

RZ/V2L, RZ/V2H

Getting Started with Flexible Software Package

Introduction

This manual describes how to use the Renesas Flexible Software Package (FSP) for writing applications for the RZ microprocessor series.

Target Device

RZ/V2L, RZ/V2H



Contents

1. Introduction	4
1.1 Overview	4
1.2 Introduction to FSP	4
1.2.1 Purpose	4
1.2.2 e ² studio IDE	4
1.3 Limitations	4
1.3.1 Peripherals and pins assignment	4
1.3.2 RAM Initialization	4
2. Starting Development Introduction	5
2.1 e ² studio setup	5
2.1.1 What is e ² studio?	5
2.1.2 e ² studio Prerequisites	5
2.1.3 e ² studio installation for Windows PC	5
2.1.4 e ² studio installation for Linux PC	12
2.2 FSP setup	18
2.2.1 Installation of FSP using Package Installer	18
2.2.2 Installation of FSP Pack using Package Zip file	20
3. Set up an SMARC EVK	21
3.1 Set up an RZ/V2L SMARC EVK	21
3.1.1 Supported Emulator	21
3.1.2 Board Setup	21
3.2 Set up an RZ/V2H EVK	25
3.2.1 Supported Emulator	25
3.2.2 Board Setup	25
4. Tutorial: Your First RZ MPU Project - Blinky	29
4.1 Tutorial Blinky	29
4.2 What Does Blinky Do?	29
4.3 Create a New Project for Blinky	30
4.3.1 Details about the Blinky Configuration	33
4.3.2 Configuring the Blinky Clocks	33
4.3.3 Configuring the Blinky Pins	33
4.3.4 Configuring the Parameters for Blinky Components	33
4.3.5 Where is main()?	33
4.3.6 Blinky Example Code	33
4.4 Build the Blinky Project	34
4.5 Debug the Blinky Project	35
4.5.1 Debug prerequisites	35
4.5.2 Debug steps	35



4.5.3	Details about the Debug Process	. 37
4.5.4	Run the Blinky Project	. 37
5.	FSP application launch with e² studio	.38
5.1	Create a Project	
5.1.1	What is a Project?	
5.1.2	Creating a New Project	
5.1.3	Duplication of Resources	
5.2	Configuring a Project	
5.2.1	Summary Tab	. 45
5.2.2	Configuring the BSP	. 45
5.2.3	Configuring Clocks	. 46
5.2.4	Configuring Pins	. 47
5.2.5	Configuring Interrupts from the Stacks Tab	. 48
5.2.6	Creating Interrupts from the Interrupts Tab	. 49
5.2.7	Viewing Event Links	. 49
5.2.8	Adding and Configuring HAL Drivers	. 50
5.3	Reviewing and Adding Components	. 51
5.4	Debugging the Project	. 52
5.5	Modifying Toolchain Settings	. 53
5.6	Importing an Existing Project into e ² studio	. 54
6.	Multi-Core Debug	.57
6.1	Project Creation for each Core	
6.2	Debugging the Project for each Core	
Revi	sion History	.62



1. Introduction

1.1 Overview

This application note describes how to use the Renesas Flexible Software Package (FSP) running on the Cortex®-M33 (hereinafter referred to as CM33) and Cortex®-R8(*)(hereinafter referred to as CR8) incorporated on RZ/V2L, and RZ/V2H. (*: CR8 is for RZ/V2H only.)

1.2 Introduction to FSP

1.2.1 Purpose

The Renesas Flexible Software Package (FSP) is an optimized software package designed to provide easy to use, scalable, high quality software for embedded system design. The primary goal is to provide lightweight, efficient drivers that meet common use cases in embedded systems.

1.2.2 e² studio IDE

FSP provides a host of efficiency enhancing tools for developing projects targeting the Renesas RZ series of MPU devices. The e² studio IDE provides a familiar development cockpit from which the key steps of project creation, module selection and configuration, code development, code generation, and debugging are all managed.

1.3 Limitations

1.3.1 Peripherals and pins assignment

RZ/V2L has a multi-core configuration of Cortex-A55 (hereinafter referred to as CA55) and CM33. Also, RZ/V2H has a multi-core configuration of CA55, CM33 and CR8. It is possible to use each peripheral and GPIO from each core. This package provides drivers for each peripheral for CM33 and CR8, but each driver can operate on the assumption that it is not used in CA55.

1.3.2 RAM Initialization

Initialization of DDR SDRAM is always carried out in CA55 bootstrap regardless of the selection of boot CPU, meanwhile Internal SRAM is initialized in the bootstrap of boot CPU.



2. Starting Development Introduction

2.1 e² studio setup

2.1.1 What is e² studio?

Renesas e² studio is a development tool encompassing code development, build, and debug. e² studio is based on the open-source Eclipse IDE and the associated C/C++ Development Tooling (CDT).

When developing for RZ MPUs, e² studio hosts the Renesas Flexible Software Package (FSP). FSP provides a wide range of time saving tools to simplify the selection, configuration, and management of modules and threads, to easily implement complex applications.

2.1.2 e² studio Prerequisites

2.1.2.1 Obtaining an RZ MPU Kit

To develop applications with RZ/V FSP, start with each Evaluation Board Kit. Start-up guide of RZ/V2L Evaluation Board Kit is available at <u>RZ/V2L SMARC EVK Start-up Guide</u>. Also, Getting Started of RZ/V2H Evaluation Board Kit is available at <u>RZ/V2H EVK Getting Started</u>.

2.1.2.2 PC Requirements

The following are the minimum PC requirements to use e² studio:

- Windows 10 with Intel i5 or i7, or AMD A10-7850K or FX
- Memory: 8-GB DDR3 or DDR4 DRAM (16-GB DDR4/2400-MHz RAM is preferred)
- Minimum 250-GB hard disk

2.1.2.3 Licensing

FSP licensing includes full source code, limited to Renesas hardware only.

2.1.3 e² studio installation for Windows PC

This chapter describes how to install the e² studio IDE on Windows PC.

2.1.3.1 Download

The latest e² studio IDE installer package can be downloaded from Renesas website for free. Please check detailed information from: <u>https://www.renesas.com/e2studio</u>. Note that user has to login to the Renesas account (in MyRenesas page) for the software download.



2.1.3.2 Installation of e² studio IDE

- 1. Double-click on e² studio installer to invoke the e² studio installation wizard page. First, you need to select Install Type. In this material, it is expected that Custom Install is selected. Then, click [Next >] to continue.
- **Note:** If e² studio was installed in your PC, the option to modify, remove the existing version or install e² studio to a different location will be displayed

Renesas e ² studio 2024-07 Setup	- 🗆 X
Renesas e ² studio 2024-07 Setup	RENESAS
Install Type	
Please select the e ² studio installation type. <u>Click here</u> for help selecting a type and to see	what features are included.
Select Install Type:	
Lite Install (Recommended) This installs e ^a studio in Lite Mode. This mode offers a simplified experience focused on simple code editing & Standard Install	debugging with only important features
This installs e ² studio in Advanced Mode. This mode offers all extended debugging functionality and other advanced	features
Custom Install Custom installation of e ² studio This mode is allows you to select which features are installed	
< <u>B</u> ack	Vext > Install Cancel

Figure 1: e² studio installation wizard

2. Welcome page

User can change the install folder by clicking [Change...].Click [Next] to continue.

Note1: If you would like to have multiple versions of e² studio, please specify new folder here. **Note2**: Multi-byte characters cannot be used for e² studio installation folder name.

Renesas e ² studio 20	24-07 Setup
() Welcome	Install directory ready
Device Families Extra Features	Install Location: [Change]
Customise Features	Prerequisite software already installed
Additional Software	
Licenses	Internet connection available
Shortcuts	Change Proxy Settings
Drivers	Ready to install
Summary	Software to install:
Installing	Renesas e2 studio v24.7.0.R20240620-1807 Java Runtime v17.0.0
Results	IAR Plugin Manager V1.1.0.202007251457 IRenesas e2 studio Common Components (Litle) v24.7.0.R20240620-1807 Renesas e2 studio Common Components (Full) v24.7.0.R20240620-1807 Renesas QE Common Components v24.7.0.R20240620-1807 Icelipse CDT Linker Script Editor and DSL v1.0.112.v20240408-2044 Iceli Advantum Complexity Editor and DSL v1.0.112.v20240408-2044

Figure 2: Installation of e² studio – Welcome page



3. Device Families

Select Devices Families to install. Click the [Next] button to continue.

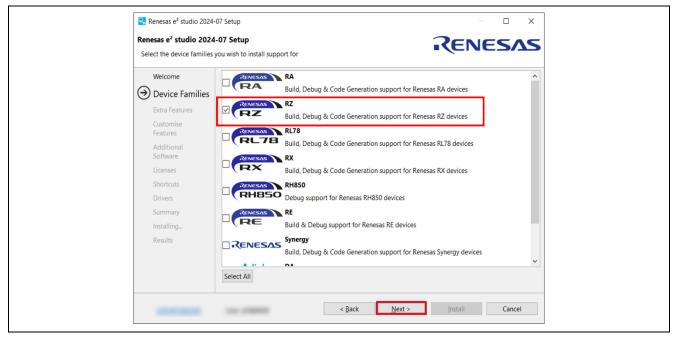


Figure 3: Installation of e² studio – Device Families

4. Extra Features

Select Extra Features (i.e., Language packs, SVN & Git support, RTOS support...) to be installed. For non-English language users, please select Language packs at this step if needed. Click the [Next] button to continue.

Renesas e ² studio 2024 Renesas e ² studio 2024 Select the extra features y	4-07 Set	tup	
Welcome Device Families		P	Japanese Language Support
Extra Features			Chinese (Simplified) Language Support
Customise Features Additional			Chinese (Traditional) Language Support
Software Licenses			Git Integration Git SCM Support
Shortcuts Drivers		P	Terminals ANSI/vt102 compatible Terminal support for Serial, ssh and Telnet
Summary Installing			
Results			
	Select	t All	
-	-		< Back Next > Install Cancel

Figure 4: Installation of e² studio – Extra Features



5. Customize Features

Select the components to install and click the [Next] button to continue. Be sure that Renesas FSP Smart Configurator Core and Renesas FSP Smart Configurator ARM are selected.

🔜 Renesas e ² studio 2024-07 Setup		– 🗆 X
Renesas e² studio 2024-07 Setup		RENESAS
Device families Extra Features Customise Features Additional Software Licenses Shortcuts Drivers Summary Installing_ Results Periodic Penesas Common Penesas Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Common Penesas Penesas Common Penesas Penesas Common Penesas Penesas Common Penesas Pene	nents you want to install. amy support to allow project generation and pulle of executable kAP projects. nergy Family Support (24.7.0.R2040620-1807) tray family support to allow project generation, build & debug Family Support (24.7.0.R2040620-1807) built compare to allow project generation, build & debug P Smart Configurator Core (96.0.v202040619-0945) monents for Remass FSP Smart Configurator P Smart Configurator ARM (96.0.v20240619-1422) & compared to Bancase FSP Smart Configurator SC VMCU Support (24.7.0.R20240620-1807) C-V MCU Support (24.7.0.R20240620-1807) monents for Remass GE ion for Eclipse (6.9.0.202403050737-r) th Git, integration with Gerrit, Gitflow, and Task repositories Tools (24.7.0.R20240620-1807) voling Deslect All Optional	
		Size of install: 911.5 MB

Figure 5: Installation of e² studio – Features

6. Additional Software

Select additional software (i.e., compilers, utilities, QE...) to be installed. Be sure to select the following item and click [Next] to continue.

GNU ARM Embedded 12.2-Rel1

Figure 6: Installation of e² studio – Additional Software

For more details on the installation of Additional Software, please see the section 2.1.4.



7. License Agreement

Read and accept the software license agreement. Click the [Next] button. Please note that user must accept the license agreement, otherwise installation cannot be continued.

🔜 Renesas e² studio 202	4-07 Setup	— 🗆
Renesas e² studio 202	24-07 Setup	RENESA
Welcome	Please read and accept the fol	lowing Software Agreements
Device Families Extra Features Customise Features Additional Software	Renesas e2 studio OpenJDK License Agreement ARM DS-5 Toolchain Integrat IAR Plugin Manager	License Terms and Conditions for RENESAS e2 studio This Renesas e2 studio license agreement ("Agreement") is between the entity on whose behalf you are entering into this Agreement ("Client") and Renesas Electronics Corporation, a Japanese company with its registered office at 3-2-24. Toyous, Koto-ku, Tokyo 135-0061, Japan ("Renesas"). YOU SHOULD READ THIS AGREEMENT CAREFULLY, AS IT CONSTITUTES A BINDING CONTRACT BETWEEN CLIENT AND RENESAS. The Renesas IDE Software (defined below) is intended for commercial use by a company or corporation only and is not designed, developed or produced for any private use or purpose. If you are an individual, or you intend to install the Renesas IDE Software is expected to be used for a private purpose directly or indirectly, you should click "No" on the installer. Otherwise, by clicking the "I accept" button or other button or mechanism designed to acknowledge agreement to the terms of an electronic copy of this Agreement, or by installing, accessing, or otherwise copying or using all or any portion of the Renesa IDE Software, you accept this Agreement on behalf of the entity for which you are authorized to act (e.g., an employer) and acknowledge that such entity is leoally bound by this Agreement and tware Agreements

Figure 7: Installation of e² studio – Licenses

8. Shortcuts

Select shortcut name for start menu and click [Next] button to continue.

Note: If e² studio was installed in another location, it is recommended to rename to distinguish from the other e² studio(s).

Renesas e ² studio 2024-07 Setup
Welcome Shortcuts to important programs and files will be created in the following locations: Device Families Image: Shortcuts Extra Features In start menu group: Customise Renesas Electronics e2 studio Features Additional Software Licenses Orivers Summary Installing Results

Figure 8: Installation of e² studio – Shortcuts



9. Summary

Components list to be installed is shown. Please confirm the contents and click the [Install] button to install the Renesas e^2 studio IDE.

Renesas e ² studio 2024-07 Setu	RENES	ΔS
Welcome Device Families Extra Features Customise	Ready to install Software to install: • Renesas e2 studio v24.7.0.R20240620-1807 • Java Runtime v17.0.0 • JAP Plugin Manager v1.1.0.202007251457	^
Features Additional Software Licenses Shortcuts	 Renessa e2 studio Common Components (Lite) v24.7.0.R20240620-1807 Renessa e2 studio Common Components (Lite) v24.7.0.R20240620-1807 Renessa e2 studio Common Components for ARM Devices v24.7.0.R20240620-1807 Renessa e2 studio Tools v24.7.0.R20240620-1807 Renessa RZ Family Support v24.7.0.R20240620-1807 Renessa RZ Family Support v24.7.0.R20240620-1807 Renessa RZ Family Support v24.7.0.R20240620-1807 Renessa ST Family Support v24.7.0.R20240620-1807 Renessa ST Family Support v24.7.0.R20240620-1807 Renessa ST Family Support v24.7.0.R20240620-1807 	
Drivers Summary Installing Results	Renesas FSP Smart Configurator ARM v9.6.0.v20240619-1422 Renesas QE Common Components v24.7.0.R20240620-1807 ARM DS-5 Toolchain Integration v1.0.5 v20240408-1026 Eclipse CDT Linker Script Editor and DSL v1.0.112.v20240408-2044 GCC for Renesas RZ Build Support v24.7.0.v20240613-0901 GCC for Renesas RZ Support Files v24.7.0.v20240513-0255 JustJ Adoptium OpenIDK Hotspot IRE Complete v17.0.10.v20240120-1143	
	Renesas Build Support Files v9.6.0.v20240408-1639 Renesas CMake Build Support Files v9.6.0.v20240408-1639 Renesas Common Project Import/Export v9.6.0.v20240606-0906 Renesas Debug Views v9.6.0.v20240607-0813	~

Figure 9: Installation of e² studio – Summary

10.Installing...

The installation is performed. Depending on selected items of additional software, new dialog prompts may appear during the installation process. Please see chapter 2.1.3.3 for more detailed information.

11.Results

Click the **OK** button to complete the installation.

	e² studio 2024-07 Setup	ENESAS
complete.	ome Installation of c2 studio is complete. te families Please click OK to close. Features □ Launch e2 studio? res ☑ View Release Notes? total ☑ View What's New? sees ses truts rs nary ling Jlts	

Figure 10: Summary Page



2.1.3.3 Installation of Additional Software

As mentioned in the section 2.1.3.2, the additional software listed below is essential for RZ/V FSP.

GNU ARM Embedded 12.2-Rel1.

In this section, the detailed procedure for installing this tool.

• GNU ARM Embedded Toolchain 12.2-Rel1

If it was selected in the Additional Software pane of e² studio, you will see the installation wizard for the GNU ARM Embedded Toolchain during the installation process.

🗑 Arm GNU Toolchain 12.2.re	I1 aarch64-none-elf —		💮 Arm GNU Toolchain 12.2.re	el1 aarch64-none-elf —	×
	Welcome to Arm GNU Tool 12.2.rel1 aarch64-none-elf Setup will guide you through the installation Toolchain 12.2.rel1 aarch64-none-elf. It is recommended that you dose all other before starting Setup. This will make it pos relevant system files without having to reb computer. Click Next to continue.	Setup n of Arm GNU applications sible to update	aarch64-none-elf. Press Page Down to see the re Contains code from project G GNU Debugger (<u>https://www.</u> GNU G Version 3, 29 J Copyright (C) 2007 Free Sofi Everyone is permitted to cop of this license document, but If you accept the terms of the	NU Binutils (<u>https://www.gnu.org/software/binutils/</u>), . <u>gnu.org/software/gdb/</u>) under the following license(s). SENERAL PUBLIC LICENSE une 2007 tware Foundation, Inc. < <u>http://fsf.org/</u> > y and distribute verbatim copies	·
Setup will install Arm GNU Too install in a different folder, die installation. Destination Folder	install Arm GNU Toolchain 12.2.rel1 aarch64-r Ichain 12.2.rel1 aarch64-none-elf in the follow ck Browse and select another folder. Click Inst	ing folder. To	Milsoft Install System v3.05-2 –	< Back I Agree Cano	×
Space required: 719.3 MB Space available: 20.7 GB Nullsoft Install System v3.05-2 –	< Back Install	Cancel		< Back Einish Cano	el

Figure 11: Installation of GNU ARM Embedded Toolchain



2.1.4 e² studio installation for Linux PC

This chapter describes how to install the e² studio IDE on Linux PC.

2.1.4.1 Download

The following files are required to download before installation.

SEGGER J-Link driver Please download the driver V7.96e or later from: https://www.segger.com/downloads/jlink/JLink Linux V796e x86 64.deb

e² studio IDE installer

e2 studio IDE installer package can be downloaded from Renesas website for free. Please check detailed information from: <u>https://www.renesas.com/e2studio</u>.

2.1.4.2 Installation

This section describes the procedure of each software installation.

Filename, version number and the file path is provided for example purpose only.

- SEGGER J-Link driver
 - Open a terminal window and enter below commands. sudo dpkg -i JLink_Linux_V796e_x86_64.deb

(If the previous install fails with unmet dependencies, retry it as follows) sudo apt-get -f install sudo dpkg -i JLink_Linux_V796e_x86_64.deb

- e² studio IDE
 - 1. Run the e² studio IDE Installer. (Before running the installer, check the execution permission of the installer.)

./e2studio_installer-2024-07_linux_host.run

2. Welcome page

User needs to select Install Type as shown below. In this material, it is expected that Custom Install is selected. Then, click [Next >] to continue.

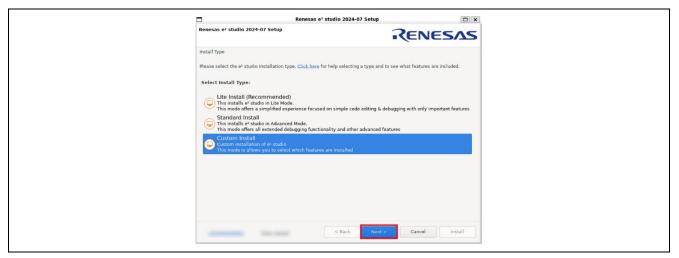


Figure 14: Installation of e² studio – Welcome page



3. Welcome page (Cont'd)

User can change the install folder by clicking [Change...]. Click [Next] to continue.

Note1: If you would like to have multiple versions of e² studio, please specify new folder here. **Note2**: Multi-byte characters cannot be used for e² studio installation folder name.

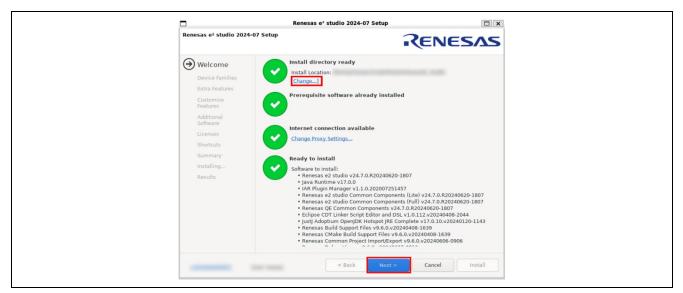


Figure 15: Installation of e² studio – Welcome page



4. Device Families

Select Devices Families to install. Click the [Next] button to continue.

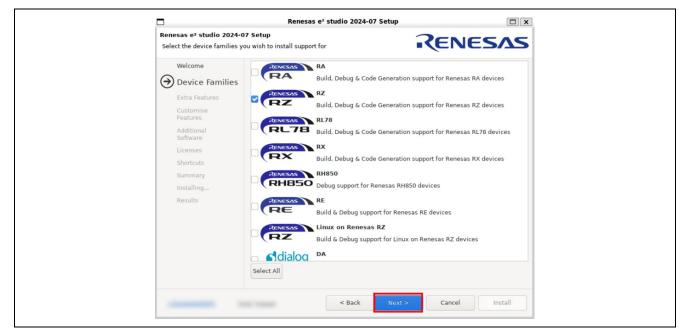


Figure 16: Installation of e² studio – Device Families

5. Extra Features

Select Extra Features (i.e., Language packs, SVN & Git support, RTOS support...) to be installed. For non-English language users, please select Language packs at this step if needed. Click the [Next] button to continue.

Renesas e ² studio 2024-07 Select the extra features you		REN	IESVS
Welcome Device Families Extra Features Customia Software Licenses Shortcuts Summary Installing Results		Japanese Language Support Chinese (Simplified) Language Support Chinese (Traditional) Language Support Git Integration Git SCM Support Terminals ANSI/vt102 compatible Terminal support for Seria	al, ssh and Telnet
	Select All		

Figure 17: Installation of e² studio – Extra Features



6. Customize Features

Select the components to install and click the [Next] button to continue.

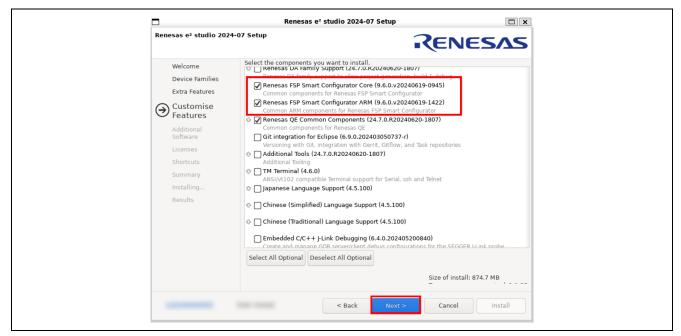


Figure 18: Installation of e² studio – Features

7. Additional Software

Select additional software (i.e., compilers, utilities, QE...) to be installed. Be sure to select the "GNU ARM Embedded 12.2-Rel1" and click [Next >] to continue.

Renesas e² studio 2024-0	7 Setup	
Select the additional software	∍ you wish to install ■	CENESAS
Welcome Device Families Extra Features Customise Features Moditional Software Licenses Shortcuts Summary Installing Results	 Renesas QE ✓ Renesas AI ✓ Renesas AI Navigator ✓ AI Transfer Learning Tool Plugin ✓ AI Model Conversion Tool Plugin Renesas Toolchains && Utilities ✓ GCC Toolchains && Utilities ✓ GOL ARM Embedded 12.2-Rel1 ✓ GNU ARM Embedded 9.3.1 2020q2 ✓ GCC ARM A-Profile (AArch64 bare-metal) 13.2.Rel1 GCC CARM A-Profile (AArch64 bare-metal) 10.3 2021 Renesas RA FSP 	
		340.2 MB download required
	< Back Next >	Cancel

Figure 19: Installation of e² studio – Additional Software



8. License Agreement

Read and accept the software license agreement. Click the [Next] button. Please note that user must accept the license agreement, otherwise installation cannot be continued.

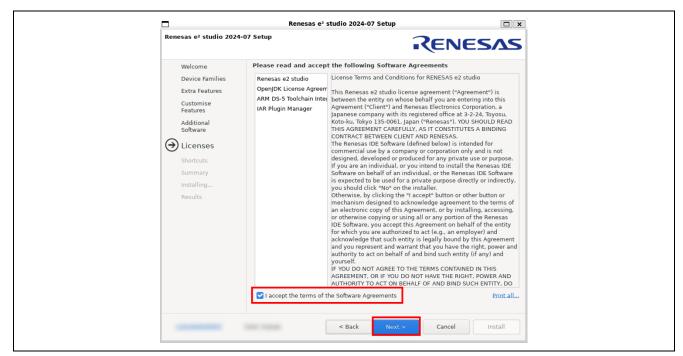


Figure 20: Installation of e² studio – Licenses

9. Shortcuts

Select shortcut name for start menu and click [Next] button to continue.

Note: If e² studio was installed in another location, it is recommended to rename to distinguish from the other e² studio(s).

Figure 21: Installation of e² studio – Shortcuts



10.Summary

Components list to be installed is shown. Please confirm the contents and click the [Install] button to install the Renesas e^2 studio IDE.

	Renesas e² studio 2024-07 Setup
Renesas e² studio 2024-07 Setup	RENESA
Welcome Device Families Extra Features Customise Features Additional Software Licenses Shortcuts Sommary Installing Results	 Ready to install: Software to install: Renesas e2 studio v24.7.0.R20240620-1807 Java Runtime v17.0.0 Renesas e2 studio Common Components (Lite) v24.7.0.R20240620-1807 Renesas e2 studio Common Components (Full) v24.7.0.R20240620-1807 Renesas e2 studio Common Components for ARM Devices V24.7.0.R20240620-1807 Renesas R2 Family Support v24.7.0.R20240620-1807 Renesas FSP Smart Configurator ARM v96.0.v20240619-1422 Renesas FSP Smart Configurator ARM v96.0.v20240619-1422 Renesas FSP Smart Configurator KAW v96.0.v20240619-1690 CAM DS-5 Toolchain Integration v1.0.S.v202404021-0100 CAC for Renesas R2 Sulid Support v24.7.0.v20240613-0901 GCC for Renesas R2 Sulid Support V24.7.0.v20240613-0901 Renesas Common Freject ImportExport v9.6.0.v20240606-6906 Renesas Debug Views v9.6.0.v20240607-0813 Renesas Debug Views v9.6.0.v20240607-0813 Renesas RIOS Debug Views v9.6.0.v20240607-0814 Renesas RIOS Debug Views v9.6.0.v20240607-0815 Renesas RIOS Debug Views v9.6.0.v20240607-0815 Renesas RIOS Debug Views v9.6.0.v20240620-0816 Renesas RIOS Debug Views v9.6.0.v20240620-0816 Renesas RI

Figure 22: Installation of e² studio – Summary

11.Installing...

The installation is performed. Depending on selected items of additional software, new dialog prompts may appear during the installation process.

12.Results

Click the **OK** button to complete the installation.

Renesas e² studio 202	24-07 Setup	
Kellesas e- studio 202	24-07 Setup	RENESAS
Welcome Device Families Extra Features Customise Features Additional Software Licenses Shortcuts Summary Installing € Results	Installation of e2 studio is complete. Please click OK to close. Launch e2 studio? View Release Notes? View What's New?	
	< Back Next	> Cancel OK

Figure 23: Summary Page



2.2 FSP setup

2.2.1 Installation of FSP using Package Installer

Package Installer **RZV_FSP_Packs_v2.0.1.exe** is showcased at <u>here.</u> This section describes the procedure for installation. Note that it's for Windows Host PC only.

- 1. Quit e2 studio.
- 2. Invoke RZV_FSP_Packs_v2.0.1.exe.
- 3. Click [Next >] to start the installation.



Figure 24: FSP Package Installer

4. See the license term and click [I Agree] if it's acceptable.

Renesas RZV FSP v2.0.1	Setup — RZV FSP v2.0.1 Please review the license terms before installin FSP v2.0.1.	g Renesas RZV	
https://github.com/renese	as/rzv-fsp/blob/master/LICENSE.md		
If you accept all terms of t	he agreement, dick I Agree.		
	< Back I Agree	Cancel	

Figure 25: FSP License Term



5. Specify e2 studio installation folder (e.g., C:\Renesas\e2studio) and click [Install].

Renesas RZV FSP v2.0.1 Setup - X Choose Install Location Choose the folder in which to install Renesas RZV FSP v2.0.1.	
The installation path must point to the root of the e2 studio installation (e.g. C:¥Renesas¥e2_studio). Please make sure e2 studio is closed before installation.	
Browse to folder where e2 studio is installed C:\Renesas\e2_studio\e4 Browse	
Space required: 51.2 MB Space available: 24.8 GB	
< Back Install Cancel	

Figure 26: Browse to the folder where e2 studio is installed

6. Click [Finish] to complete the installation.



Figure 27: Completion of FSP Installation

If the box **Open up documentation for this release** is checked at that time, FSP documentation for the installed version of FSP should be opened.



2.2.2 Installation of FSP Pack using Package Zip file

No package installer is available for Linux Host PC. Thus, you need to install FSP with the zip file **RZV_FSP_Packs_v2.0.1.zip**. This section describes the procedure for installation.

- 1. Download RZV_FSP_Packs_v2.0.1.zip from here.
- 2. Extract the zip file to e2 studio installation directory. If it's successfully extracted, **rz_fsp/rzv/packs** should be placed at **<e2 stduio installation directory>/Internal/projectgen**.

↑ 📕 « internal > projectgen ≯ rz_fsp > rzv > packs	
- menuer, historikar, mith in hours	
Amazon.FreeRTOS-Kernel.10.4.6+fsp.2.0.1.pack	
Arm.CMSIS5.5.8.0+renesas.0.fsp.2.0.1.pack	
Linaro.OpenAMP.1.0.0+fsp.2.0.1.pack	
Renesas.RZV.2.0.1.pack	
Renesas.RZV_baremetal_blinky.2.0.1.pack	
Renesas.RZV_baremetal_blinky_cm33boot.2.0.1.pack	
Renesas.RZV_baremetal_blinky_cr8.2.0.1,pack	
Renesas.RZV_baremetal_minimal_cm33boot.2.0.1.pack	
Renesas.RZV_baremetal_minimal_cr8.2.0.1.pack	
Renesas.RZV_board_custom.2.0.1.pack	
Renesas.RZV_board_rzv2h_evk2.0.1.pack	
Renesas.RZV_board_rzv2l_smarc.2.0.1.pack	
Renesas.RZV_freertos_blinky.2.0.1.pack	
Renesas.RZV_freertos_blinky_cm33boot.2.0.1.pack	
Renesas.RZV_freertos_blinky_cr8.2.0.1.pack	
Renesas.RZV_freertos_minimal_cm33boot.2.0.1.pack	
Renesas.RZV_freertos_minimal_cr8.2.0.1.pack	
Renesas.RZV_mcu_rzv2h.2.0.1.pack	
Renesas.RZV_mcu_rzv2l.2.0.1.pack	
SEGGERJLink.7.96.5.pack	

Figure 28: FSP Packs on e2studio installation directory

- 3. At the 1st invocation of e2 studio after the extraction, FSP should be automatically installed.
- 4. You can check if the installation is successfully done by the procedure below:
 - Click Help > CMSIS Packs Management > Renesas RZ/V

	e - test/configuration.xml - e ^a studio		
Renesas Views Run Renesas Al Win			
r .	🚳 Welcome	Q i 🖻	
德 [test] FSP Configuration ×	 Help Contents Search 		BE Out
Summary	Show Context Help	>	There i
	Show Active Keybindings Shift+Ctrl+L	ject Content	outline
	Cheat Sheets		
	Renesas Help		
Project Summary	🖉 Toolchain Help 🕨 🕨		
	CMSIS Packs Management	Renesas R	Z/A
	🖉 Add Renesas Toolchains	Renesas R	Z/G
	Eclipse User Storage	Renesas R	Z/N
	🧐 Perform Setup Tasks	Renesas R	Z/T
	🍫 Check for Updates	Renesas R	Z/V
Board: : RZ/V2H Evaluation Kit Device: : R9A09G057H44GBG	Install New Software		
Device:: N9A09005/H44086 Core:: Core 6(CM33_0) Toolchain:: GCC for Renesas RZ Toolchain Version:: 12.2.1.arm-12-24 FSP Version:: : 2.0.0	Eclipse Marketplace		
	lAR Embedded Workbench plugin manager		
	About e ² studio		
Project Type: : Flat			

Figure 29: CMSIS Packs Management (1)

• If FSP is successfully installed, 2.0.1 should be listed under FSP as shown below:

	CMSIS Packs Manage	ement - Renesas RZ	/ v	
Packs location:				
Show in System Ex				
Available Packs			•	(i) +
Category		Version	Status	
▼ 🐻 FSP				
▶ III 2.0.1				
▶ 🖽 Generic				

Figure 30: CMSIS Packs Management (2)



3. Set up an SMARC EVK

3.1 Set up an RZ/V2L SMARC EVK

Below is an example of a typical system configuration.

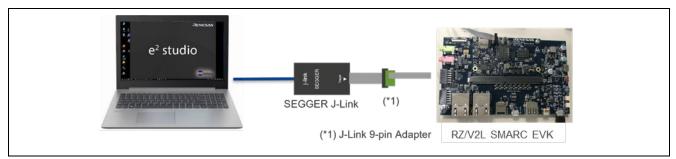


Figure 31: System Configuration Example – RZ/V2L SMARC EVK

3.1.1 Supported Emulator

• SEGGER J-Link

For details on SEGGER J-Link, please see <u>J-Link Debug Probes by SEGGER – the Embedded Experts</u>.

3.1.2 Board Setup 3.1.2.1 Boot MODE

To set the board to Boot mode 3(QSPI Boot(1.8V) Mode), set the SW11 as below.

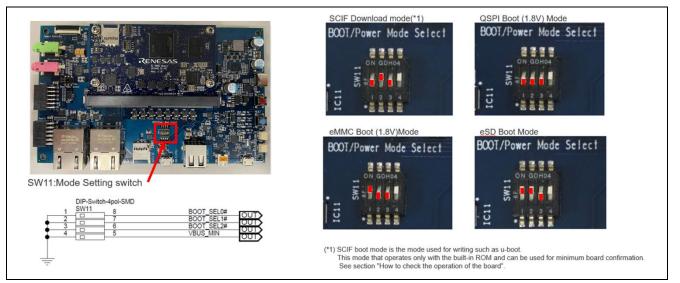


Figure 32: Boot MODE



3.1.2.2 JTAG connection

When connecting JTAG, you must set the DIP SW1 settings as follows:

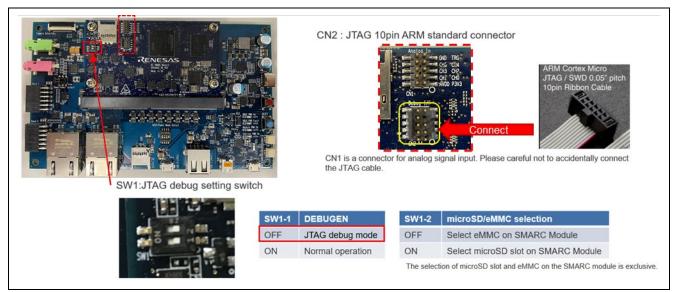


Figure 33: JTAG connection

Please note that RZ/V2L SMARC EVK has CoreSight 10 connector and therefore, the following adapter must be needed to connect Segger J-Link.

https://www.segger.com/products/debug-probes/j-link/accessories/adapters/9-pin-cortex-m-adapter/

3.1.2.3 Debug Serial (console output)

Debug serial uses CN14. The baud rate is 115200bps.

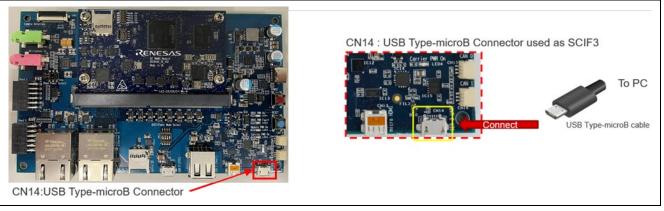


Figure 34: Debug Serial(console output)



3.1.2.4 Power Supply

Here are the power supply related goods to be used in Renesas' development. Please prepare for the equivalent ones for your development.

- USB Type-C cable CB-CD23BK (manufactured by Aukey)
- USB PD Charger Anker PowerPort III 65W Pod (manufactured by Anker)

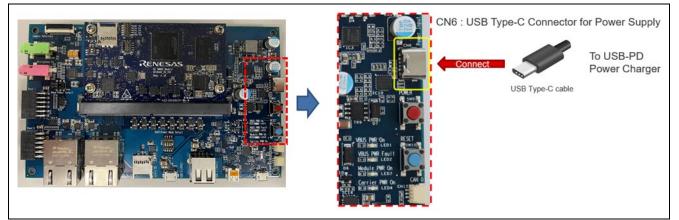


Figure 35: Power Supply

Connect USB-PD Power Charger to USB Type-C Connector. Then LED1(VBUS PWR On) and LED3 (Module PWR On) lights up. Press SW9 to turn on the power. Then LED4(Carrier PWR On) lights up.

Note: When turn on the power, press and hold the power button for 1 second. When turn off the power, press and hold the power button for 2 seconds

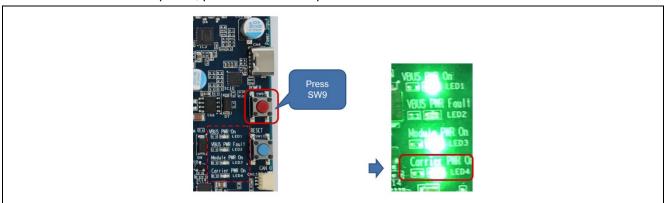


Figure 36: LED Status after Turning on EVK

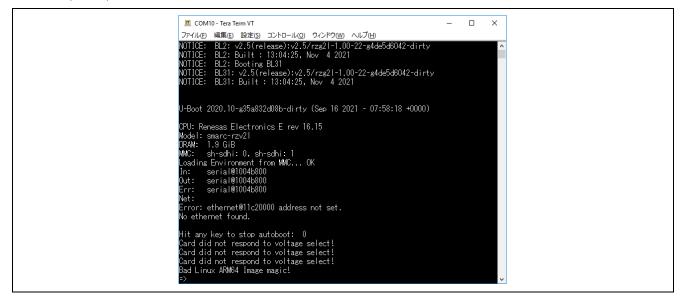


3.1.2.5 How to check the operation of the board

First, check the board for problems. There are two ways to do this. Please check with either.

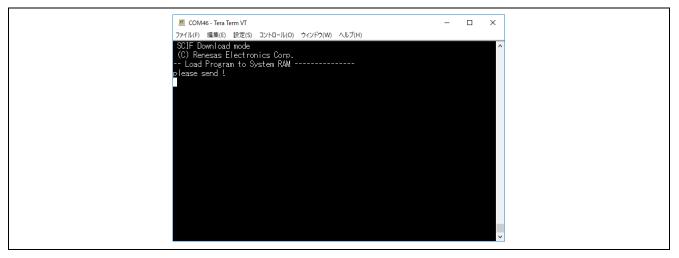
BOOT MODE: QSPI Boot(1.8V) Mode

If u-boot is written to the serial flash, when the power is turned on, the following will be output to the console(CN14).



BOOT MODE: SCIF Download Mode

When the power is turned on, the following will be output to the console (CN14).





3.2 Set up an RZ/V2H EVK

Below is an example of a typical system configuration.

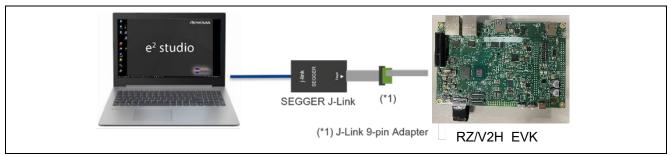


Figure 37: System Configuration Example – RZ/V2L SMARC EVK

3.2.1 Supported Emulator

SEGGER J-Link

For details on SEGGER J-Link, please see <u>J-Link Debug Probes by SEGGER – the Embedded Experts</u>.

3.2.2 Board Setup 3.2.2.1 Boot MODE

Set the boot mode using the DSW1 shown in the figure below.

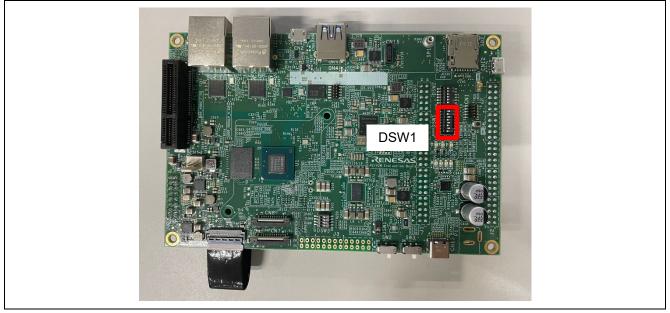


Figure 38: Boot MODE



Switch No.	Function
1	Select the cold boot CPU
	OFF: CM33 / ON: CA55
2	Input the CA55 frequency at the CA55 cold boot.
	[SW2 : SW3] = [OFF : OFF] : 1.6GHz
	= [OFF : ON] : 1.7GHz [default]
3	= [ON : OFF] : 1.1GHz
	= [ON : ON] : 1.5GHz
4	Input the boot mode select signal.
	[SW4 : SW5] = [OFF : OFF] : xSPI
	= [OFF : ON] : SCIF
5	= [ON : OFF] : SD
	= [ON : ON] : eMMC
6	OFF: SSCG OFF / ON: SSCG ON
7	OFF: Normal mode / ON: Debug mode
8	Fix OFF

Select the boot mode with the following settings.

3.2.2.2 JTAG connection

When connecting to JTAG, you must set the DSW1 Switch No.7 settings as ON.

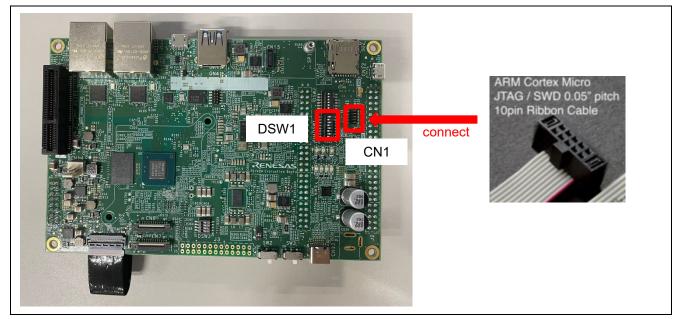


Figure 39: JTAG connection

Please note that RZ/V2H EVK has CoreSight 10 connector and therefore, the following adapter must be needed to connect Segger J-Link.

https://www.segger.com/products/debug-probes/j-link/accessories/adapters/9-pin-cortex-m-adapter/



3.2.2.3 Debug Serial (console output)

Debug serial uses CN12. The baud rate is 115200bps.

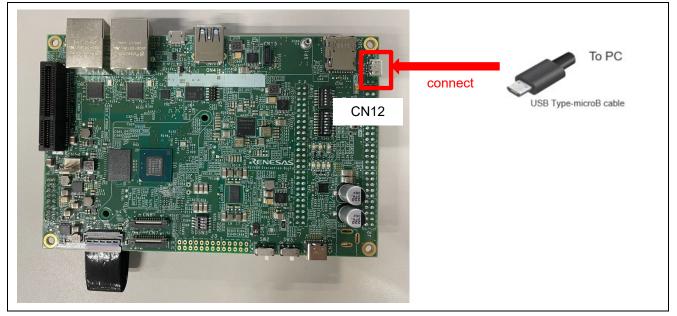


Figure 40: Debug Serial(console output)

3.2.2.4 Power Supply

Here are the power supply related goods to be used in Renesas' development. Please prepare for the equivalent ones for your development.

- USB PD Charger MAGCUBE PD 100W (AOC-C005) (manufactured by AOHI)
- USB Type-C cable included with MAGCUBE PD 100W (AOC-C005) (manufactured by AOHI)

Check that the power slide switch SW2 and SW3 are turned OFF.

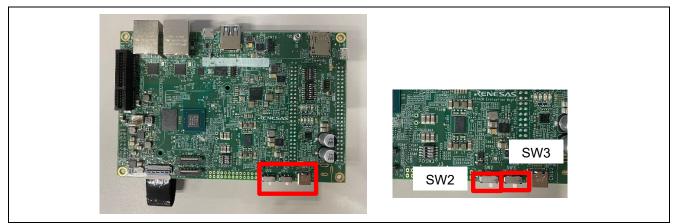


Figure 41: Power Supply



Connect USB-PD Power Charger to CN13. Turn the SW3 ON, then LD2 and LD7 light up. Turn the SW2 ON, then LD1, LD3 and LD4 light up.

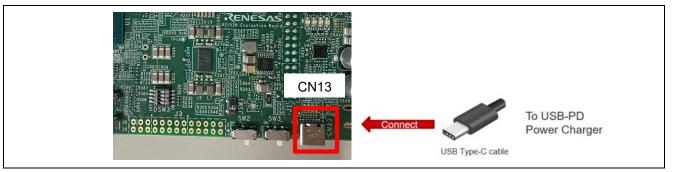


Figure 42: Power Supply



4. Tutorial: Your First RZ MPU Project - Blinky

4.1 Tutorial Blinky

The goal of this tutorial is to quickly get acquainted with the Flexible Platform by moving through the steps of creating a simple application using e² studio and running that application on an RZ MPU board.

4.2 What Does Blinky Do?

The application used in this tutorial is Blinky, traditionally the first program run in a new embedded development environment.

Blinky is the "Hello World" of microprocessors. If the LED blinks you know that:

- The toolchain is setup correctly and builds a working executable image for your chip.
- The debugger has installed with working drivers and is properly connected to the board.
- The board is powered up and its jumper and switch settings are probably correct.
- The microprocessor is alive, the clocks are running, and the memory is initialized.
- Timer (GTM) interrupt is intentionally fired and GPIO is properly controlled.

Note: RZ/V2H SMARC EVK has on-board LED but RZ/V2L SRMAC EVK board does not have any LED. Thus, Blinky sample application for RZ/V2L SRMAC EVK is designed to use the Pmod module described below alternatively:

Pmod LED (Four High-brightness LEDs): <u>https://reference.digilentinc.com/pmod/pmodled/start</u>

This module is not included on the RZ/V2L SRMAC EVK board and so, please prepare it beforehand.

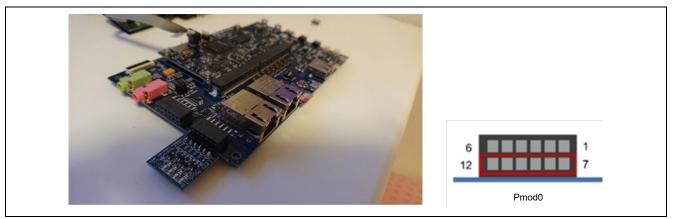


Figure 43: Connection Pmod LED module (410-076)

In the case of RZ/V2H EVK, the on-board LED is placed as below.

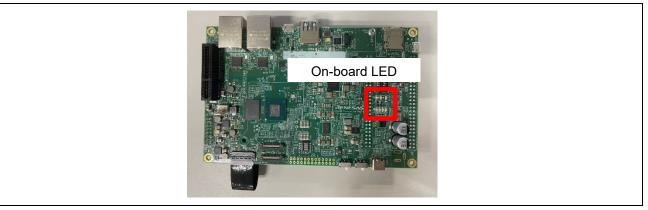


Figure 44: On-board LED of RZ/V2H EVK



4.3 Create a New Project for Blinky

The creation and configuration of an RZ/V C/C++ FSP Project is the first step in the creation of an application.

The base RZ/V pack includes a pre-written Blinky example application.

Follow these steps to create an RZ MPU project:

1. In e² studio, click **File** > **New** > **C/C++ Project**.

2	work - e² studio					
File	Edit Source Refactor Navigate	Search Project	t R	enesas Views Run Window Help		
	New	Alt+Shift+N >		Renesas C/C++ Project	> -	010
	Open File		C++	Makefile Project with Existing Code		
È,	Open Projects from File System		¢	C/C++ Project	Ģ	¥ (
	Recent Files	>	2	Project		

Figure 45: New C/C++ Project

2. Select [Renesas RZ] > [Renesas RZ/V C/C++ FSP Project] and Click Next.

All CMake Make Make Create an executable or static library C/C++ FSP project for Renease RZ/G (C++ FSP Project Renease RA Renease RZ/T C/C++ FSP Project Create an executable or static library C/C++ FSP project for Renease RZ/V (C++ FSP Project Renease RZ/V (C++ FSP Project Create an executable or static library C/C++ FSP project for Renease RZ/V.

Figure 46: Renesas RZ/V C/C++ FSP Project

- 3. Assign a name to this new project. Blinky is a good name to use for this tutorial.
- 4. Click Next. The Project Configuration window shows your selection.

Figure 47 : e² studio Project Configuration window (part 1)



5. Select the board support package by selecting the name of your board from the Device Selection dropdown list. Select **GNU ARM Embedded** in Toolchains and version is **12.2.1.arm-12-24** and Click **Next**.

Reveals RZ/V C(C++ FSP Project Reveals RZ/V C(C++ FSP Project Device and Tool Selection Selection Selection Selection Selection Core: Core: R0A33 Core: M33 Core: M3 Core: M4 Core: M4 Core: M4 Core: M4 C
Device and Tools Selection Device Selection FSP Venicer Device Selection Board Device Details Corre: Corre: Corre: Corre: Corre: Corre: Device Details Tractone No Precesor Corte: Corte: Device Details Device Details Device No Device Venils Devicesor Corte: Devicesor Corte: Devicesor Corte: Devicesor Corte: Debugger Pluit: Public RAM
Device and Tools Selection PSP Version: Device Read-00000070+44683 Core: Device: Toolchains Conta-M33
PSP Version Board Description Board Description Device R540900007944089 Core: Core (KM11,0) Language:
Parket International Context International Context Bearding International Context Context Context Language: International Context Tout/colorine Debugger International Context Debugger International Context Debugger International Context Debugger
Device: R5A05057H446BG Core: Core: KCM31 (0) Device Details Language: ⊕C ○ C++ ToutZone No Procesor Corter M33 ToutChains ToutChains Debugger June ABM → ✓
Device R84050337H44363 Core Core 6(CM13_0) Device bittalis Language: Imazone No Fina 1188 Processor Contex/M33 Debugger Junca Participant Colvaria Debugger Junca Participant
Core Core 6(CM13_0) ○ Device Details Language: ● C () C++ InstZone No Toolchains Debugger Cortex-M33 Toolchains Debugger ↓Link ABM
Toolchains Debugger
12.21.arm-12-24 Manage Toolsham.
Cancel

Figure 48 : e² studio Project Configuration window (part 2)

6. Select the **build artifact** and **RTOS**. Be sure that on the current version, **Secure** should always be chosen at the **Sub-core start state**. Otherwise, the created project can't be built successfully.

Renesas RZ/V C/C++ FSP Project	- 🗆 X
Renesas RZ/V C/C++ FSP Project	
Build Artifact, RTOS Selection and Sub-Core Selection	
Build Artifact Selection	RTOS Selection
Executable Project builds to an executable file	No RTDS ~
Static Library Project builds to a static library file	
 Executable Using an R2/F Static Library Project builds to an executable file Project uses an exoling R2/F value library project 	
Sub-core start state	
Secure Start sub-core in secure state	
Non-secure Start sub-core in non-secure state	
0	< <u>Back</u> <u>Next</u> Einish Cancel

Figure 49 : e² studio Project Configuration window (part 3)

7. Select the **Blinky** template for your board and click **Finish**.

Renesas RZ/V C/C++ FSP Project Project Template Selection Project Template Selection Selection Bare Metal (CM33) - Blinky Bare metal FSP project of Cortex-M33 core that includes BSP and will blink LEDs if available. This project will initialize clocks, pins, stacks, and the C nutrime environment. Remesas RZV 2.0.1 pack] Bare Metal (CM33) - Minimal Bare Metal SP project of Cortex-M33 core that includes BSP. This project will initialize clocks, pins, stacks, and the C nutrime environment. [Remesas RZV 2.0.1 pack]
Project Template Selection Project Template Selection Bare Metal (CM33) - Blinky Bare metal TSP project of Cortex-M33 core that includes BSP and will blink LEDs if available. This project will initialize clocks, plins, stacks, and the C nuttime environment. [RemeasER2/2.23 L pack] Bare Metal (CM33) - Minimal Bare Metal (CM33) - Minimal environment. Bare Metal (CM33) - Minimal Bare Metal
Bare Metal (CM33) - Blinky Bare metal FSP project of Cortex-M33 core that includes BSP and will blink LEDs if available. This project will initialize clocks, pins, stacks, and the C- number environment. [Renesas:R2V2.0.1 pack] Bare Metal (CM33) - Minimal Bare metal SP project of Cortex-M33 core that includes BSP. This project will initialize clocks, pins, stacks, and the C runtime environment.
Base metal FSP project of Cortex-M33 core that includes BSP and will blink LEDs if available. This project will initialize clocks, pink, stacks, and the C nutrime environment. [Renesas:RZV2.0.1 pack] Base Metal (CM33) - Minimal Base metal FSP project of Cortex-M33 core that includes BSP. This project will initialize clocks, pins, stacks, and the C nutrime environment.
environment.
Code Generation Settings
(2) < Back Next> Emith Cancel
Templates for RZ/V2H

Figure 50 : e² studio Project Configuration window (part 4)



Once the project has been created, the name of the project will show up in the **Project Explorer** window of e² studio. Now click the **Generate Project Content** button in the top right corner of the **Project Configuration** window to generate your board specific files.

(Blinky) FSP Configura	tion ×		- 8
Summary			Generate Project Content
Project Summar	у	Rene	sas ^
Board:	RZ/V2H Evaluation Kit		
Device:	R9A09G057H44GBG		
Core:	Core 6(CM33_0)		
Toolchain:	GCC for Renesas RZ		
Toolchain Version:	12.2.1.arm-12-24		
FSP Version:	2.0.1		
Project Type:	Flat		
Location:	C:/work/v2h/Blinky <		
Selected software of	components		

Figure 51 : e² studio Project Configuration tab

• Your new project is now created, configured, and ready to build.



4.3.1 Details about the Blinky Configuration

The Generate Project Content button creates configuration header files, copies source files from templates, and generally configures the project based on the state of the Project Configuration screen.

For example, if you check a box next to a module in the Components tab and click the Generate Project Content button, all the files necessary for the inclusion of that module into the project will be copied or created. If that same check box is then unchecked those files will be deleted.

4.3.2 Configuring the Blinky Clocks

By selecting the Blinky template, the clocks are configured by e² studio for the Blinky application. The clock configuration tab (see 5.2.3 Configuring Clocks) shows the Blinky clock configuration. The Blinky clock configuration is stored in the BSP clock configuration file.

4.3.3 Configuring the Blinky Pins

By selecting the Blinky template, the GPIO pins used to toggle the LED1 are configured by e² studio for the Blinky application. The pin configuration tab shows the pin configuration for the Blinky application (see 5.2.4.Configuring Pins). The Blinky pin configuration is stored in the BSP configuration file.

4.3.4 Configuring the Parameters for Blinky Components

The Blinky project automatically selects the following HAL components in the Components tab:

- r_gtm
- r_ioport

To see the configuration parameters for any of the components, check the Properties tab in the HAL window for the respective driver (see 5.2.8.Adding and Configuring HAL Drivers).

4.3.5 Where is main()?

The main function is located in < project >/rzv_gen/main.c. It is one of the files that are generated during the project creation stage and only contains a call to hal_entry(). For more information on generated files, see Adding and Configuring HAL Drivers.

4.3.6 Blinky Example Code

The blinky application is stored in the hal_entry.c file. This file is generated by e² studio when you select the Blinky Project template and is located in the project's src/ folder.

The application performs the following steps:

- 1. Get the LED information for the selected board by bsp_leds_t structure.
- 2. Set the configuration of Timer (GTM) and the callback function that is called when interrupt is fired.
- 3. Define the output level HIGH for the GPIO pins controlling the LEDs for the selected board.
- 4. Toggle the LEDs by writing to the GPIO pin with "R_BSP_PinWrite((bsp_io_port_pin_t) pin, pin_level)" in callback function of GTM that is called with the specified interval.



4.4 Build the Blinky Project

Highlight the new project in the Project Explorer window by clicking on it and build it.

There are three ways to build a project:

- 1. Click on Project in the menu bar and select Build Project.
- 2. Click on the hammer icon.
- 3. Right-click on the project and select Build Project.

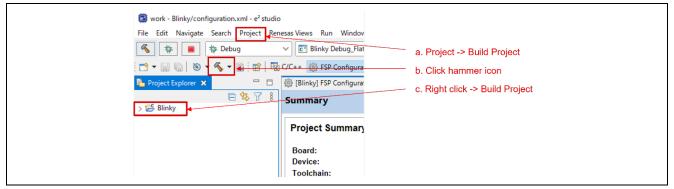


Figure 52 : e² studio Project Explorer window

Once the build is complete a message is displayed in the build Console window that displays the final image file name and section sizes in that image.

🎦 Pin Conflicts 📮 Console 🗙
CDT Build Console [Blinky]
'arm-none-eabi-gcc -mthumb -mcpu=cortex-m33+nodsp+notp -tdiagnostics-pa
'arm-none-eabi-gcc -mthumb -mcpu=cortex-m33+nodsp+nofp -fdiagnostics-pa
arm-none-eabi-gcc @"Blinky.elf.in"
arm-none-eabi-objcopy -0 srec "Blinky.elf" "Blinky.srec"
arm-none-eabi-sizeformat=berkeley "Blinky.elf"
text data bss dec hex filename
4812 2072 16784304 16791188 1003694 Blinky.elf
05:35:14 Build Finished. 0 errors, 0 warnings. (took 7s.823ms)

Figure 53 : e² studio Project Build console



4.5 Debug the Blinky Project

4.5.1 Debug prerequisites

To debug the project on a board, you need

- The board to be connected to e² studio
- The debugger to be configured to talk to the board
- The application to be programmed to the microprocessor

Applications run from the internal ram or external ram of your microprocessor. To run or debug the application, the application must first be programmed to ram by JTAG debugger. SMARC EVK board has an JTAG header and requires an external JTAG debugger to the header.

4.5.2 Debug steps

To debug the Blinky application, follow these steps:

1. Configure the debugger for your project by clicking Run > Debugger Configurations ...

Run	Window Help	
	Renesas Device Partition Manager	
	TraceX	>
e j	Tracealyzer	>
Q	Run	Ctrl+F11
检	Debug	F11
	Run History	>
0	Run As	>
1	Run Configurations	
	Debug History	>
*	Debug As	>
	Debug Configurations	
9	External Tools	>

Figure 54 : e² studio Debug icon

or by selecting the drop-down menu next to the bug icon and selecting Debugger Configurations ...

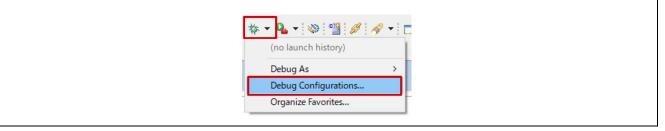


Figure 55 : e² studio Debugger Configurations selection option

2. Select your debugger configuration in the window. If it is not visible, then it must be created by clicking the New icon in the top left corner of the window. Once selected, the **Debug Configuration** window displays the **Debug configuration** for your **Blinky** project.



 Debug Configurations Create, manage, and run configurations 				□ ×
Image: Second	Name: Blinky Debug Main % Debugger Startup ty Sour Project: Blinky	ce 🔲 Common		Browse
Doc Scipt Doc Scipt	C/C++ Application: Debug/Blinky.elf Build (if required) before launching <u>Build Configuration:</u> Use Active O Enable auto build © Use workspace settings	O Disable auto buil Configure Workspa		Browse
Filter matched 13 of 15 items			Revert Debug	Apply Close

Figure 56 : e² studio Debugger Configurations window with Blinky project (1)

- 3. Select the debug configuration for the generated project and select the **Debugger** tab.
- 4. Click **Debug** to begin debugging the application.
- 5. Extracting **RZ Debug**.

Progress Information	, 		×
Preparing launch delegate			
Configuring GDB			
	Cancel	Details >	



4.5.3 Details about the Debug Process

In debug mode, e² studio executes the following tasks:

- 1. Downloading the application image to the microprocessor and programming the image to the internal and/or external memory.
- 2. Setting a breakpoint at main().
- 3. Setting the stack pointer register to the stack.
- 4. Loading the program counter register with the address of the reset vector.
- 5. Displaying the startup code where the program counter points to.

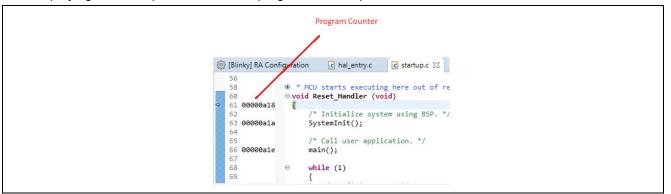


Figure 57 : e² studio Debugger memory window

4.5.4 Run the Blinky Project

While in Debug mode, click **Run > Resume** or click on the **Play** icon twice.



Figure 58 : e² studio Debugger Play icon

The LED (RZ/V2L SRMAC EVK: Pmod LED, RZ/V2H SRMAC EVK: On bord LED) should now be blinking.



5. FSP application launch with e² studio

This section describes how to create project and debug in single core environment. Please refer to the section 6 for the multi-core project creation and debug in RZ/V2H.

5.1 Create a Project

5.1.1 What is a Project?

In e² studio, all FSP applications are organized in RZ MPU projects. Setting up an RZ MPU project involves:

1. Create a Project

2. Configuring a Project

These steps are described in detail in the next two sections. When you have existing projects already, after you launch e² studio and select a workspace, all projects previously saved in the selected workspace are loaded and displayed in the **Project Explorer** window. Each project has an associated configuration file named configuration.xml, which is located in the project's root directory.

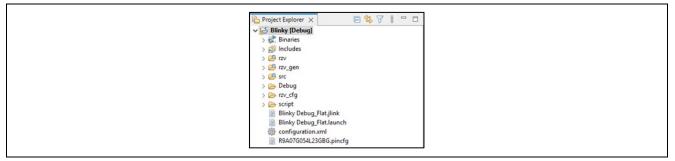


Figure 59 : e² studio Project Configuration file

Double-click on the configuration.xml file to open the RZ MPU Project Editor. To edit the project configuration, make sure that the **FSP Configuration** perspective is selected in the upper right-hand corner of the e² studio window. Once selected, you can use the editor to view or modify the configuration settings associated with this project.



Figure 60 : e² studio FSP Configuration Perspective

Note: Whenever the RZ project configuration (that is, the configuration.xml file) is saved, a verbose RZ Project Report file (rzv_cfg.txt) with all the project settings is generated. The format allows differences to be easily viewed using a text comparison tool. The generated file is located in the project root directory.

🏊 Project Explorer 🗴 🛛 🗟 🛱 🖇 🕈	□ 📄 rzv_cfg.txt ×		- 6
Slinky [Debug]	1	FSP Configuration	^
> 🗟 Binaries	2	Board "RZ/V2L Evaluation Kit (SMARC)"	
> 🔊 Includes	3	R9A07G054L23GBG	
> 🙆 rzv	4	part_number: R9A07G054L23GBG	
> 😂 rzv_gen	5	rom_size_bytes: 0	
	6	ram_size_bytes: 131072	
> 😂 src	7	<pre>package_style: LFBGA</pre>	
> 🗁 Debug	8	package_pins: 456	
> 🗁 rzv_cfg	9		
> 🦢 script	10	RZV2L	
Blinky Debug_Flat.launch	11 12	series: 2	
configuration.xml	12	RZV2L Family	
R9A07G054L23GBG.pincfg	14	RZ/V2L Common	
☐ rzv_cfg.txt	15	Secure stack size (bytes): 0x200	
	16	Main stack size (bytes): 0x200	
	17	Heap size (bytes): 0	
	18	MCU Vcc (mV): 3300	
	19	Parameter checking: Disabled	
	20	Assert Failures: Return FSP_ERR_ASSERTION	
	21	Error Log: No Error Log	
	22	PFS Protect: Enabled	
	23	C Runtime Initialization : Enabled	~

Figure 61 : RZ Project Report



The RZ Project Editor has several tabs. The configuration steps and options for individual tabs are discussed in the following sections.

Note: The tabs available in the RZ Project Editor depend on the e² studio version and the layout may vary slightly, however the functionality should be easy to follow.

Figure 62 : RZ Project Editor tabs

5.1.2 Creating a New Project

For RZ MPU applications, generate a new project using the following steps:

1. Click on File > New > C/C++ Project.

۲	work - e² studio						
File	Edit Source Refactor Navigate S	Search Project	R	enesas Views Run Window Help			
	New	Alt+Shift+N >		Renesas C/C++ Project	>	1-	010
	Open File		C ++	Makefile Project with Existing Code			
	Open Projects from File System		¢	C/C++ Project		Þ	× (
	Recent Files	>		Project			

Figure 63 : New RZ MPU Project

2. Then click on the Renesas RZ/V C/C++ FSP Project template for the type of project you are creating.

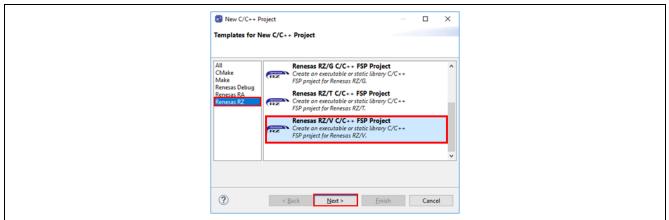


Figure 64 : New Project Templates



3. Select a project name and location.

Renesas RZ/V C/C++ FSP Project	- 0 X
Renesas RZ/V C/C++ FSP Project	
Project Name and Location	1
Project name	
Binky	
Use gefault location	
Location: C/works_RZV/work/Blinky	Biowse-
Chodse file system: default	
You can download more Roman packs here	
② < Back Next > En	Sish Cancel

Figure 65 : RZ MPU Project Generator (Screen 1)

4. Click Next.

5.1.2.1 Selecting a Board and Toolchain

In the Project Configuration window select the hardware and software environment:

- 1. Select the **FSP version**.
- 2. Select the **Board** for your application. You can select an existing RZ MPU Evaluation Kit or select **Custom User Board** for any of the RZ MPU devices with your own BSP definition.
- 3. Select the **Device**. The **Device** is automatically populated based on the **Board** selection. Only change the **Device** when using the **Custom User Board (Any Device)** board selection.
- 4. Select the **Core**. You can select Core 6(CM33_0), Core 4(CR8_0) or Core 5(CR8_1) if you selected RZ/V2H for the **Device**.
- 5. To add threads, select RTOS, or No RTOS if an RTOS is not being used.
- 6. The Toolchain selection defaults to GNU Arm Embedded.
- 7. Select the Toolchain version. This should default to the installed toolchain version.
- 8. Select the **Debugger**. The J-Link Arm Debugger is preselected.
- 9. Click Next.

📴 Renesas Rž	Z/V C/C++ FSP Project		- O X
	V C/C++ FSP Project ools Selection		
Device: Core:		Board Description Device Details TrustZone Pins Processor	No 1368 Conter-M33
Toolchains		Debugger	
GNU ARM E	mbedded	J-Link ARM	~
12.2.1.arm-12	-24 V Manage Toolchains		
?		< <u>B</u> ack	Next > Einish Cancel

Figure 66 : RZ MPU Project Generator (Screen 2-1)

If Core 4(CR8_0) or Core 5(CR8_1) is selected in procedure 4, you need to select the preceding project. To select the preceding project when creating the Core 4(CR8_0) or Core 5(CR8_1) project, it is required to prepare Core 6(CM33_0) before Core 4(CR8_0) or Core 5(CR8_1) project creation.



🕲 Renesas RZ/V C/C	++ FSP Project — 🗆 🗙
Renesas RZ/V C/C	++ FSP Project
Preceding Project or	Smart Bundle Selection
Preceding Project	Rlink
© rreccomy rroject	Choose this option if you have access to the project source code of the preceding processor core or security context.
O Smart Bundle:	anose this option in you have access to the project source code of the processing processor core of second contents
0	
	Workspace File System Variables
	Choose this option if you only have access to a Smart Bundle describing the configuration of the preceding processor
	core or security context.
Preceding Project/Sr	art Bundle Details
FSP version	2.0.1
Toolchain Toolchain version	GNU ARM Embedded 12.2.1.arm-12-24
Board	RZ/V2H Evaluation Kit
Device	R9A09G057H44GBG
Core	CM33_0
Zones	CM33_0_S
?	< Back Next > Finish Cancel
J	

Figure 67 : RZ MPU Project Generator (Screen 2-2)

5.1.2.2 Selecting a Project Template

In the next window, select the build artifact, **Sub-core start state** and **RTOS**. Be sure that you select **Secure** as **Sub-core start state** in the current version.

 Renesas RZ/V C/C++ FSP Project Renesas RZ/V C/C++ FSP Project 		- • ×
Build Artifact, RTOS Selection and Sub-Core Selection		
Build Artifact Selection	RTOS Selection	
Executable Project builds to an executable file	No RTOS	~
 Static Library Project builds to a static library file 		
Executable Using an RZ/V Static Library Project builds to an executable file Project uses an existing RZ/V static library project		
Sub-core start state		
Secure Start sub-core in secure state		
 Non-secure Start sub-core in non-secure state 		
?	< <u>B</u> ack <u>N</u> ext >	Einish Cancel

Figure 68 : RZ MPU Project Generator (Screen 3)



In the next window, select a project template from the list of available templates. By default, this screen shows the templates that are included in your current RZ/V MPU Pack. Once you have selected the appropriate template, click **Finish**.

Note: If you want to develop your own application, select the basic template for your board, **Bare Metal -Minimal** or **FreeRTOS - Blinky**.

Renesas RZ/V C/C++ FSP Project Project Template Selection	Renesas RZ/V C/C++ FSP Project Project Template Selection
Project Template Selection	Project Template Selection
Bare Metal - Blinky Bare metal PB project thai includes BSP and will blink LEDs if available. This project will initialize clocks, pins, stacks, and the cruster environment. (Renesas-RZV2L.10.0 pack) Bare Metal - Minimal Bare metal PSP project that includes BSP. This project will initialize clocks, pins, stacks, and the C runtime environment. [Renesas-RZV2L.10.0 pack]	PreeRTOS 5- Blinky - Dynamic Allocation PreeRTOS 4- Blinky - Dynamic Allocation PreeRTOS 5- Blinky - Static Allocation PreeRTOS 5- PreeRTOS
Code Generation Settings Use Renesse Code Formatter	Code Generation Settings Use Remeses Code Formatter
() < Back Einst Cancel	(2) < Back int > Emish Cancel
us RZ/V C/C++ FSP Project — 🗌 🗙	Renesas RZ/V C/C++ FSP Project Renesas RZ/V C/C++ FSP Project Project Template Selection
RZ/V C/C++ FSP Project	Project Template Selection
emplate Selection	Image: State of the second state of
neration Settings enesas Code Formatter	
< Back Next > Einish Cancel	Code Generation Settings ☑ Use Renetas Code Formatter
	(?) < Back Next > Finish Can

Figure 69 : RZ MPU Project Generator (Screen 4)



When the project is created, e² studio displays a summary of the current project configuration in the RZ MPU Project Editor.

Figure 70 : RZ MPU Project Editor and available editor tabs

On the bottom of the RZ MPU Project Editor view, you can find the tabs for configuring multiple aspects of your project:

- With the **Summary** tab, you can see all the key characteristics of the project: board, device, toolchain, and more.
- With the **BSP** tab, you can change board specific parameters from the initial project selection.
- With the Clocks tab, you can configure the MPU clock settings for your project.
- With the Interrupts tab, you can add new user events/interrupts.
- With the Event Links tab, you can configure events used by the Event Link Controller.
- With the **Stacks** tab, you can add and configure FSP modules. For each module selected in this tab, the **Properties** window provides access to the configuration parameters, interrupt selections.
- The **Components** tab provides an overview of the selected modules. Although you can also add drivers for specific FSP releases and application sample code here, this tab is normally only used for reference.

The functions and use of each of the supported tabs is explained in detail in the next section.



5.1.3 Duplication of Resources

In the case of RZ/V2H Core 4(CR8_0) or Core 5(CR8_1) project, duplicate resources are indicated as red character in **Stacks** tab when using resources that are used in the linked Core 6(CM33_0) project.

The following image is an example of both Core $6(CM33_0)$ and Core $4(CR8_0)$ projects using same resource. The duplication of r_gtm is indicated in **Stacks** tab. To avoid this duplication, please change the channel resource in **Properties** of r_gtm.

	arch Project Renesas Views Run Windov	v Help				
	0 14 毛 武 母 株・Գ-・				Q I III III C/C++ @ FSP Configuration 存 Debu	
Project Explorer ×		Blinky] FSP Configuration Image: Configuration Image: Configuration	[R8] FSP Configuration \times	- 1	FSP Visualization ×	
> 😸 Blinky ~ 🔂 Project_CR8		Stacks Configuration		Generate Project Conten	The active editor element does not use this view	
> 🔊 Includes > 🤒 rzv		Threads 🚯 New Thread 🚯 Remove 🗎	HAL/Common Stacks	🐑 New Stack > 😤 Extend Stack > 🛞 Remove		
> @ rzv_gen > @ sc > @ rzv_cfg > @ script ∰ configurationxml		HAL/Common g_joport I/O Port (r_joport) g_timer2 Timer (r_gtm)	g_ioport I/O Port (r_ioport)	di th g_timer2Timer (r_gtm)		
		R. Transis and C'Rout	٥	A stack element with a bar of this color indica This instance may be referenced by one other	'g_timer2 Timer (r_gtm)' is a Module instance. odule instance only.	
	Project_CR8 Debug_FlatJaunch O Developer Assistance			Error: Peripheral 'GTM0' is allocated within a p	receding project or smart bundle	
	Objects 🕢 New Object > 🛍 Remove					
		Summary BSP Clocks Pins Interrupts Event Lin	iks 3 Stacks Components			
🔲 Properties 🗙 🛃 Pr	oblems 🌒 Smart Browser 📮 Console				🛃 🐨 🖉 🖬 🖓 🐘 🖛 🛙	
g_timer2 Timer (r_g	tm)					
Settings Property			Value		1	
securitys	1					
API Info Param	eter Checking		Default (BSP)			
✓ Module g	g_timer2 Timer (r_gtm)					
✓ Gener	al					
Na			g timer2 GTM0			
	annel					
Mo	de		Periodic			
Per	iod		8			
			Hertz			
Per	iod Unit		P1CLK for GTM0 and 1. P5CLK fo			

Figure 71 : Duplication of resource between Core 6(CM33_0) and Core 4(CR8_0) projects



5.2 Configuring a Project

Each of the configurable elements in an FSP project can be edited using the appropriate tab in the RZ Configuration editor window. Importantly, the initial configuration of the MPU after reset and before any user code is executed is set by the configuration settings in the **BSP** tab. When you select a project template during project creation, e² studio configures default values that are appropriate for the associated board. You can change those default values as needed. The following sections detail the process of configuring each of the project elements for each of the associated tabs.

5.2.1 Summary Tab

Figure 72 : Configuration Summary tab

The **Summary** tab, seen in the above figure, identifies all the key elements and components of a project. It shows the target board, the device, toolchain and FSP version. Additionally, it provides a list of all the selected software components and modules used by the project. This is a more convenient summary view when compared to the **Components** tab.

5.2.2 Configuring the BSP

The **BSP** tab shows the currently selected board (if any) and device. The Properties view is located in the lower left of the Project Configurations view as shown below.

Note:	If the Properties	view is not visible	, click Window > S	Show View > Pro	perties in the to	p menu bar.
-------	-------------------	---------------------	--------------------	-----------------	-------------------	-------------

Summary BSP Locks Pins Interrupts Event Links Stacks Components Properties X Problems Smart Browser Console							
RZ/V2H Evaluation Kit							
Settings	Property	Value					
Settings	✓ R9A09G057H44GBG						
	part_number	R9A09G057H44GBG					
	rom_size_bytes	0					
	ram_size_bytes	131072					
	package_style	LFBGA					
	package_pins	1368					
	V RZ Common (CR8)						
	> stack size (bytes)						
	Heap size (bytes)	0x8000					
	MCU Vcc (mV)	3300					
	Parameter checking	Disabled					
	Assert Failures	Return FSP_ERR_ASSERTION					
	Error Log	No Error Log					
	PFS Protect	Enabled					
	C Runtime Initialization	Enabled					
	Caches	Enabled					
	✓ RZV2H (CR8_1)						
	series	2					

Figure 73 : Configuration BSP tab



The **Properties** view shows the configurable options available for the BSP. These can be changed as required. The BSP is the FSP layer above the MPU hardware. e² studio checks the entry fields to flag invalid entries. For example, only valid numeric values can be entered for the stack size.

When you click the **Generate Project Content** button, the BSP configuration contents are written to rzv_cfg/fsp_cfg/bsp/bsp_cfg.h This file is created if it does not already exist.

Warning

Do not edit this file as it is overwritten whenever the Generate Project Content button is clicked.

5.2.3 Configuring Clocks

The **Clocks** tab presents a graphical view of the MPU's clock tree, and each HAL driver uses the settings for dedicated numerical calculation. For example, scif_uart driver calculates the communication rate from the settings in Clocks tab. Please note that the clock configuration is carried out on the main core (CA55) in advance when CM33 work as sub core. Thus, clocks configuration here must align with the settings on CA55.

In the case of CM33 cold boot, BSP will configure each clock setting in start-up process according to content of **Clocks** tab.

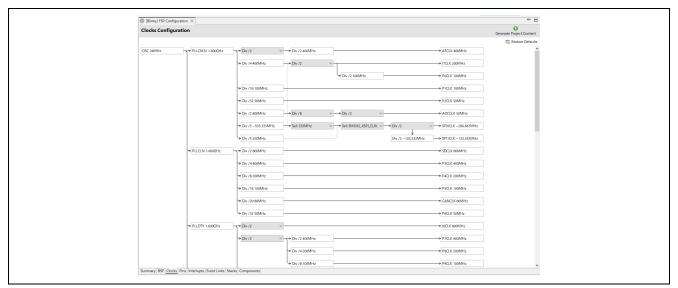


Figure 74 : Configuration Clocks tab

When you click the **Generate Project Content** button, the clock configuration contents are written to: rzv_gen/bsp_clock_cfg.h

This file will be created if it does not already exist.

Warning

Do not edit this file as it is overwritten whenever the Generate Project Content button is clicked.



5.2.4 Configuring Pins

The pins tab provides flexible configuration of the MPU's pins. As many pins can provide multiple functions, they can be configured on a peripheral basis. For example, selecting a serial channel via the SCIF peripheral offers multiple options for the location of the receive and transmit pins for that module and channel. The location and function of the pins are shown in the **FSP Visualization** view. For more information on the function and color coding of the pins, please check the Legend in the **FSP Visualization** view.

[Blinky] FSP Configuration $ imes$					% FSP Visualization $ imes$	-
in Configuration			Generate Project Co	ntent	Type pin function	
Select Pin Configuration		🔛 Export to CSV file 🚦	Configure Pin Driver Warn	ings		
RZV2H-EVK.pincfg		configurations				
Pin Selection $\models \textcircled{E} \textcircled{P}_{z}^{a}$	Pin Configuration		Cycle Pin Gro	oup		
Type filter text	Name	Value	Link			
✓ ✓ Ports						
> P1 > V P2						
> P2 > p2 > Y	<			>		
in Function Pin Number						

Figure 75 : Pin Configuration

The pin configurator includes built-in conflict checker. So, if the same pin is allocated to another peripheral or I/O function, the pin will be shown as red in the **FSP Visualization** view and with white cross in a red square in the **Pin Selection** pane and **Pin Configuration** pane in the main **Pins** tab.

In the example shown below, port P70 is already used by the IRQ_IRQ0, and the attempt to connect to this pin to the GPT results in dangling connection error. To fix this error, select another port from the pin drop-down list or disable the IRQ_IRQ0.

Select Pin Configuration Export to CSV file Configure Pin Driver Warnings R2V2H-EVK.pincfg Manage configurations. Generate data:	*[Blinky] FSP Configuration	ion ×		(Generate Project Cont	tent
Type filter text Name Value Lock Lock ImercGPT Operation Mixed Lock Lock GPT1 Operation Mode Custom GPT2 GPT3 GTICODA None GPT4 GTICODA None GTICODB GPT6 GPT7 GTICODBN None GPT7 GPT8 GPT0 Module name:	RZV2H-EVK.pincfg			o CSV file 🗄 Config	ure Pin Driver Warnin	ngs
Pla Group Selection Mied Operation Mode Custom GPT1 GPT2 GPT3 GPT4 GPT5 GPT7 GPT7 GPT8 GPT9 GPT9 GPT9 GPT9 GPT0 GPT0 GPT2 GPT3 GPT0 GPT2 GPT2 GPT3 GPT2 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT2 GPT3 GPT2 GPT3 GPT2 GPT2 GPT3 GPT2 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT2 GPT3 GPT3 GPT2 GPT3 GPT3 GPT2 GPT3 GPT2 GPT3 GPT3 GPT2 GPT3	Pin Selection	E 🕀 🖻 🖓	Pin Configuration		😲 Cycle Pin Grou	up
	 ✓ Immer.GPT ✓ GPT0 GPT1 GPT3 GPT4 GPT5 GPT5 GPT7 		Pin Group Selection Operation Mode Input/Output GTIOC0A GTIOC0AN GTIOC0B GTIOC0B COBN	Mixed Custom * P70 None None		125
Pin Function Pin Number		~				
	Pin Function Pin Number					

Figure 76 : e2 studio Pin Configurator

When you click the **Generate Project Content** button, the pin configuration contents are written to: rzv_gen\pin_data.c. This file will be created if it does not already exist.

Warning: Do not edit this file as it is overwritten whenever the Generate Project Content button is clicked.



In the case of versions earlier than RZ/V FSP v2.0.0, It does not support **Pins** tab and If user would like to use I/O port, I/O Port setting should be applied to "src/pin_data.c" manually. For details on I/O Port setting and, please refer to <u>Setting GPIO with Flexible Software Package.</u>

5.2.5 Configuring Interrupts from the Stacks Tab

You can use the **Properties** view in the **Stacks** tab to enable interrupts by setting the interrupt priority. Select the driver in the **Stacks** pane to view and edit its properties.

🕸 (Blink)] FSP Configuration \times			- 0
Stacks	Configuration		Generati	e Project Content
Threads	🐔 New Thread 😭 Remove 📋	HAL/Common Stacks	🔊 New Stack > 🚊 Extend Stack	> 👔 Remove
4	AL/Common 9 gipper UP OPH Driver on r_geport 9 gitme2 Timer Driver on r_gem © New Object > © Remove	g ioport VO Port Driver on r_ioport	er2 Timee Drives trn	
	BSP Clocks Pins Interrupts Event Links S ties × 😰 Problems 🌸 Smart Browser			
	2 Timer Driver on r_gtm			
Settings	Property		Value	
	✓ Common			
	Parameter Checking		Default (BSP)	
	 Module g_timer2 Timer Driver on r_gtm 			
	> General			
	> Interrupts > Extra Features			

Figure 77 : Configuring Interrupts in the Stacks tab

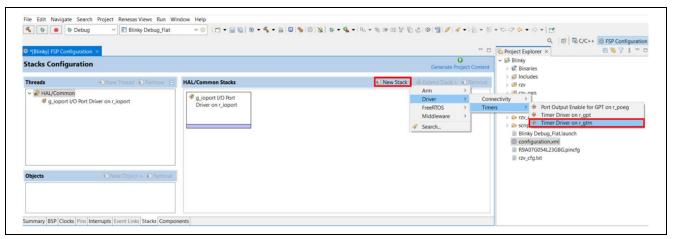


Figure 78: Add new stack Timer (GTM)



5.2.6 Creating Interrupts from the Interrupts Tab

On the **Interrupts** tab, the interrupt of the driver selected in the **Stacks** tab is registered.

(Blinky] FSP C	onfiguration 🛛		
Interrupts C	onfiguration		Generate Project Content
User Events			New User Event > 🔬 Remove
Event			ISR
Allocations			
Interrupt	Event		ISR
0	ID:46 GTM0 INT (GTM0 Interrupt)		gtm_int_isr
0 000 0			
Summary BSP C	locks Pins Interrupts Event Links Stacks Componen	5	



And on the Interrupts tab, the user can add a peripheral interrupt create by user. This can be done by adding a new event via the **New User Event** button.

5.2.7 Viewing Event Links

The Event Links tab can be used to view the Event Link Controller events. The events are sorted by peripheral to make it easy to find and verify them.

Event Links Configuration	1		Generate Project	Content
User Events Produced	🔊 New User Event > 🔬 Remove	User Events Consumed	🔊 New User Event 🕡 R	
User Events Floudced	Will new oser event > N. Remove	User Events Consumed	Ver New Oser Event N	
Event		Peripheral Function	Event	
GPT_U0_GPT_GTCIL_N_0 (GPT0 A a	nd B both low interrupt)	Single port 0	GPT_U0_GPT_GTCIL_N_0 (GPT0 A and B both low int	err
Allocations				
Peripheral Function		Event		^
Input port group 2 (Port 8)		No allocation		
Input port group 1 (Port 6)		No allocation		
Single port 0			(GPT0 A and B both low interrupt)	
Single port 1		No allocation		
Single port 2		No allocation		~

Figure 80 : e2 studio Project configurator – Viewing Event Links

Like the Interrupts tab, user-defined event sources and destinations (producers and consumers) can be defined by clicking the relevant **New User Event** button. Once a consumer is linked to a producer the link will appear in the Allocations section at the bottom.

Note1: When selecting an ELC event to receive for a module (or when manually defining an event link), only the events that are made available by the modules configured in the project will be shown.

Note2: On devices that do not have ELC, this tab is not available and it is grayed out.



5.2.8 Adding and Configuring HAL Drivers

For applications that run outside or without the RTOS, you can add additional HAL drivers to your application using the HAL/Common thread. To add drivers, follow these steps:

- 1. Click on the HAL/Common icon in the **Stacks** pane. The Modules pane changes to **HAL/Common** Stacks.
- 2. Click New Stack to see a drop-down list of HAL level drivers available in the FSP.
- 3. Select a driver from the menu **New Stack > Driver**.

*[Blinky] FSP Config	uration ×		(
itacks Configur			Generate Project Content → 😸 Blinky
Threads	New Object> Remove	HAL/Common Stacks	New Stack Arm Arm Arm Arm Connectivity Priver Connectivity Priver Pr

Figure 81 : e² studio Project configurator - Adding drivers

4. Select the driver module in the **HAL/Common Modules** pane and configure the driver properties in the **Properties** view.

e² studio adds the following files when you click the Generate Project Content button:

- The selected driver module and its files to the rzv/fsp directory
- The main() function and configuration structures and header files for your application as shown in the table below.

File	Contents	Overwritten by Generate Project Content?
rzv_gen/main.c	Contains main() calling generated and user code. When called, the BSP already has Initialized the MPU.	Yes
rzv_gen/hal_data.c	Configuration structures for HAL Driver only modules.	Yes
rzv_gen/hal_data.h	Header file for HAL driver only modules.	Yes
src/hal_entry.c	User entry point for HAL Driver only code. Add your code here.	No

The configuration header files for all included modules are created or overwritten in this folder: rzv_cfg/fsp_cfg



5.3 Reviewing and Adding Components

The **Components** tab enables the individual modules required by the application to be included or excluded. Modules common to all RZ/V MPU projects are preselected. All modules that are necessary for the modules selected in the **Stacks** tab are included automatically. You can include or exclude additional modules by ticking the box next to the required component.

Components Configuration		Generate	O Project Conte	ent
		Filter All V	earch	
Component	Version	Description	Variant	^
V 🔮 Board				
Custom		Custom Board Support Files		
rzv2l_smarc		Evaluation Kit RZ/V2L Support Files (RZ		
✓ ♥ rzv2l				
Ø device		Board support package for R9A07G054	R9A07G05	
device		Board support package for RZV2L		
device		Board support package for R9A07G054	R9A07G05	
device		Board support package for R9A07G054	R9A07G05	
device		Board support package for R9A07G054	R9A07G05	
✓ fsp		Board support package for RZ/V2L (RZ		
V 🕫 CMSIS				
V CMSIS5				
CoreM		Arm CMSIS Version 5 - Core (M)		
DSP		Arm DSP Library Source		
II NN		Arm NN Library Source		
🗸 🛷 Common				
🗸 🔮 all				
fsp_common		Board Support Package Common Files		
V 🛷 HAL Drivers				
🗸 🔮 all				
🖾 r_gpt		General PWM Timer		
☑ r_gtm		General Timer		
☑ r_ioport		I/O Port		~

Figure 82 : Components Tab

Clicking the **Generate Project Content** button copies the .c and .h files for each selected component into the following folders:

- rzv/fsp/inc/api
- rzv/fsp/inc/instances
- rzv/fsp/src/bsp
- rzv/fsp/src/<Driver_Name>

 e^2 studio also creates configuration files in the rzv_cfg/fsp_cfg folder with configuration options set in the **Stacks** tab.

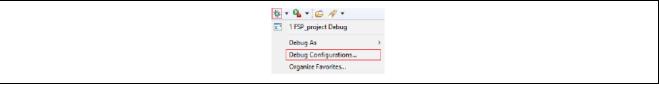


5.4 Debugging the Project

Once your project builds without errors, you can use the Debugger to download your application to the board and execute it.

To debug an application, follow these steps:

1. On the drop-down list next to the debug icon, select **Debug Configurations**.



2. In the Debug Configurations view, click on your project listed as MyProject Debug.

Create, manage, and run configurations
Bype filter ted. Main Startup Constraints C / C++ Application Project:
C C(++ Appliciation C
C OC++ Servers Application Meriopert ODE Stratuktor Debuggin C ODE Stratuktor Debuggin C ODE Stratuktor Debuggin C ODE Stratuktor Debuggin C ODE Stratuktor Debuggin D ODE St
FARE Sorger MyProject Browne CC 068 Hardware Debuggin Cole+ Applicatione Debuggin CO 080 QuenCQ Debuggin Debuggin Debuggin Cole+ Applicatione Debuggin Debuggin More Applet Debuggin Debuggin Java Applet Debuggin Debuggin Ball Configures Debuggin Debuggin Debuggin Statistics Debuggin Debuggin More Debuggin Debuggin Debuggin Statistics Debuggin Debuggin More Debuggin Debuggin Debuggin Statistics Debuggin Debuggin Debuggin Debuggin Debuggin Debuggin Debuggin Debuggin Debuggin Debuggin
C GOB Vardware Chebryghin G GOB Sindware Chebryghin G GOB Sindware Chebryghin G GOB Sindware Chebryghin Level Application Level Kongo (Expender Level Application) Level Kongo (Expender Application) Le
CODS Sinulator Debuggin Leasing for spectral Java Application Leasing for spectral Leasing for spectral Build of required bard for large comparison Leasing for spectral Build of required bard for large comparison Leasing for spectral Build of required bard for large comparison Decrease CDD Hordward Debug Decrease CDD Hordward Debug Process Smulter Debug Obset and build
Java Applet Java Appletsin Lunack Group Lunack Group Lunack Group El Lunack Group El Lunack Group El Lunack Group El Lunack Group El Lunack Group El Lonack Group El
Rendet Iva Application C Endele auto build Disable auto build C Endele auto build C Endel
Reyert Apply

3. Secure and Non-secure Vector Address are configured in the Connection Settings tab of the Debugger tab. The settings in below image are for setting the address of Secure and Non-secure Vector Offset (*) mapped in Blinky project. Please note that these addresses vary in accordance with linker settings. (*: In the case of RZ/V2H project, Non-secure Vector is empty since it is not used.)

Debug Configurations			– 🗆 X		
Create, manage, and run configu	irations		-	Filter	
				Symbol	Address ^
			2	 SystemInit_S 	0x72eff549
				UsageFault_Handler_NS	0x6001084d
	Name: Blinky Debug_Flat			UsageFault_Handler_S	0x72eff535
type filter text	📄 Main 🎋 Debugger 🍉 Startup 🔲 Comm	on 4/ Source		Marm Porat S	0.72-6691
C/C++ Application				_Secure_Vectors	0x1001ff80
C C/C++ Remote Application	Debug hardware: J-Link ARM V Target De	vice: R9A07G044L23GBG_(Vectors	0x00010000
EASE Script				bsp_clock_init_veneer	0x72eff5b9
GDB Hardware Debugging	GDB Settings Connection Settings Debug T	pol Settings		bsp_clock_freq_var_init	0x600107b1
GDB OpenOCD Debugging	✓ JTAG Scan Chain		^	bsp_clock_init	0x60010835
GDB Simulator Debugging (I	Multiple Devices	No	*	bsp_init	0x600108d9
🜌 Java Applet	IRPre	0			
Java Application	DRPre	0		_	
🙀 Launch Group	✓ Connection				OK Cancel
Remote Java Application	Register initialization	No	~		
✓ CT Renesas GDB Hardware Deb	Reset on connection	No	×		
C [®] Blinky Debug_Flat	Reset before run	No	~		
C [*] Renesas Simulator Debuggir	ID Code (Bytes)	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF		Select Symbol	×
	Hold reset during connect	No	~		
	Reset before download	No	~	Filter	
	Prevent Releasing the Reset of the CM3 Secure Vector Address	Lore Yes	<u> </u>		
	Non-secure Vector Address			Symbol	Address ^
	✓ SWV			UsageFault_Handler_NS	0x6001084d
	Core clock (MHz)	0		UsageFault_Handler_S	0x72eff535
	✓ TrustZone			Warm_Reset_S	0x72eff581
	Set TrustZone secure/non-secure bound	laries No	× .	Serve Vester	0-1001#80
			÷	_Vectors	0x00010000
				_bsp_clock_init_veneer	0x72e#5b9
< >		Reye	rt Apply	bsp_clock_freq_var_init	0x600107b1
Filter matched 13 of 15 items		neve	nt AbbiX	bsp_clock_init	0x60010835
·				bsp_init	0x600108d9
		Det	Close	bsp_irg_cfg	0x60010849
0		Rec.	core		
?					

4. Connect the board to your PC via a standalone Segger J-Link debugger and click **Debug**. **Note**: For details on using J-Link and connecting the board to the PC, see 3.1.2.2.JTAG connection.



5.5 Modifying Toolchain Settings

There are instances where it may be necessary to make changes to the toolchain being used (for example, to change optimization level of the compiler or add a library to the linker). Such modifications can be made from within e² studio through the menu **Project > Properties > Settings** when the project is selected. The following screenshot shows the settings dialog for the GNU Arm toolchain. This dialog will look slightly different depending upon the toolchain being used.

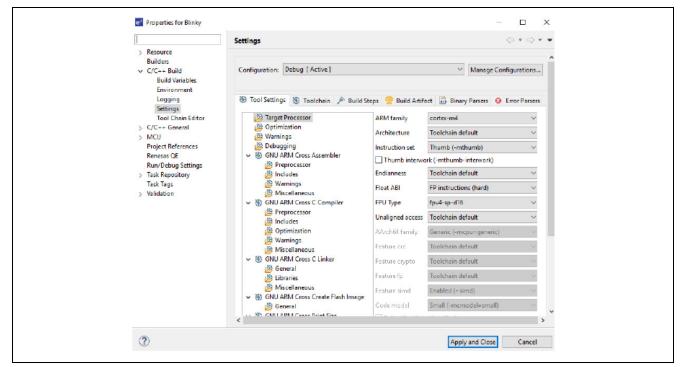


Figure 83 : e² studio Project toolchain settings

The scope for the settings is project scope which means that the settings are valid only for the project being modified.

The settings for the linker which control the location of the various memory sections are contained in a script file specific for the device being used. This script file is included in the project when it is created and is found in the script folder (for example, /script/fsp.ld).



5.6 Importing an Existing Project into e² studio

- 1. Start by opening e² studio.
- 2. Open an existing Workspace to import the project and skip to step d. If the workspace does not exist, proceed with the following steps:
 - a. At the end of e² studio startup, you will see the Workspace Launcher Dialog box as shown in the following figure.

e ² Eclipse Launcher	×
Select a directory as workspace	
e ² studio uses the workspace directory to store its prefere	ences and development artifacts.
Workspace: C\Users\ <user_name\e2studio\workspace< th=""><th>✓ <u>B</u>rowse</th></user_name\e2studio\workspace<>	✓ <u>B</u> rowse
Use this as the default and do not ask again	
<u>R</u> ecent Workspaces	
	Launch Cancel

Figure 84 : Workspace Launcher dialog

b. Enter a new workspace name in the Workspace Launcher Dialog as shown in the following figure. e² studio creates a new workspace with this name.

Clipse Launcher X
Select a directory as workspace
e ² studio uses the workspace directory to store its preferences and development artifacts.
Workspace: C\Users\ <username>\e2studio\new_workspace</username>
Use this as the default and do not ask again Recent Workspaces
Launch Cancel

Figure 85 : Workspace Launcher dialog - Select Workspace

- c. Click Launch.
- d. When the workspace is opened, you may see the Welcome Window. Click on the **Workbench** arrow button to proceed past the Welcome Screen as seen in the following figure.

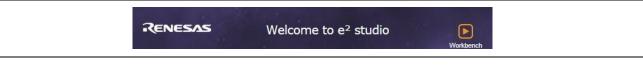


Figure 86 : Workbench arrow button

3. You are now in the workspace that you want to import the project into. Click the **File** menu in the menu bar, as shown in the following figure.



Figure 87 : Menu and tool bar



4 Click Import on the File menu or "Import project" on Project Explorer, as shown in the following figure.

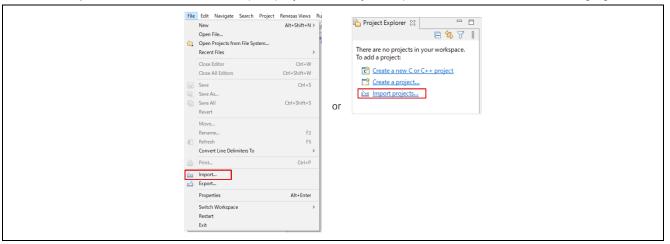


Figure 88 : File drop-down menu

5. In the **Import** dialog box, as shown in the following figure, choose the **General** option, then **Existing Projects into Workspace**, to import the project into the current workspace.

S Import - X
Select
Create new projects from an archive file or directory.
Select an import wizard:
type filter text
 ✓ Seneral Archive File CMSIS Pack Existing Projects into Workspace File System GRUARM-NONE/RZ(DS-5) project conversion to GCC ARM Embedded Preferences Projects from Folder or Archive Rename & Import Existing CC++ Project into Workspace Reneas Web97/h_D0977/h_7D9527/h Reneas CA78K0R (CS+) 7D9527/h Reneas CC-RX/CC-RL (CS+) 7D9527/h Reneasa GitHub FreeRTOS (with IoT libraries) 7D9527/h > C/C++ Install
? < Back Next > Finish Cancel

Figure 89 : Project Import dialog with "Existing Projects into Workspace" option selected

- 6. Click Next.
- 7. To import the project, use either Select archive file or Select root directory.



a. Click **Select root directory** file as shown in the following figure.

Import Import Projects Select a directory to search for existing Eclipse projects. Select archive file Select archive file Projects	
Select a directory to search for existing Eclipse projects.	
Select root directory: Browse Select archive file: Browse	
O Select archive file:	
O Select archive file:	
Projects:	
Select All	
Destlect All	
Refresh	
Options Search for nested projects	
Copy projects into workspace	
Gase newly imported projects upon completion	
Hide projects that already exist in the workspace	
Working sets	
Add project to working sets New	
Working sets: v Select	
(?) < Back Next > Finish Cancel	
Concernance	

Figure 90 : Import Existing Project dialog 1 - Select root directory

- 8. Click Browse.
- 9. For **Select root directory**, browse to the project folder that you want to import.
- 10. Select the file for import.
- 11.Click Open.
- 12. Select the project to import from the list of **Projects**, as shown in the following figure.

Import		-		×
Import Projects Select a directory to sea	rch for existing Eclipse projects.			
 Select root directory: Select archive file: Projects: 	C:\works_RZV\work\Blinky	~		owse
Blinky (C:\works_F	λΖV\work\Blinky)		Dese	ect All elect All efresh

Figure 91 : Import Existing Project dialog 2

13. Click **Finish** to import the project.



6. Multi-Core Debug

In the case of RZ/V2H, FSP supports multi-core (CM33 core, CR8 core0 and CR8 core1) operation.

This section describes how to debug multi-core environment by running Blinky project for each core.

6.1 **Project Creation for each Core**

1. Create workspace for each core.

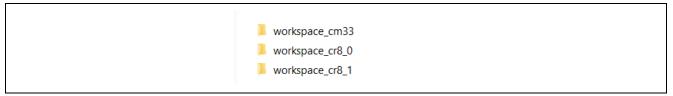


Figure 92 : Workspace creation

2. Launch e2studio and specify workspace for CM33 core.

e² studio Launcher	- [×
Select a directory as workspace			
\mathbf{e}^{2} studio uses the workspace directory to store its preferences and development artifacts.			
vworkspace_cm33	~	<u>B</u> rov	vse
Use this as the default and do not ask again			
<u>R</u> ecent Workspaces			
Launch		Cancel	

Figure 93 : Launching e2studio for CM33 Core Workspace

3. Generate new project for CM33 core in accordance with section 4.3. In the case of CM33 core project, ensure to select **Core** and **Project Template Selection** as below.

Renesas RZ/V C/C++ FSP Project	- 0	X 📴 Renesas RZ/V C/C++ FSP Project – 🗆 X
Renesas RZ/V C/C++ FSP Project Device and Tools Selection		Renesas RZ/V C/C++ FSP Project Project Template Selection
Device Selection FSP Version: 20.1 Board: RZ/V2H Evaluation Kit Device: R9A090057H446BG Core: Core 6(CM33.0)	Board Description	Project Template Selection Project Template Selection Bare Metal (CM33) - Blinky Bare metal FSP project of Contex-M33 core that includes BSP and will blink LEDs if available. This project will initialize clocks, pins, stacks, and the C nutrime environment. (Renesas RZV2.0.1 pack)
Language: @C OC++	Device Details TrustZone No Pins 1368 Processor Contex-M33	Bare Metal (CM33) - Minimal Bare metal ISP project of Cortex-M33 core that includes BSP. This project will initialize clocks, pins, stacks, and the C runtime environment. IRenesas RZV2.0.1 pack]
Toolchains	Debugger	
GNU ARM Embedded 12.2.1.arm-12-24 V Menage Toolchains.	J-Link ARM	Code Generation Settings
0	< <u>B</u> ack <u>N</u> ext > Einish Car	4 (Back Next > Emish Cancel

Figure 94 : Setting of CM33 core project

4. Build the Blinky project for CM33 core.



 Generate new project for CR8 core0 in accordance with section 4.3. In the case of CR8 core0 project, ensure to select Core, Preceding Project, and Project Template Selection as below.

Renesas RZ/V C/C++ FSP Project	— 🗆 X	Renesas RZ/V C/C++ FSP Project	- 🗆 ×
Renesas RZ/V C/C++ FSP Project	-	Renesas RZ/V C/C++ FSP Project	
Device and Tools Selection		Preceding Project or Smart Bundle Selection	
Device Selection	Board Description	Preceding Project: Blinky_cm33	,
FSP Version: 2.0.1	Soard Description	Choose this option if you have access to the project source code of the preceding pr	processor core or security context.
Board: RZ/V2H Evaluation Kit ~		O Smart Bundle: Resolved location:	
			File System Variables
Core: Core 4(CR8_0)	Device Details TrustZone No	Choose this option if you only have access to a Smart Bundle describing the configu	
Language: C C++	Pins 1368	core or security context.	
	Processor Cortex-R8	Preceding Project/Smart Bundle Details	
		FSP version 2.0.1 Toolchain GNU ARM Embedded	
Toolchains	Debugger	Toolchain version 12.2.1.arm-12-24	
GNU ARM Embedded	J-Link ARM	Board RZ/V2H Evaluation Kit Device R9A09G057H44G8G	
		Core CM33_0	
12.2.1.arm-12-24 V Manage Toolchair	<u>15</u>	Zones CM33_0_S	
	Reneass RZ/V C/C++ FSP Project Project Template Selection Project Template Selection Bare Metal (CR8 core0) - Blinky Bare Metal (SP project of Corex-RB cored that includes BSP and clock, pins, stack, and the C runtime environment.	will blink LEDs if available. This project will initialize	
	[Renesas:RZV.2.0.1.pack] Image: Cross-RZV.2.0.1.pack] Image: Cross-RZV.2.0.1.pack]	i project will initialize clocks, pins, stacks, and the C	
	Code Generation Settings Use Renesas Code Formatter		
	?	k Next> Einish Cancel	

Figure 95 : Setting of CR8 core0 project

6. Build the Blinky project for CR8 core0.



 Generate new project for CR8 core1 in accordance with section 4.3. In the case of CR8 core1 project, ensure to select Core, Preceding Project, and Project Template Selection as below.

Renesas RZ/V C/C++ FSP Project	— 🗆 X	Renesas RZ/V C/C++ FSP Project	
Renesas RZ/V C/C++ FSP Project		Renesas RZ/V C/C++ FSP Project	
Device and Tools Selection		Preceding Project or Smart Bundle Select	ction
Device Selection		Preceding Project: Blinky_cr8_0	
FSP Version: 2.0.1	Board Description		on if you have access to the project source code of the preceding processor core or security context.
a company and a straight strai		O Smart Bundle:	,,
		Resolved location	
	Device Details		Workspace File System Variables
Core: Core 5(CR8_1) ~	TrustZone No	Choose this optio core or security co	on if you only have access to a Smart Bundle describing the configuration of the preceding processo
Language: C C++	Pins 1368		
	Processor Cortex-R8	Preceding Project/Smart Bundle Details FSP version	2.0.1
	< >>	Toolchain	GNU ARM Embedded
Toolchains	Debugger	Toolchain version	12.2.1.arm-12-24
GNU ARM Embedded	J-Link ARM 🗸	Board Device	RZ/V2H Evaluation Kit R9A09G057H44GBG
		Core	CR8_0
12.2.1.arm-12-24 V Manage Toolchains		Zones	CM33_0_S, CR8_0_S
	Code Generation Settings Code Generation Sett		
	0	< Back Next > Einish	Cancel

Figure 96 : Setting of CR8 core1 project

- 6. Build the Blinky project for CR8 core1.
- 3 projects are created in CM33 core workspace. Copy all projects to CR8 core0 workspace and CR8 core1 workspace.

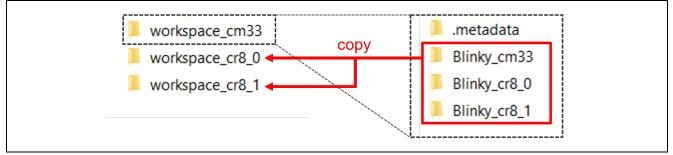


Figure 97 : Copy the project



6.2 Debugging the Project for each Core

1. Select the **Debug Configuration** on e2studio of CM33 core workspace and click **Debug**.

	Debug Configurations		
	Create, manage, and run configur	rations	Ť.
Fun Window Help Reneas Device Partition Manager (2) (2) Tracext >> (3) Traceatyzer >> (4) Run A (5) Debug (7) Run A (8) Run	P P	Name: Bilinky_cm33 Debug.Flat Main: Debugger Startup Source Common Broject: Bilinky_cm33 C/C++ Application: DebugBinky_cm33.eff Build Configuration; Use Active C Enable auto build Use workspace Settings Configure Workspace Settings.	Browse Browse
	< >> Filter matched 11 of 13 items	Reyort	Apply
	?	Debug	Close

Figure 98 : CM33 Core Debug Configuration

2. Launch new e2stuio for CR8 core0 workspace.

e² studio Launcher	_		×
Select a directory as workspace			
e ² studio uses the workspace directory to store its preferences and development artifacts.			
\workspace_cr8_0		Bro	wse
Use this as the default and do not ask again			
<u>R</u> ecent Workspaces			
Launch		Cance	I

Figure 99 : Launching e2studio for CR8 Core0 Workspace

3. Select the **Debug Configuration** on e2studio of CR8 core0 workspace and click **Debug**.

	Debug Configurations Create, manage, and run configurations	×
Run Window Help Reneass Device Partition Manager ItraceX Tracealyzer TraceX Patternia Run Ctrl+F1 Debug F11 Run Kintory > Run Configurations > Debug As > Debug Configurations > Lebug Configurations. >	Taure and to cool	Startup Common Startup Common Startup Common Startup Source Browse Browse Browse Browse Browse Constants
	<	Regett Apply Debug Close

Figure 100 : CR8 Core0 Debug Configuration



4. Launch new e2stuio for CR8 core1 workspace.

👩 e ^z studio Launcher	_		×	
Select a directory as workspace				
e ² studio uses the workspace directory to store its preferences and deve artifacts.	lopment			
۰ /workspace_cr8_1		Brow	/se	
Use this as the default and do not ask again				
▶ <u>R</u> ecent Workspaces				
	aunch	Cancel		

Figure 101 : Launching e2studio for CR8 Core1 Workspace

5. Select the Debug Configuration on e2studio of CR8 core1 workspace and click Debug.

Run Window Help Renesas Device Partition Manager Image: Startup Name: Blinky_cr8_1 Debug_Flat Renesas Device Partition Manager Image: Startup Con If TaceX C/C++ Application Project Image: TaceX C/C++ Application Project Image: Script CGDB Hardware Debugging Project Image: Script CGDB Simulator Debugging C/C++ Application: Image: Script Debug/Blinky_cr8_1 Leff	
Run A listory > Run Configurations. > Debug History > * Debug Sconfigurations. > Debug Configurations. > Blinky.cr8 0 Debug.Flat Blinky.cr8 0 Debug.Flat Blinky.cr8 1 Deb	Browse_ Variables Search Project Bgowse_ O Disable auto build Configure Workspace Settings Reyert Apply

Figure 102 : CR8 Core1 Debug Configuration

6. You can debug each core project.



Revision History

	Date	Description		
Rev.		Page	Summary	
2.01	Aug, 07 .24	6 to 20	Updated the description and figure based on the latest	
		29 to 32	development environment.	
		39 to 43		
		45, 49,		
		57 to 59		
2.00	May,31 .24	1	Added RZ/V2H to the target device.	
		4 to 20	Updated the description and figure based on the latest	
		29 to 31	development environment.	
		41 to 43		
		46		
		25 to 28	Added description and figure for RZ/V2H EVK.	
		40	Updated the description of project creation.	
		44	Added the section of how to avoid resource duplication	
			description in the case of RZ/V2H project creation.	
		47 to 48	Added description about the Pins tab.	
		58 to 62	Added description about the multi-core debugging of RZ/A1H.	
1.10	Jan,31.23	5 to 10	Updated e2sutio version to install for Windows PC and	
			changed images of e2studio installation	
		16 to 20	Updated e2studio version to install for Linux PC and changed	
			images of e2studio installation	
		23 to 24	Updated pack version to install	
		34 to 36	Changed Chapter 4.6 to 4.9 to a sub-chapter of Chapter 4.5	
			(These have been changed to Chapters 4.5.1 to 4.5.4)	
		35, 49	Updated the method of specifying Secure Vector Address and	
			Non-secure Vector Address	
1.00	Jan.14.22	-	First Edition issued	



General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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. Precaution against Electrostatic Discharge (ESD)

A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on

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

3. Input of signal during power-off state

Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. 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 the 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.

5. Clock signals

After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.

6. Voltage application waveform at input pin

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).

7. Prohibition of access to reserved addresses

Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products

Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 systemevaluation test for the given product.

Notice

- 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
- Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
- 3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
- 5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
- Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended 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; industrial robots; etc.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.

- 7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
- 8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
- 9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
- 10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
- 11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
- 12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
 Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.
- (Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
- (Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.