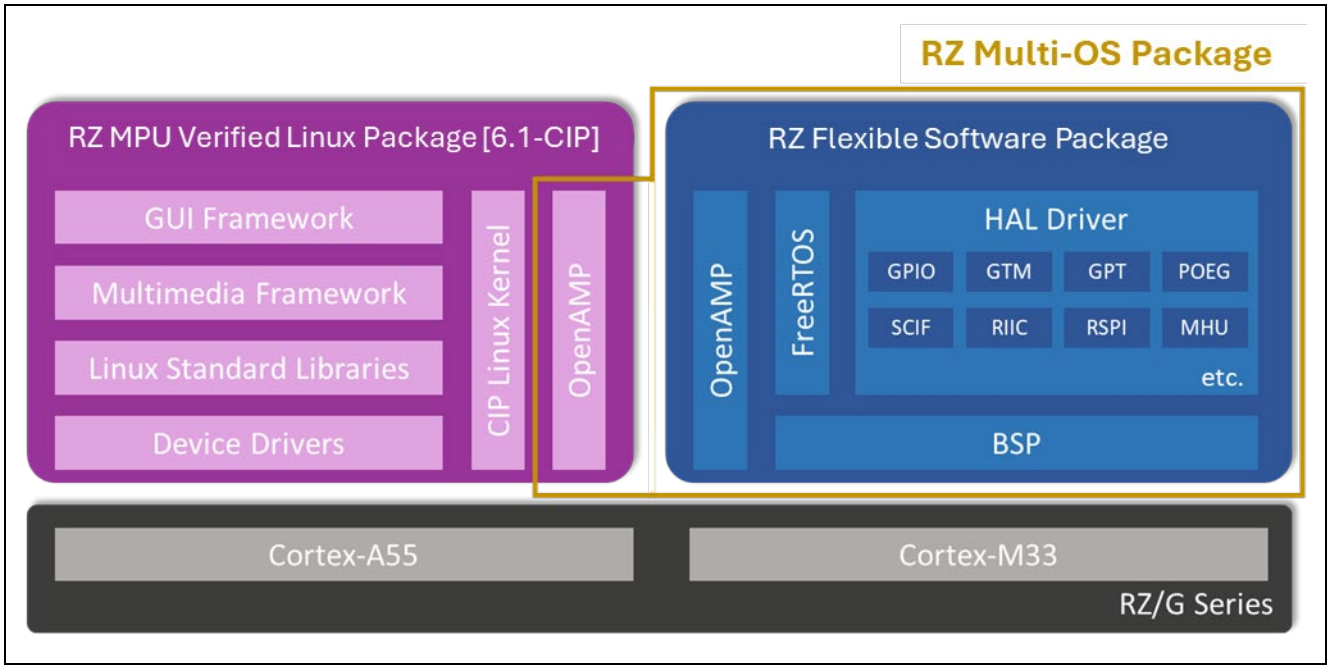
RZ/G3S

Quick Start Guide for RZ Multi-OS Package

Introduction

This document outlines the procedure for integrating the RZ Multi-OS Package into the RZ Verified Linux Package (referred to as VLP) for the RZ/G3S. By integrating the Multi-OS Package, users can efficiently establish a Multi-OS environment wherein Linux operates on the Cortex®-A55 and FreeRTOS/BareMetal runs on the Cortex-M33, with support for Inter-Processor Communication between these CPU cores.

This package requires the RZ Flexible Software Package (FSP) for an RTOS/BareMetal environment. The figure below illustrates the software stack for integrating the RZ Multi-OS Package with the RZ/G3S:



Here are brief descriptions of each component related to RZ Multi-OS Package:

- RZ FSP
  This software package consists of production ready peripheral drivers, FreeRTOS and portable middleware stacks and best in-case HAL drivers with low memory footprint.
- OpenAMP
  The framework includes the software components required for Asymmetric Multiprocessing (AMP) systems, such as Inter-Processor Communication.

Target Device

RZ/G3S

Contents

1. Specifications 3
2. Verified Operation Conditions 3
3. Sample Program Setup 3
3.1 Flexible Software Package Setup 3
3.2 Integration of Multi-OS Package related stuff 3
3.2.1 Using RZ MPU VLP v4.0.1 3
3.2.2 Using RZ/G VLP v3.0.7 4
3.3 Note for integration 5
3.4 Deployment of RZ/G VLP 5
4. Sample Program Invocation on RZ/G3S 6
4.1 Hardware Setup 6
4.2 CM33 Sample Program Setup 7
4.3 CM33 Sample Program Invocation for communicating with Linux 10
4.3.1 CM33 Sample Program Invocation using Segger J-Link 10
4.3.2 CM33 Sample program invocation with u-boot 16
4.3.3 CM33 Sample Program Invocation from remoteproc 17
4.3.4 CM33 Sample Program Invocation with BL2 of Trusted Firmware-A 17
4.4 CA55 Sample Program Invocation 18
4.5 Overview of Sample Program's behavior 20
5. CM33 cold boot support 21
5.1 RZ/G VLP Setup 21
5.2 Deployment of CA55 Build Artifacts to SMARC EVK 21
5.3 Setup and deployment of CM33 related stuff 23
6. Assignment of peripherals 24
7. Appendix 24
7.1 Setting for RPMsg Example supporting OpenAMP v2024.05 24
8. Reference Documents 24
Revision History 25

1. Specifications

Table 1-1 lists the on-chip peripheral modules to be used in this package.

Table 1-1 Peripheral modules to be used in this package

Peripheral module	Usage
Message Handling Unit (MHU)	Configure Inter-Processor Interrupt.
Serial Communications Interface with FIFO (SCIFA)	Perform standard serial communications sending and receiving console messages.
Interrupt controller (INTC)	Handle the following types of interrupts as shown below for example: • Processors should receive interrupts during buffered serial communications. • MHU module fires Inter-Processor Interrupt.
General Purpose Input Output (GPIO)	Configure I/O lines used by serial communications.
General Timer (GTM)	Configure the tick for FreeRTOS.

2. Verified Operation Conditions

Table 2-1 shows the verified operation conditions.

Table 2-1 Verified Operation Conditions

Item	Contents
Integrated Development Environment	e^2 studio 2026-07 or later
Toolchain	GNU Arm Toolchain 13.3.Rel1 AArch32 bare-metal target (arm-none-eabi)
Dependent Software	• RZ Flexible Software Package (FSP) v4.2.0 • RZ/G Verified Linux Package v3.0.7-update5 • RZ MPU Verified Linux Package v4.0.1

3. Sample Program Setup

3.1 Flexible Software Package Setup

Multi-OS Package expects RZ Flexible Software Package (FSP) to be installed in advance. For details on the installation, please refer to Getting Started with Flexible Software Package.

3.2 Integration of Multi-OS Package related stuff

3.2.1 Using RZ MPU VLP v4.0.1

This section describes how to integrate OpenAMP related stuff to RZ Verified Linux Package [6.1-CIP] (hereinafter referred to as VLP). The steps are based on **SMARC EVK of RZ/G3S Linux Start-up Guide** (hereinafter referred to as **Linux Start-up Guide**) included in **RZ MPU VLP v4.0.1**.

1. Follow the procedure stated from the beginning of **2.2 Building Images** to **(3) Add layers** of **Linux Start-up Guide**.
2. Download Multi-OS Package (r01an8260ej0420-rz-multi-os-pkg.zip) to a working directory and run the commands stated below:

```
cd ~/rzg_vlp_<pkg ver>
$ unzip <Multi-OS Dir>/r01an8260ej0420-rz-multi-os-pkg.zip
$ tar zxvf r01an8260ej0420-rz-multi-os-pkg/meta-rz-features_multi-os_v4.2.0.tar.gz
```

Here, <Multi-OS Dir> indicates the path to the directory where Multi-OS Package is placed.

3. Add the layer for Multi-OS Package, simply run the helper script and select the **meta-rzg3s-vlp4** layer from the list.

```
$ cd build
$ bash ../meta-rz-features/meta-rz-multi-os/add_meta_layer.sh
```

(Optional for remoteproc support)

4. Uncomment the following line in **meta-rz-features/meta-rz-multi-os/meta-rzg/meta-rzg3s/meta-rzg3s-vlp4/conf/layer.conf** for enabling remoteproc support:

```
MACHINE_FEATURES:append = " RZ_REMOTEPROC"
#MACHINE_FEATURES:append = " RZ_CM33_FIRMWARE_LOAD"
#MACHINE_FEATURES:append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES:append = " RZ_AWO_SUPPORT"
```

5. Start a build as described in **(5) Start a build** of **2.2 Building Images** as shown below:

```
$ MACHINE=smarc-rzg3s bitbake core-image-<target>
```

For details on the allowable value of <target>, please refer to **Linux Start-up Guide**.

3.2.2 Using RZ/G VLP v3.0.7

This section describes how to integrate OpenAMP related stuff to RZ/G Verified Linux Package [5.1-CIP] (hereinafter referred to as VLP). The steps are based on **SMARC EVK of RZ/G3S Linux Start-up Guide** (hereinafter referred to as **Linux Start-up Guide**) included in **RZ/G VLP v3.0.7-update5**.

1. Follow the procedure stated from the beginning of **2.2 Building Images** to **(3) Add layers** of **Linux Start-up Guide**.
2. Download Multi-OS Package (r01an8260ej0420-rz-multi-os-pkg.zip) to a working directory and run the commands stated below:

```
cd ~/rzg_vlp_<pkg ver>
$ unzip <Multi-OS Dir>/r01an8260ej0420-rz-multi-os-pkg.zip
$ tar zxvf r01an8260ej0420-rz-multi-os-pkg/meta-rz-features_multi-os_v4.2.0.tar.gz
```

Here, <Multi-OS Dir> indicates the path to the directory where Multi-OS Package is placed.

3. Add the layer for Multi-OS Package, simply run the helper script and select the **meta-rzg3s-vlp3** layer from the list.

```
$ cd build
$ bash ../meta-rz-features/meta-rz-multi-os/add_meta_layer.sh
```

(Optional for remoteproc support)

4. Uncomment the following line in **meta-rz-features/meta-rz-multi-os/meta-rzg/meta-rzg3s/meta-rzg3s-vlp3/conf/layer.conf** for enabling remoteproc support:

```
MACHINE_FEATURES_append = " RZ_REMOTEPROC"
#MACHINE_FEATURES_append = " RZ_CM33_FIRMWARE_LOAD"
#MACHINE_FEATURES_append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES_append = " RZ_AWO_SUPPORT"
```

5. Note: If any Patch files at '5. Notes' in **RZ/G Verified Linux Package Version 3.0.7-update5 Release Note** - Rev. 1.20 (hereinafter referred to as VLP3.0.7-update5 Release Note) are applied, please follow the below notes:

Note 5.1: please make sure that the layer for Multi-OS Package is added before applying the Patch files at '5. Notes' in **VLP3.0.7-update5 Release Note**.

Note 5.2: please apply the same procedure (Note 5.1) for part **4.3.4**, **5.1** in this document.

6. Start a build as described in **(5) Start a build** of **2.2 Building Images** as shown below:

```
$ MACHINE=smarc-rzg3s bitbake core-image-<target>
```

For details on the allowable value of <target>, please refer to **Linux Start-up Guide**.

3.3 Note for integration

By default, all clock-related configuration is handled on the Linux (main core) side. If the system design is changed such that clocks for specific drivers are configured by the RTOS (sub core) side, the implementation must be customized to ensure these settings do not conflict with Linux's clock initialization sequence.

The peripherals which are NOT enabled enter Module Standby Mode after Linux kernel is booted up. That means the peripherals used on sub core (CM33) might stop working at that time. To avoid such a situation, Multi-OS Package incorporates the patch below:

- 0001-Set-SCIF1-and-OSTM1-OSTM2-as-critical-clock.patch

This patch prevents SCIF channel1, GTM channel 1 and GTM channel 2 used in RPMsg demo program from entering Module Standby Mode. If you have any other peripherals which you would like to stop entering Module Standby implicitly, please update the patch as shown below:

```
diff --git a/drivers/clk/renesas/r9a08g045-cpg.c b/drivers/clk/renesas/r9a08g045-cpg.c
index 33a204fbe25c..dba2b4925e66 100644
--- a/drivers/clk/renesas/r9a08g045-cpg.c
+++ b/drivers/clk/renesas/r9a08g045-cpg.c
@@ -381,6 +381,9 @@ static const unsigned int r9a08g045_crit_mod_clks[] __initconst = {
  MOD_CLK_BASE + R9A08G045_IA55_CLK,
  MOD_CLK_BASE + R9A08G045_DMAC_ACLK,
  MOD_CLK_BASE + R9A08G045_VBAT_BCLK,
+ MOD_CLK_BASE + R9A08G045_SCIF1_CLK_PCK,
+ MOD_CLK_BASE + R9A08G045_OSTM1_PCLK,
+ MOD_CLK_BASE + R9A08G045_OSTM2_PCLK,
+ MOD_CLK_BASE + R9A08G045_xxxx,
 };
```

With respect to the allowable value for **xxxx** above, please refer to the source code below:

https://github.com/renesas-rz/rz_linux-cip/blob/rz-6.1-cip43/drivers/clk/renesas/r9a08g045-cpg.c#L375-L381

3.4 Deployment of RZ/G VLP

With respect to the deployment of Linux kernel, device tree and root filesystem for RZ/G3S, please refer to **Linux Start-up Guide**.

4. Sample Program Invocation on RZ/G3S

4.1 Hardware Setup

1. Connect J-Link to RZ/G3S SMARC EVK. For details, please refer to Getting Started with Flexible Software Package.
2. Connect Pmod USBUART to the upper side of PMOD 3A connector of Smarc EVK as shown below for securing the console for sample program.

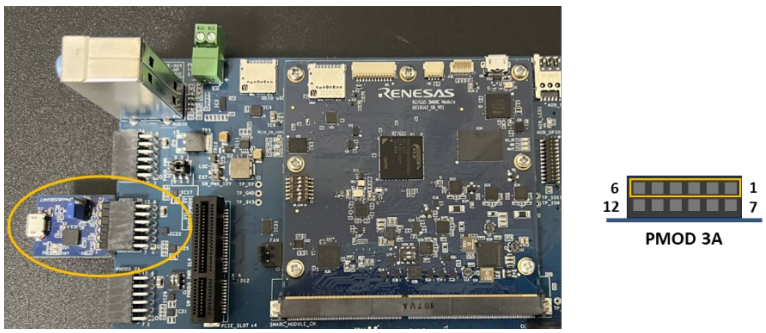


Figure 4-1. Connection between Pmod USBUART and RZ/G3S SMARC EVK

4.2 CM33 Sample Program Setup

Here are the procedures for setting up the sample program running on CM33:

1. Extract **r01an8260ej0420-rz-multi-os-pkg.zip** on your development PC.
2. Extract **<device>_rpmsg_<com_type>_example.zip** included in **r01an8260ej420-rz-multi-os-pkg**. Here, <device> should be any of **rzg3s_cm33** or **rzg3s_cm33_fpu**. Also, <com_type> should either of **linux_rtos** or **rtos_rtos**.
3. Open e^2 studio and click **File** > **Import**.
4. Double-click **General** and select **Existing Projects into Workspace** as shown in Figure 4-2:

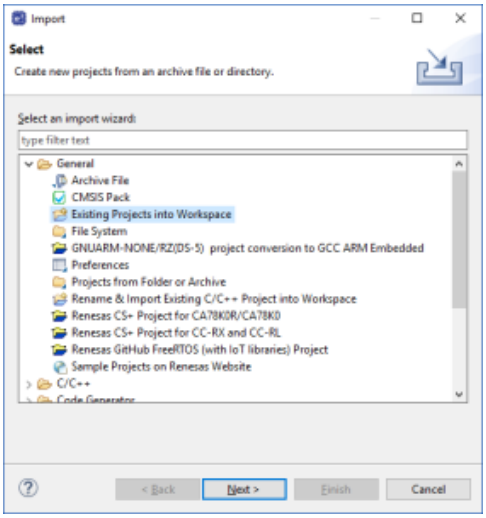


Figure 4-2. Import sample project (1)

5. Input the path to the directory of sample project you would like to import to **Select root directory**, press **Enter** key and click **Finish** button.

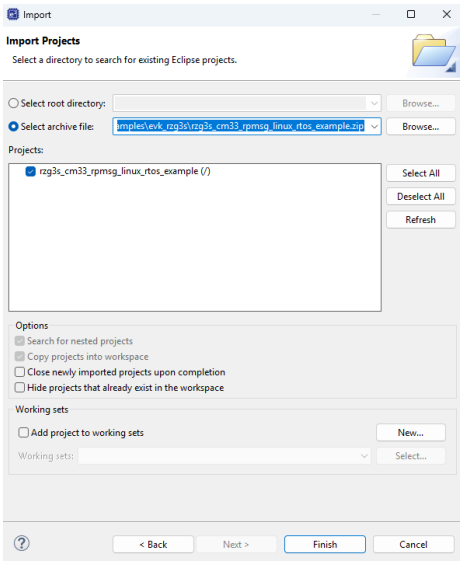


Figure 4-3. Import sample project (2)

(Optional for configuring RPMsg channel)

6. By default, RPMsg channel 0 is configured to be used on CM33. If you would like to change the channel, you need to open the property of **MainTask#0** on FSP Smart Configurator, specify the channel number you would like to use for **Thread Context** and push **Generate Project Content** button. If **Generate Project Content** pop-up is shown, click **Proceed** to reflect the changes to source code.

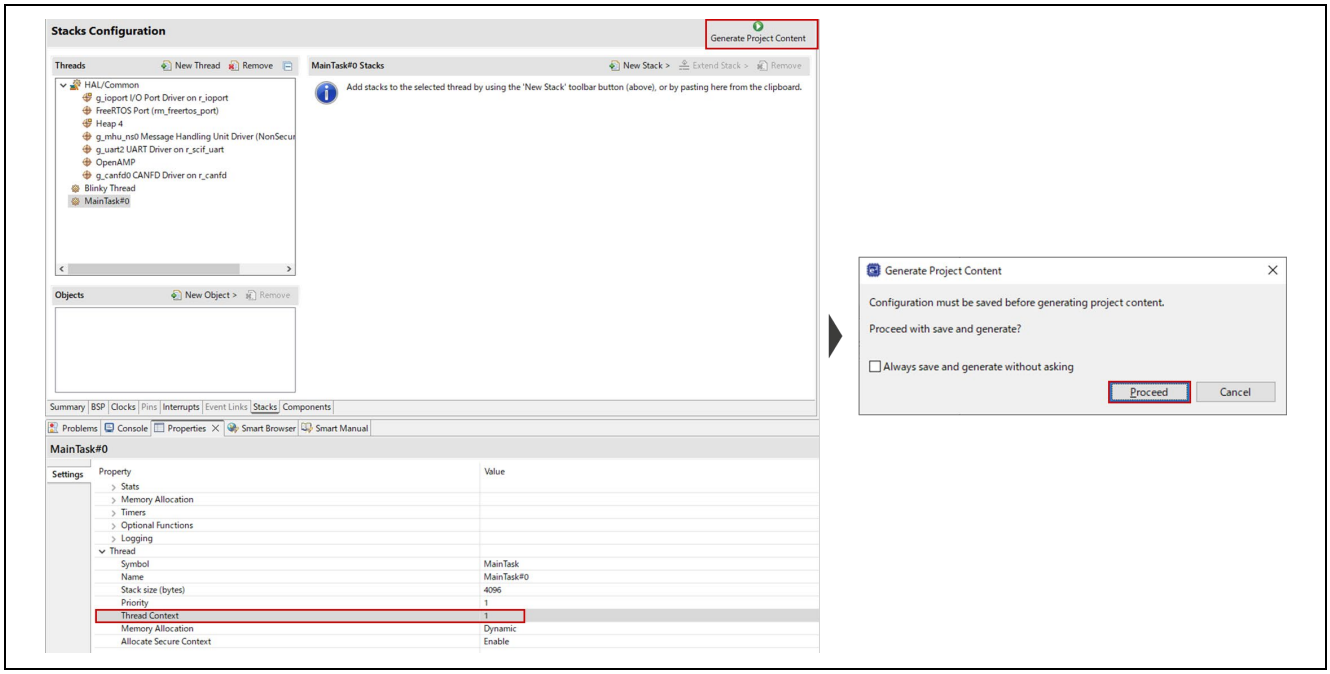


Figure 4-4. RPMsg Channel Setting

(Optional for remoteproc support)

7. Change the value for **ENABLE_REMOTEPROC** defined in **platform_info.h** from 0 to 1 as shown below:

```
#define ENABLE_REMOTEPROC (1U)
```

(Optional for CM33 cold boot support)

8. Configure **Launch Cortex-A55(core0)** as **Enabled** and Click **Generate Project Content** to generate the updated source code.

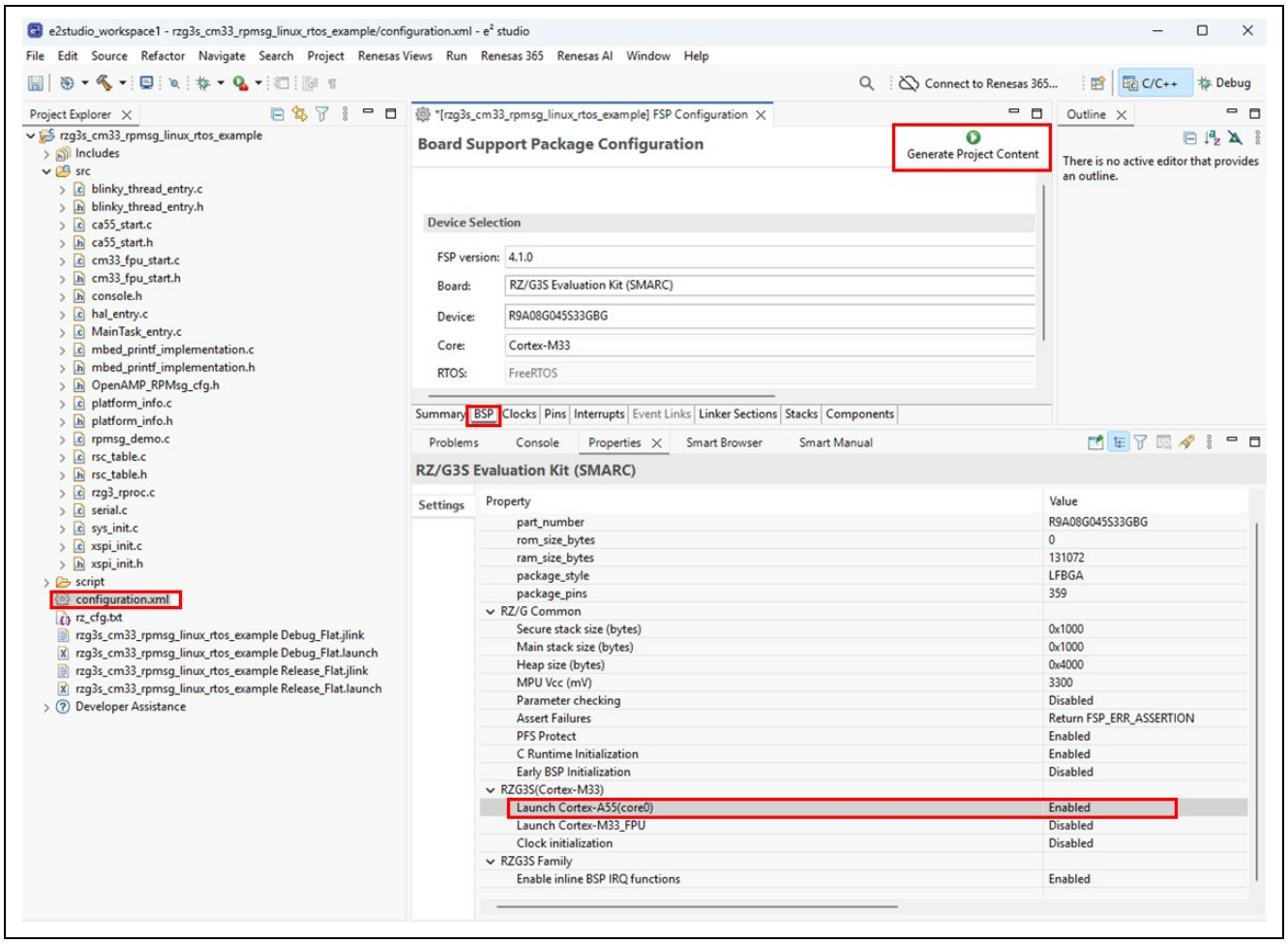


Figure 4-5. Linux boot configuration from CM33 cold boot

9. Build the project from **Choose Project** > **Build Project**.
10. If the build is successfully completed, the following files should be generated in Debug and/or Release folder in accordance with the active Build Configuration.
    - <device>_rpmsg_linux_rtos_example.elf (Note 1)
    - <device>_rpmsg_linux_rtos_example.srec (Note 1)
    - <device>_rpmsg_rtos_rtos_example.elf (Note 1)
    - <device>_rpmsg_rtos_rtos_example.srec (Note 1)

Notes: 1. The possible string for <device> is any of **rzg3s_cm33** and **rzg3s_cm33_fpu**.

4.3 CM33 Sample Program Invocation for communicating with Linux

4.3.1 CM33 Sample Program Invocation using Segger J-Link

You need to follow the following steps to invocate CM33 sample program with Segger J-Link.

(Optional for CM33 cold boot support)

1. Select **Run** > **Debug Configurations...** if CM33 is configured as Boot CPU for RZ/G3S.

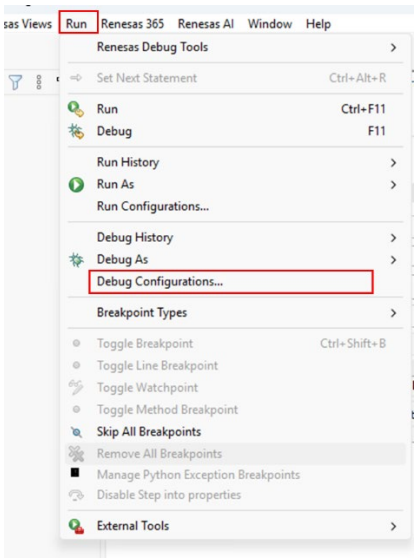


Figure 4-7. Debug Configuration (1)

(Optional for CM33 cold boot support)

2. Click **rzg3s_cm33_rpmsg_<com_type>_example Debug_Flat** or **rzg3s_cm33_rpmsg_<com_type>_example Release_Flat**.

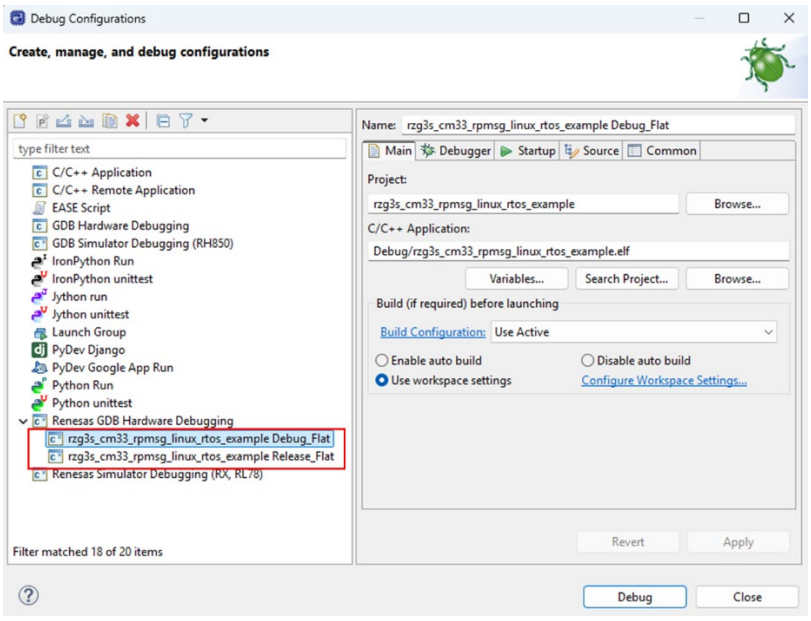


Figure 4-8. Debug Configuration (2)

(Optional for CM33 cold boot support)

3. Click **Debugger** tab and select **Connection Settings** as shown below:

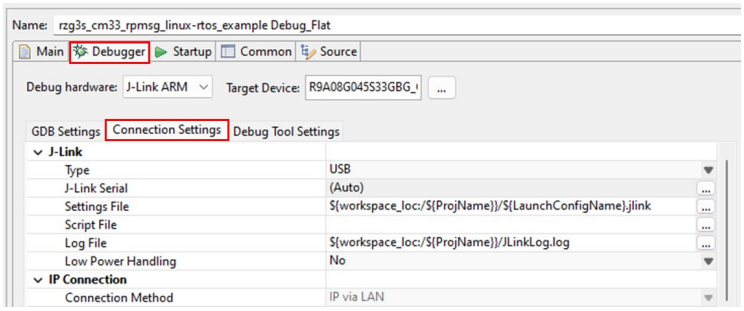


Figure 4-9. Debugger - Connection Settings

(Optional for CM33 cold boot support)

4. Configure **Reset after download** as **Yes**.

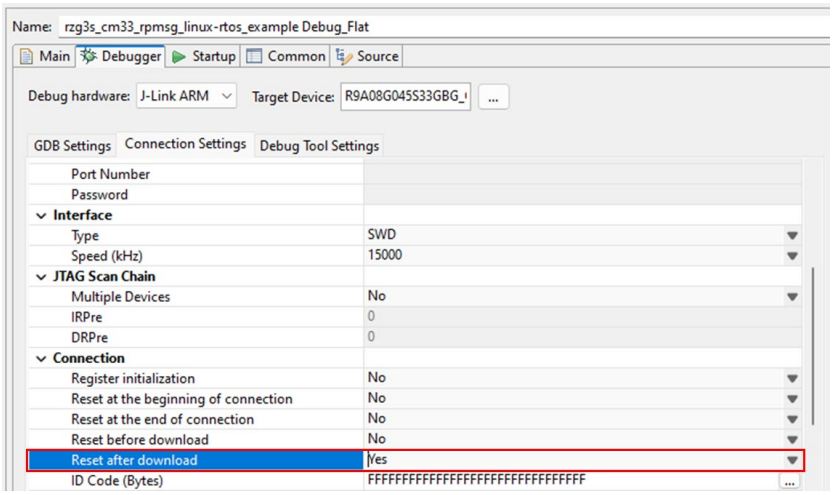


Figure 4-10. Reset after download Setting

(Optional for CM33 cold boot support)

5. Select **Startup** tab and click **Add...** as shown below:

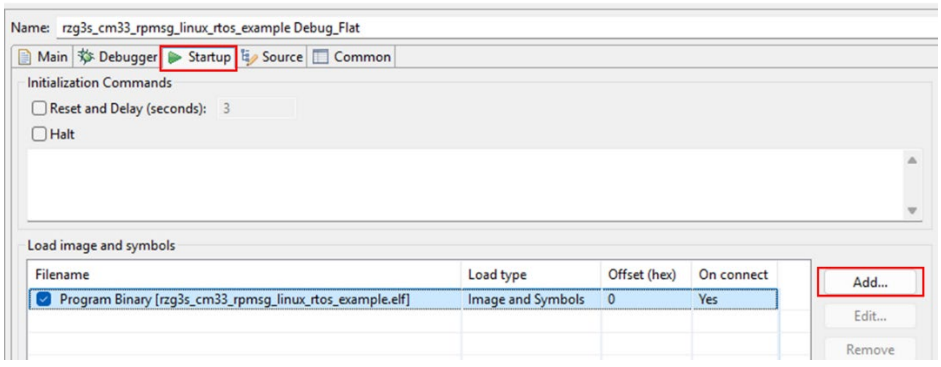


Figure 4-11. Add download module (1)

(Optional for CM33 cold boot support)

6. Click **Workspage...**, choose **rzg3s_cm33_rpmsg_<com_type>_example_cm33boot.srec** and click **OK**.

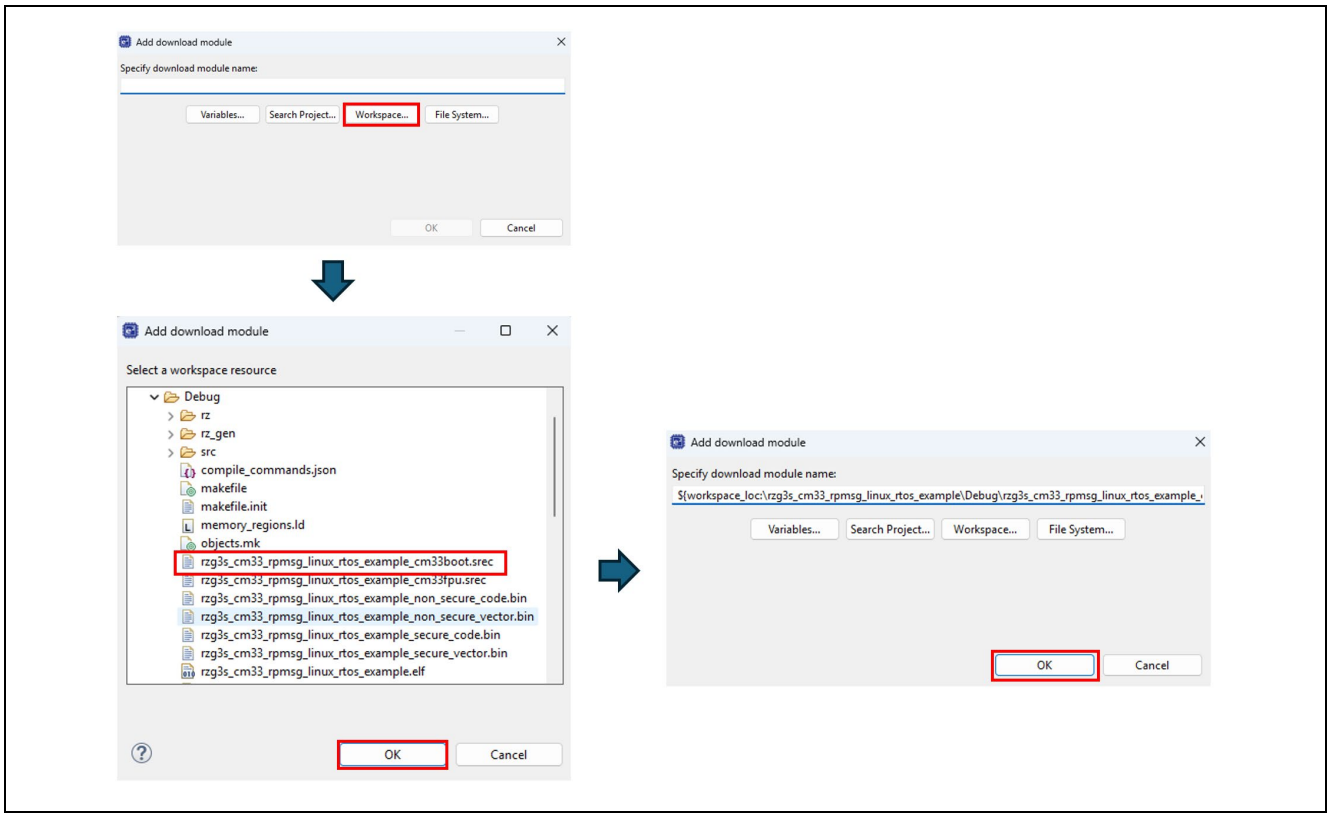


Figure 4-12. Add download module (2)

(Optional for CM33 cold boot support)

7. Configure **Load type** of **Program Binary [rzg3s_cm33_rpmsg_<com_type>_example.elf]** and **rzg3s_cm33_rpmsg_<com_type>_example_cm33boot.srec** as **Symbols only** and **Image only**, respectively.

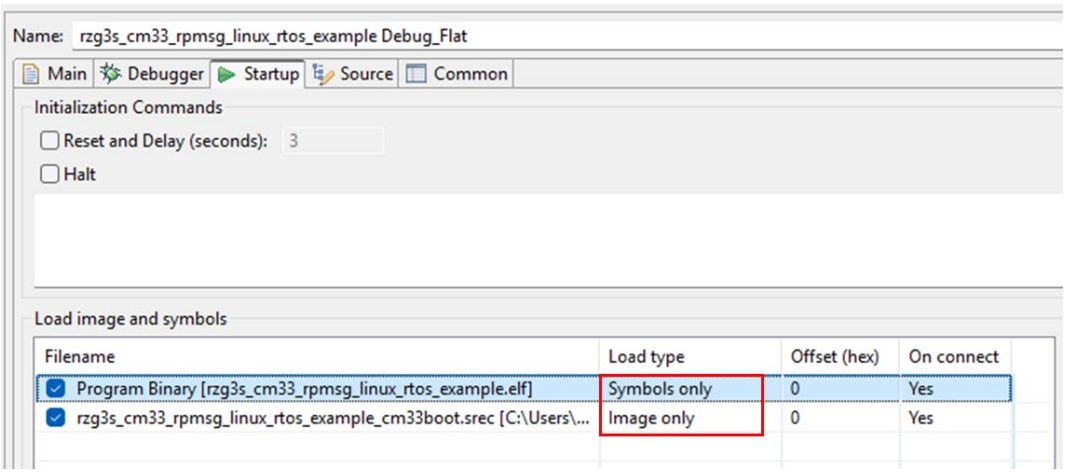


Figure 4-13. Configuration of Load type

8. Click Debug button to launch Debug Perspective.

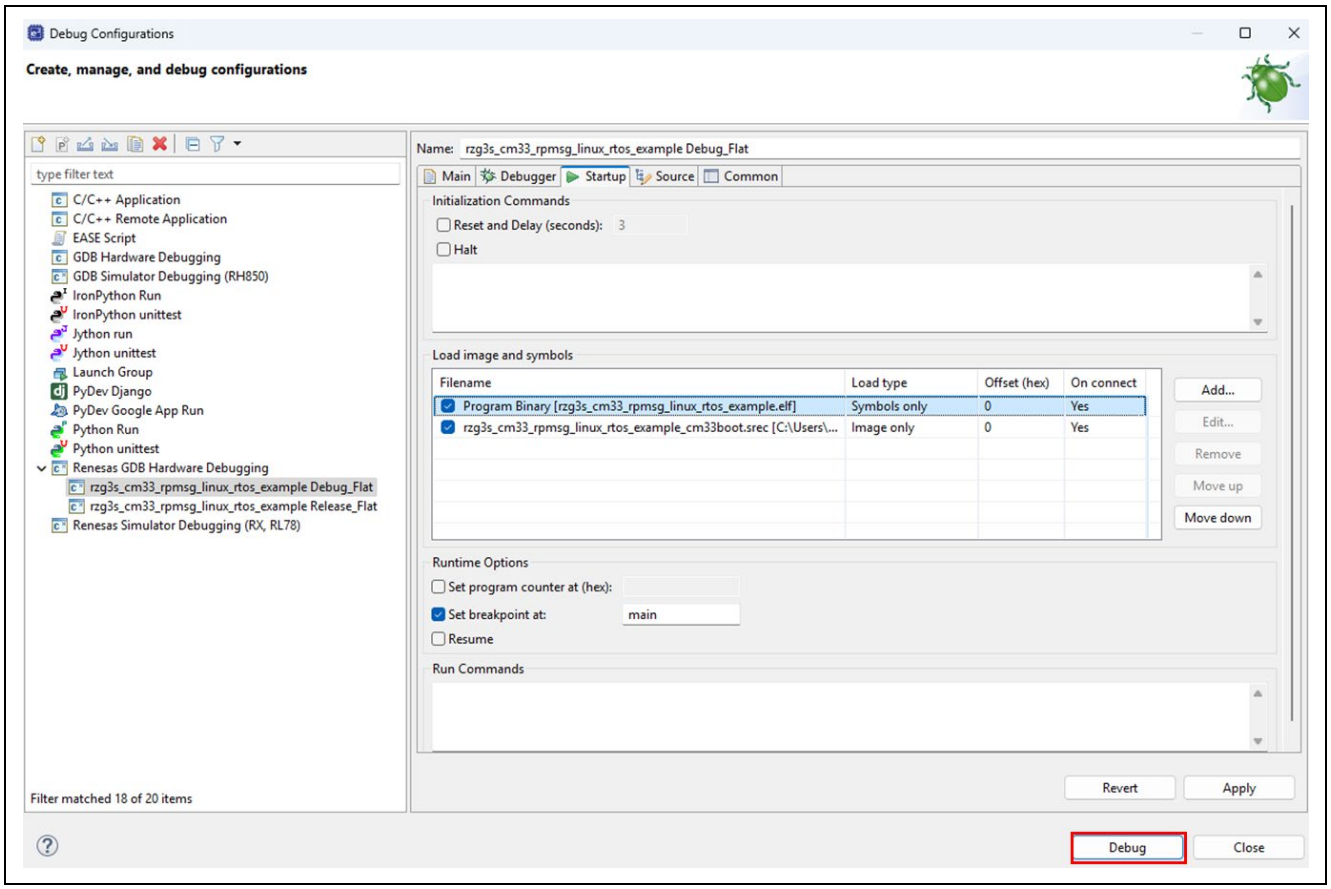


Figure 4-14. Debug Perspective Launch

If **Confirmation Perspective Switch** window below appears, press **Switch** to continue:

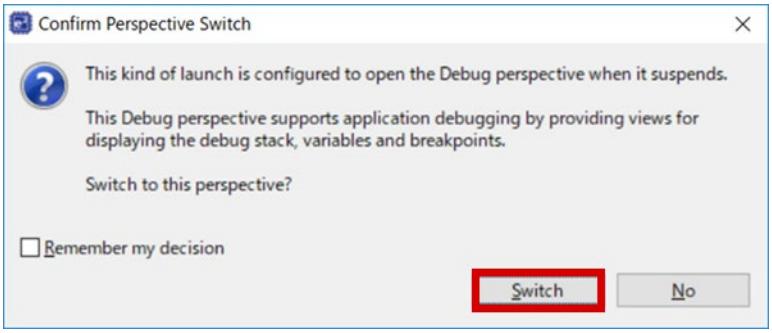


Figure 4-15. Confirm Perspective Switch

9. When the debug perspective is opened, Program Counter (PC) should be located at the top of **Entry_Function_S** function. Then, you need to press the button indicated by a red arrow in Figure 4-16:

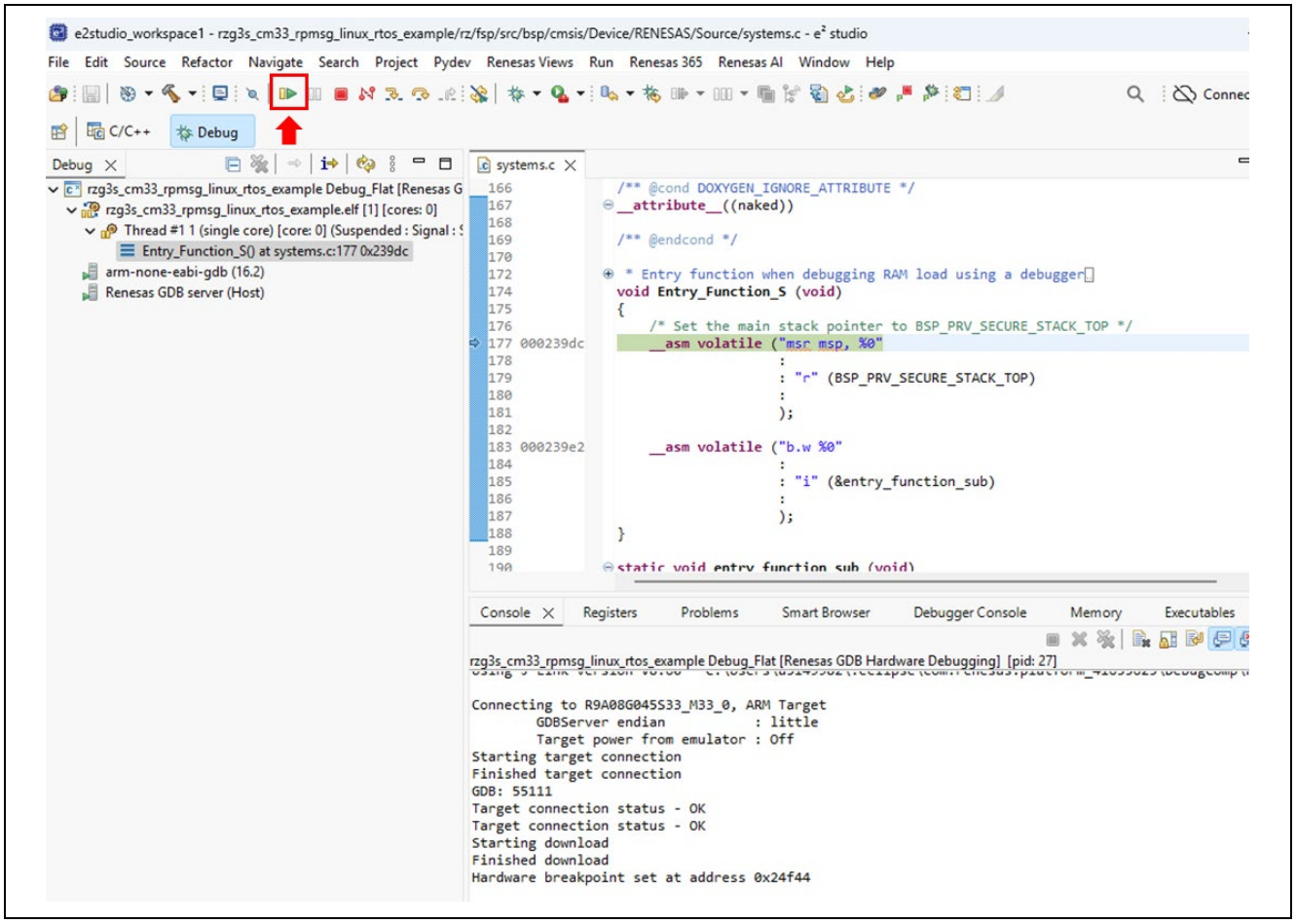


Figure 4-16. Start to debug RPMsg Sample Program (1)

10. The program should stop at the top of the **main** function. Then, click the same button as the previous step.

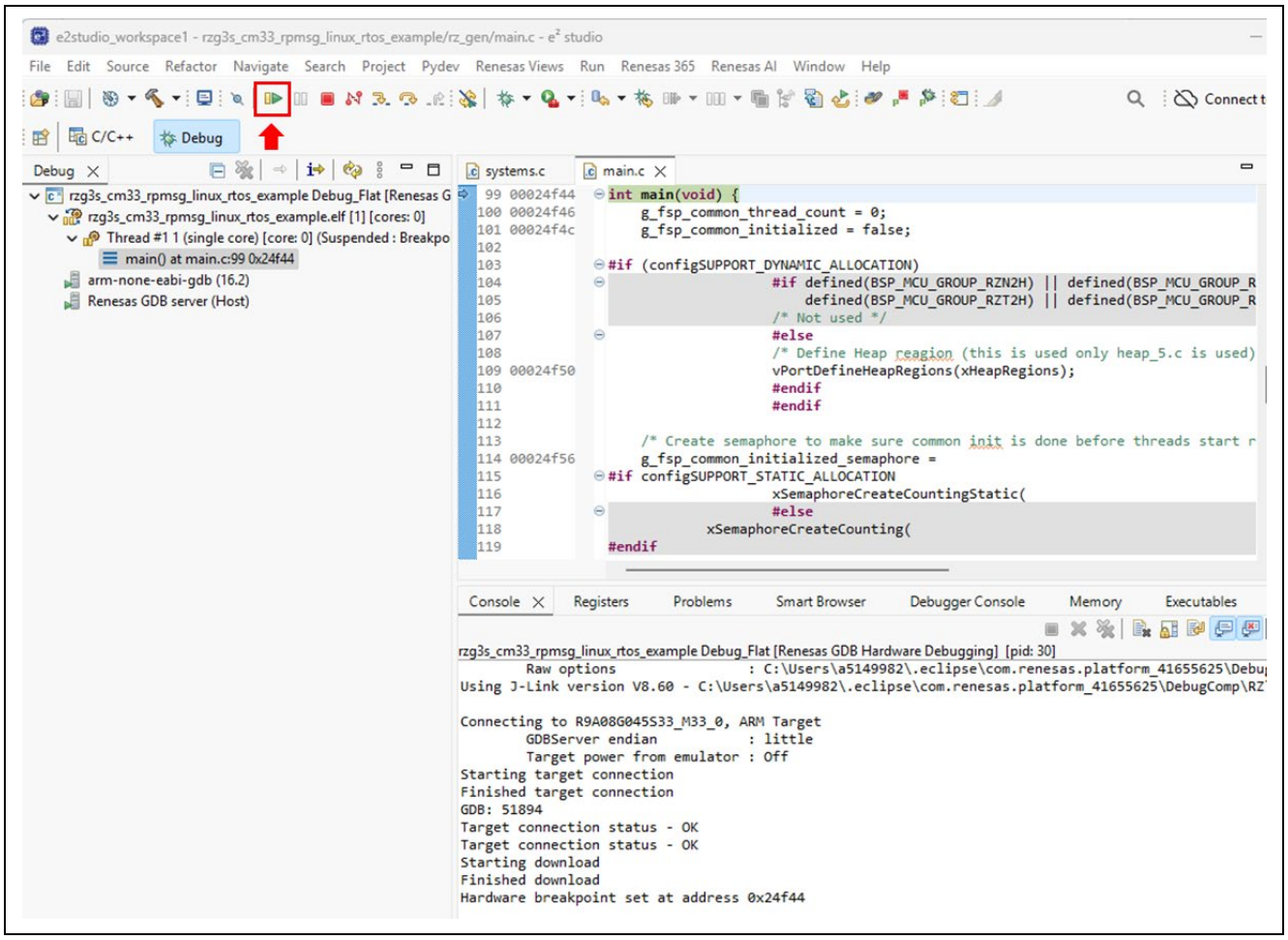


Figure 4-17. Start to debug RPMsg Sample Program (2)

11. Now that CM33 sample program has been started, the following message is shown on the console connected to Pmod USBUART: (Note)

```
Successfully probed IPI device
Successfully open uio device: 42F00000.rsctbl.
Successfully added memory device 42F00000.rsctbl.
Successfully open uio device: 43000000.vring-ctl0.
Successfully added memory device 43000000.vring-ctl0.
Successfully open uio device: 43200000.vring-shm0.
Successfully added memory device 43200000.vring-shm0.
Initialize remoteproc successfully.
creating remoteproc virtio
initializing rpmsg vdev
```

At this point of time, the CM33 program is waiting for the establishment of rpmsg channel with CA55.

4.3.2 CM33 Sample program invocation with u-boot

You can invoke CM33 sample program from u-boot by following the procedure described below:

1. Copy the binary files generated at 8 of section 4.2 to microSD card.
2. Insert the microSD card into SD1 of RZ/G3S SMARC EVK.
3. Turn on SMARC EVK by pressing the Power button for a few seconds.
4. Hit any key to stop autoboot within 3 seconds after the following message is shown in the console

```
U-Boot 2024.07 (Sep 22 2025 - 10:51:32 +0000)
CPU:   Renesas Electronics CPU rev 1.0
Model: smarc-rzg3s
DRAM:  896 MiB
Core:  31 devices, 15 uclasses, devicetree: separate
MMC:   sd@11c00000: 0, sd@11c10000: 1, sd@11c20000: 2
Loading Environment from MMC... Reading from MMC(0)... OK
In:    serial@1004b800
Out:   serial@1004b800
Err:   serial@1004b800
Net:
Error: ethernet@11c30000 No valid MAC address found.
No ethernet found.

Hit any key to stop autoboot:  0
=>
```

5. Load the binary files you copied at step1 from microSD card to RAM by executing the commands stated below on the console. Here, N stands for the partition number in which you stored the binary files.

- **For CM33**

```
=> setenv cm33_start 'dcache off; mmc dev 1;
fatload mmc 1:N 0x00023000 rzg3s_cm33_rpmsg_linux_rtos_example_secure_vector.bin;
fatload mmc 1:N 0x00023890 rzg3s_cm33_rpmsg_linux_rtos_example_secure_code.bin;
fatload mmc 1:N 0x10023800 rzg3s_cm33_rpmsg_linux_rtos_example_non_secure_vector.bin;
fatload mmc 1:N 0x10023840 rzg3s_cm33_rpmsg_linux_rtos_example_non_secure_code.bin;
mw.l 0x11020844 0x0001312C; mw.l 0x11020848 0x0001312C;
mw.l 0x1102084C 0x00023000; mw.l 0x11020850 0x10023800;
mw.l 0x11010504 0x00010001; mw.l 0x11010804 0x00040004;
mw.l 0x11010804 0x00070007; dcache on'
=> saveenv
=> run cm33_start
```

- **For CM33-FPU**

```
=> setenv cm33fpu_start 'dcache off; mmc dev 1;
fatload mmc 1:N 0x00060000 rzg3s_cm33-fpu_rpmsg_linux_rtos_example_secure_vector.bin;
fatload mmc 1:N 0x00060890 rzg3s_cm33-fpu_rpmsg_linux_rtos_example_secure_code.bin;
fatload mmc 1:N 0x10060800 rzg3s_cm33-fpu_rpmsg_linux_rtos_example_non_secure_vector.bin;
fatload mmc 1:N 0x10060840 rzg3s_cm33-fpu_rpmsg_linux_rtos_example_non_secure_code.bin;
mw.l 0x11020884 0x0001312C; mw.l 0x11020888 0x0001312C;
mw.l 0x1102088C 0x00060000; mw.l 0x11020890 0x10060800;
mw.l 0x11010504 0x01000100; mw.l 0x11010804 0x04000400;
mw.l 0x11010804 0x07000700; dcache on'
=> saveenv
=> run cm33fpu_start
```

Note: If N = 1, using “`fatload`”. If N = 2, using "`ext4load`”.

6. CM33/CM33-FPU sample program is now started.

4.3.3 CM33 Sample Program Invocation from remoteproc

Please enable option remoteproc support in section **3.2 Integration of Multi-OS Package related stuff** when building Linux image for running this feature.

On RZ/G3S SMARC EVK, you can invoke RPMsg sample program with remoteproc by following the procedure stated below:

1. Booting up Linux by following **5. Booting and Running Linux** of **Linux Start-up Guide**
2. Invoke the command below to specify RPMsg sample program to be loaded:

```
root@smarc-rzg3s:~# echo <device>_rpmsg_linux_rtos_example.elf >
/sys/class/remoteproc/remoteprocX/firmware

(Note 1)(Note 2)
```

3. Kick CM33 by invoking the command below:

```
root@smarc-rzg3s:~# echo start > /sys/class/remoteproc/remoteprocX/state
```

If CM33 starts to work successfully, the following message should be shown:

```
root@smarc-rzg3s:~# echo start > /sys/class/remoteproc/remoteproc0/state
[  185.663206] remoteproc remoteproc0: powering up cm33
[  185.731650] remoteproc remoteproc0: Booting fw image
rzg3s_cm33_rpmsg_linux_rtos_example.elf, size 1211812
[  185.747645]  remoteproc0#vdev0buffer: assigned reserved memory node
vdev0buffer@0x43200000
[  185.764284]  remoteproc0#vdev0buffer: registered virtio0 (type 7)
[  185.771036] remoteproc remoteproc0: remote processor cm33 is now up
```

Note: 1. **<device>** should be either rzg3s_cm33 or rzg3s_cm33-fpu.
2. **remoteprocX** can be either remoteproc0 or remoteproc1. When you would like to kick CM33, remoteproc0 should be specified. Also, remoteproc1 should be specified for kicking CM33_FPU.

Note: The firmware components are stored in the file system at **/lib/firmware** folder. Please replace it if any updates to firmware.

4.3.4 CM33 Sample Program Invocation with BL2 of Trusted Firmware-A

On RZ/G3S SMARC EVK, you can invoke RPMsg sample program from BL2 of Trusted Firmware-A by the procedure stated below. Be sure to configure CA55 as boot CPU when enabling this feature.

1. If you build **RZ MPU VLP v4.0.1**, please uncomment the following line in meta-rz-multi-os/meta-rzg/meta-rzg3s/meta-rzg3s-vlp4/conf/layer.conf

```
#MACHINE_FEATURES:append = " RZ_REMOTEPROC"
MACHINE_FEATURES:append = " RZ_CM33_FIRMWARE_LOAD"
#MACHINE_FEATURES:append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES:append = " RZ_AWO_SUPPORT"
```

If you build **RZ/G VLP v3.0.7-update5**, please uncomment the following line in meta-rz-multi-os/meta-rzg/meta-rzg3s/meta-rzg3s-vlp3/conf/layer.conf

```
#MACHINE_FEATURES_append = " RZ_REMOTEPROC"
MACHINE_FEATURES_append = " RZ_CM33_FIRMWARE_LOAD"
#MACHINE_FEATURES_append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES_append = " RZ_AWO_SUPPORT"
```

2. Rebuild Trusted Firmware-A as shown below.

```
MACHINE=smarc-rzg3s bitbake trusted-firmware-a -c cleansstate
MACHINE=smarc-rzg3s bitbake firmware-pack -c cleansstate
MACHINE=smarc-rzg3s bitbake core-image-minimal
```

3. Re-program the resultant BL2 and FIP binary files using FlashWriter.
4. Program rzg3s_cm33_rpmsg_<com_type>_example.srec using FlashWriter with the parameter stated below:

- For eMMC Boot mode:

Partition Area	Start Address in sector	Program Start Address
1	1000	80200000

- For xSPI Boot Mode:

Address to load to RAM	Address to save to ROM
80200000	200000

4.4 CA55 Sample Program Invocation

On RZ/G3S SMARC EVK, you need to follow the procedure shown below to invoke CA55 sample program running on Linux.

1. Boot up Linux by executing the following command on u-boot:

```
run bootcmd
```

2. Login as **root**.

```
smarc-<device> login: root
```

3. Run CA55 sample program by executing the following command on Linux.

```
root@smarc-<device>:~# rpmsg_sample_client
```

4. Then, you can see the following message on the console of Linux side.

```
****************************************
*   rpmsg communication sample program   *
****************************************

1. communicate between CM33 cores
2. communicate between CM33 and CA55

e. exit

please input
>
```

5. Input the number which performs the communication you would like to try on the console. Please note that 1 is allowable ONLY when remoteproc support is enabled. Also, you must NOT invoke the CM33

program in advance. Meanwhile, in case of selecting 2, you need to invoke the CM33 program in advance.

6. In case of typing 1, the communication between CM33 cores should be established and the communication log is repeatedly displayed via the Pmod USBUART. Also, when 2 is typed, you can see the following message on Linux console:

```
[XXX] proc_id:0 rsc_id:0 mbx_id:0
metal: info:      metal_uio_dev_open: No IRQ for device 10400000.mbox-uio.
metal: info:      metal_uio_dev_open: No IRQ for device 11010000.cpg-uio.
[XXX] Successfully probed IPI device
metal: info:      metal_uio_dev_open: No IRQ for device 42f00000.rsctbl.
[XXX] Successfully open uio device: 42f00000.rsctbl.
[XXX] Successfully added memory device 42f00000.rsctbl.
metal: info:      metal_uio_dev_open: No IRQ for device 43000000.vring-ctl0.
[XXX] Successfully open uio device: 43000000.vring-ctl0.
[XXX] Successfully added memory device 43000000.vring-ctl0.
metal: info:      metal_uio_dev_open: No IRQ for device 43200000.vring-shm0.
[XXX] Successfully open uio device: 43200000.vring-shm0.
[XXX] Successfully added memory device 43200000.vring-shm0.
metal: info:      metal_uio_dev_open: No IRQ for device 43100000.vring-ctl1.
[XXX] Successfully open uio device: 43100000.vring-ctl1.
[XXX] Successfully added memory device 43100000.vring-ctl1.
metal: info:      metal_uio_dev_open: No IRQ for device 43500000.vring-shm1.
[XXX] Successfully open uio device: 43500000.vring-shm1.
[XXX] Successfully added memory device 43500000.vring-shm1.
metal: info:      metal_uio_dev_open: No IRQ for device 42f01000.mhu-shm.
[XXX] Successfully open uio device: 42f01000.mhu-shm.
[XXX] Successfully added memory device 42f01000.mhu-shm.
[XXX] Initialize remoteproc successfully.
[XXX] proc_id:1 rsc_id:1 mbx_id:0
[XXX] Initialize remoteproc successfully.
[XXX] proc_id:0 rsc_id:0 mbx_id:1
[XXX] Initialize remoteproc successfully.
[XXX] proc_id:1 rsc_id:1 mbx_id:1
[XXX] Initialize remoteproc successfully.

**************************************
*   rpmsg communication sample program   *
**************************************

1. communicate with CM33 ch0
2. communicate with CM33 ch1
3. communicate with CM33_FPU ch0
4. communicate with CM33_FPU ch1
5. communicate with CM33 ch0 and CM33_FPU ch1

e. exit

please input
>
```

7. Input the number which performs the communication between CM33 and CA55 you would like to try on the console.

4.5 Overview of Sample Program's behavior

This section describes the overview of sample program's behavior.

1. When the CA55 sample program is successfully executed, the communication channel among CA55 and CM33 is established.
2. The CA55 sample program starts to send the message to CM33 with incrementing the message size from the minimum value 17 to the maximum value 488. During the communication, the message as shown below is displayed on your console:

```
[xxx] Sending payload number 148 of size 165
```

3. When CM33 sample program receives the message sent from CA55, the echo reply is sent back to CA55 sample program.
4. When CA55 receives the echo reply, the message below should be displayed on your console:

```
[xxx] received payload number 148 of size 165
```

5. After the 488-byte sized payload is sent from CA55 to CM33, CM33 sends back the echo reply, the message indicating the termination of the communication channel is sent from CA55 to CM33. Then, the CA55 sample program outputs the following log messages to your console:

- **Termination message on CA55**

```
**********************************
Test Results: Error count = 0
**********************************
Quitting application .. Echo test end
Stopping application...
```

- **Termination message on CM33**

```
De-initializating remoteproc
```

Then, CM33 side re-waits for the establishment of connection channel. You can see the following log on the console a short time later:

```
creating remoteproc virtio
initializing rpmsg vdev
```

5. CM33 cold boot support

This chapter describes how CA55 and CM33 related stuff should be deployed for CM33 cold boot.

Using RZ MPU VLP v4.0.1

5.1 RZ/G VLP Setup

This section described how to integrate CM33 cold boot related stuff to RZ MPU VLP v4.0.1.

1. Follow the procedure from the beginning of **2.2 Building Images** to **(3) Add layers** of **SMARC EVK of RZ/G3S Linux Start-up Guide**.

2. Download Multi-OS Feature Package (r01an8260ej0420-rz-multi-os-pkg.zip) to your working directory and run the commands stated below:

```
$ cd ~/rzg_vlp_<pkg ver>
$ unzip <Multi-OS download dir>/r01an8260ej0420-rz-multi-os-pkg.zip
$ tar zxvf r01an8260ej0420-rz-multi-os-pkg/meta-rz-features_multi-os_v4.2.0.tar.gz
```

3. If you build **RZ MPU VLP v4.0.1**, please uncomment the following line to use cm33 coldboot support:

```
#MACHINE_FEATURES:append = " RZ_REMOTEPROC"
#MACHINE_FEATURES:append = " RZ_CM33_FIRMWARE_LOAD"
MACHINE_FEATURES:append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES:append = " RZ_AWO_SUPPORT"
```

   If you build **RZ/G VLP v3.0.7**, please uncomment the following line to use cm33 coldboot support:

```
#MACHINE_FEATURES_append = " RZ_REMOTEPROC"
#MACHINE_FEATURES_append = " RZ_CM33_FIRMWARE_LOAD"
MACHINE_FEATURES_append = " RZ_CM33_COLDBOOT"
#MACHINE_FEATURES_append = " RZ_AWO_SUPPORT"
```

4. Rebuild Trusted Firmware-A as shown below.

```
MACHINE=smarc-rzg3s bitbake trusted-firmware-a -c cleansstate
MACHINE=smarc-rzg3s bitbake firmware-pack -c cleansstate
MACHINE=smarc-rzg3s bitbake core-image-minimal
```

5.2 Deployment of CA55 Build Artifacts to SMARC EVK

If CM33 is configured as Boot CPU, you need to follow the procedure stated below for deploying bootloader files, Linux kernel image, device tree and rootfs:

1. Follow **3. Preparing the SD Card**, **4.1 Preparation of Hardware and Software**, **4.2 Startup Procedure** and **4.3 Download Flash Writer to RAM** of **SMARC EVK of RZ/G3S Linux Start-up Guide** to Invoke Flash Writer.

(Optional for Flash Writer settings)

2. Change the transfer rate of Flash Writer from the default one (115200bps) to the high speed one (921600bps) as shown below:

```
> SUP
Scif speed UP
Please change to 921.6Kbps baud rate setting of the terminal.
```

3. Program **bl2_no_bp_spi-smarc-rzg3s.srec** with Flash Writer as shown below:

```
> XLS2
===== Qspi writing of RZ/G3 Board Command =============
Load Program to Spiflash
Writes to any of SPI address.
Program size & Qspi Save Address
===== Please Input Program Top Address ============
Please Input : H'a1e00
===== Please Input Qspi Save Address ===
Please Input : H'200000
Please send ! ('.' & CR stop load)
```

Send **bl2_no_bp_spi-smarc-rzg3s.srec** via terminal software (e.g., TeraTerm) after the message **Please send ! ('.' & CR stop load)** is output on your console as shown above.

When **bl2_no_bp_spi-smarc-rzg3s.srec** is programmed successfully, the following message is shown on your console:

```
Erase SPI Flash memory...
Erase Completed
Write to SPI Flash memory.
======= Qspi Save Information =================
SpiFlashMemory Stat Address : H'00200000
SpiFlashMemory End Address  : H'0021EB58
================================================

SPI Data Clear(H'FF) Check : H'00000000-0000FFFF,Clear OK?(y/n)
```

Finally, type **y** to continue

4. Program **fip-smarc-rzg3s.srec** with Flash Writer as shown below:

- **For RZ MPU VLP v4.0.1:**

```
> XLS2
===== Qspi writing of RZ/G3 Board Command =============
Load Program to Spiflash
Writes to any of SPI address.
Program size & Qspi Save Address
===== Please Input Program Top Address ============
Please Input : H'0
===== Please Input Qspi Save Address ===
Please Input : H'260000
Please send ! ('.' & CR stop load)
```

- **For RZ/G VLP v3.0.7-update5:**

```
> XLS2
===== Qspi writing of RZ/G3 Board Command =============
Load Program to Spiflash
Writes to any of SPI address.
Program size & Qspi Save Address
===== Please Input Program Top Address ============
Please Input : H'0
===== Please Input Qspi Save Address ===
Please Input : H'264000
Please send ! ('.' & CR stop load)
```

Send **fip-smarc-rzg3s.srec** via terminal software (e.g., TeraTerm) after the message **Please send ! ('.' & CR stop load)** is output on your console as shown above.

When **fip-smarc-rzg3s.srec** is programmed successfully, the following message is shown on your console:

```
Erase SPI Flash memory...
Erase Completed
Write to SPI Flash memory.
======= Qspi Save Information =================
SpiFlashMemory Stat Address : H'00264000
SpiFlashMemory End Address  : H'00342B6E
================================================

SPI Data Clear(H'FF) Check : H'00000000-0000FFFF,Clear OK?(y/n)
```

Finally, type **y** to continue.

5.3 Setup and deployment of CM33 related stuff

Refer to **4.2 CM33 Sample Program Setup** and follow the procedure including the item "**(Optional for CM33 cold boot support)**".

Note: Do not follow the procedure "**(Optional for remoteproc support)**" at this time.

6. Assignment of peripherals

Table 6.1 shows the expected assignment of peripherals to sub core (CM33) on this example project while all other devices remain assigned to the Linux (main core) side.

Table 6.1 Peripherals assignment on RZ/G3S Multi-OS Example Project

Peripherals	CPU		Remarks
	Main core CA55	Sub core CM33	
GTM	ch0, ch3-7	ch1-2	0001-Set-SCIF1-and-OSTM1-OSTM2-as-critical-clock.patch 0002-Disabled-OSTM1-OSTM2-for-the-use-on-the-target-core.patch
SCIF	ch0, ch2-4	ch1	0001-Set-SCIF1-and-OSTM1-OSTM2-as-critical-clock.patch 0002-Disabled-SCIF1-for-the-use-on-the-target-core.patch

7. Appendix

7.1 Setting for RPMsg Example supporting OpenAMP v2024.05

For support OpenAMP v2024.05, the RPMsg examples require additional preprocessor to build and run successfully with the updated virtio/OpenAMP configuration as below:

`VIRTIO_DRIVER_SUPPORT=1` for the Master core.
`VIRTIO_DEVICE_SUPPORT=1` for the Slave core.

Configuration path:
e² studio (GCC): *Properties → Cross ARM C Compiler → Preprocessor → Defined symbols*

8. Reference Documents

- R01AN5924 RZ/G2L RZ/G2LC RZ/G2UL RZ/G3S RZ/G3E Getting Started with Flexible Software Package
- R01UH1076 RZ/G3S SMARC Module Board User's Manual: Hardware
- R01UH1077 RZ SMARC Series Carrier Board II User's Manual: Hardware

Revision History

Rev.	Date	Description	
		Page	Summary
3.00	Jul.22.2025	-	First edition.
3.10	Dec.26.2025	-	Updated to align with RZ MPU Verified Linux Package v4.0.1.
3.20	Feb.06.2026	-	Update Multi-OS package version to 3.2.0.
4.00	Mar.31.2026	-	Update Multi-OS package version to 4.0.0.
4.10	Jun.30.2026	-	Update Multi-OS package version to 4.1.0.
4.20	Sep.03.2026	-	Update Multi-OS Package version to 4.2.0.

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.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation 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.
2. 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.
6. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. 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/.