RZ/G2L RZ/G2LC RZ/G2UL RZ/G3S RZ/G3E

Getting Started with Flexible Software Package

Introduction

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

Target Device

RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/G3S, RZ/G3E

Contents

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) incorporated on RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/G3S and RZ/G3E.

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/G2L, RZ/G2LC, RZ/G2UL, RZ/G3S and RZ/G3E have a multi-core configuration of Cortex®-A55 (hereinafter referred to as CA55) and CM33. It is possible to use each peripheral and GPIO from each core. This package provides drivers for the peripheral for CM33, 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 the software 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/G FSP, start with each Evaluation Board Kit.
Start-up guide of each Evaluation Board Kit is available at the below page.

- RZ/G2L
  RZ/G2L Evaluation Board Kit Quick Start Guide
- RZ/G2LC and RZ/G2UL
  SMARC EVK of RZ/G2L, RZ/G2LC, RZ/G2UL Linux Start-up Guide
- RZ/G3S
  RZ/G3S Evaluation Board Kit Quick Start Guide
- RZ/G3E
  www.renesas.com/rzg3e-evk

2.1.2.2 PC Requirements

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

- Windows 10 or Ubuntu 20.04 LTS Desktop(64-bit) 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: https://www.renesas.com/e² studio. 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 will be selected. Then, click **[Next >]** button 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

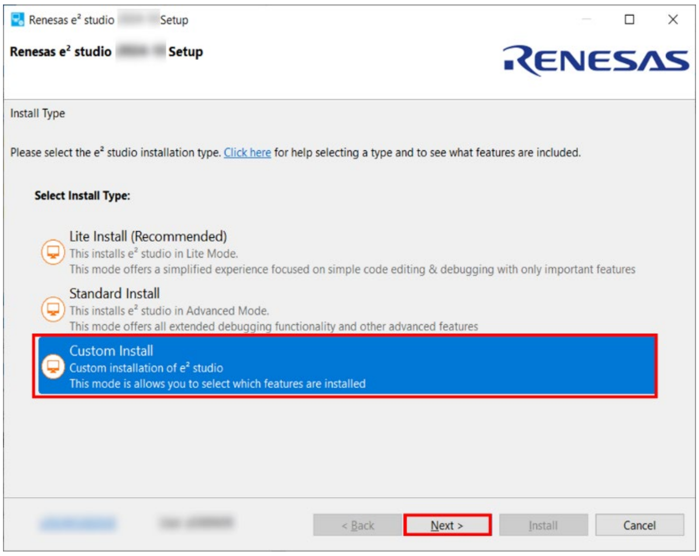


Figure 1: Installation of e² studio – Install Type

2. Welcome page
   User can change the install folder by clicking **[Change…]** button. Click **[Next >]** button to continue.

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

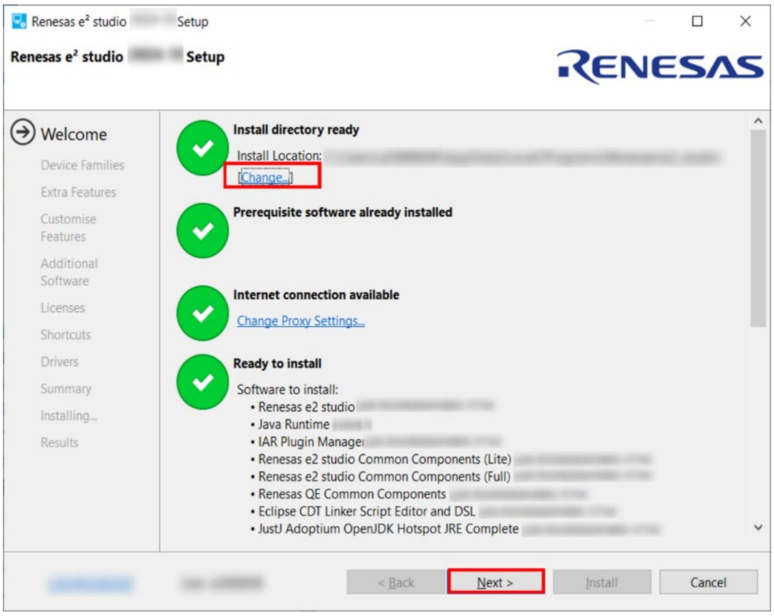


Figure 2: Installation of e² studio – Welcome page

3. Device Families
   Select Devices Families to install. Click **[Next >]** button to continue.

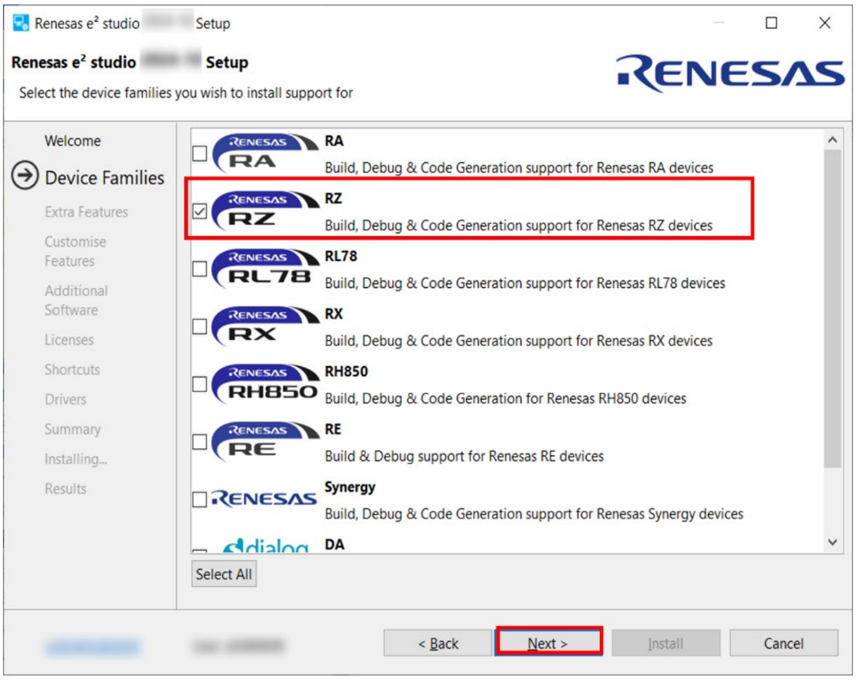


Figure 3: Installation of e² studio – Device Families

4. Extra Features
   Select Extra Features (e.g., Language packs, SVN & Git support…) to be installed. For non-English language users, please select Language packs at this step if needed. Then, click **[Next >]** button to continue.

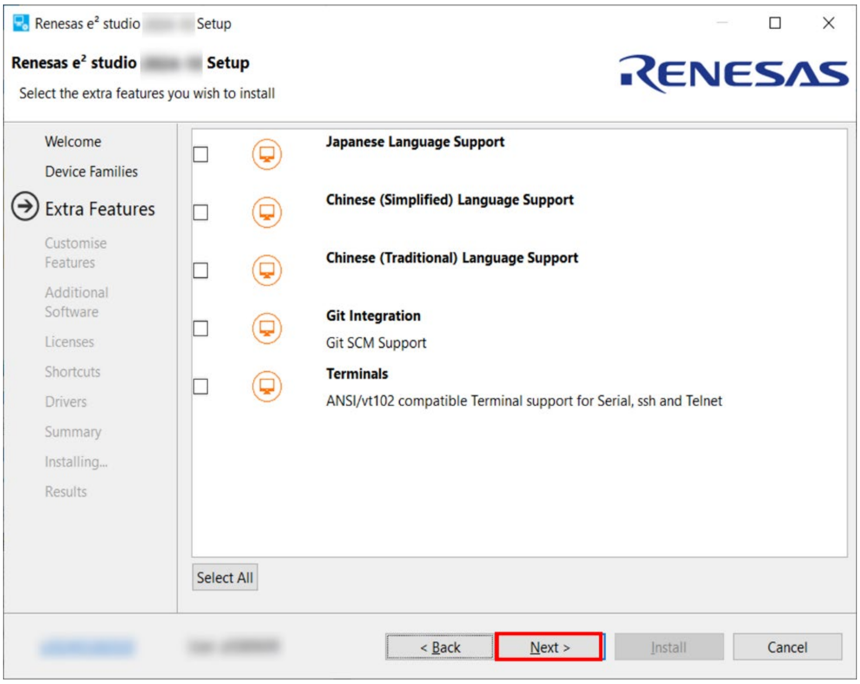


Figure 4: Installation of e² studio – Extra Features

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

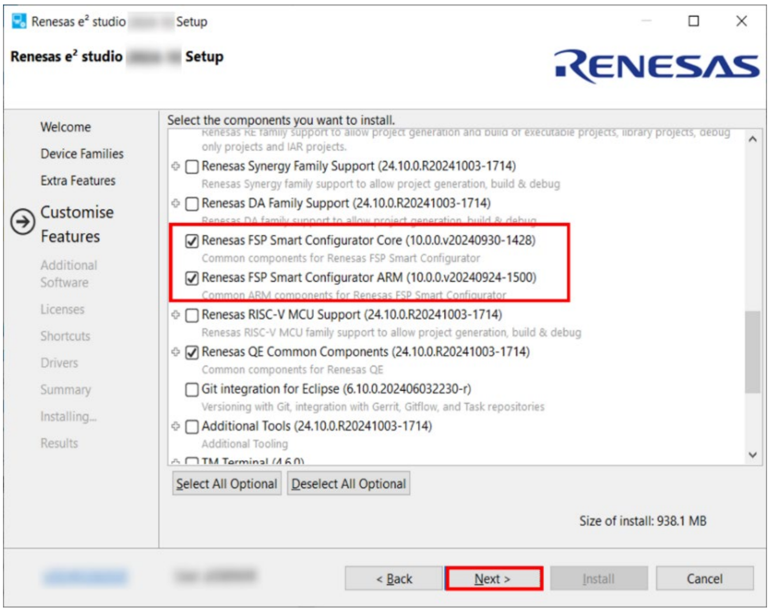


Figure 5: Installation of e² studio – Customize Features

6. Additional Software
   Select additional software (i.e., compilers, utilities, QE…) to be installed. Be sure to select the following item and click **[Next >]** button to continue.

   - GNU ARM Embedded 13.3-Rel1

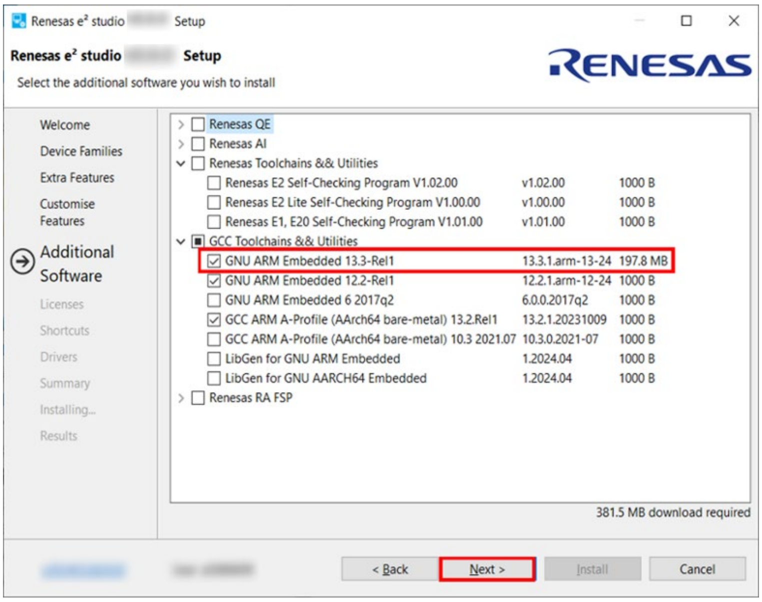


Figure 6: Installation of e² studio – Additional Software

For more details on the installation of Additional Software, please see section 2.1.3.3.

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

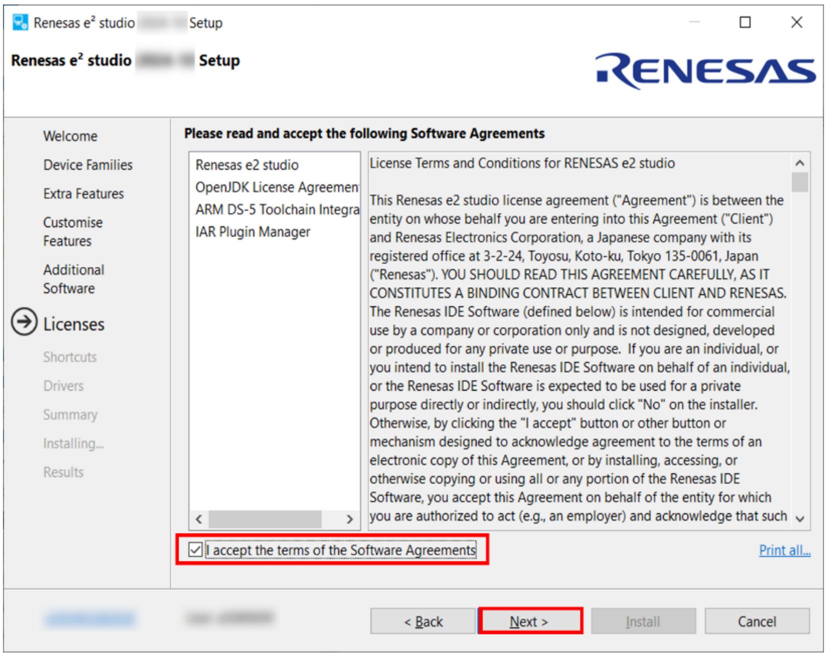


Figure 7: Installation of e² studio – Licenses

8. Shortcuts
   Select shortcut name for start menu and click **[Next >]** button to continue.

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

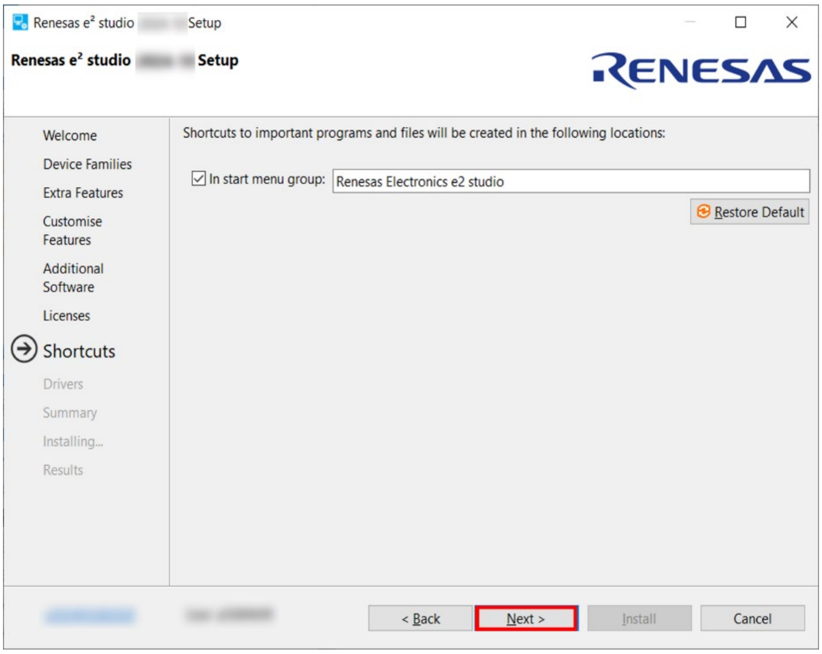


Figure 8: Installation of e² studio – Shortcuts

9. Summary

   Components list to be installed is shown. Please confirm the contents and click **[Install]** button to install the Renesas e² studio IDE.

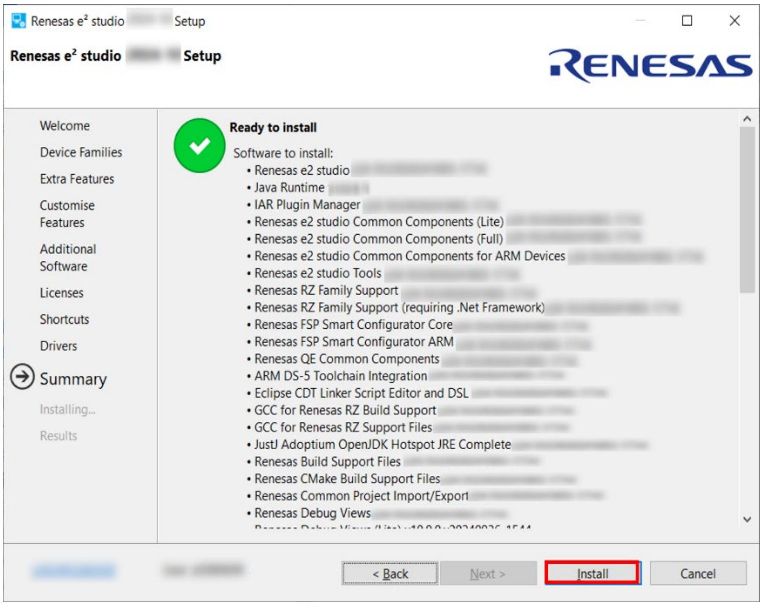


Figure 9: Installation of e² studio – Summary

10. Installing...

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

11. Results

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

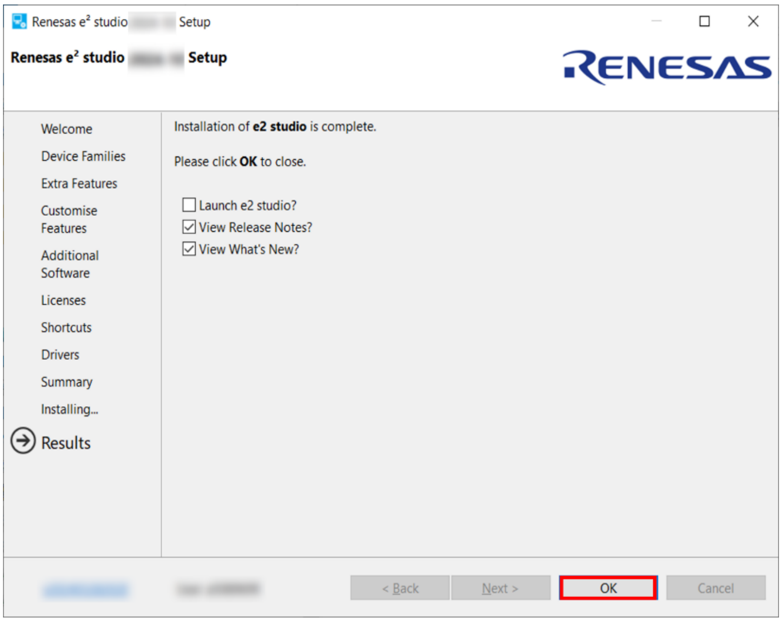


Figure 10: Summary Page

2.1.3.3 Installation of Additional Software

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

- GNU ARM Embedded 13.3-Rel1.

In this section, the detailed procedure for installing these tools.

- **GNU ARM Embedded Toolchain 13.3-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.

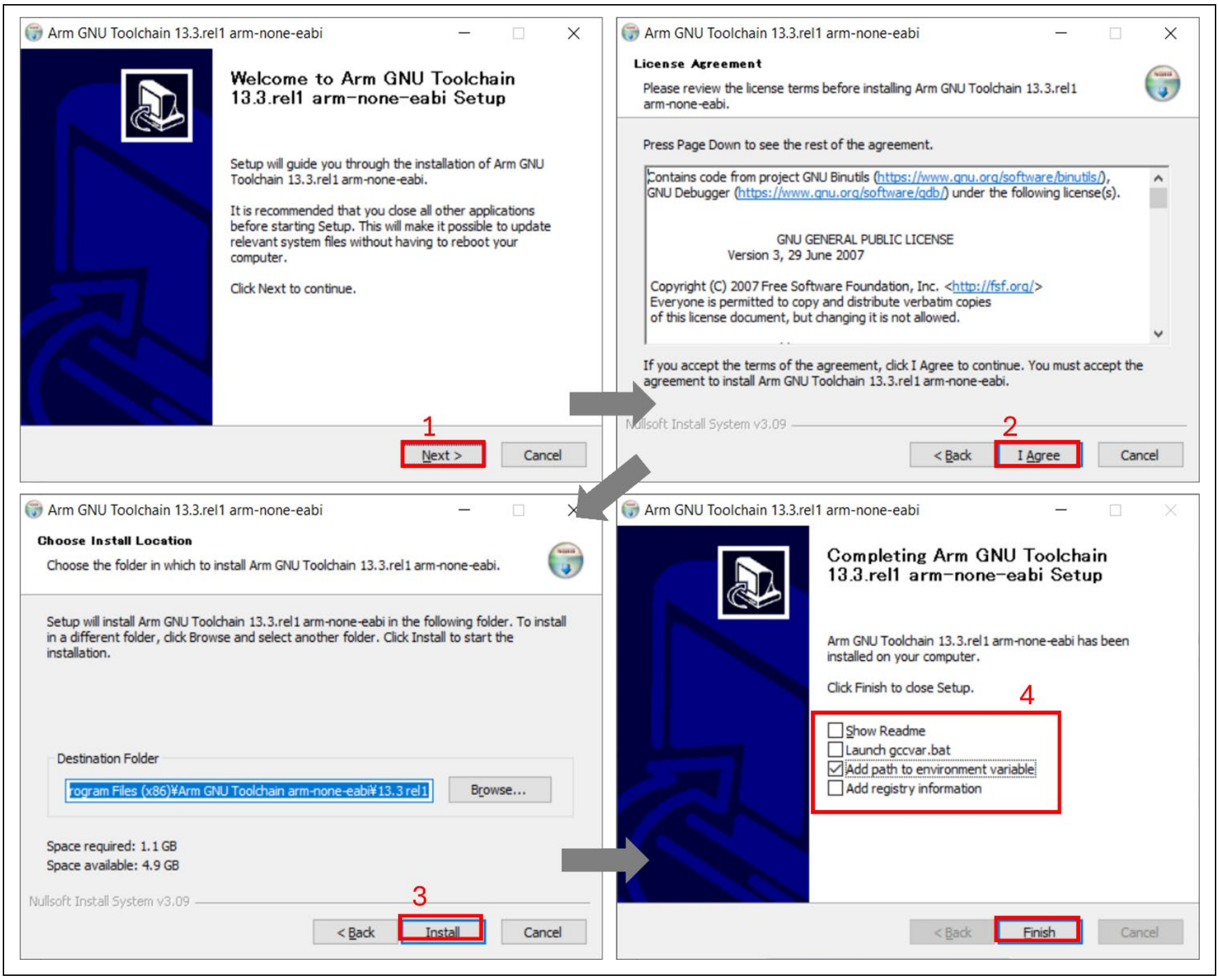


Figure 11: Installation of GNU ARM Embedded Toolchain

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 choose the version V7.96j or later and download Linux 64-bit DEB Installer at the URL below:
  https://www.segger.com/downloads/jlink/
- e² studio IDE installer
  e² studio IDE installer package can be downloaded from Renesas website for free. Please check detailed information from: https://www.renesas.com/e² studio.

2.1.4.2 Installation

This section describes the procedure of each software installation.
Filename, version number and the file path are provided for example purpose only.

- SEGGER J-Link driver

  1. Open a terminal window and enter commands stated below.
     sudo dpkg -i JLink_Linux_V796j_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_V796j_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-2025-01_linux_host.run

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

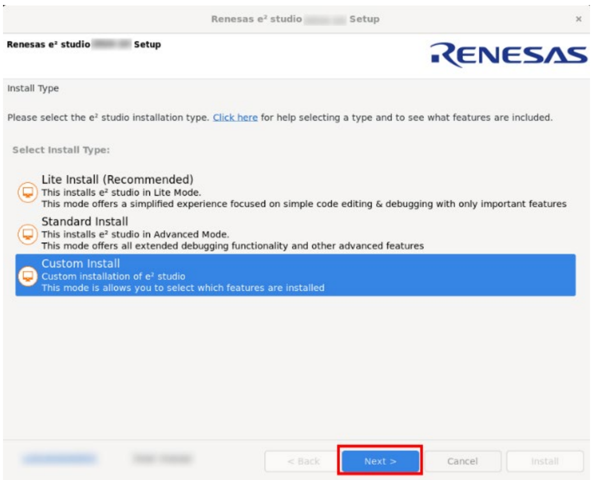


Figure 12: Selection of Install Type

3. **Welcome page (Cont'd)**
   User can change the install folder by clicking **[Change…]** button.
   Click **[Next >]** button 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.

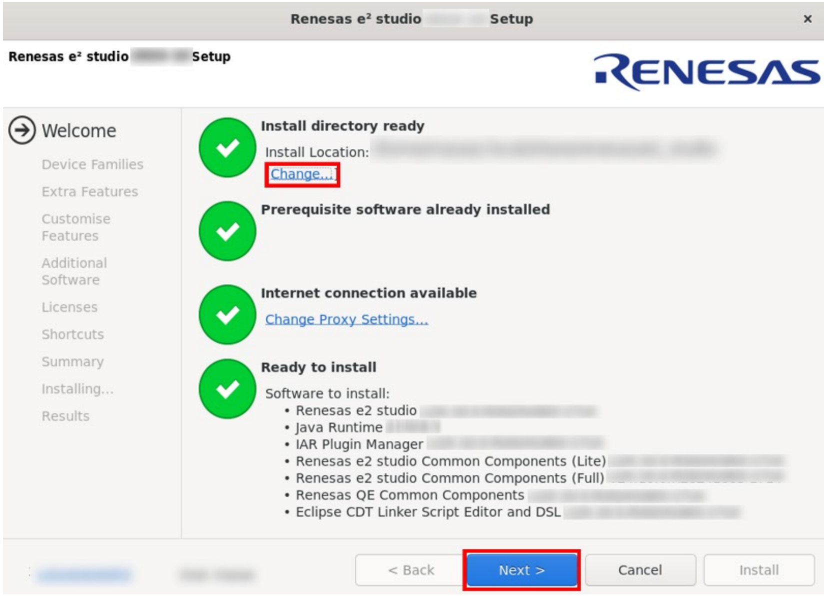


Figure 13: Installation of e² studio – Welcome page

4. **Device Families**
   Select Devices Families to install. Click **[Next >]** button to continue.

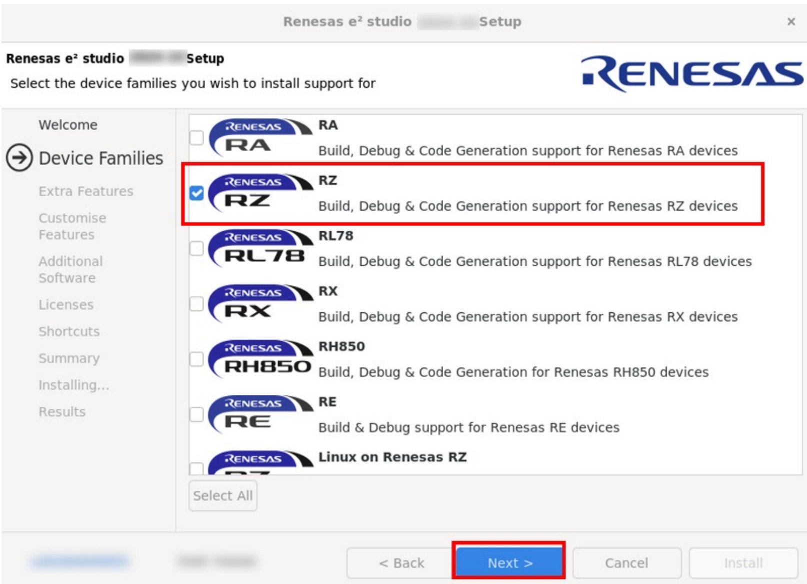


Figure 14: Installation of e² studio – Device Families

5. **Extra Features**
   Select Extra Features (e.g., Language packs, SVN & Git support…) to be installed.
   For non-English language users, please select Language packs at this step if needed.
   Then, click **[Next >]** button to continue.

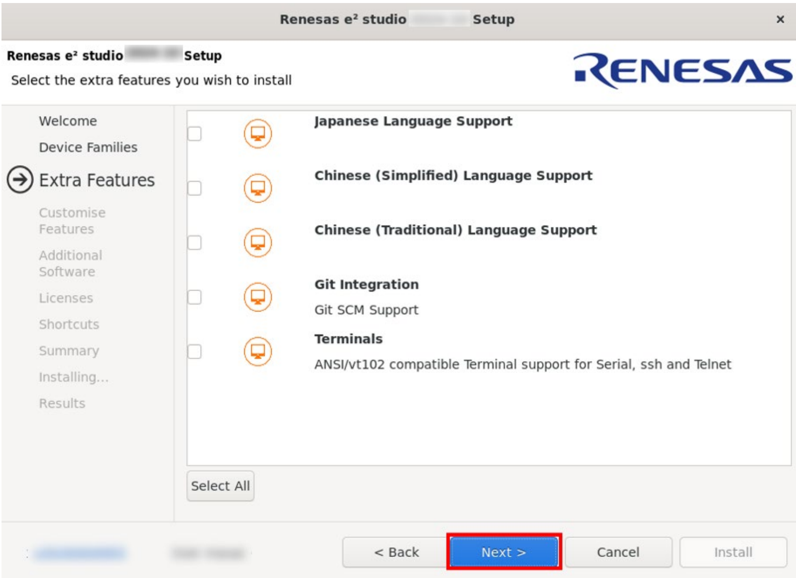


Figure 15: Installation of e² studio – Extra Features

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

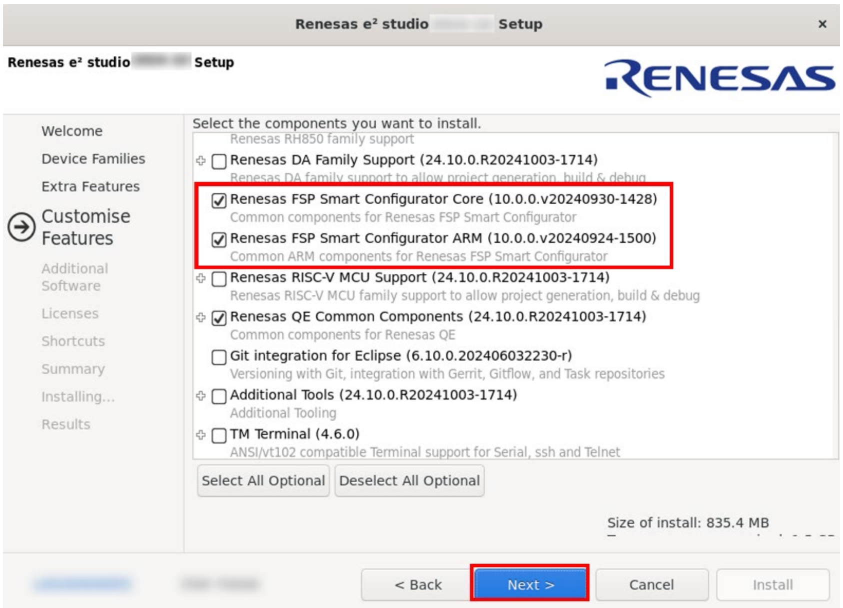


Figure 16: Installation of e² studio – Customize Features

7. **Additional Software**
   Select the additional software (i.e., compilers, utilities, QE…) to be installed.
   Be sure to select the "GNU ARM Embedded 13.3-Rel1" and click **[Next >]** button to continue.

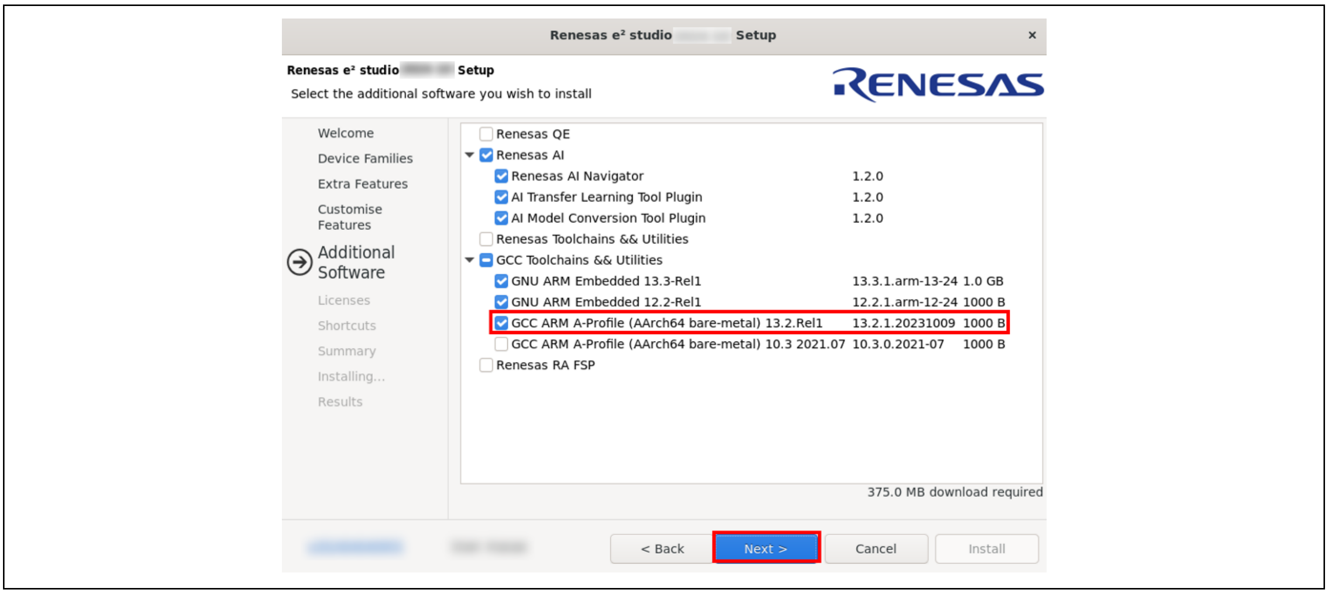


Figure 17: Installation of e² studio – Additional Features

8. **License Agreement**
   Read and accept the software license agreement. Click **[Next >]** button to continue.
   Please note that the user must accept the license agreement, otherwise installation cannot be continued.

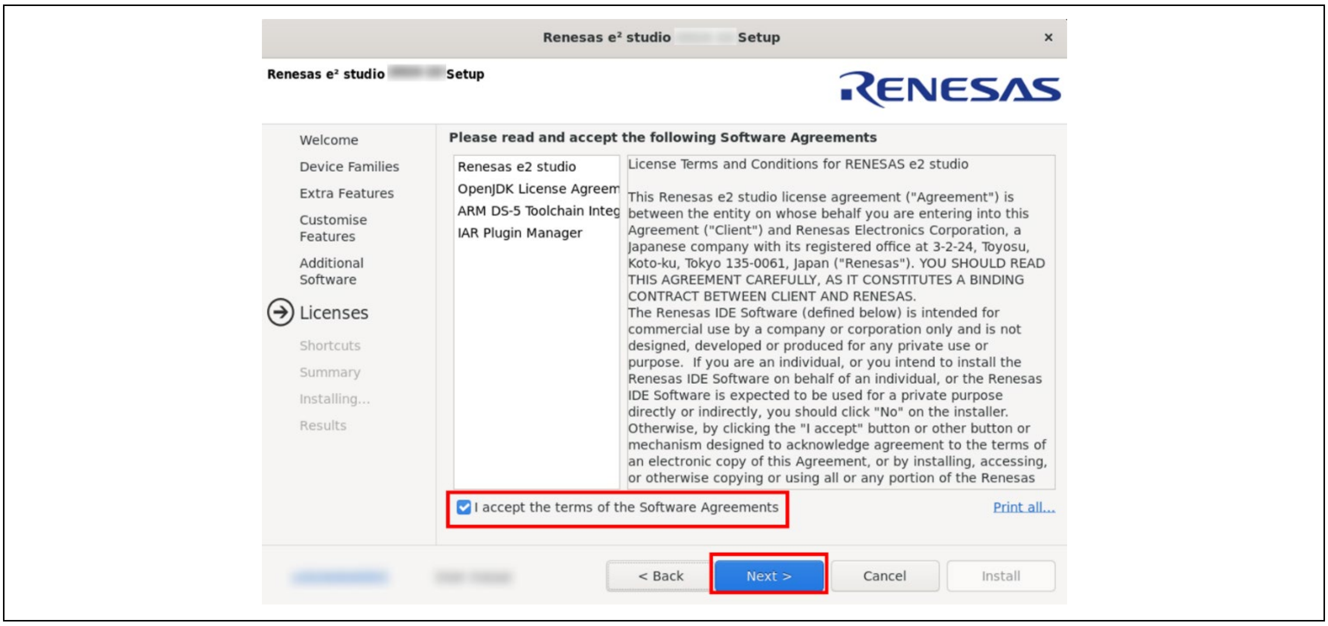


Figure 18: Installation of e² studio – Licenses

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).

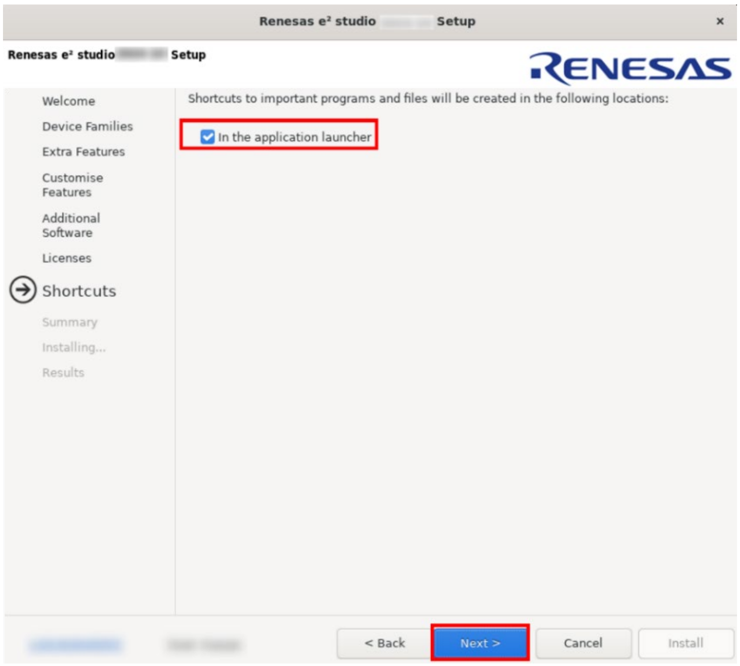


Figure 19: Installation of e² studio – Shortcuts

10. Summary
    Components list to be installed is shown. Please confirm the contents and click **[Install]** button to install the Renesas e² studio IDE.

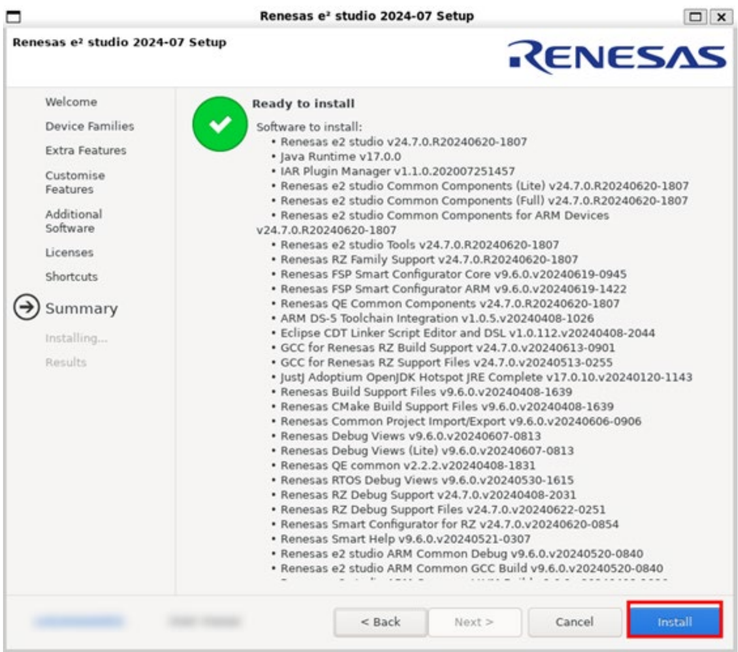


Figure 20: Installation of e² studio – Summary

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 **[OK]** button to complete the installation.

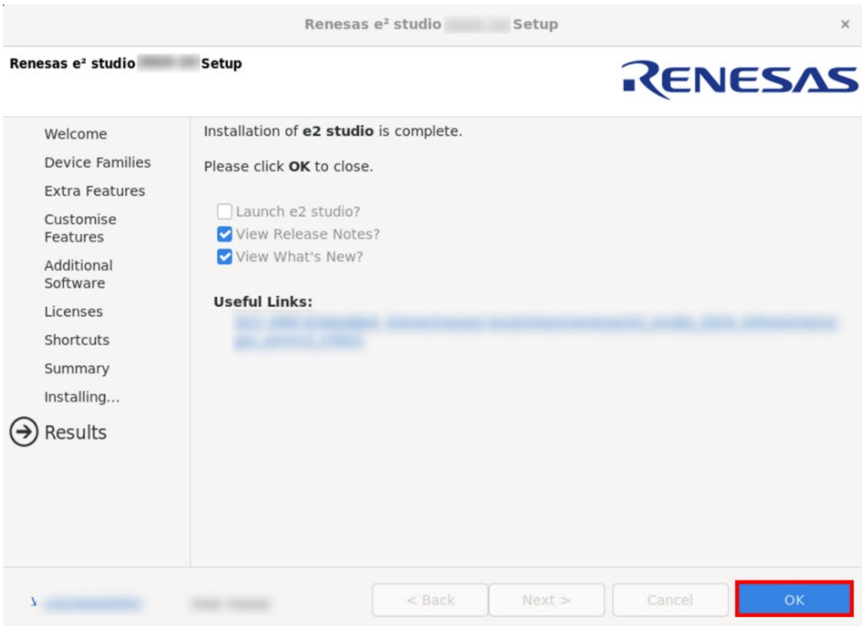


Figure 21: Installation of e² studio – Results

2.2 FSP setup

2.2.1 Installation of FSP using Package Installer

Package Installer **RZG_FSP_Packs_v3.1.0.exe** is showcased at here. This section describes the procedure for installation. Note that it's for Windows Host PC only.

1. Quit e² studio.
2. Invoke **RZG_FSP_Packs_v3.1.0.exe**.
3. Click **[Next >]** button to start the installation.

Figure 22: FSP Package Installer

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

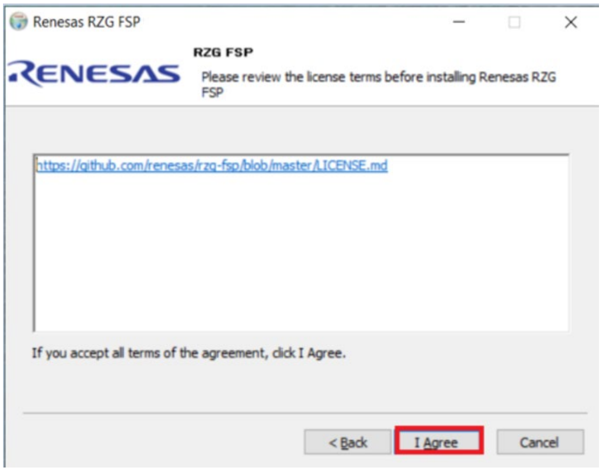


Figure 23: FSP License Term

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

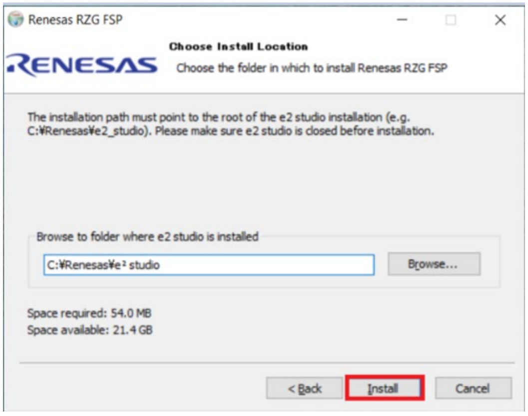


Figure 24: Browse to the folder where e² studio is installed

6. Click **[Finish]** button to complete the installation.

Figure 25: 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 Packs using Package Zip file

No package installer is available for Linux Host PC. Thus, you need to install FSP with the zip file **RZG_FSP_Packs_v3.1.0.zip**. This section describes the procedure for installation.

1. Download RZG_FSP_Packs_v3.1.0.zip from here.
2. Extract the zip file to e² studio installation directory. If it's successfully extracted, **rz_fsp/rzg/packs** should be placed at **<e2 stduio installation directory>/Internal/projectgen**.

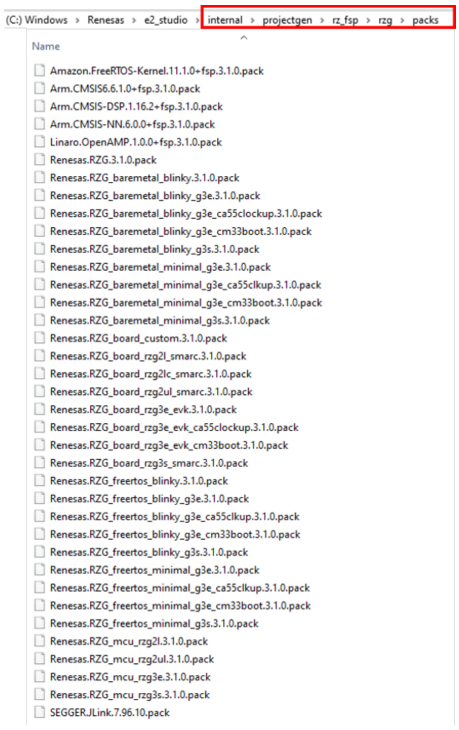


Figure 26: FSP Packs on e² studio installation directory

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/G

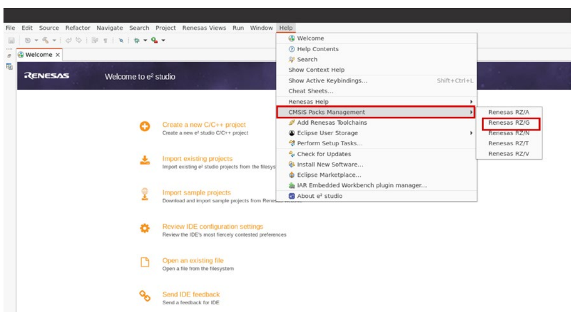


Figure 27: CMSIS Packs Management (1)

- If FSP is successfully installed, 3.1.0 should be listed under FSP as shown below:

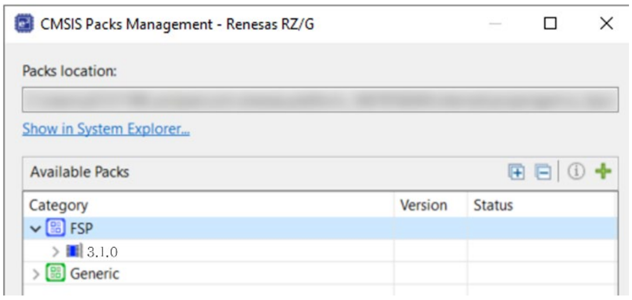


Figure 28: CMSIS Packs Management (2)

3. Set up a SMARC EVK

3.1 RZ/G2L SMARC EVK

Below is an example of a typical system configuration.

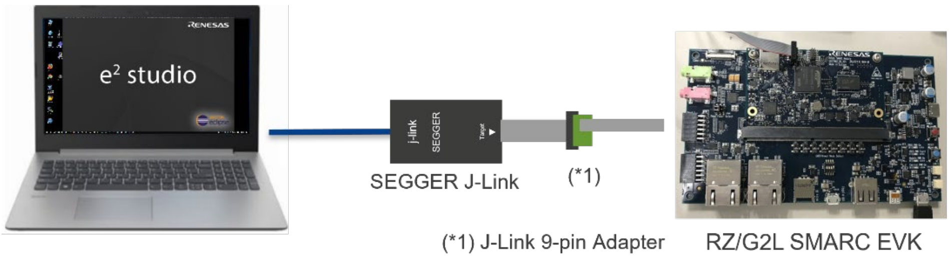


Figure 29: System Configuration Example – RZ/G2L SMARC EVK

3.1.1 Supported Debugger

- SEGGER J-Link

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

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.

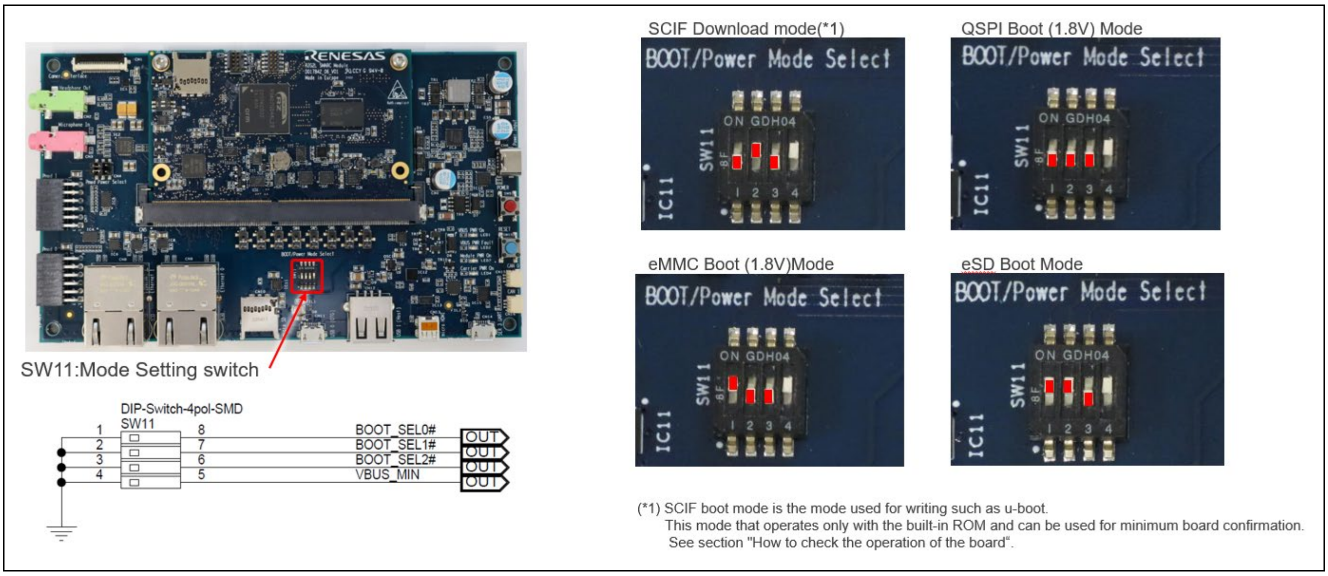
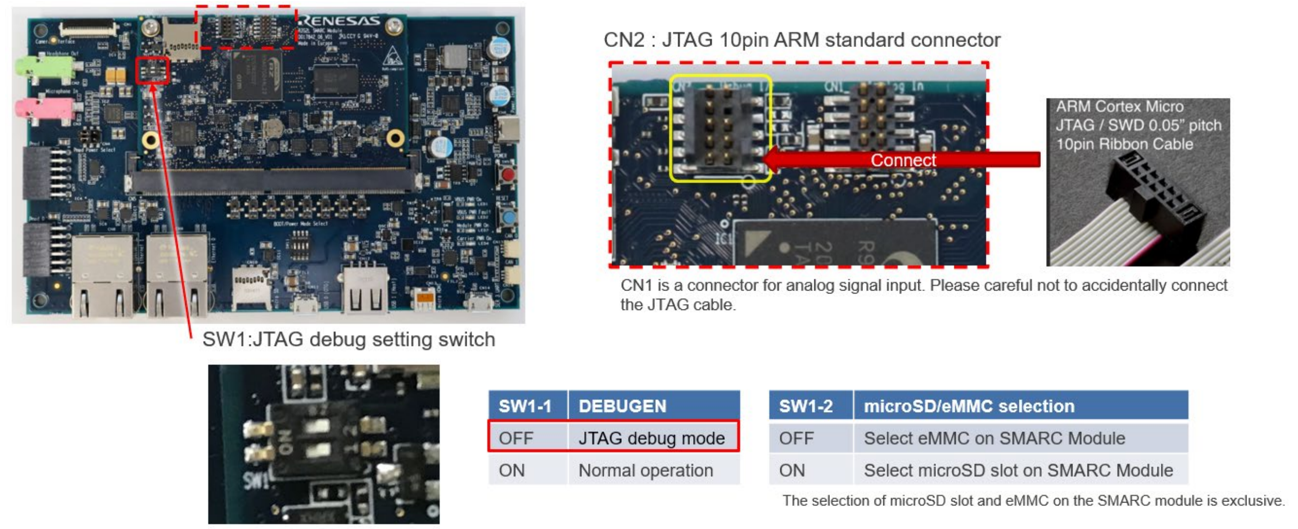


Figure 30: Boot MODE

3.1.2.2 JTAG connection

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



SW1-1	DEBUGEN
OFF	JTAG debug mode
ON	Normal operation

SW1-2	microSD/eMMC selection
OFF	Select eMMC on SMARC Module
ON	Select microSD slot on SMARC Module

The selection of microSD slot and eMMC on the SMARC module is exclusive.

Figure 31: JTAG connection

Please note that RZ/G2L 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.

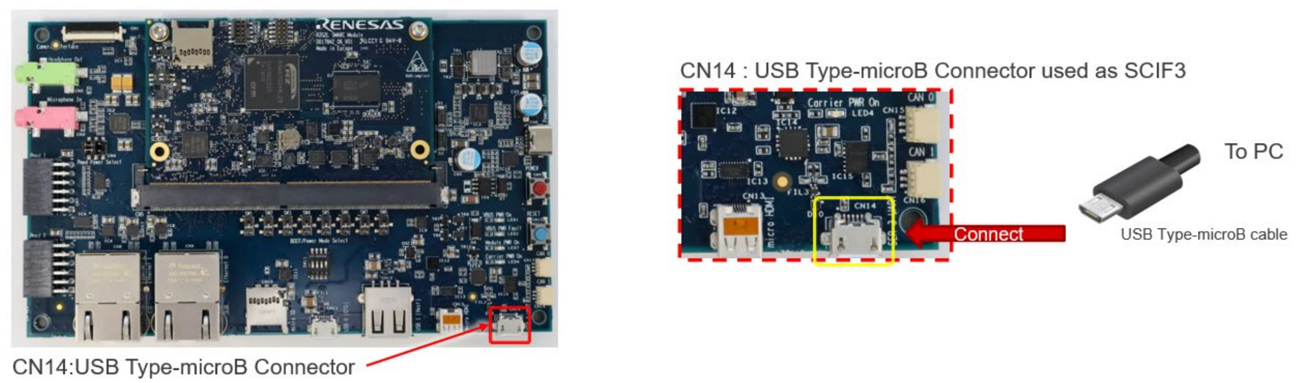


Figure 32: Debug Serial (console output)

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)

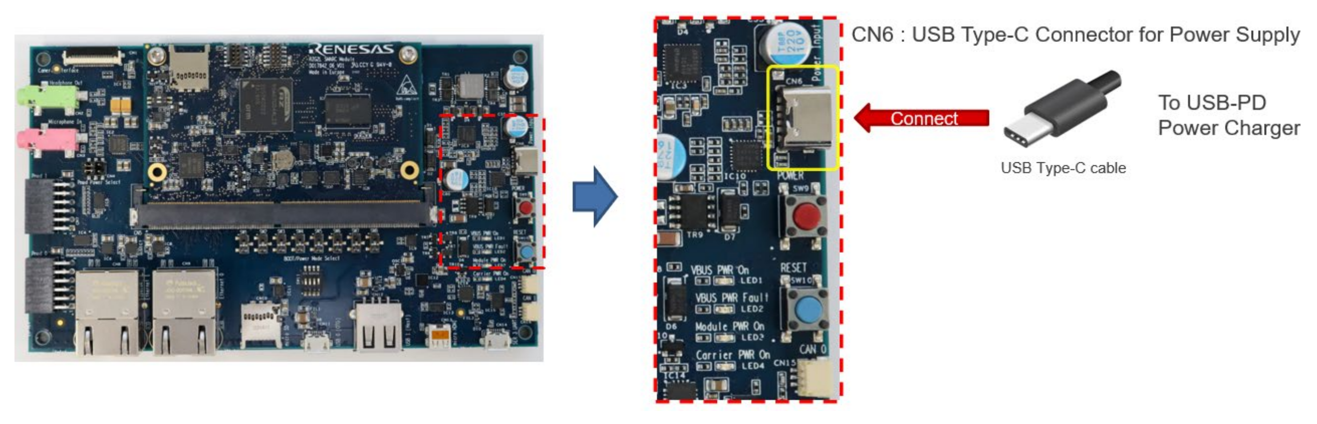


Figure 33: Power Supply

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

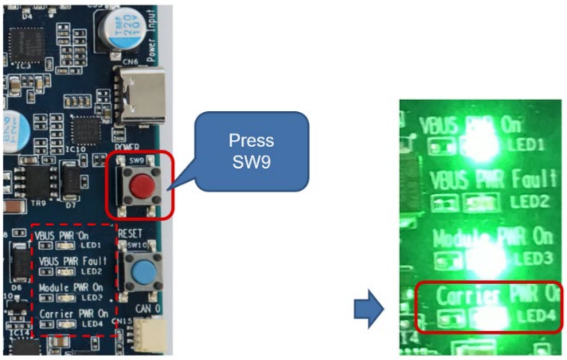


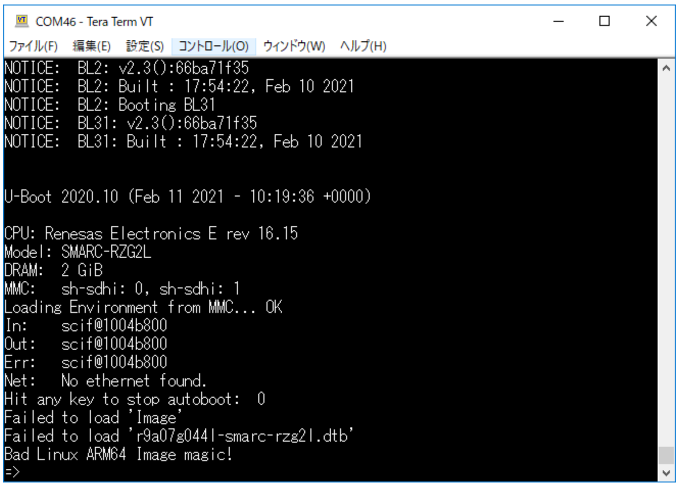
Figure 34: LED Status after Turning on EVK

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.

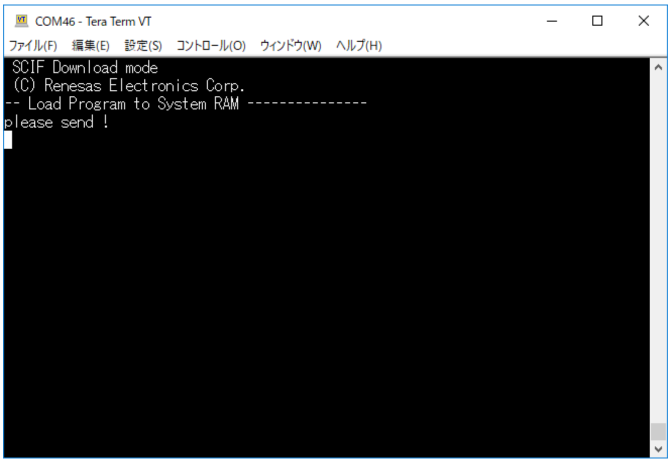
BOOT MODE: QSPI Boot (1.8V) Mode

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 RZ/G3S SMARC EVK

Below is an example of a typical system configuration.

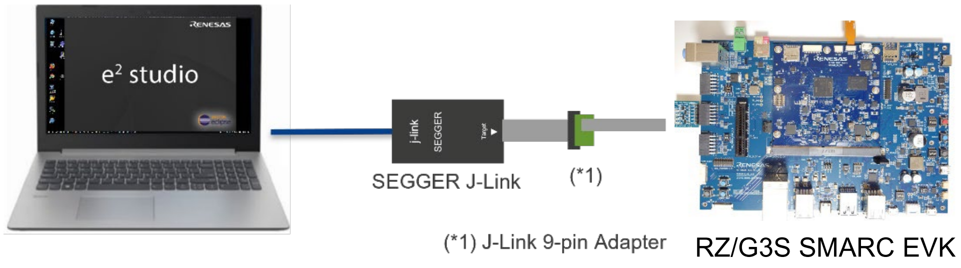


Figure 35: System Configuration Example – RZ/G3S SMARC EVK

3.2.1 Supported Debugger

- SEGGER J-Link

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

3.2.2 Board Setup

3.2.2.1 Boot MODE

Set the boot mode using the two DIP SWITCHs shown in the figure below.

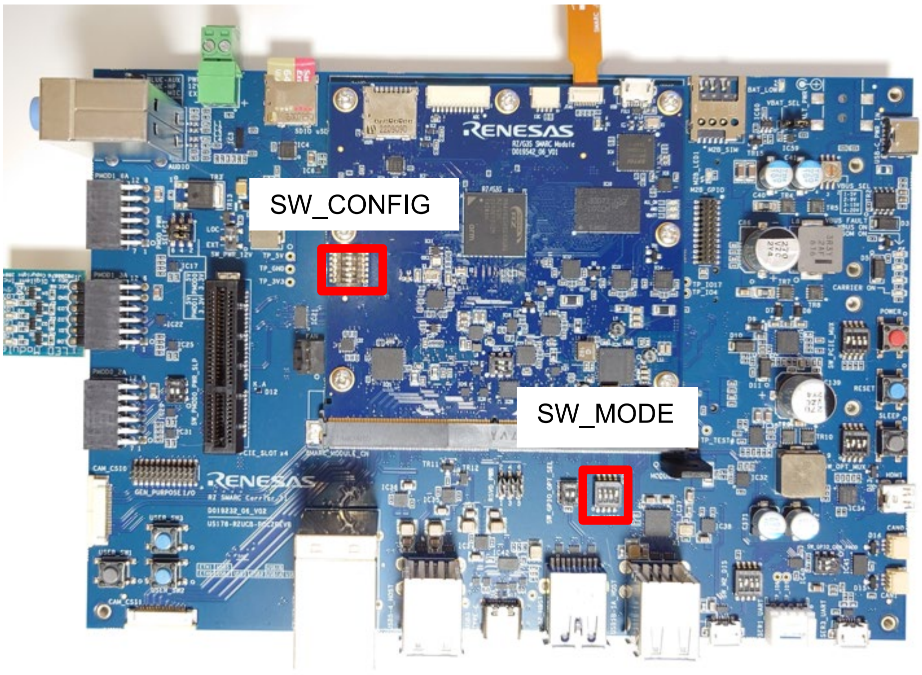


Figure 36: Boot MODE

In SW_CONFIG, select the boot CPU with the following settings.

1	2	3	4	5	6
OFF	OFF	ON	OFF	OFF	ON: CM33 boot OFF: CA55 boot

In SW_MODE, select the boot device with the following settings.

- Boot Mode 1: Booting from eMMC

1	**2**	**3**	**4**
ON	OFF	OFF	ON

- Boot Mode 2: Booting from serial flash memory

1	**2**	**3**	**4**
OFF	OFF	OFF	ON

- Boot Mode 3: Booting from the program downloaded through the serial communications with FIFO (SCIF)

1	**2**	**3**	**4**
OFF	ON	OFF	ON

3.2.2.2 JTAG connection

For JTAG connection, connect the included “SMARC JTAG ADAPTOR” to the board.

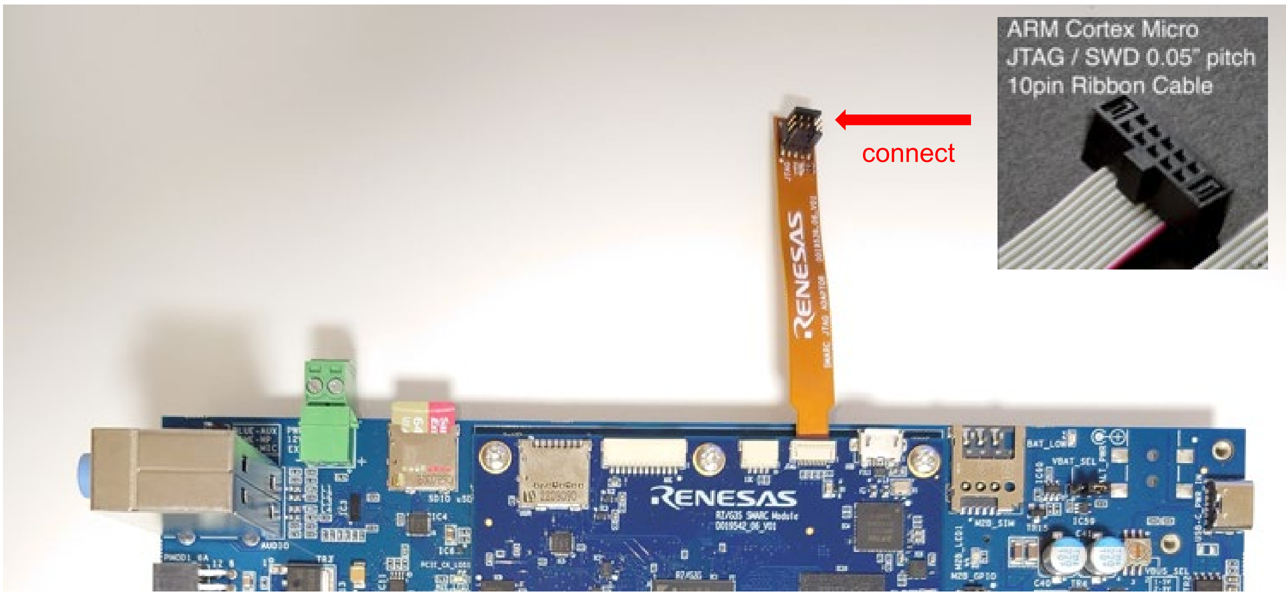


Figure 37: JTAG connection

Please note that RZ/G3S 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/

3.2.2.3 Debug Serial (console output)

Debug serial uses SER3_UART (SCIFA ch0). The baud rate is 115200bps.

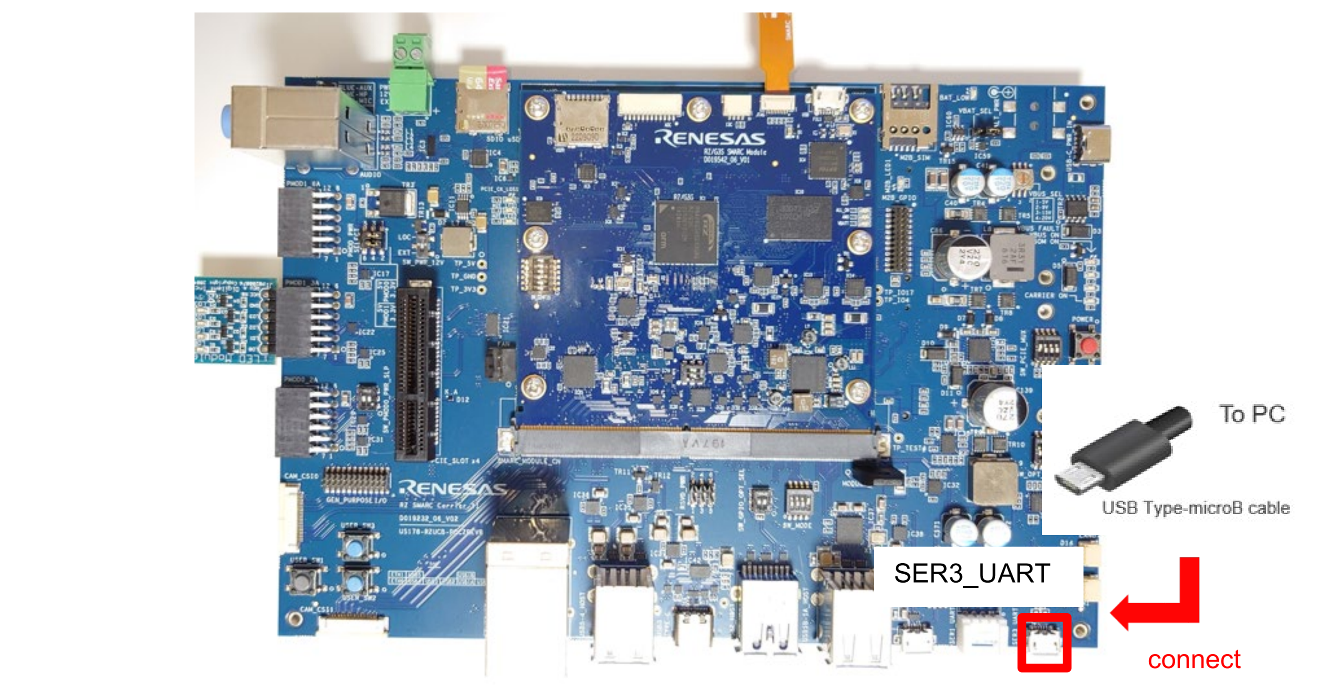


Figure 38: Debug Serial (console output)

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

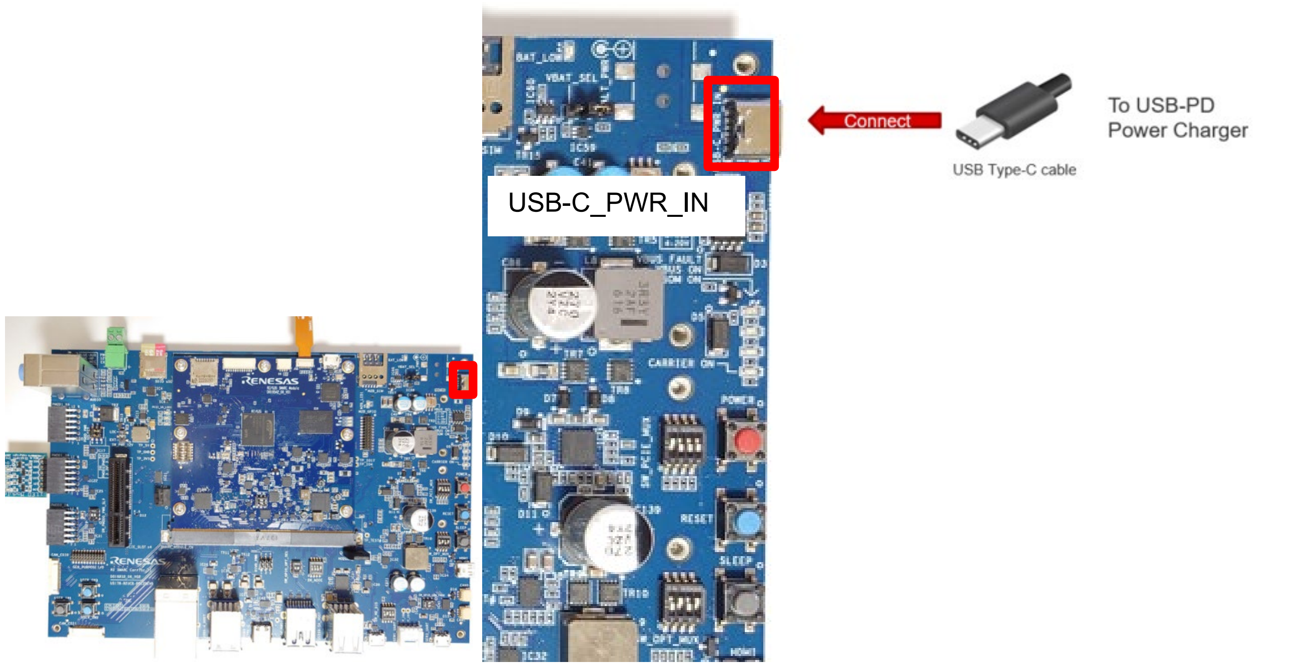


Figure 39: Power Supply

Connect USB-PD Power Charger to USB Type-C Connector. Then VBUS ON and SOM ON light up. Press POWER to turn on the power. Then CARRIER ON lights up.

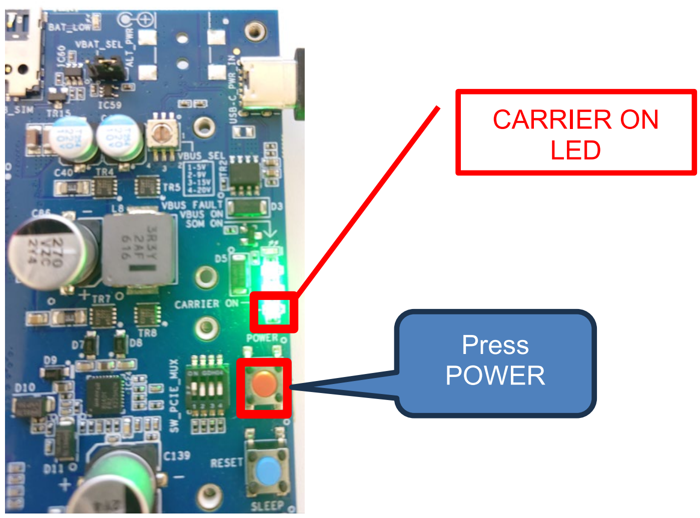


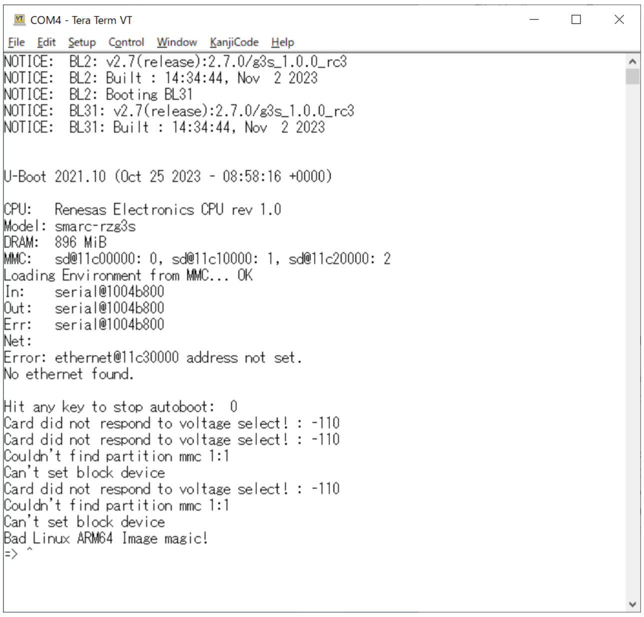
Figure 40: LED Status after Turning on EVK

3.2.2.5 How to check the operation of the board

First, check the board for problems. There are three ways to do this. Please check with any of them.

BOOT MODE1: Booting from eMMC

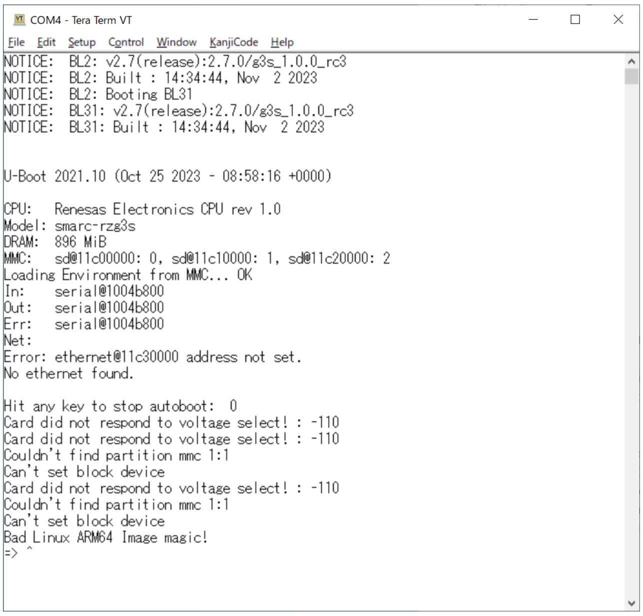
If u-boot is written to the eMMC, When the power is turned on, the following will be output to the console (SER3_UART).




```
NOTICE:  BL2: v2.7(release):2.7.0/g3s_1.0.0_rc3
NOTICE:  BL2: Built : 14:34:44, Nov  2 2023
NOTICE:  BL2: Booting BL31
NOTICE:  BL31: v2.7(release):2.7.0/g3s_1.0.0_rc3
NOTICE:  BL31: Built : 14:34:44, Nov  2 2023


U-Boot 2021.10 (Oct 25 2023 - 08:58:16 +0000)

CPU:   Renesas Electronics CPU rev 1.0
Model: smarc-rzg3s
DRAM:  896 MiB
MMC:   sd@11c00000: 0, sd@11c10000: 1, sd@11c20000: 2
Loading Environment from MMC... OK
In:    serial@1004b800
Out:   serial@1004b800
Err:   serial@1004b800
Net:
Error: ethernet@11c30000 address not set.
No ethernet found.

Hit any key to stop autoboot:  0
Card did not respond to voltage select! : -110
Card did not respond to voltage select! : -110
Couldn't find partition mmc 1:1
Can't set block device
Card did not respond to voltage select! : -110
Couldn't find partition mmc 1:1
Can't set block device
Bad Linux ARM64 Image magic!
=> ^
```

BOOT MODE2: Booting from serial flash memory

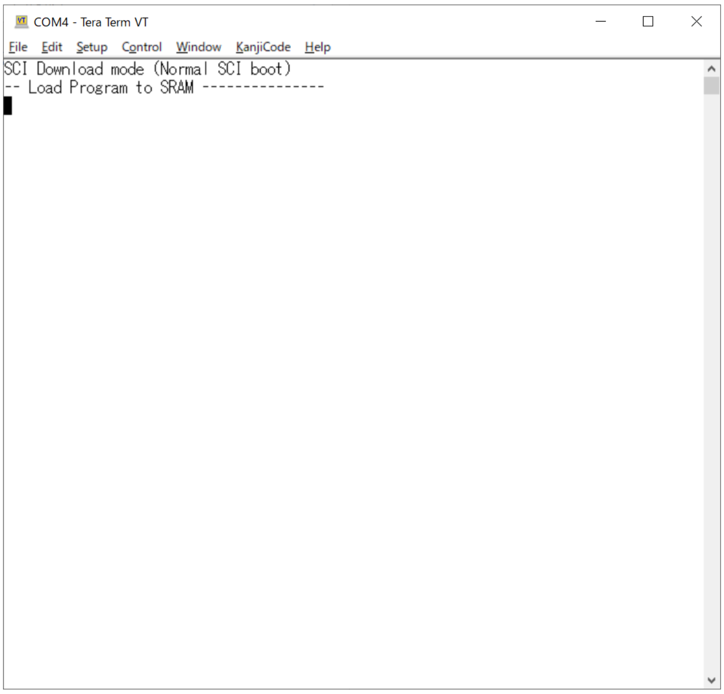
If u-boot is written to the serial flash, When the power is turned on, the following will be output to the console (SER3_UART).




```
NOTICE:  BL2: v2.7(release):2.7.0/g3s_1.0.0_rc3
NOTICE:  BL2: Built : 14:34:44, Nov  2 2023
NOTICE:  BL2: Booting BL31
NOTICE:  BL31: v2.7(release):2.7.0/g3s_1.0.0_rc3
NOTICE:  BL31: Built : 14:34:44, Nov  2 2023


U-Boot 2021.10 (Oct 25 2023 - 08:58:16 +0000)

CPU:   Renesas Electronics CPU rev 1.0
Model: smarc-rzg3s
DRAM:  896 MiB
MMC:   sd@11c00000: 0, sd@11c10000: 1, sd@11c20000: 2
Loading Environment from MMC... OK
In:    serial@1004b800
Out:   serial@1004b800
Err:   serial@1004b800
Net:
Error: ethernet@11c30000 address not set.
No ethernet found.

Hit any key to stop autoboot:  0
Card did not respond to voltage select! : -110
Card did not respond to voltage select! : -110
Couldn't find partition mmc 1:1
Can't set block device
Card did not respond to voltage select! : -110
Couldn't find partition mmc 1:1
Can't set block device
Bad Linux ARM64 Image magic!
=> ^
```

BOOT MODE3: Booting from the program downloaded through the serial communications with FIFO (SCIF)

When the power is turned on, the following will be output to the console (SER3_UART).



3.3 RZ/G3E SMARC EVK

Below is an example of a typical system configuration.

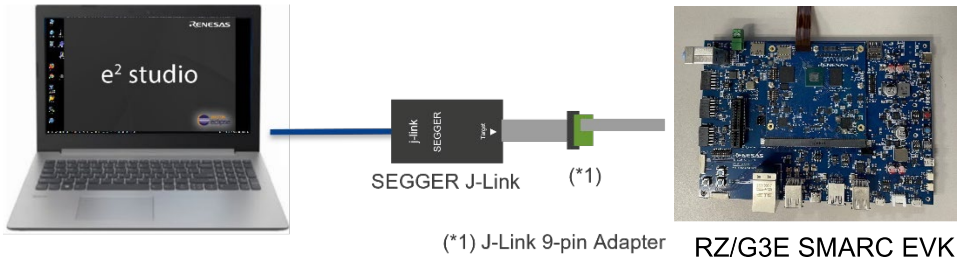


Figure 41: System Configuration Example – RZ/G3S SMARC EVK

3.3.1 Supported Debugger

- SEGGER J-Link

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

3.3.2 Board Setup

3.3.2.1 Boot MODE

Set the boot mode using the two DIP SWITCHs shown in the figure below.

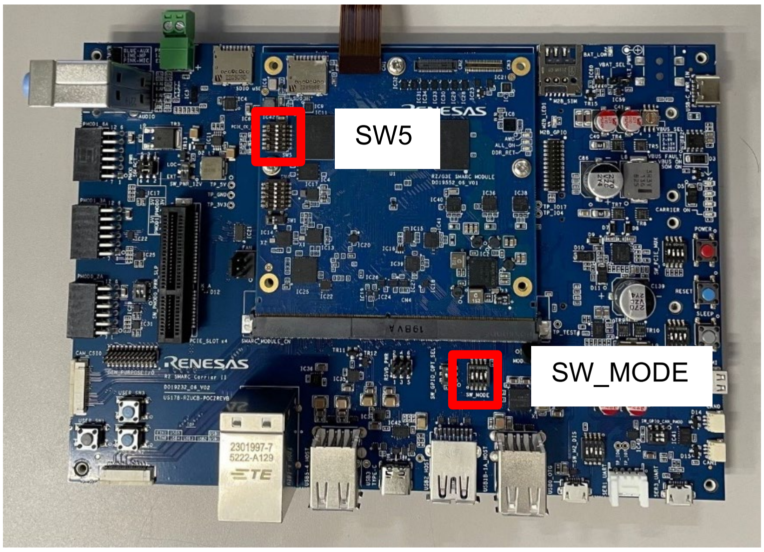


Figure 42: Boot MODE

In SW5, select the boot CPU with the following settings.

1	2	3	4	5	6
OFF	ON: CM33 boot OFF: CA55 boot	ON	OFF	OFF	OFF

In SW_MODE, select the boot device with the following settings.

- Boot Mode 3: Booting from SCIF download

1	2	3
OFF	ON	OFF

- Boot Mode 4: Booting from USB download

1	2	3
ON	ON	ON

- Boot Mode 5: Booting from eMMC 1.8 V

1	2	3
ON	OFF	OFF

- Boot Mode 6: Booting from a serial flash memory connected to the xSPI bus space 1.8 V

1	2	3
OFF	ON	OFF

3.3.2.2 JTAG connection

For JTAG connection, connect the included “Breakout board” to the board.

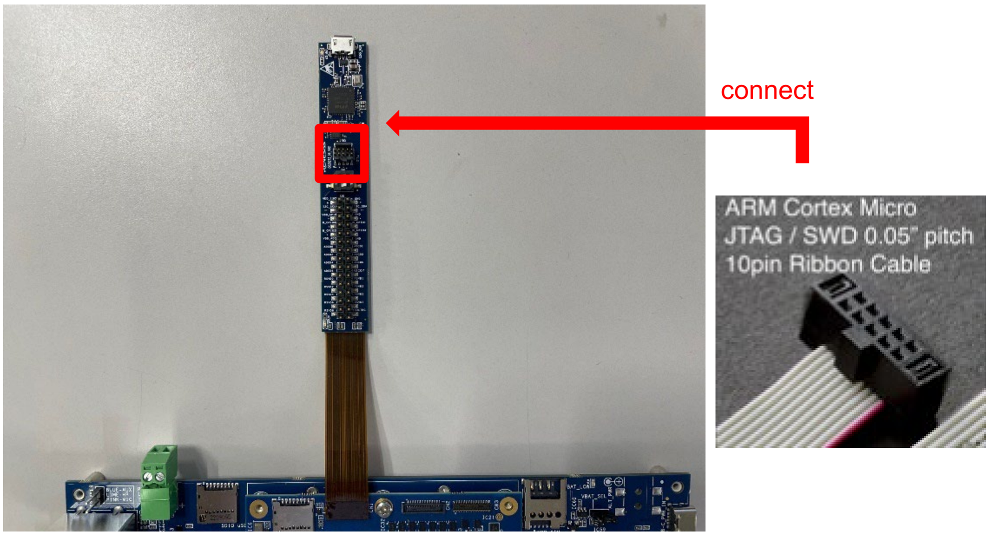


Figure 43: JTAG connection

Note1: RZ/G3E SMARC EVK has CoreSight 10 connector. 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/

Note2: It is strongly recommended that do not detach the breakout board once it is connected to the CN1 connector on the module board.

3.3.2.3 Debug Serial (console output)

Debug serial uses Debug_UART (SCIFA ch0). The baud rate is 115200bps.

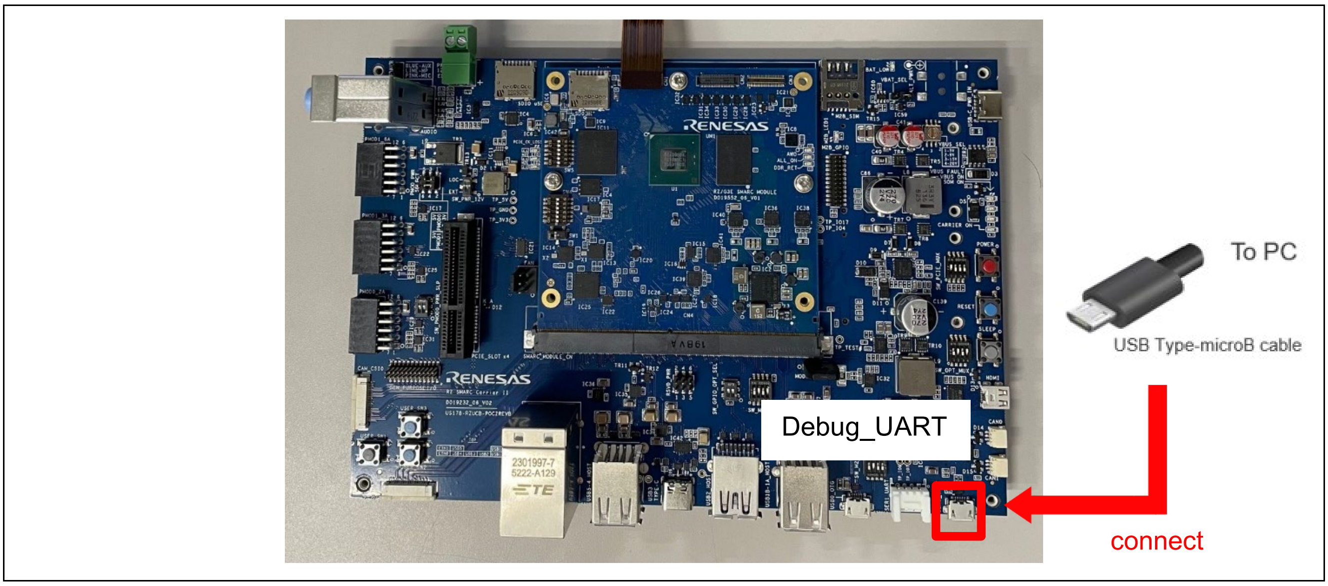


Figure 44: Debug Serial (console output)

3.3.2.4 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)

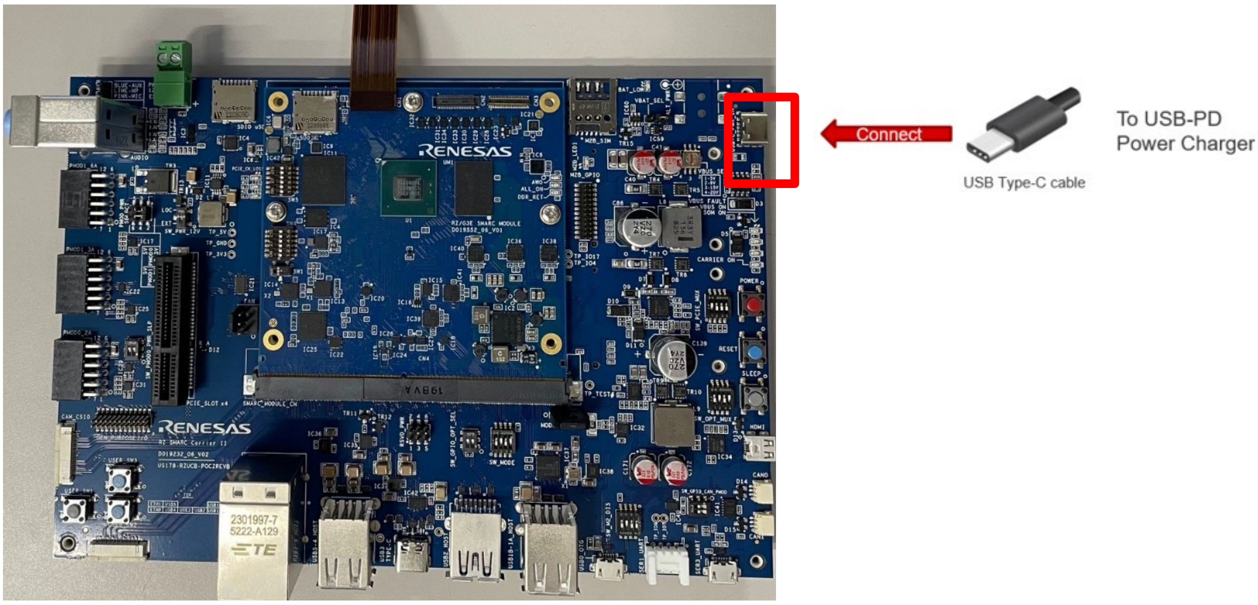


Figure 45: Power Supply

Connect USB-PD Power Charger to USB Type-C Connector. Then VBUS ON and SOM ON light up. Press POWER to turn on the power. Then CARRIER ON lights up.

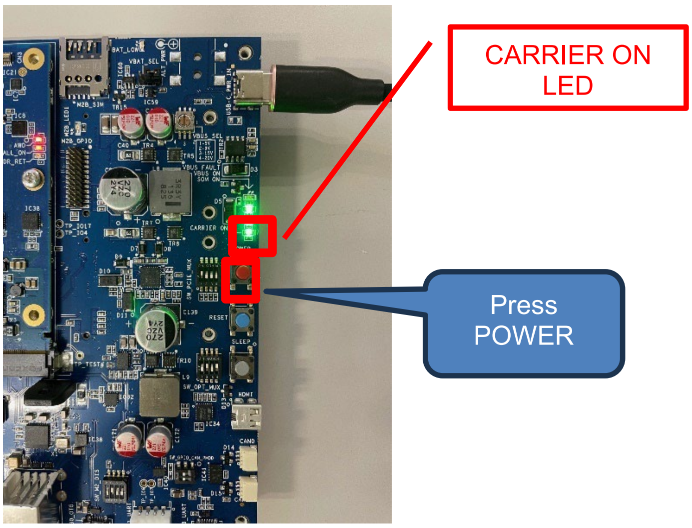


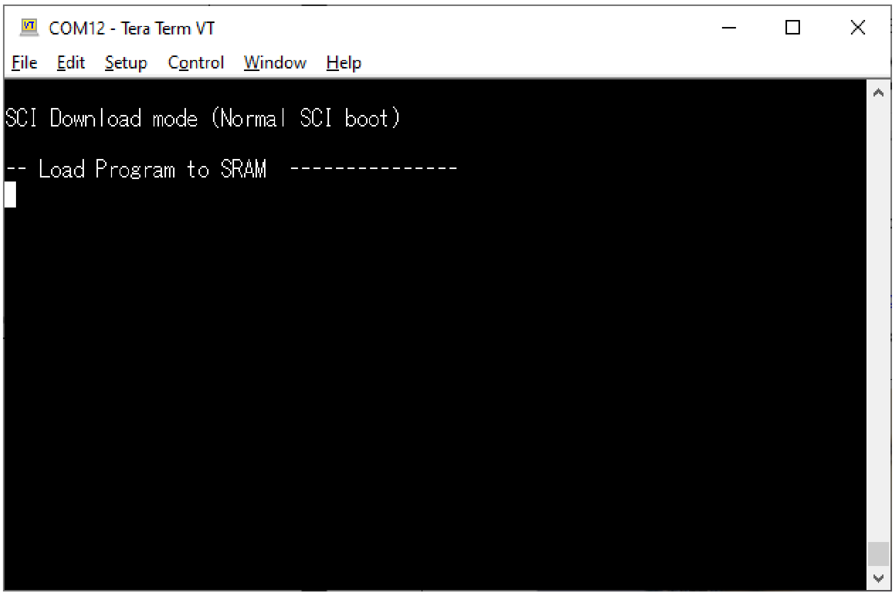
Figure 46: LED Status after Turning on EVK

3.3.2.5 How to check the operation of the board

First, check the board for problems.

BOOT MODE3: Booting from the program downloaded through the serial communications with FIFO (SCIF)

When the power is turned on, the following will be output to the console (SER3_UART).



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.

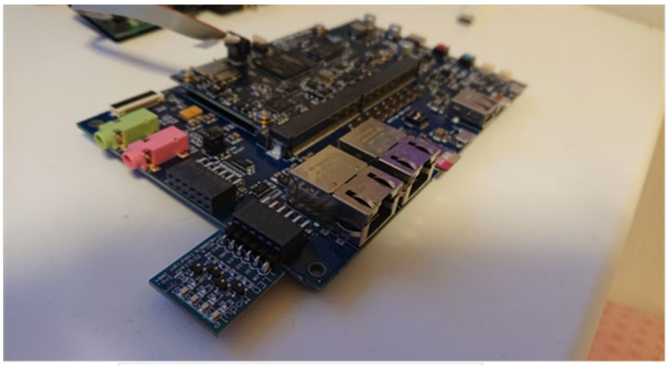
Blinky is the "Hello World" of microprocessors. If the LED blinks you know that:

- 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: SMARC EVK board does not have any LED.
Thus, Blinky sample application used in this tutorial is designed to use the Pmod module described below alternatively:

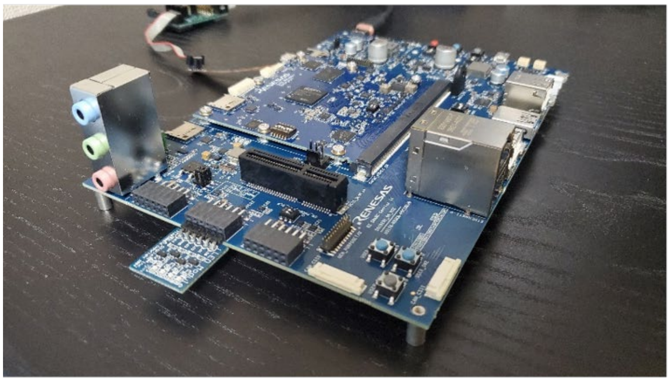
- Pmod LED (Four High-brightness LEDs): https://reference.digilentinc.com/pmod/pmodled/start

This module is not included on the SMARC EVK board and so, please prepare it beforehand.

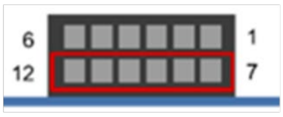


RZ/G2L SMARC EVK

Pmod1

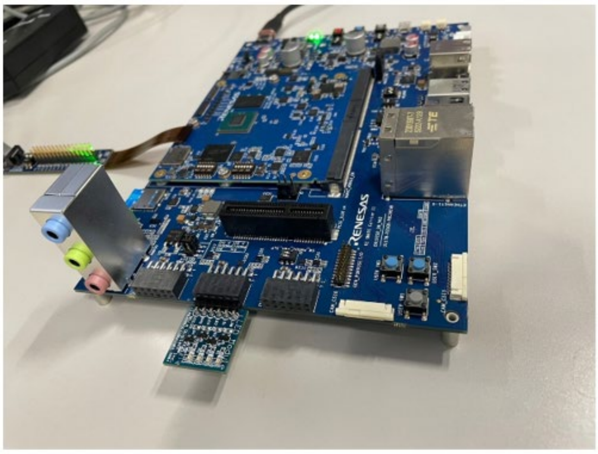


RZ/G3S SMARC EVK



PMOD1_3A

Figure 47: Connection Pmod LED module to RZ/G2L or RZ/G3S



RZ/G3E SMARC EVK

PMOD1_3A

Figure 48: Connection Pmod LED module to RZ/G3E

4.3 Create a New Project for Blinky

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

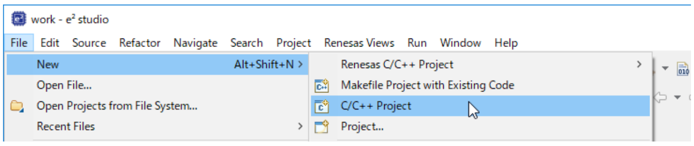


Figure 49: New C/C++ Project

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

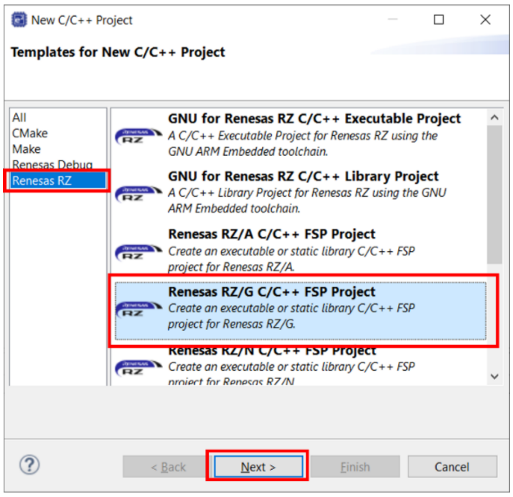


Figure 50: Renesas RZ/G C/C++ FSP Project

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

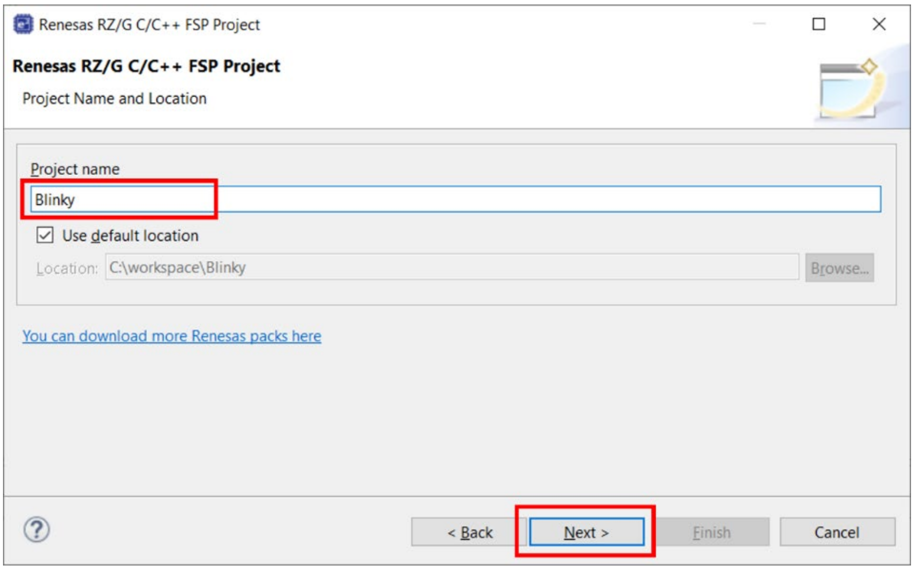


Figure 51 : e² studio Project Configuration window (part 1)

5. Select the board support package by selecting the name of your board from the Device Selection drop-down list. Select **GNU ARM Embedded** in Toolchains and version is **13.3.1 arm-13-24** and click **[Next >]** button.

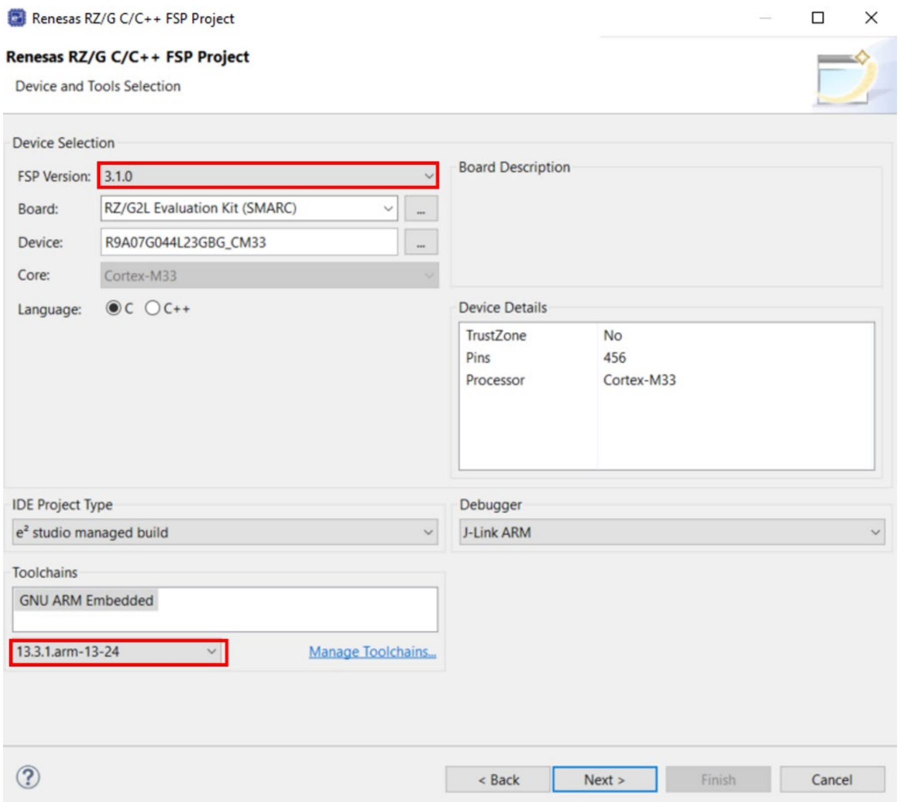


Figure 52 : e² studio Project Configuration window (part 2)

6. (Optional) When creating an RZ/G3E project, support for CM33 cold boot is also available. For further information, refer to the RZ/G Multi-OS Package documentation.

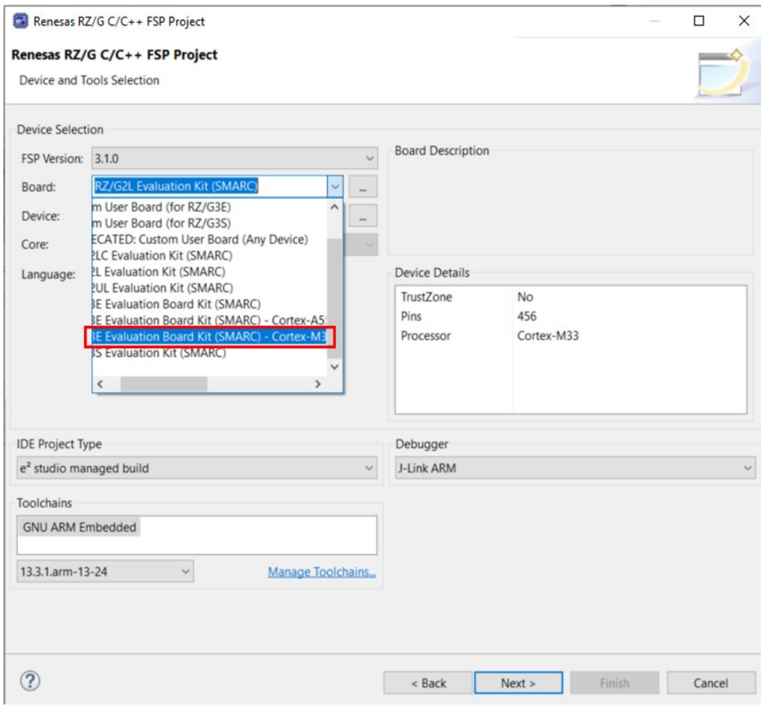


Figure 53: e² studio Project Configuration window (part 3)

7. (Optional) When creating RZ/G Multi-Core project, choose the right bundled project as **Preceding Project**, click **[Next >]** button.

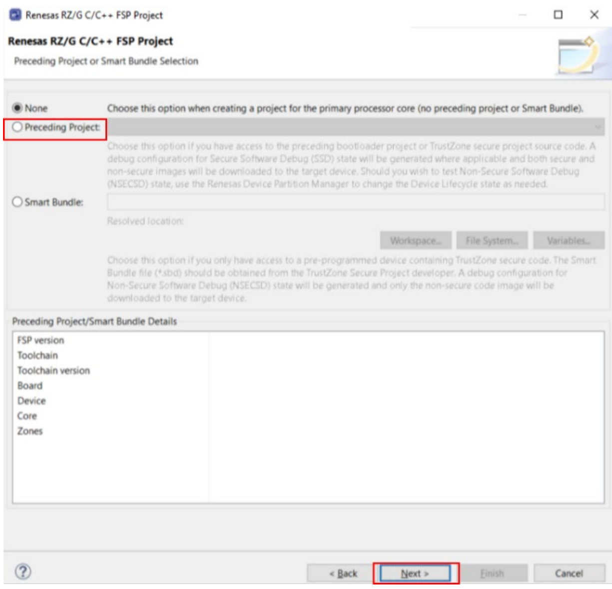


Figure 54: e² studio Project Configuration window (part 4)

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

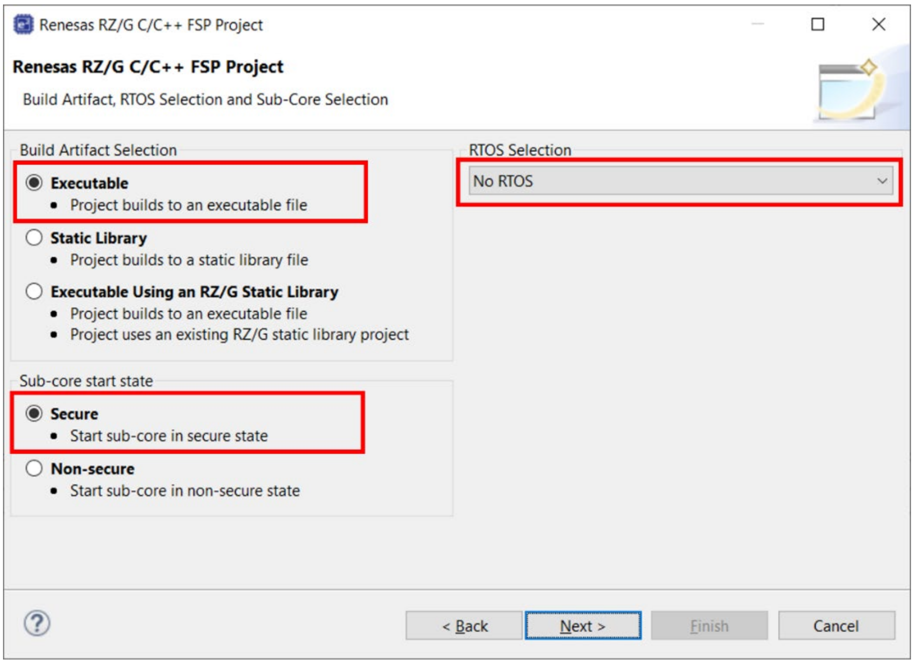


Figure 55 : e² studio Project Configuration window (part 4)

8. Select the **Blinky** template for your board and click **[Finish]** button.

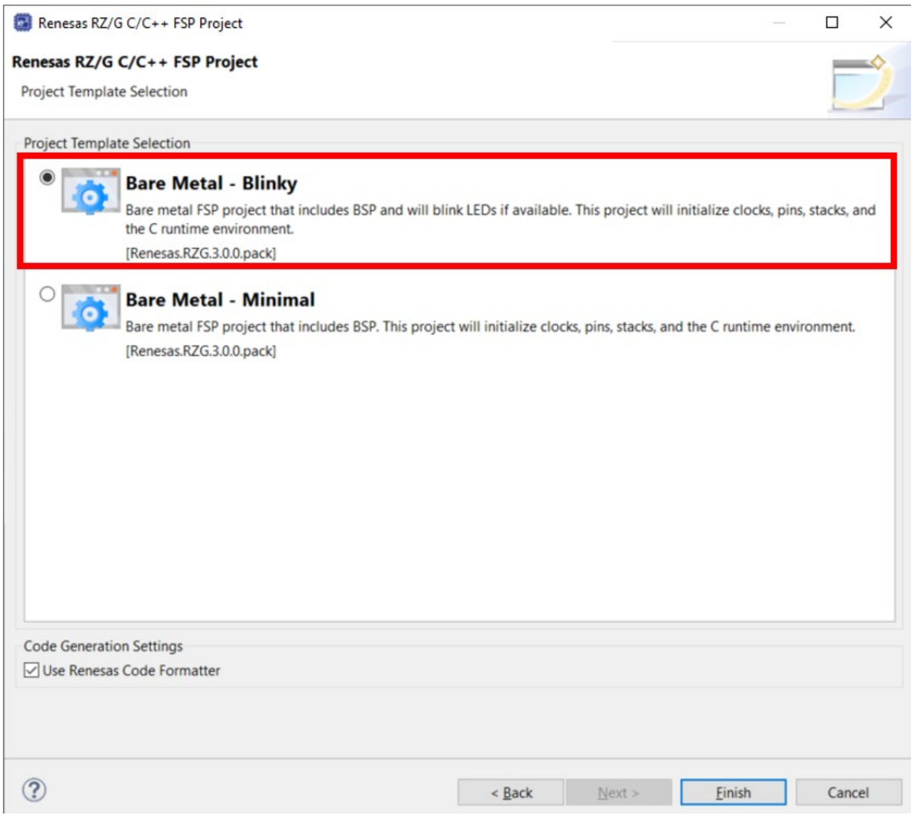


Figure 56 : e² studio Project Configuration window (part 4)

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

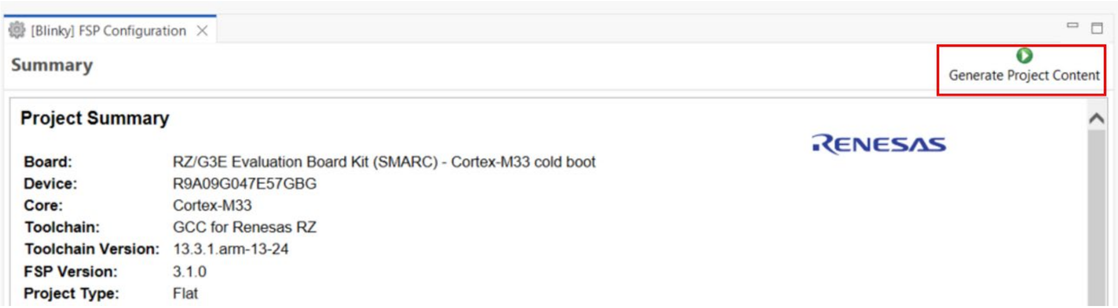


Figure 57 : e² studio Project Configuration tab

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>/rzg_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.

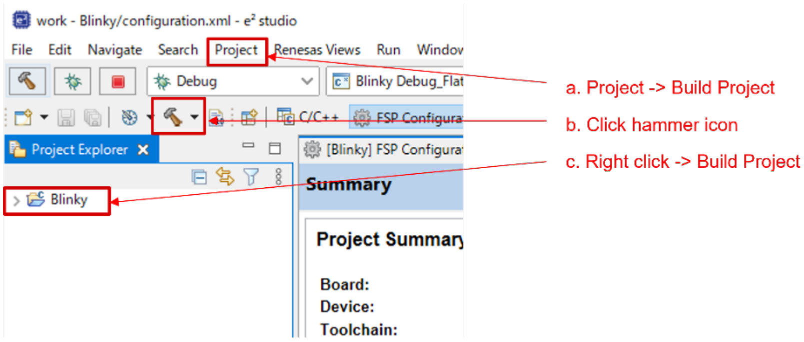


Figure 58 : e² studio Project Explorer window

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.

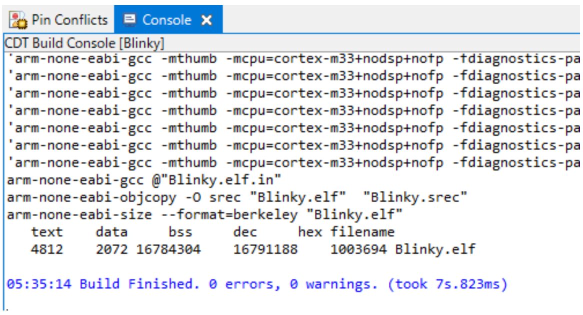


Figure 59 : e² studio Project Build console

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 a 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** ...

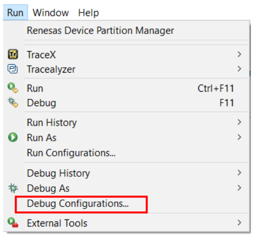


Figure 60 : e² studio Debug icon

or by selecting the drop-down menu next to the bug icon and selecting **Debugger Configuration**s ...

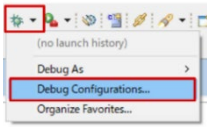


Figure 61 : e² studio Debugger Configurations selection option

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

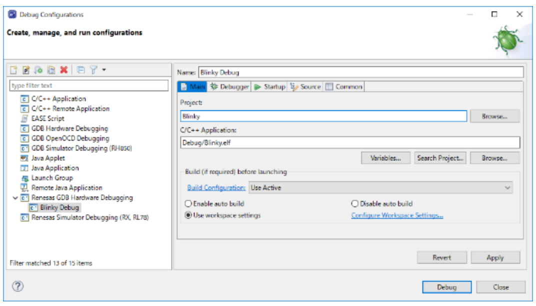


Figure 62 : e² studio Debugger Configurations window with Blinky project (1)

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

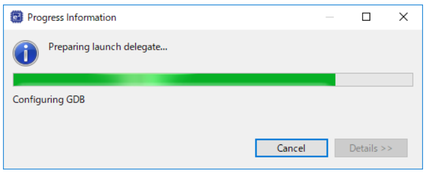


Figure 63 : e² studio Debugger Configurations window with Blinky project (2)

4.5.2.1 Debug multi-core steps

To debug both the RZ/G3S CM33 core and the RZ/G3S CM33_FPU core simultaneously, follow these steps:

1. Create a separate e² studio workspace for each core. For example, use **workspace_cm33** for the CM33 core and **workspace_cm33_fpu** for the CM33_FPU core.

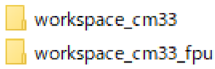


Figure 64: Workspace creation

2. Launch e² studio and select the workspace for the CM33 core.

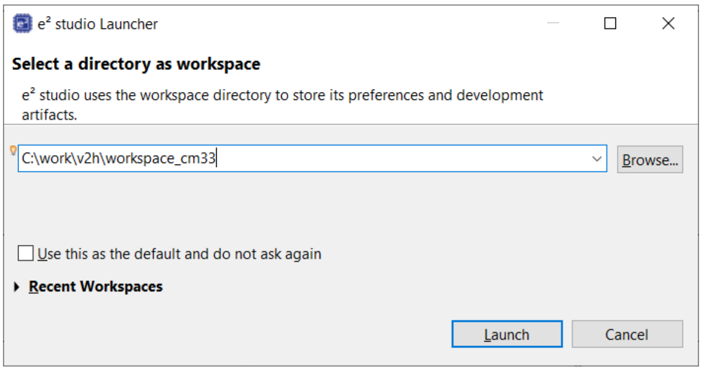


Figure 65: Launch e² studio (for CM33 core)

3. Create an FSP project for the CM33 core. In the configuration window, select the appropriate **Core** and **Project Template Selection** as shown in the figure below.

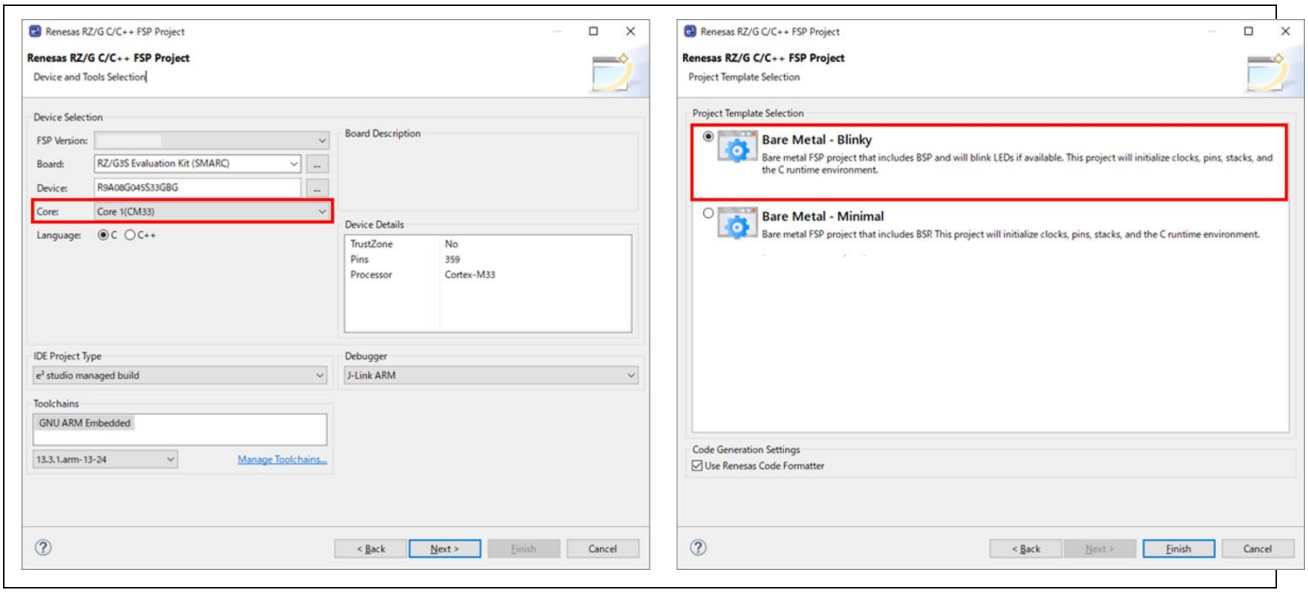


Figure 66: CM33 Core Project Creation

4. Build the FSP project for the CM33 core.

5. Create an FSP project for the CM33 core. In the configuration window, select the appropriate **Core** and **Project Template Selection** as shown in the figure below.

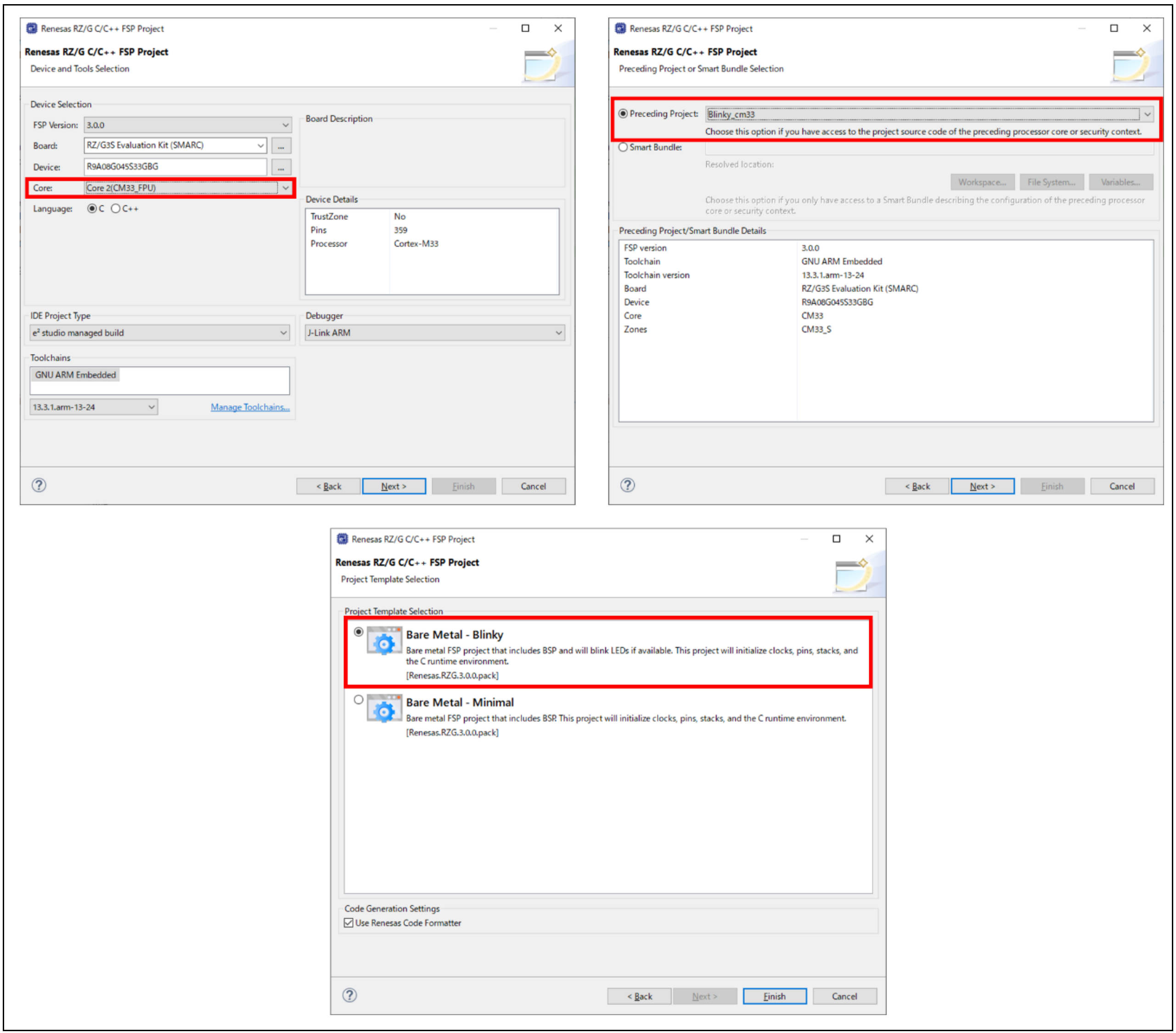


Figure 67: CM33 Core Project Creation

6. Build the FSP project for the CM33_FPU core.
7. Copy the two projects created in the CM33 core workspace into the CM33_FPU core workspace.

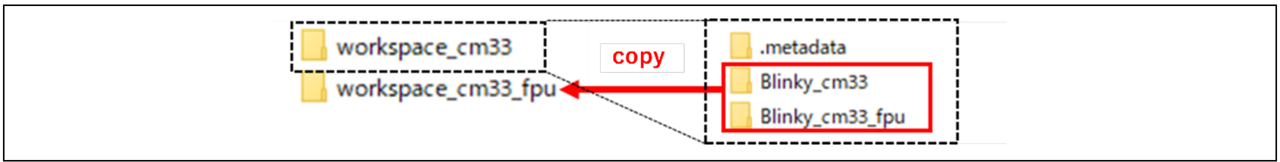


Figure 68: Project Copy

8. Click the **Debug Configuration** in the workspace for the CM33 core, then click **Debug**.

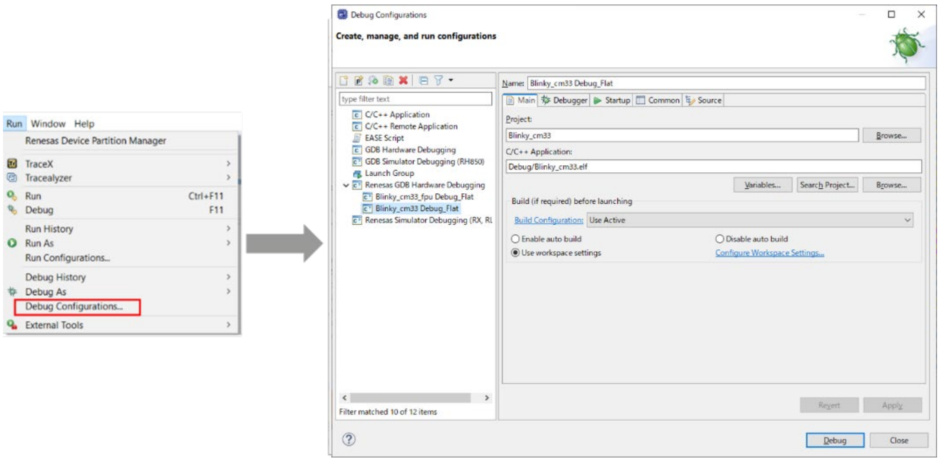


Figure 69: CM33 core Debug Configuration

9. Launch e² studio anew and select the workspace for the CM33_FPU core.

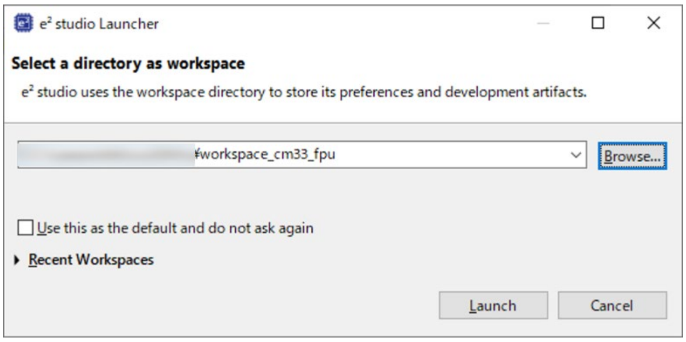


Figure 70: Launch e² studio (for CM33_FPU core)

10. Import the project for the CM33_FPU core.

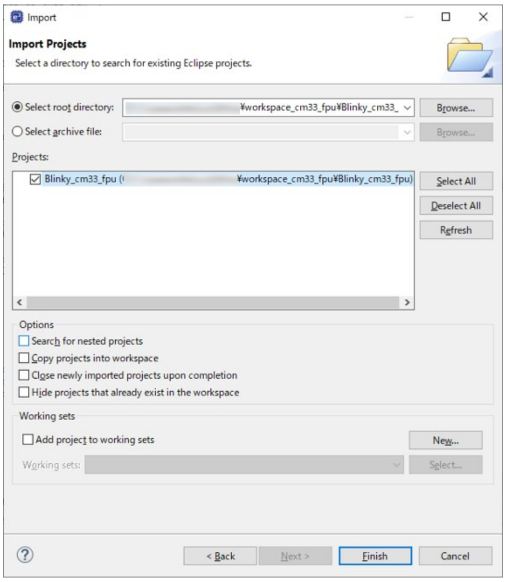


Figure 71: Import CM33_FPU Core Project

11. In the CM33_FPU core workspace, click **Debug Configuration** and then click **Debug**.

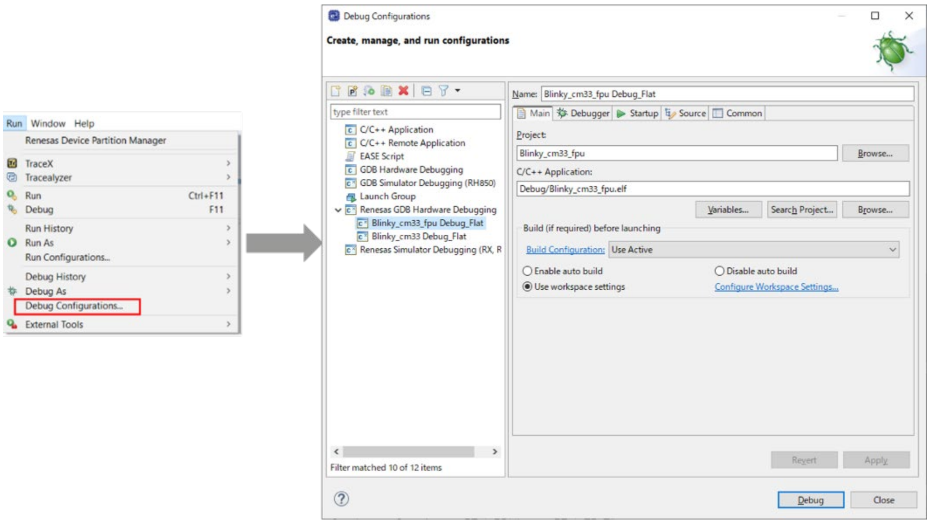


Figure 72: CM33_FPU core Debug Configuration

12. This will enable the debugger functions for each core in their respective e² studio workspaces for the CM33 core and the CM33_FPU core.

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.

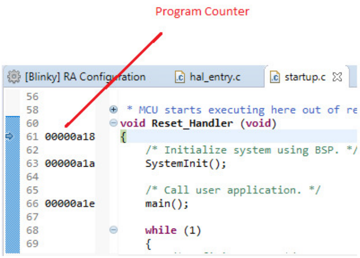


Figure 73 : e² studio Debugger memory window

4.5.4 Run the Blinky Project

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

Figure 74 : e² studio Debugger Play icon

The LED on the Pmod LED should now be blinking.

5. FSP application launch with e² studio

5.1 Creating 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. **Creating 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.

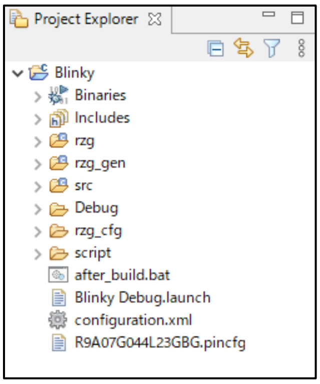


Figure 75 : e² studio Project Configuration file

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 76 : 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 (rzg_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.

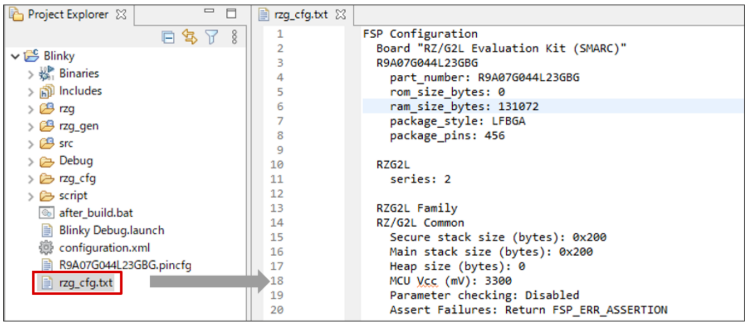


Figure 77 : RZ Project Report

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.

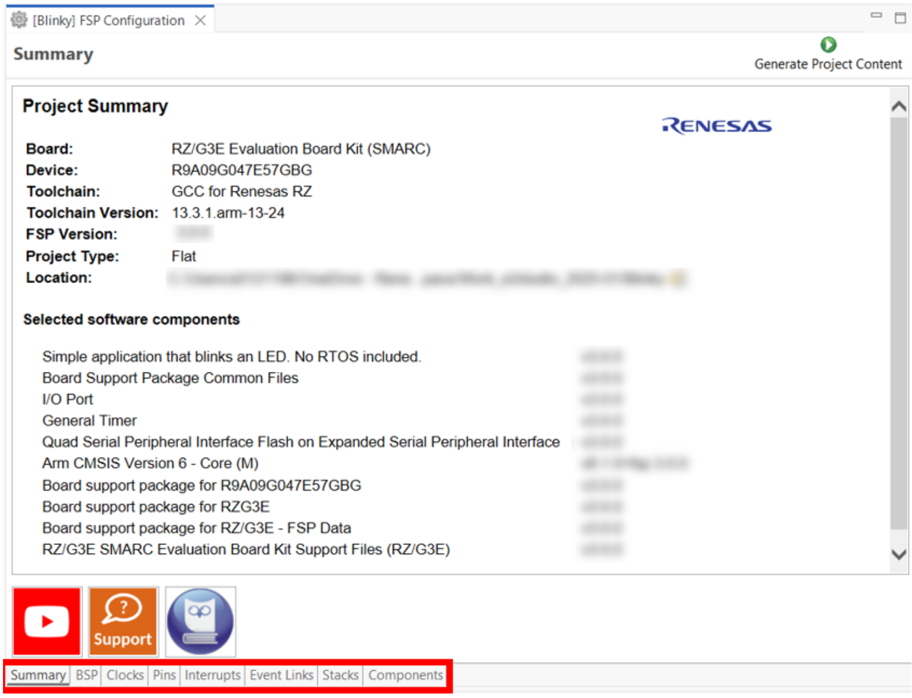


Figure 538 : RZ Project Editor tabs

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.

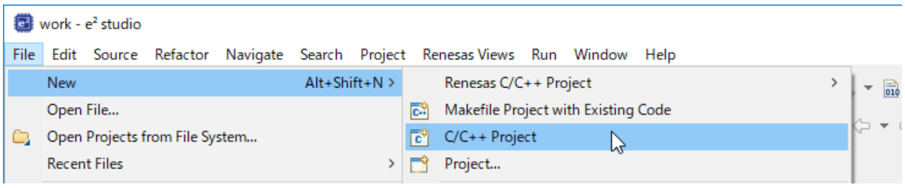


Figure 79 : New RZ MPU Project

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

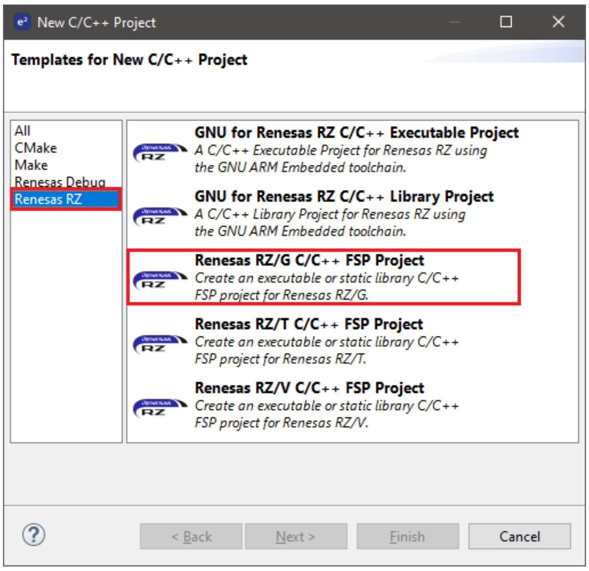


Figure 80 : New Project Templates

3. Select a project name and location.

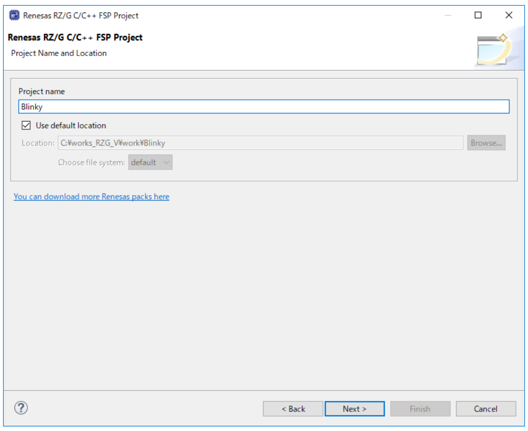


Figure 81 : RZ MPU Project Generator (Screen 1)

4. Click **[Next >]** button.

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** for the board selection.
4. Select the **Core**. You can select Core 1(CM33) or Core 2(CM33_FPU) if you selected RZ/G3S 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 >]** button.

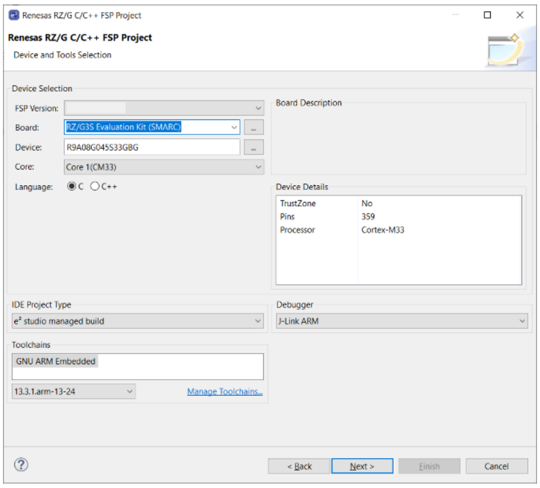


Figure 82 : RZ MPU Project Generator (Screen 2-1)

If Core 2(CM33_FPU) is selected in procedure 4, you need to select the preceding project.
To select the preceding project when creating the Core 2(CM33_FPU) project, it is required to prepare Core 1(CM33) before Core 2(CM33_FPU) project creation.

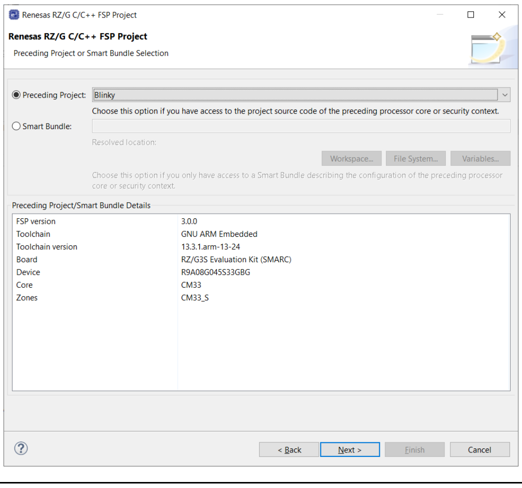


Figure 83 : RZ MPU Project Generator (Screen 2-2)

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.

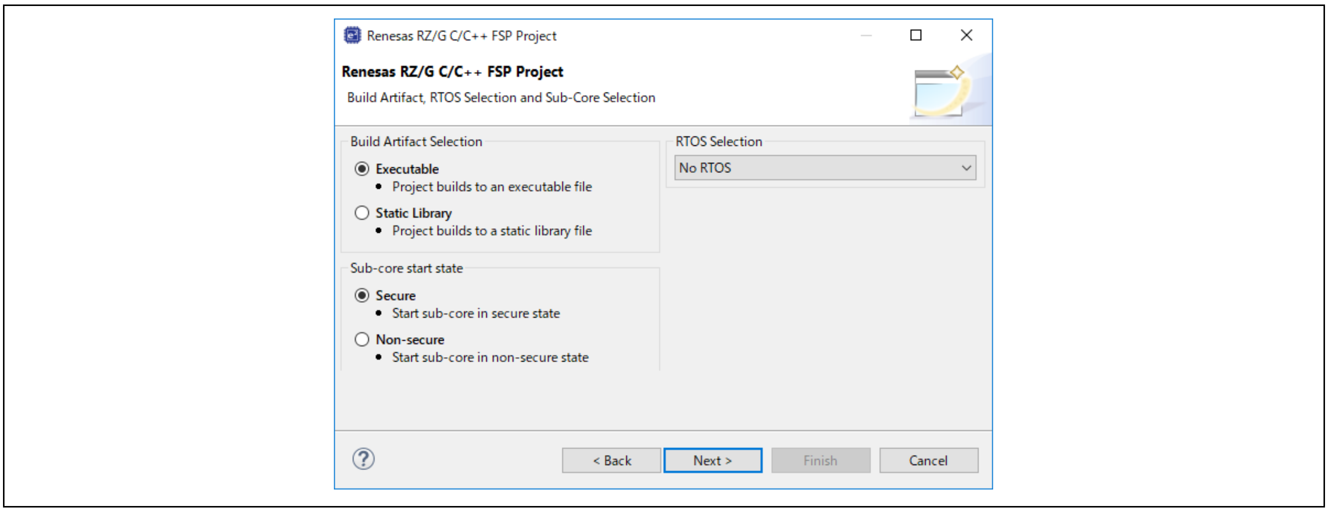


Figure 84 : RZ MPU Project Generator (Screen 3)

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/G MPU Pack. Once you have selected the appropriate template, click **[Finish]** button.

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

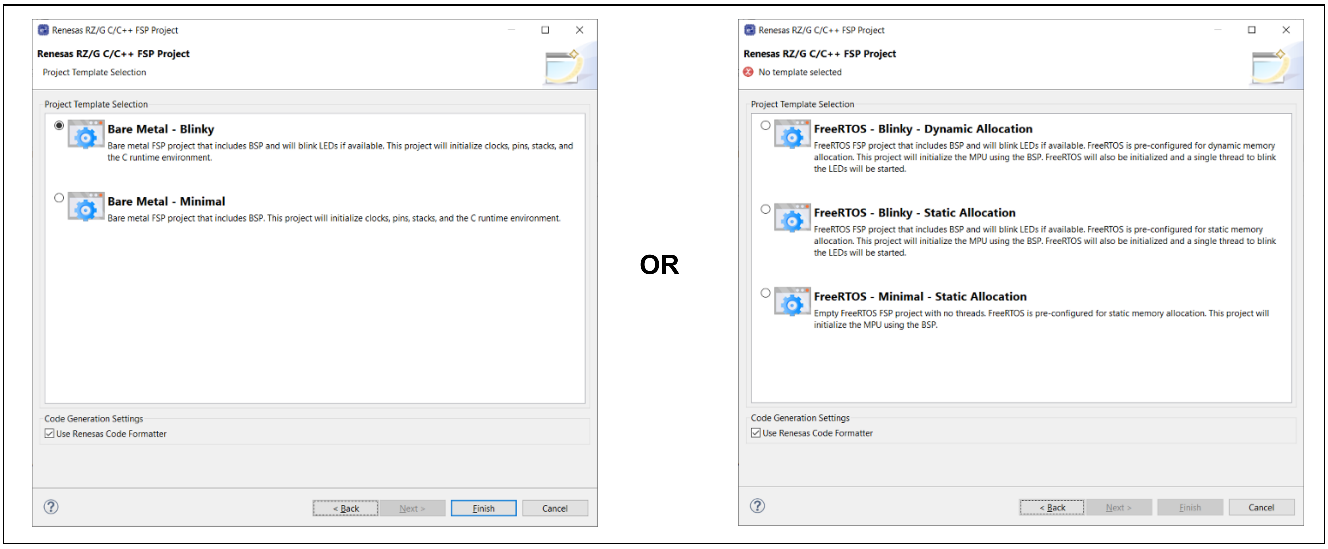


Figure 85 : RZ MPU Project Generator (Screen 4)

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

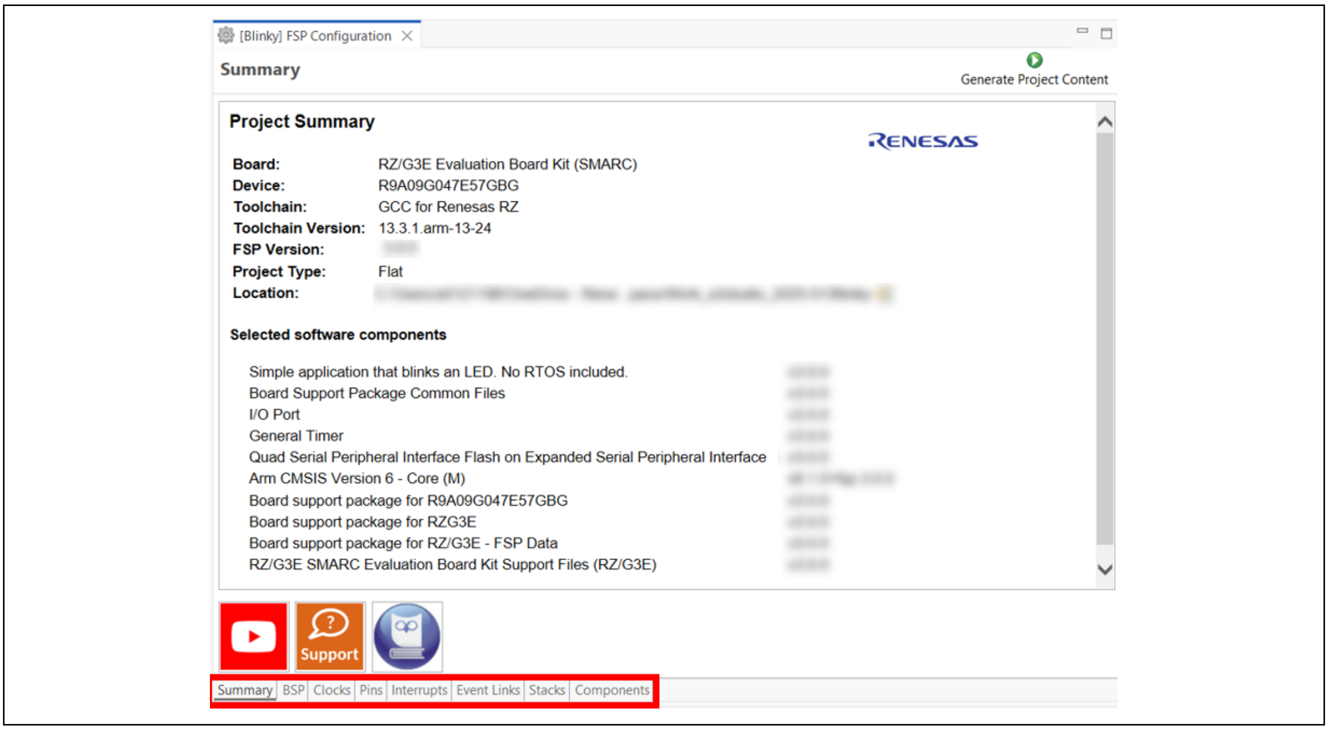


Figure 86 : RZ MPU Project Editor and available editor tabs

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 refer to the MPU clock settings for your project. In the case of CM33 cold boot, you can configure the MPU clock settings for your project.
- With the **Pins** tab, you can configure the electrical characteristics and functions of each port pin.
- 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 are explained in detail in the next section.

Note: RZ/G2L, RZ/G2LC, RZ/G2UL and RZ/G3S don't support **Event Links** tab.

5.1.3 Duplication of Resources

In the case of RZ/G3S Core 2(CM33_FPU) project, duplicate resources are indicated as red character in **Stacks** tab when using resources that are used in the linked Core 1(CM33) project.

The following image is the example that both of Core 1(CM33) and Core 2(CM33_FPU) projects are created with Blinky template. The duplication of r_gtm is indicated in **Stacks** tab. To avoid this duplication, please change the channel resource in **Properties** of r_gtm.

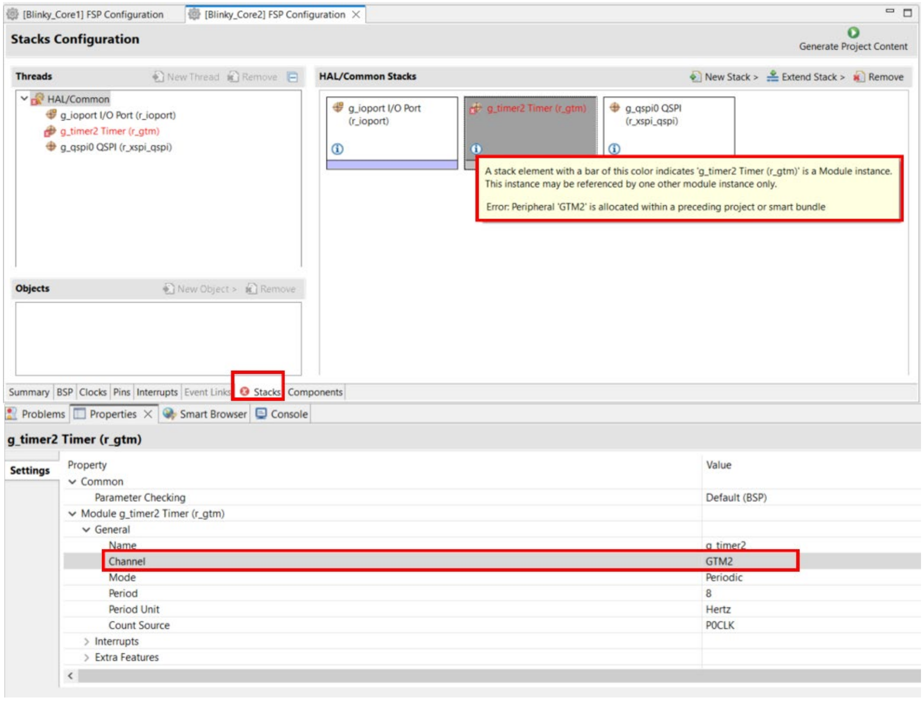


Figure 87 : Duplication of resource between Core 1(CM33) and Core 2(CM33_FPU) projects

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

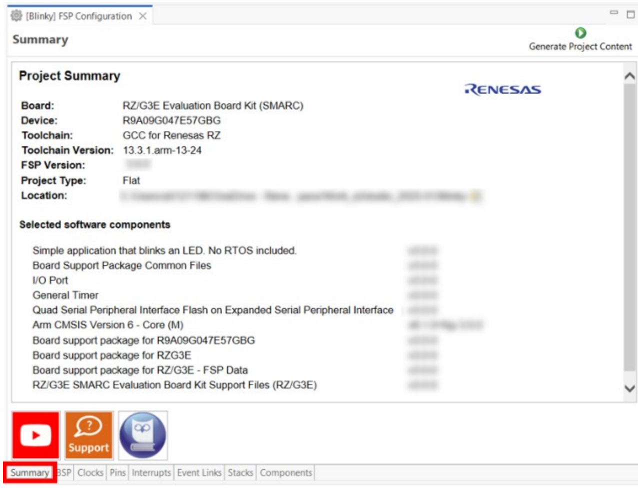


Figure 88 : Configuration Summary tab

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 in the lower left of the Project Configurations view as shown below.

Note: If the Properties view is not visible, click **Window > Show View > Properties** in the top menu bar.

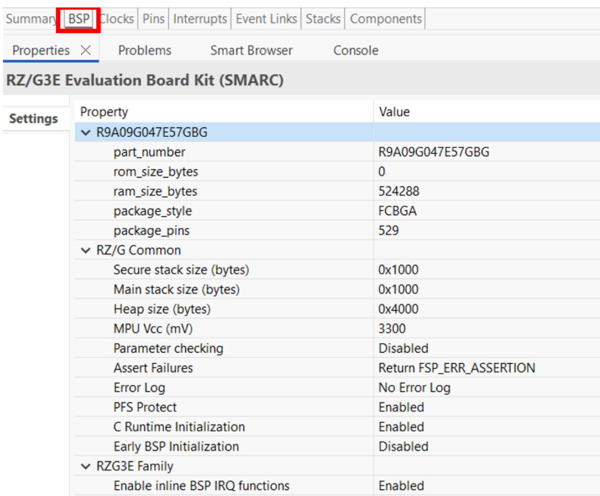


Figure 89 : Configuration BSP tab

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 [**Generate Project Content]** button, the BSP configuration contents are written to rzg_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. If a clock setting is invalid, the offending clock value is highlighted in red. It is still possible to generate code with this setting, but correct operation cannot be guaranteed. In the figure below, the xSPI clock SPI0CLK has been changed so the resulting clock frequency is 400 MHz instead of the required less than 267 MHz. This parameter is colored red.

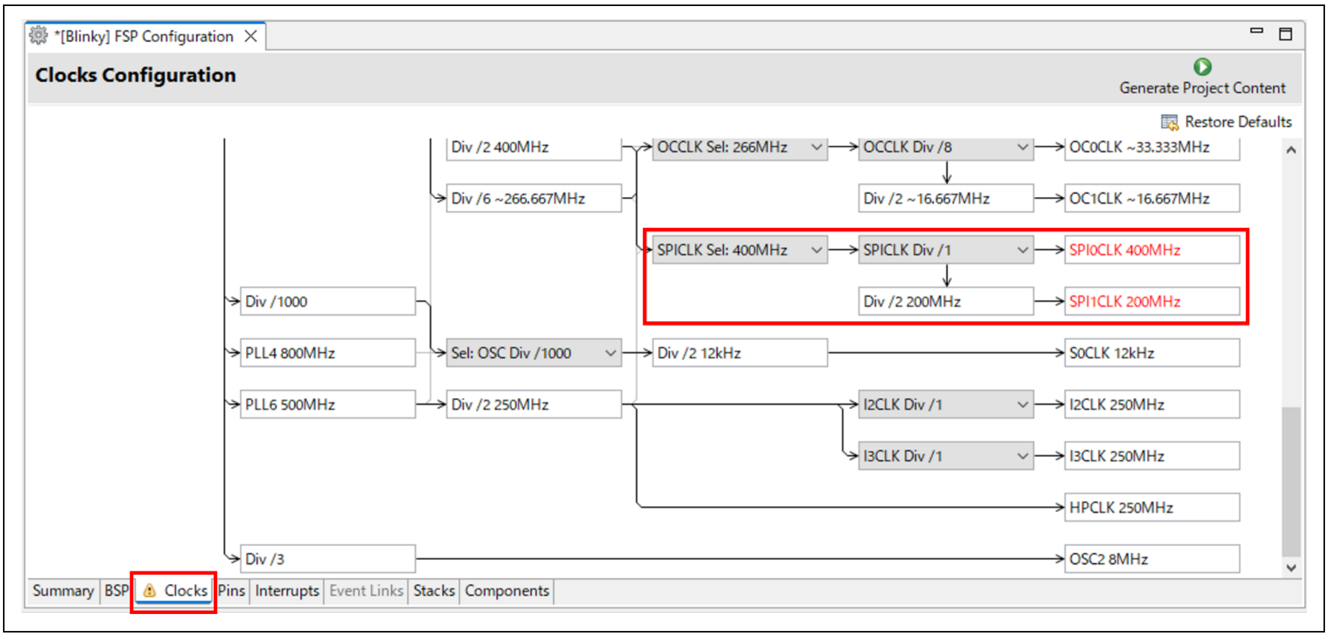


Figure 540 : Configuration Clocks tab

When you click **[Generate Project Content]** button, the clock configuration contents are written to rzg_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.

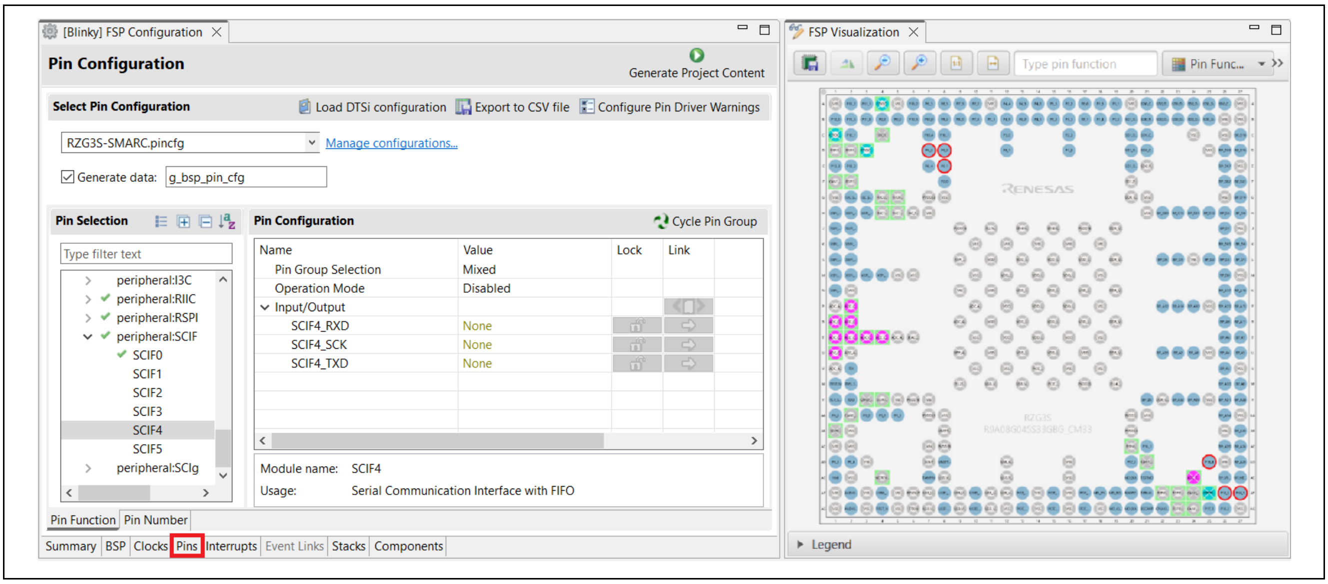


Figure 91 : Pin Configuration

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 P13_1 is already used by the GPT, and the attempt to connect to this pin to the Serial Communication Interface with FIFO (SCIF) results in dangling connection error. To fix this error, select another port from the pin drop-down list or disable the GPT.

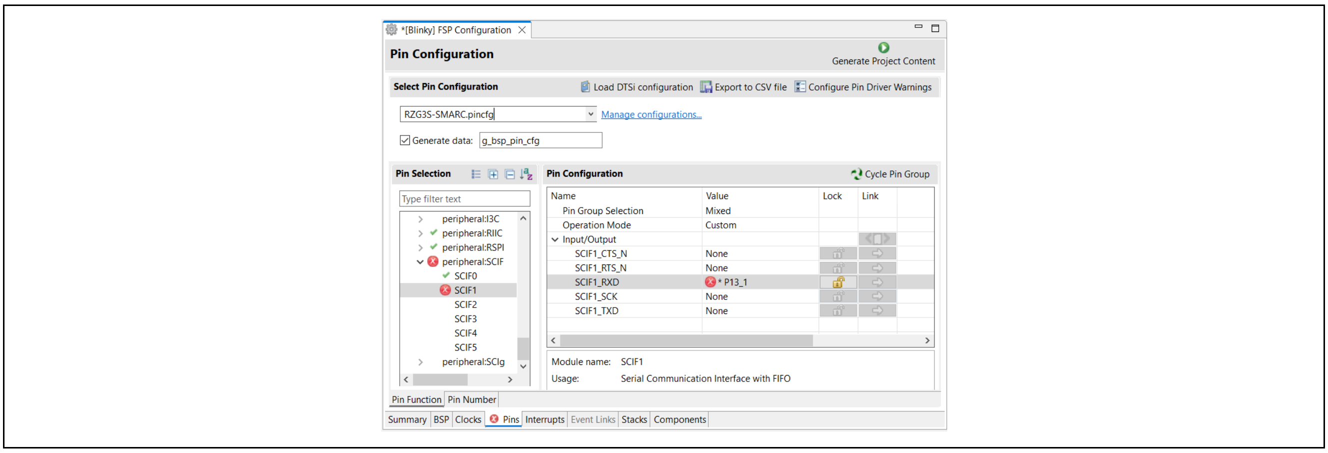


Figure 92 : e² studio Pin Configurator

When you click **[Generate Project Content]** button, the pin configuration contents are written to: rzg_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/G 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 how to apply the setting of “src/pin_data.c” to **Pins** tab, please refer to Setting GPIO with Flexible Software Package.

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.

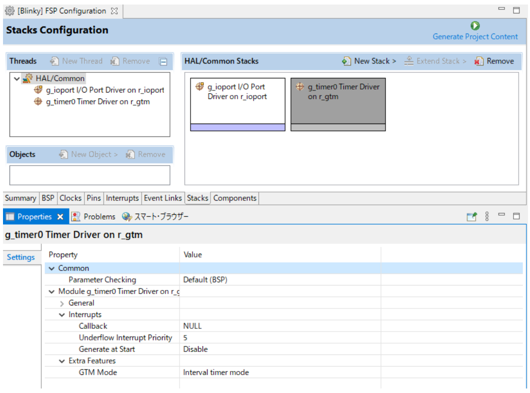


Figure 93 : Configuring Interrupts in the Stacks tab

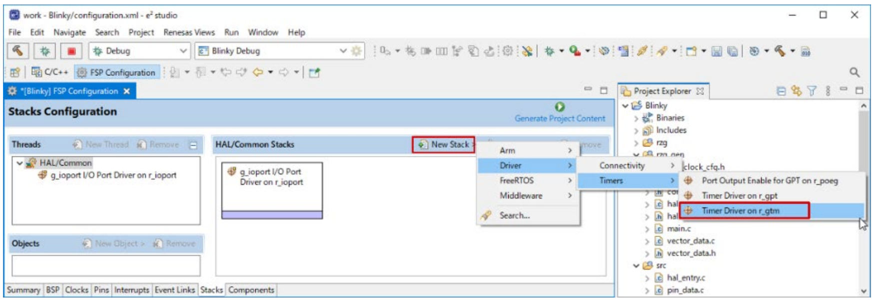


Figure94: Add new stack Timer (GTM)

5.2.6 Creating Interrupts from the Interrupts Tab

On the **Interrupts** tab, the interrupts of the driver which user selected in the **Stacks** tab are registered.

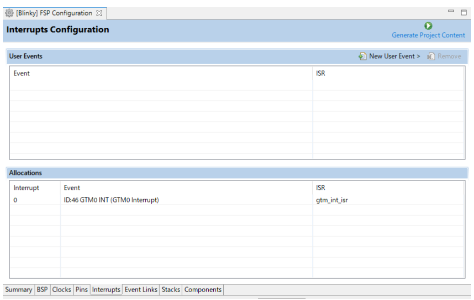


Figure 95 : Configuring interrupt in Interrupt Tab

Also, on the Interrupts tab, the user can add user's own peripheral interrupts. This can be achieved 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.

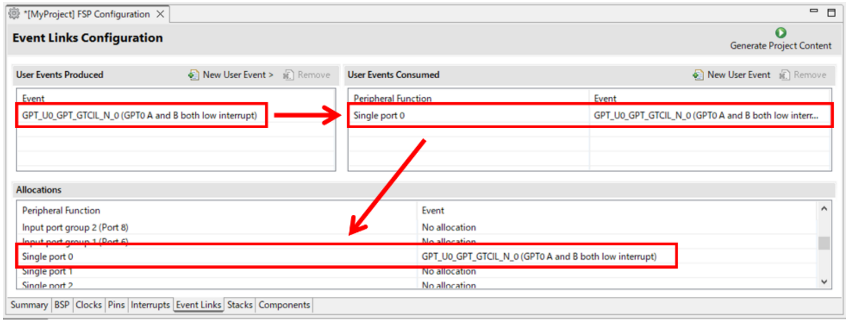


Figure 96 : e2 studio Project configurator – Viewing Event Links

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.

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**.

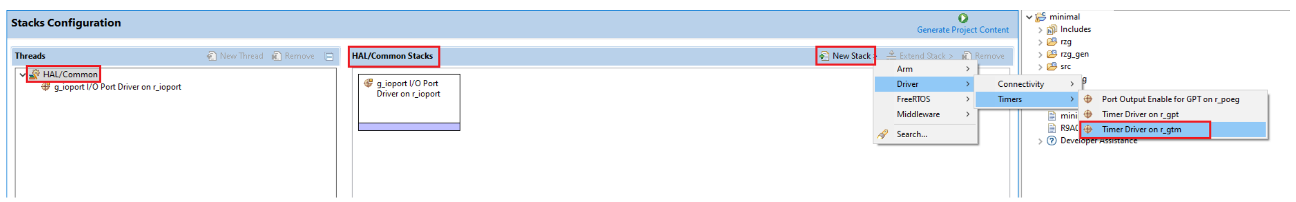


Figure 97 : e² studio Project configurator - Adding drivers

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 **[Generate Project Content]** button:

- The selected driver module and its files to the rzg/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?
rzg_gen/main.c	Contains main() calling generated and user code. When called, the BSP has already initialized the MPU.	Yes
rzg_gen/hal_data.c	Configuration structures for HAL Driver only modules.	Yes
rzg_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:
rzg_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/G 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.

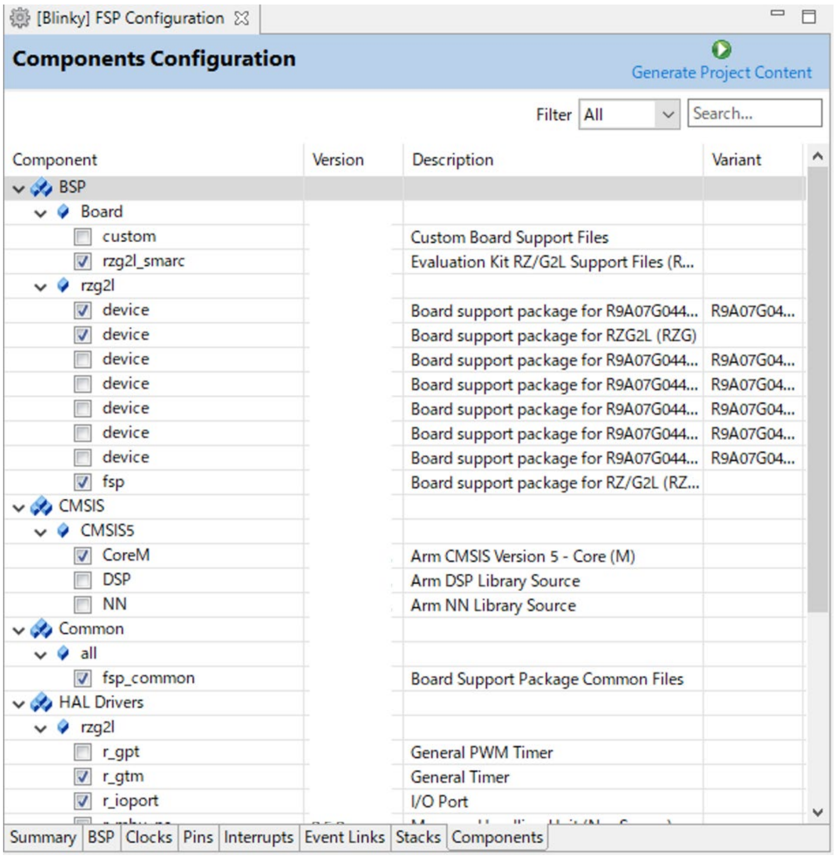


Figure 98 : Components Tab

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

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

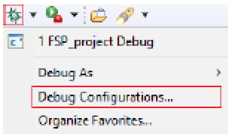
e² studio also creates configuration files in the rzg_cfg/fsp_cfg folder with configuration options set in the **Stacks** tab.

5.4 Debugging the Project

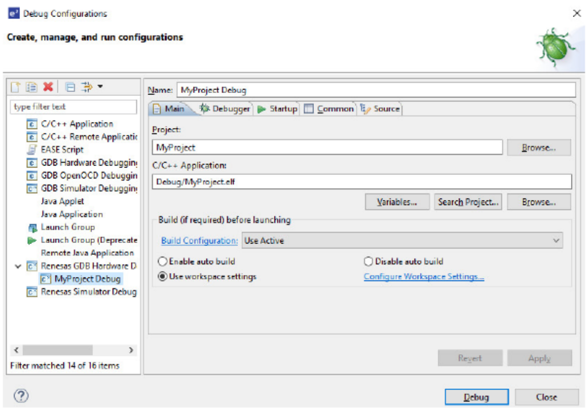
Once your project builds without errors, you can use the Debugger to download your application to the board and execute it.

To debug an application, follow these steps:

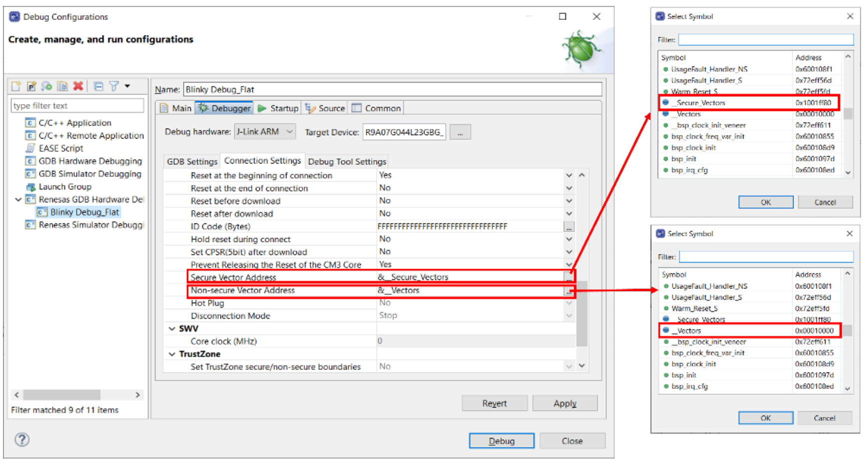
1. On the drop-down list next to the debug icon, select **Debug Configurations**.



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



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.



4. Connect the board to your PC via a standalone Segger J-Link debugger and click **[Debug]** button.

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 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.

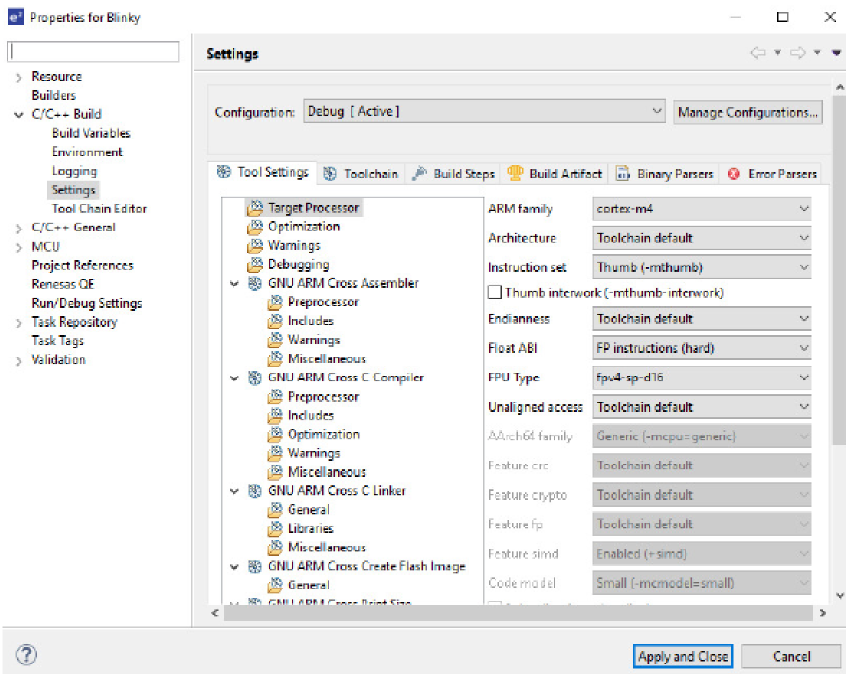


Figure 99 : e² studio Project toolchain settings

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 created project. (for example, script/fsp.ld).

5.1 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.

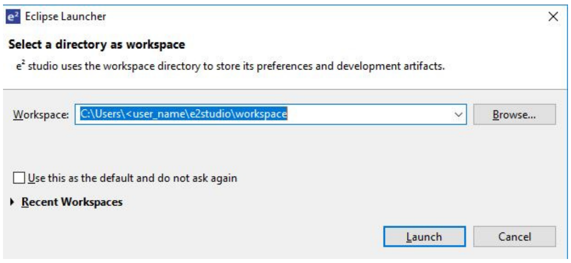


Figure 100 : Workspace Launcher dialog

   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.

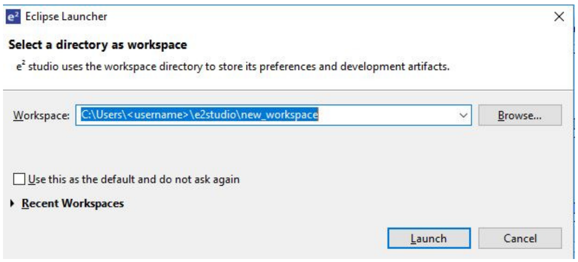


Figure 102 : Workspace Launcher dialog - Select Workspace

   c. Click **[Launch]** button.
   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.

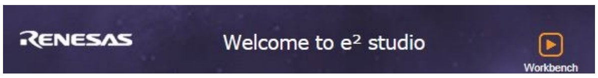


Figure 102 : Workbench arrow button

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.

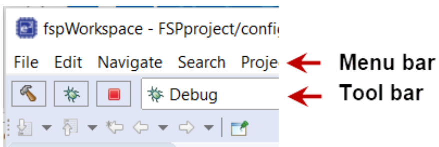


Figure 103 : Menu and tool bar

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

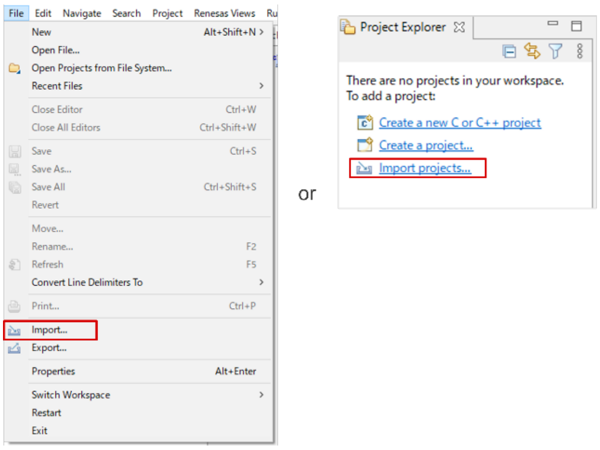


Figure 104 : File drop-down menu

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.

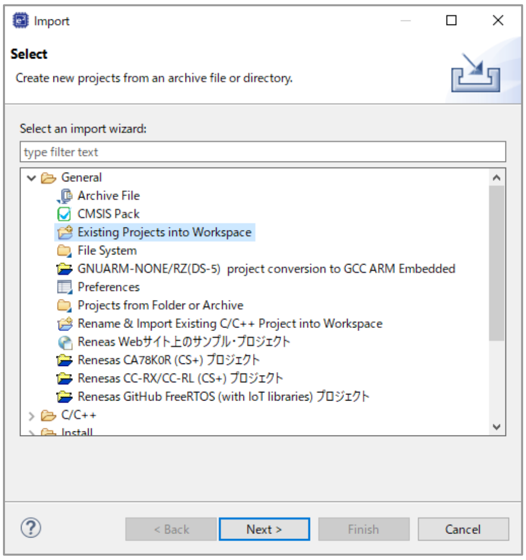


Figure 55 : Project Import dialog with "Existing Projects into Workspace" option selected

6. Click **[Next >]** button.
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.

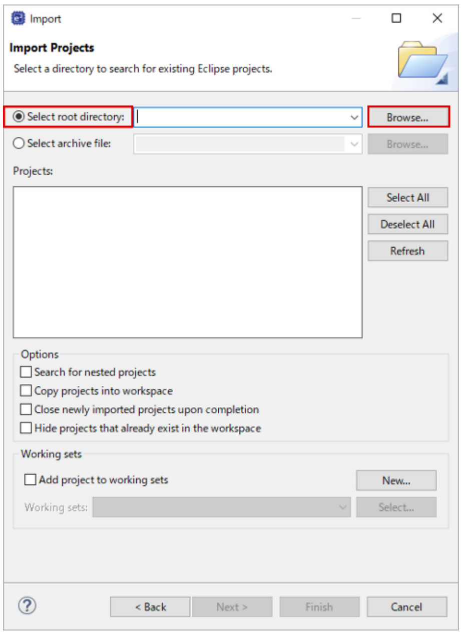


Figure 56 : Import Existing Project dialog 1 - Select root directory

    b. Click [**Browse]** button.
    c. For **Select root directory**, browse to the project folder that you want to import.
    d. Select the file for import.
    e. Click [**Open]** button.
    f. Select the project to import from the list of **Projects**, as shown in the following figure.

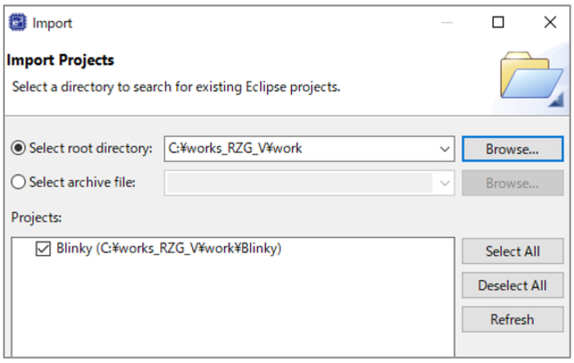


Figure 57 : Import Existing Project dialog 2

8. Click [**Finish]** button to import the project.

Revision History

Rev.	Date	Description	
		Page	Summary
3.1.0	Jul.22.25	40, 46 to 50,	Added the description for the template of the RZ/G3E CM33 cold boot Added the description how to debug the RZ/G3S CM33 and CM33_FPU at the same time. Added the description whether a project is bundled to another project when creating new project.
3.0.0	Mar.31.25	4 to 20, 31 to 35, 37 to 40, 46, 48, 50 to 52, 56	Updated the description and figure based on the latest development environment.
2.1.0	Sep.30.24	6 to 11, 13 to 22, 35 to 36, 42,44, 46,48, 59 to 67	Updated the description and figure based on the latest development environment.
2.01	Feb.13.24	6 to 11, 13 to 22,	Updated the description and figure based on the latest development environment.
		45	Updated the description of project creation.
		48	Added the section of how to avoid resource duplication description in the case of RZ/G3S project creation.
		60 to 69	Added the section of how migrate the project which using previous FSP version to latest FSP version.
2.00	Jan.9.24	1	Added RZ/G3S to the target device.
		5	Updated the description of the RAM initialization section.
		6 to 11	Updated the description and figure based on the latest development environment.
		12 to 22	Updated the development setup for Linux Host PC and FSP installation.
		26 to 32	Added description and figure for RZ/G3S SMARC EVK.
		33 to 36	Updated the description and figure based on the latest development environment.
		40	Removed the steps to configure Secure Vector and Non-secure Vector from the Debug step.
		46 to 51	Added description about the Pins and Clocks tabs.
		55	Updated the instructions for configuring Secure Vector and Non-secure Vector.
1.20	Nov.30.22	12	Updated version information of SEGGER J-Link driver and Libgen Update for GNU ARM Embedded Toolchains.
		18, 24	Updated installation procedure for FSP packs.
		67	Updated the method of specifying Secure Vector Address and Non-secure Vector Address.
1.10	Apr.27.22	5 to 53	Updated the description and figure based on the latest development environment.
1.01	Dec.3.21	14 to 22	Added a section of e² studio installation for Linux PC.
1.00	Jul.30.21	-	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 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 system-evaluation test for the given product.

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

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/.