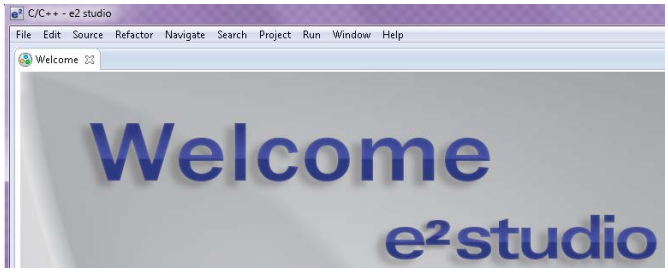


3.4 Starting e² studio and Importing Sample Code

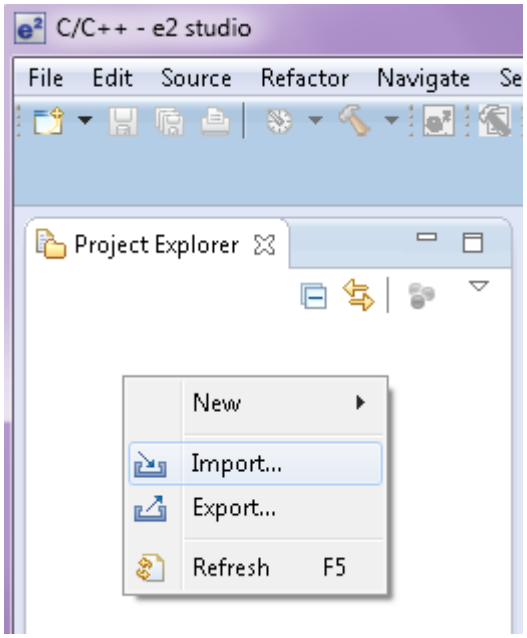
- Start Eclipse for e² studio by selecting it from the Start Menu -> All Programs -> Renesas Electronics e² studio 4.01 -> Renesas e² studio. The first dialog box to appear will be the Workspace Launcher.
- Click 'Browse' and select the desired location to store your workspace. Click 'OK'.



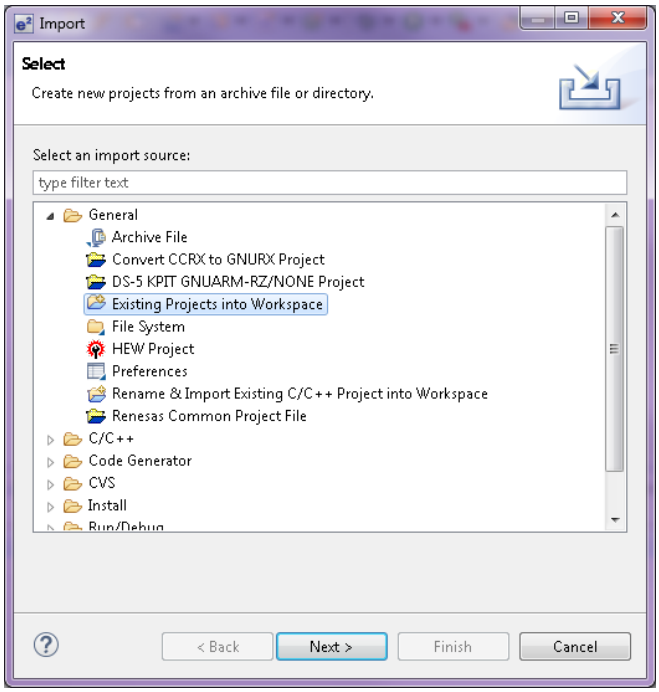
- e² studio will open with the 'Welcome…' tab as shown opposite.
- Close the tab by clicking on the ✖ cross.



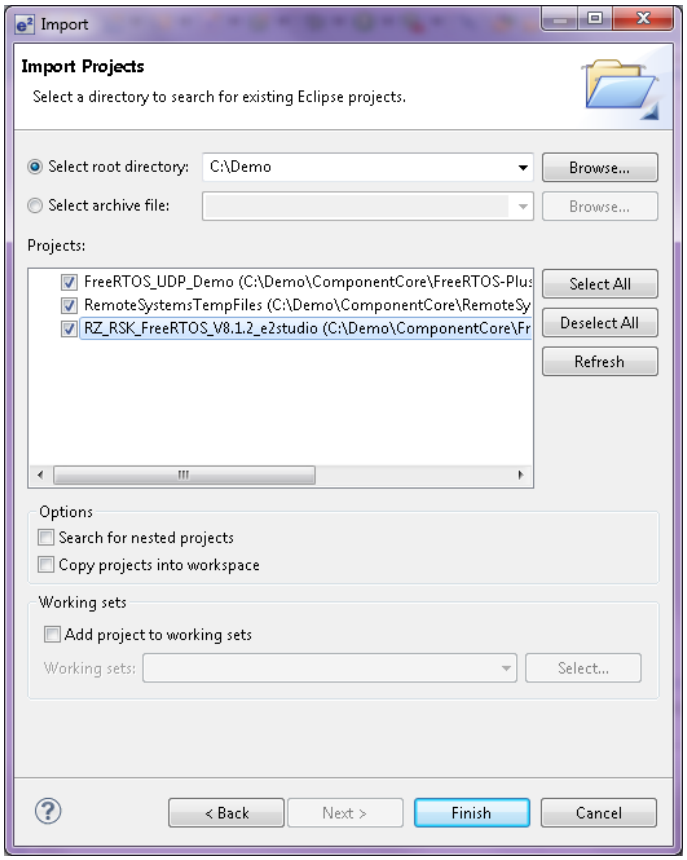
- Right click in the 'Project Explorer' window and select 'Import…'



- 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 your desired directory:
- Select the project
  RZ_RSK_FreeRTOS_V8.1.2_e2studio
- Ensure that the ‘Copy projects into workspace’ option is un-ticked.
- Caution: Ticking this box will copy the projects from the location where they were installed. It is important NOT to select this option.
  This is because the FreeRTOS build structure include paths that exist outside of the current project and e2sudio copy project function will break the links.
- Click ‘Finish’.



3.5 Build Configurations and Debug Sessions

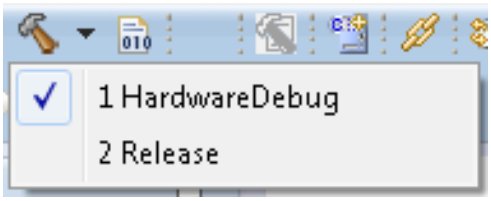
3.5.1 Build Configuration

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

HardwareDebug
This default 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. The ‘HardwareDebug’ build configuration provided for this Tutorial program is configured to load the code directly into RAM.

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

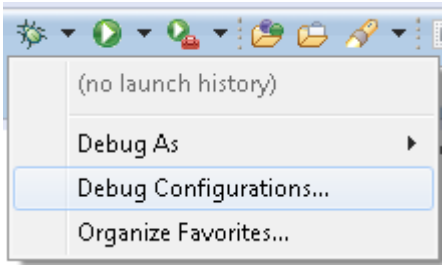
e^2 studio will now build the code.



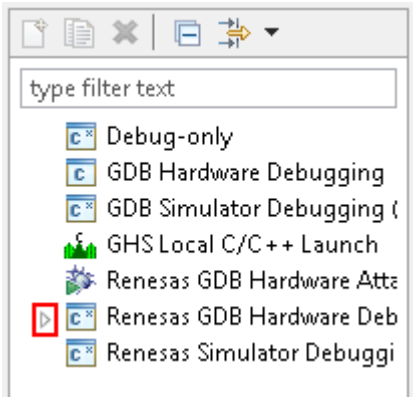
- The output from the build process will be presented in the console window of e^2 studio.

3.5.2 Debug Configuration

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



- The ‘Debug Configurations’ dialog box will appear. Click on the button next to ‘Renesas GDB Hardware Debugging’ to expand the view.
- By default, e2 studio creates Debug Configurations for each existing build mode. The RSK+RZA1H project have pre-configured debug configurations that are ready to use.

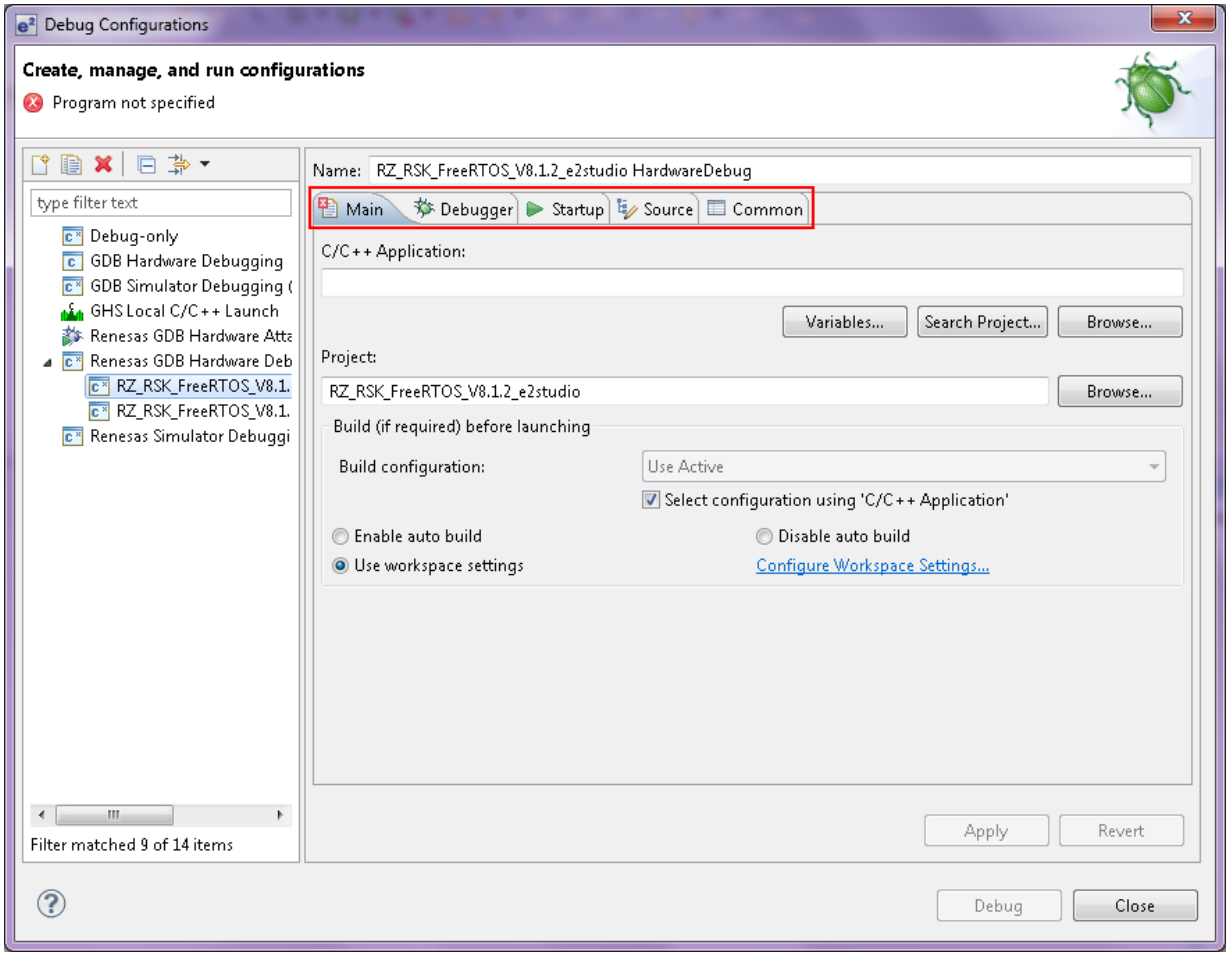


Note:
To manually create a new e2 studio debug configuration, click on ‘Renesas GDB Hardware Debugging’ then click on the ‘New’ button  .

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

- The debug configurations control page will open. Observe the settings under each tab.
- Under the ‘Debugger’ tab, ensure the ‘Debugger hardware’ option is set to ‘J-Link ARM’.
- The ‘Target Device’ is preset to ‘R7S721001’.

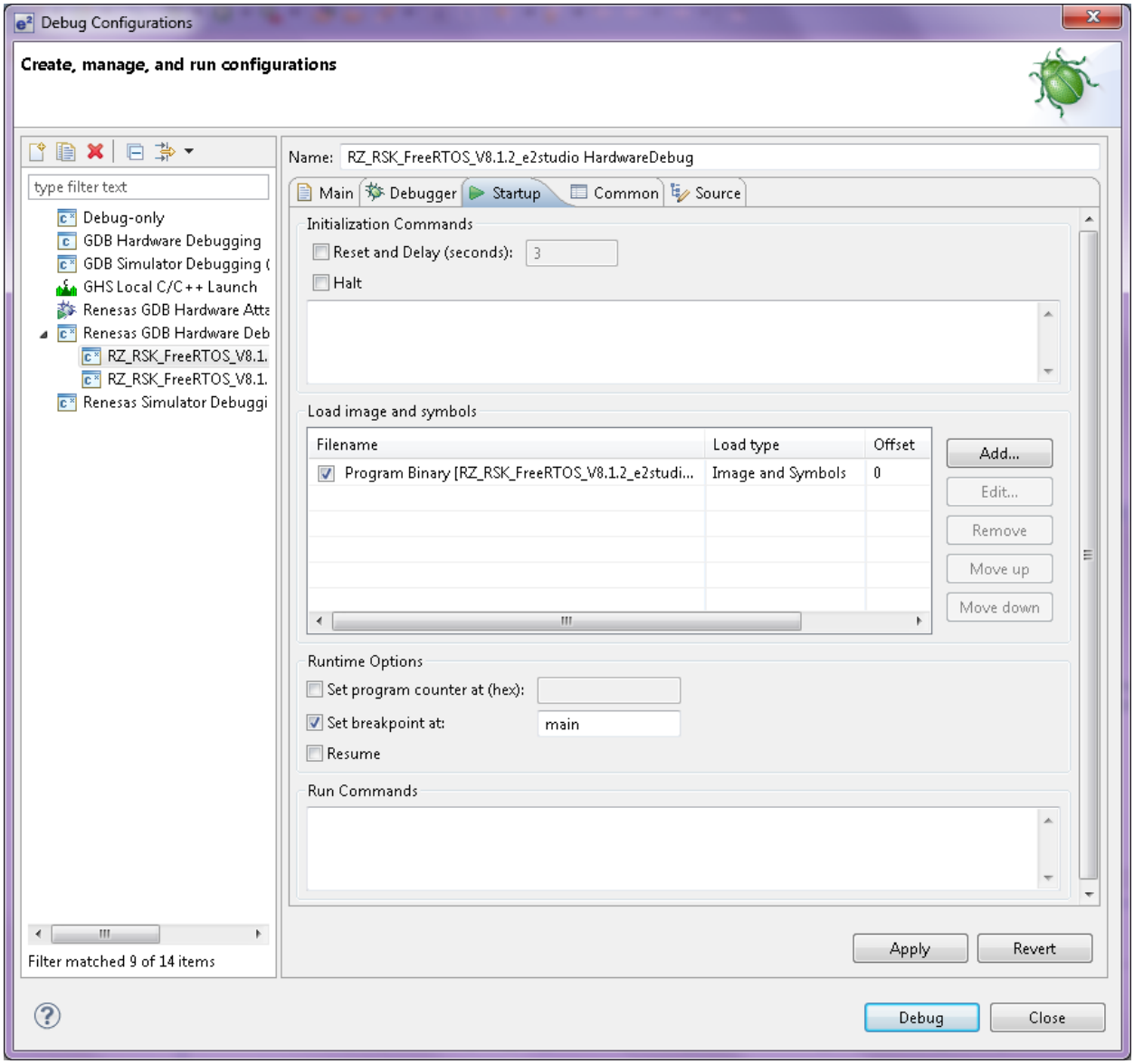
Note:
Do **NOT** modify any settings.



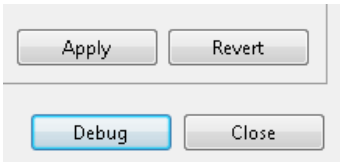
- A security dialog may appear indicating that the Windows Firewall has blocked some features of the eclipse platform.
- Under the text ‘Allow eclipse to communicate on these networks’, ensure the check box next to ‘Private networks, such as my home or work network’ is ticked. Click ‘Allow access’

- Select the ‘Startup’ tab.

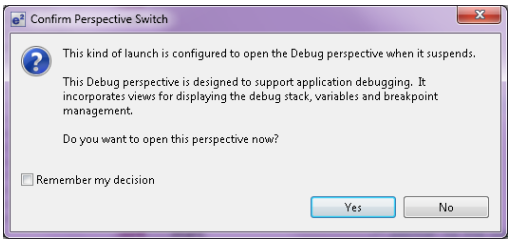
- Ensure the ‘Runtime Option’ ‘Set breakpoint at:’ is specified as ‘main’.



- Click on ‘Debug’.



- Before downloading the code a dialog box will appear asking if you would like to switch to the ‘e^2 studio Debug perspective’. If you agree 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 Perpsective’ dialog box will appear. Click on the desired perspective to select it then ‘OK’.
- Alternatively, click on the button within the top right corner of the screen, as shown opposite, and select the ‘C/C++’ perspective.

4. Testing CommonCore project

This section will cover basic testing of the CommonCore project, and covers

- Locating sample software
- Connecting to the target board
- Interacting with the software

4.1 Locating sample software

Once software modules have been added to CommonCore sample project they are place into the Sample folder:

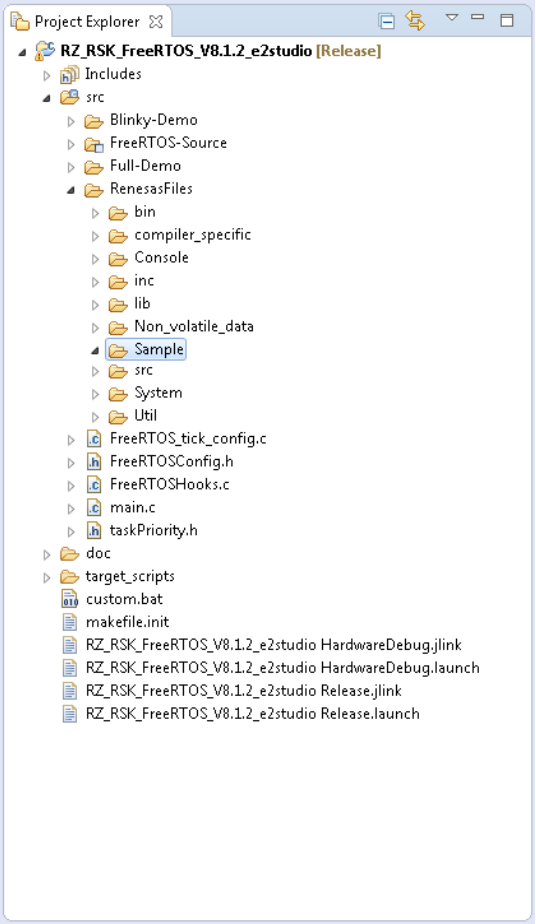
ComponentCore\FreeRTOS\Demo\CORTEX_A9_RZ_R7S72100_e2studio\src\RenesasFiles\Sample

By default no supplemental samples are installed so this folder is empty.

Samples can be added using the Renesas SDC Module wizard located in 'File -> New' Menu.

For detailed instructions please refer to the quick start guide

r20ut3490eg0100_rsk+rza1h_FreeRTOS_qsg.pdf



4.2 Connecting to target board

When the CommonCore project is downloaded to the target board and the device is in execution the software interacts with the user in a number of ways.

- On board LEDs (LED0,LED1,LED2,LED3) on the target device
- Serial output via USB Mini-B connector (CN18)
- PMOD display (supplied with RSK kit) connected to either PMOD1, or PMOD2.

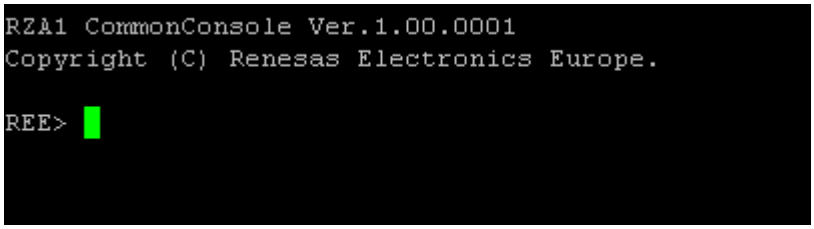
4.2.1 On board LED's

Applications can use the 4 on board LED's to indicate status. The LED's are not used in the samples provided to allow users to customise their use in an application.

4.2.2 Serial output via USB Mini-B

Serial Console output is provided via the RL78G1C USB to Serial converter connected to CN18 on the RSK. The serial connection required is **Baud** 115200, **Data bits** 8, **Stop bits** 1, **Parity** None, **Flow Control** None. By installing the RSK the required software for RSK serial port should already be available and selecting whichever com port maps to 'RSK USB Serial Port' should be used to establish the connection.

Once connection is established and the CommonCore project is downloaded to the RSK and the device is running the following should be shown on the console (note version updates on each build):



4.2.3 PMOD display

Components can use the PMOD display (supplied with the RSK) to provide feedback to the customer. (Example below)

With the default configuration of the CommonCore project the PMOD display is used to indicate the that software is running successfully, once FreeRTOS startup has completed and the scheduler is running a single use task is created to display a welcome message on the PMOD display, once its job has completed and the message is shown the task deletes itself, This is to allow other software to use the display in any way they see fit.

The configuration uses two functions to utilise the PMOD display, (see Removing the PMOD sample commands)
Function '`lcd_pmod_init()`' initialises the pmod interface and MUST BE CALLED prior to any use of the PMOD display. This function is not optional if PMOD use is desired.

Function '`R_PMOD_Sample_Main()`' adds the PMOD demonstration commands

- pmodstart - Run PMOD sample
- pmodstop - Terminate PMOD sample
- pmodrotate - Rotate image on PMOD display by 90 degrees clockwise.

Which provide a simple demonstration of the PMOD use.

Command pmodstart
Initialises a sequence of images to be shown on the display with about 1 second delay between each image.

The images display in the following order



Desert → Hydrangeas → Penguins

Command pmodstop
Terminates the pmodstart command.

Command pmodrotate
Rotates PMOD output 90 degrees clockwise, note that the associated function '`R_LCD_ImageRotation()`' can be used by customised code to rotate the pmod output at any time.

The display rotates in the following order:




4.2.3.1 Removing the PMOD sample commands

The PMOD sample commands can be removed from the CommonCore project by removing the function call '`R_PMOD_Sample_Main()`' in the file '`RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/main.c`'
Simply remove the line marked with a red box, to remove the commands from the console, though the PMOD interface shall still be available to use in code.

```
/******************************************************************************
Function Name: main
Description:   Main entry point of the application
Arguments:     none
Return value:  none
******************************************************************************/
int main(void)
{
    IOSTR       p_iostr[3];

    /* Assign the streams for the serial console */
    p_iostr[0].pIn = stdin;
    p_iostr[0].pOut = stdout;

    fprintf(p_iostr->pOut,"\r\n");
    fflush(p_iostr->pOut);

    r_RIIC3_Initialise();

    /* Initialise PMOD */
    lcd_pmod_init();
    R_PMOD_SampleMain(p_iostr->pIn,p_iostr->pOut);

    /* START INSERT SAMPLE MAIN FUNCTION CALLS HERE */

    /* END INSERT SAMPLE MAIN FUNCTION CALLS HERE */

    /* Do the serial console */
    command_shell(p_iostr);

    /* Close the streams */
    fclose(p_iostr[2].pIn);

    /* Exit from main returns to rstprg.c where low power mode is entered */
}
/******************************************************************************
End of function  main
******************************************************************************/
```

4.2.3.2 Customised access to the PMOD interface

Display access to the PMOD is managed by the interface file '`r_lcd_pmod.h`' can be found in the folder: '`RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/Device_Drivers/LCDPMod`'

4.3 Interacting with software

The serial console is the primary form of user input to the software, use the console to execute available commands, each command comes with simple instructions on use using the help or '?' command.

Following commands are available in the default project:

```
RZA1 CommonCore Demo Ver.1.00.0001
Copyright (C) Renesas Electronics Europe.

REE> help
The commands available are:
F1 or ? <CR> - print the command help strings
F2 - brings up the last command line
F3 - repeats the last command
date DD/MM/YYYY<CR> - Set the date
time hh:mm:ss<CR> - Set the time
mem a l<CR> - Reads memory from address H'a length H'l
sys <CR> - Shows the system resource usage information
task n<CR> - List n tasks
ver <CR> - Show the application version
mperf <CR> - Check the speed of writing to SDRAM
led n s <CR> - Sets LED n state to On s = 1,
            Off s = 0, Toggle s = ^
led0 s <CR> - Sets LED0 state to On s = 1,
              Off s = 0, Toggle s = ^
led1 s <CR> - Sets LED1 state to On s = 1,
              Off s = 0, Toggle s = ^
led2 s <CR> - Sets LED2 state to On s = 1,
              Off s = 0, Toggle s = ^
led3 s <CR> - Sets LED3 state to On s = 1,
              Off s = 0, Toggle s = ^
restart <CR> - Restart the system with a WDT reset
dma -o<CR> where o is one of the or more following options
    -r = repeat test until key press
    -s = transfer size (default 256KByte)
    -a = source in SDRAM
    -b = destination in SDRAM
    -m = use C memcpy function in place of the DMA
    -x = execute
    (no option) = Command list display

pmodstart <CR> - Run PMOD sample
pmodstop <CR> - Terminate PMOD sample
pmodrotate <CR> - Rotate image on PMOD display by 90 degrees clockwise.
```

Note
No additional documentation of the commands are available.
When adding sample code please refer to the associated ‘Description.txt’ file that shall be provided with the sample or type ‘help’ and see the new commands and their associated instructions.

5.Interaction between Samples

This section detail how the independent samples utilise share resources.

5.1 Available resources for sharing

The samples share the following peripherals and as such their access is restricted. When access is required to the noted peripherals please use the associated interface:

5.1.1 Display Access Locking Functions

As the PMOD Display and LCD panel use some of the same I/O pins both cannot be driven at the same time. Use the following interface to provide exclusion.
/RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/inc/r_scr_resource.h

```
/***********************************************************************
* Function Name : R_SCR_ResourceLockDisplay
* Description   : Attempts to lock screen resources for requesting task
* Argument      : none
* Return value  : true for success, false if the event could not be created
*                 or the wait timed out
***********************************************************************/
_Bool R_SCR_ResourceLockDisplay (void);

/***********************************************************************
* Function Name : R_SCR_ResourceUnlockDisplay
* Description   : Frees screen resources
* Argument      : none
* Return value  : none
***********************************************************************/
void R_SCR_ResourceUnlockDisplay (void);

/***********************************************************************
* Function Name : R_SCR_ResourceDisplayStatus
* Description   : Check to see if Display is in use
* Argument      : none
* Return value  : 0 Not in use
*                 non-zero taskID of display resource holder
***********************************************************************/
uint32_t R_SCR_ResourceDisplayStatus (void);
```

5.1.2 User LED functions

The LED's (LED0,1,2,3) are driven by the interface
/RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/Hardware_Interfaces/inc/hwLED.h

```
/***********************************************************************
Function Name: led_on
Description:   LED specified for iLedNum is turned on.
Arguments:     IN iLedNum : LED number
Return value:  none
***********************************************************************/
void led_on(int32_t iLedNum);

/***********************************************************************
Function Name: led_toggle
Description:   LED specified for iLedNum is toggled.
Parameters:    IN iLedNum : LED number
Return value:  none
***********************************************************************/
void led_toggle(int32_t iLedNum);
```

```
/***********************************************************************
Function Name: led_off
Description:   LED specified for iLedNum is tuned off.
Arguments:     IN iLedNum : LED number
Return value:  none
***********************************************************************/
void led_off(int32_t iLedNum);

/***********************************************************************
Function Name: led_status
Description:   LED status specified for "iLedNum".
Arguments:     IN iLedNum : LED number
Return value:  0 led on
               1 led off
***********************************************************************/
int32_t led_status(int32_t iLedNum);
```

5.1.3 I²C Functions

The I²C channel 0 is driven by the interface
/RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/Hardware_Interfaces/inc/hwRIIC0.h

```
/***********************************************************************
* Function Name: r_RIIC0_Open
* Description  : Function to open RIIC channel 0
* Arguments    : none
* Return Value : 0 for success or -1 on error
***********************************************************************/
int32_t r_RIIC0_Open(void);

/***********************************************************************
* Function Name: r_RIIC0_Close
* Description  : Function to close RIIC channel 0
* Arguments    : none
* Return Value : none
***********************************************************************/
void r_RIIC0_Close(void);

/***********************************************************************
 * Function Name: R_RIIC_CAMERA_write
 * Description  : Configuration register for Camera
***********************************************************************/
int32_t R_RIIC_CAMERA_write(uint32_t channel,
                            uint8_t d_adr,
                            uint16_t w_adr,
                            uint8_t data);

/***********************************************************************
 * Function Name: R_RIIC_HDMI_Enable
 * Description  : Enable HDMI module on RSK TFT APP BOARD via I2C
***********************************************************************/
int32_t R_RIIC_HDMI_Enable(void);
```

The I²C channel 3 is driven by the interface
/RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/Hardware_Interfaces/inc/hwRIIC3.h

```
/***********************************************************************
* Function Name: r_RIIC3_Open
```

```
* Description  : Function to open RIIC channel 3
* Arguments    : none
* Return Value : 0 for success or -1 on error
***********************************************************************/

int32_t r_RIIC3_Open(void);

/***********************************************************************
* Function Name: r_RIIC3_Close
* Description  : Function to close RIIC channel 3
* Arguments    : none
* Return Value : none
***********************************************************************/

void r_RIIC3_Close(void);

/***********************************************************************
* Function Name: r_RIIC3_UpdateState_PE1
* Description  : Function to modify output state of Port Expander 1
*                on RIIC channel 3
* Arguments    : IN  mask - which pin(s) to modify
*                IN  state - new state for pin(s)
* Return Value : none
***********************************************************************/

void R_RIIC3_UpdateState_PE1(uint8_t mask, uint8_t state);

/***********************************************************************
* Function Name: r_RIIC3_UpdateState_PE2
* Description  : Function to modify output state of Port Expander 2
*                on RIIC channel 3
* Arguments    : IN  mask - which pin(s) to modify
*                IN  state - new state for pin(s)
* Return Value : none
***********************************************************************/

void R_RIIC3_UpdateState_PE2(uint8_t mask, uint8_t state);
```

5.2 Memory resources

The CommonCore project provides five heaps only two are used.

Heap 0: Default size 128KB used for stacks allocation/deallocation for any created tasks.
As each task has access to the System Memory (heap 3) using malloc, calloc, realloc, and free, it is advised that these memory utilities are to be used and specific task stacks are kept small.

Heap 1: Default size 256KB, Reserved for future use

Heap 2: Default size 512KB, Reserved for future use

Heap 3: System Memory, Remaining free RAM is accessed through this heap, by default malloc, free etc. use this heap, this behaviour can be modified in the file

/RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/CRT/RF/inc/malloc.h

```
/* Define the memory that is used for malloc, calloc, realloc and free */
#define MALLOC_MEMORY_TYPE          HEAP_SRAM
```

Heap 4: Default size 8.000MB, SDRAM can be accessed through this heap.

6.Project Interaction with FreeRTOS

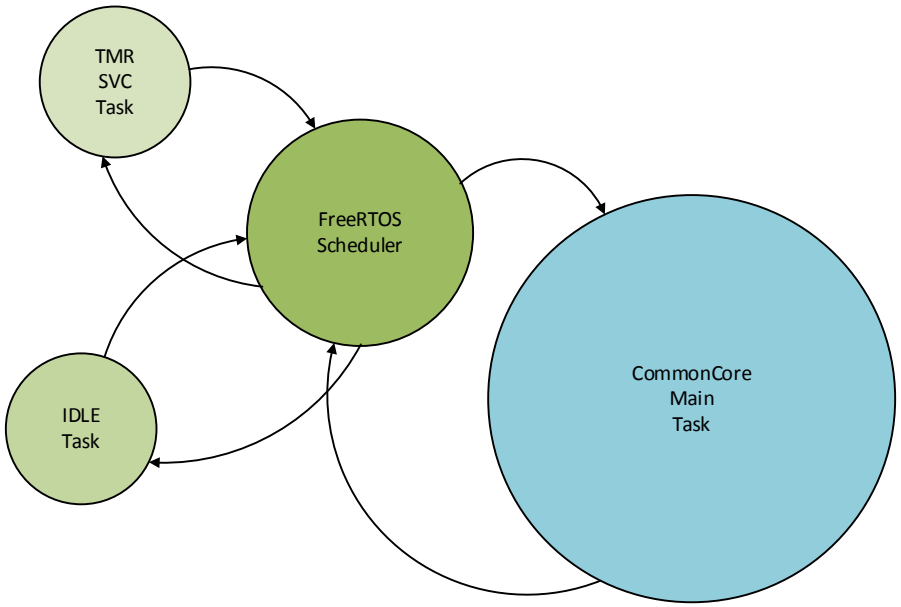
This section will examine how the CommonCore project and the sample modules interacts with FreeRTOS

6.1 Overview

The RSK CommonCore code runs in one FreeRTOS task, but has the ability to spawn supplemental tasks as required during execution.

FreeRTOS runs using two tasks
- TMR SVC which services the tick timer
- IDLE which services the processor free time.

CommonCore Runs single task
- Main task services the RSK and all its peripherals, as required,



Typical task list for CommonCore FreeRTOS (scheduler is hidden from the user)

6.2 Sample Modules

The samples supplied with CommonCore project that can be added as required. Instructions have been provided to explain how to add them but this section details how the CommonCore projects changes during execution.

The standard operation of a sample will add a number of commands (command table) to the CommonCore command line and optionally start a monitor task associated with the sample. The user can then operate the sample by invoking the relevant command.

The following code snippet shows how a typical example (in this case emmc) sample declares a new command table, adds to a free slot in the console, and creates a monitor task.

There are 24 free slots available in the console for new command tables. Each table can contain a significant number of commands as the memory for each command group is statically defined in the CMDFNASS structure (below)

```
/* Table that associates command letters, function pointer and a little
 description of what the command does */
CMDFNASS gpcmdEMCSample[] =
{
    "mmchi",
    cmd_mmchi,
    "<CR> - Run MMCHI sample commandline interface \n",

};

/* Table that points to the above table and contains the number of entries */
static CMDTAB gcmdEMCSample =
{
    gpcmdEMCSample,
    ((sizeof(gpcmdEMCSample))/(sizeof(CMDFNASS)))
};

void R_EMMC_SampleMain(FILE *pIn, FILE *pOut)
{
```

■ Code Cut for Clarity

```
   /* Add commands to commandline */
   CommandAddCmdTab(&gcmdEMCSample);

   /* Create monitor task */
   taskCreate((PTASKFN)task_monitor_emmc,
      NULL,
      TC_DEFAULT_STACK_SIZE,
      TC_DEFAULT_HEAP,
      TC_EMMC_TASK_NAME,
      TC_EMMC_DEFAULT_PRIORITY);
}
```

The command interface use by the CommonCore project to process the command line is /RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/Console/inc/command.h

```
/*****************************************************************************
Function Name: CommandAddCmdTab
Description:   Function to insert supplemental commands to the main console
Arguments:     IN  pIn - Pointer to the input file stream
Return value:  0 = ERROR
               1 = handle to command line (use when removal is required)
*****************************************************************************/
int32_t CommandAddCmdTab(CMDTAB *cmd);
```

Some samples (including emmc sample) add monitor tasks to the scheduler. These task once started stay resident regardless of whether the sample is executing.

The task by the interface use by the CommonCore project to interface with FreeRTOS /RZ_RSK_FreeRTOS_V8.1.2_e2studio/src/RenesasFiles/System/Task_Control/inc/r_task.h

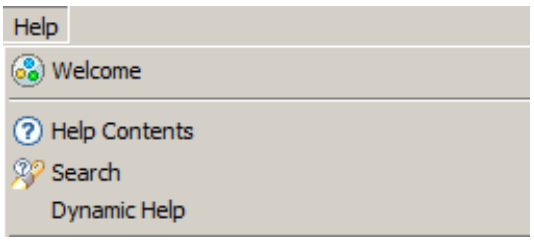
```
/*****************************************************************************
 Function Name: taskCreate
 Description:   Function to create a task
 Arguments:     IN  pTaskFunction - Pointer to the task function
 IN  pParameters - Pointer to the task paramaters
 IN  stStackSize - The required stack size for the task
 IN  memType - The memory type to allocate the stack from
```

```
 IN  pszTaskName - The task name
 IN  uiPriority - The priority of the task
 Return value:  The task ID or -1U on error
 ***************************************************************************/
uint32_t taskCreate(PTASKFN pTaskFunction,
                            void *pParameters,
                            size_t stStackSize,
                            MEMTYPE memType,
                            char *pszTaskName,
                            uint32_t uiPriority);

/***************************************************************************
 Function Name: taskDestroy
 Description:   Function to destroy a task
 Arguments:     IN  uiTaskID - The ID of the task to destroy
 Return value:  true if the task was destroyed
 ***************************************************************************/
_Bool taskDestroy(uint32_t uiTaskID);

/***************************************************************************
 Function Name: taskGetID
 Description:   Function to get the task ID
 Arguments:     none
 Return value:  The ID of the task
 ***************************************************************************/
uint32_t taskGetID(void);

/***************************************************************************
 Function Name: taskSleep
 Description:   Function to sleep for a number of mS
 Arguments:     IN  dwSleep_mS - The time to sleep in mS
 Return value:  none
 ***************************************************************************/
void taskSleep(uint32_t dwSleep_mS);

/***************************************************************************
 Function Name: taskGetPriority
 Description:   Function to get the task priority
 Arguments:     IN uiTaskID - The ID of the task to get the priority of
 Return value:  The priority of the task or -1U if not found
 ***************************************************************************/
uint32_t taskGetPriority(uint32_t uiTaskID);

/***************************************************************************
 Function Name: taskSetPriority
 Description:   Function to set the priority of a task
 Arguments:     IN  uiTaskID - The ID of the task to get the priority for
 IN  uiPriority - The priority of the task
 Return value:  true if the priority was set
 ***************************************************************************/
_Bool taskSetPriority(uint32_t uiTaskID, uint32_t uiPriority);

/***************************************************************************
 Function Name: taskSetState
 Description:   Function set a task suspend / resume state
 Arguments:     IN  uiTaskID - The ID of the task to change state
 IN  bfSuspend - true to suspend the task
 false to resume operation
 Return value:  true if the state was changed
```

```
 ***********************************************************************/
_Bool taskSetState(uint32_t uiTaskID, _Bool bfSuspend);

/***********************************************************************
 Function Name: taskName
 Description:   Function to get the task name
 Arguments:     OUT pszName - Pointer to the destination name
 IN  stLength - The length of the buffer
 IN  uiTaskID - The task ID
 Return value:  The number of chars delivered
 ***********************************************************************/
size_t taskName(char *pszName, size_t stLength, uint32_t uiTaskID);

/***********************************************************************
Function Name: taskInformation
Description:   Function to get information on the running tasks in the form of
               a dynamically allocated string. The string should be freed with
               a call to free
Arguments:     none
Return value:  Pointer to a zero terminated string
***********************************************************************/
char *taskInformation(void);

/***********************************************************************
 Function Name: taskSwitch
 Description:   Function to cause a task switch
 Arguments:     none
 Return value:  none
 ***********************************************************************/
void taskSwitch(void);

/***********************************************************************
 Function Name: taskExit
 Description:   Function to cause the current task to exit
 Arguments:     none
 Return value:  none
 ***********************************************************************/
void taskExit(void);
```

7. 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.



For information about the RZA1H series microcontrollers refer to the RZA1H Group Hardware 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/

A real time operating system demonstration for the Renesas RZ microcontrollers (ARM Cortex-A9) is provided free of charge by FreeRTOS. This can be found on the FreeRTOS website at:

http://www.freertos.org/Renesas_RZ_Cortex-A9-RTOS.html

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

Copyright

REVISION HISTORY	RSK+RZA1H FreeRTOS Integration Manual

Rev.	Date	Description	
		Page	Summary
1.00	April 28, 2015	—	First Edition issued

Renesas Starter Kit Manual: FreeRTOS Integration Manual

Publication Date: Rev. 1.00 Jun 25, 2015

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.
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
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.
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 Rd., Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

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

RZA1H Group

Renesas Electronics Corporation

R20UT3491EG0100