

# Linux Interface Specification Power Management

User's Manual: Software

RZ/G2L Group, RZ/V2L Group, RZ/G3E Group and  
RZ/Five

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (<http://www.renesas.com>).

## 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. 1.11 Jul 2025)

## Corporate Headquarters

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

[www.renesas.com](http://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/](http://www.renesas.com/contact/).

## 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.

# How to Use This Manual

## 1. Purpose and Target Readers

This manual is designed to provide the user with an understanding of the hardware functions and electrical characteristics of the MPU. It is intended for users designing application systems incorporating the MPU.. It is intended for users developing software incorporating the processors. A basic knowledge of software development and Linux systems is necessary in order to use this document.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

The following documents apply to the RZ/G2L Group, RZ/V2L Group, RZ/G3E Group and RZ/Five Group. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document Title	Document No.
User's manual for Hardware	Hardware specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description  Note: Refer to the application notes for details on using peripheral functions.	RZ/G2L Group User's Manual: Hardware	---
		RZ/V2L Group User's Manual: Hardware	---
		RZ/Five Group User's Manual: Hardware	---
		RZ/G3E Group User's Manual: Hardware	---
User's manual for Software	Description of Power Management Linux interface Specification	Linux interface Specification Power Management	This user's manual
Application Note	Information on using peripheral functions and application examples Sample programs Information on writing programs in assembly language and C	Available from Renesas Electronics Web site.	
Renesas Technical Update	Product specifications, updates on documents, etc.		

2. Notation of Numbers and Symbols

3. Register Notation

#### 4. List of Abbreviations and Acronyms

Abbreviation	Full Form
DU	Display Unit on RZ/G Series, 2nd Generation
VSPD	Video signal processing for Display
DSI	Display Serial Interface
FBDev	Framebuffer Device
DRM	Direct Rendering Manager
KMS	Kernel Mode Setting
DRI	Direct Rendering Infrastructure
FB	Framebuffer
LIF	LCDC Interface (VSP-DU connect mode)
RPF	Read Pixel Formatter
WPF	Write Pixel Formatter
BRS	Blend ROP Sub-Unit
CRTC	Cathode Ray Tube Controller
CVT	Coordinated Video Timings
GTF	General Timing Formula

# Table of Contents

1. Overview.....	1
1.1 Overview .....	1
1.2 Function .....	1
1.3 Reference.....	3
1.3.1 Standard.....	3
1.3.2 Related documents .....	3
1.4 Restrictions .....	3
1.5 Notice .....	3
2. Terminology.....	4
3. Operating Environment.....	5
3.1 Hardware Environment .....	5
3.2 Software configuration .....	6
4. Function.....	8
4.1 CPU Hotplug .....	8
4.1.1 CPU Offline .....	8
4.1.2 CPU Online .....	9
4.2 CPU Freq .....	10
4.2.1 DFS .....	10
4.2.2 Frequency .....	12
4.3 Runtime PM .....	13
4.4 Thermal Management.....	14
4.5 Module Standby Mode .....	16
4.6 Suspend to Idle .....	17
4.7 Suspend to Ram .....	18
5. External Interface.....	20
5.1 CPU Hotplug .....	21
5.1.1 CPU Hotplug definition .....	21
5.1.2 CPU Hotplug operation .....	21
5.2 CPU Freq .....	22
5.2.1 CPU Freq definition .....	22
5.2.2 CPU Freq operation .....	23
5.3 Runtime PM .....	24
5.3.1 Runtime PM definition .....	24
5.3.2 Runtime PM operation .....	25
5.4 System Shutdown .....	27
5.5 Module Standby Mode .....	28
5.6 Suspend to Idle .....	29
5.7 Suspend to Ram .....	29
6. Integration.....	30

# 1. Overview

## 1.1 Overview

This manual explains the power management for RZ/G2L Group, RZ/V2L Group and RZ/Five.

Note:

Currently, this device is supported in two kernel versions v5.10 and v6.1 with the information below:

- v5.10: RZ/G2L Group, RZ/V2L group and RZ/Five.
- v6.1: RZ/G2L, RZ/G2LC and RZ/G3E.

## 1.2 Function

Power management supports the following functions:

**Table 1-1 Power management functions**

Function name	Overview of software specification
CPU Hotplug	CPU Hotplug is a function that user controls online and offline for CPU.
CPU Freq	CPU Freq is to control frequency for CPU.
Runtime PM	Runtime PM is a function to manage the clock of each device, and to control on or off for them according to the usage status of each device.
Module Standby Mode	The Module Standby Mode is a mode that requests the clock stop of the module specified by the master. The purpose of this mode is to reduce power consumption by stopping unnecessary functions.
Suspend to Idle	A generic, pure software, light-weight variant of system suspend. It allows more energy to be saved relative to runtime idle by freezing user space, suspending the timekeeping and putting all I/O devices into low-power states.
Suspend to Ram	System suspends its operation and saves its state into RAM, then powers down most hardware components to conserve energy. When a user action or a wake-up event occurs, the system is "woken up" and resumes operation from its previous state
System Shutdown	System shutdown will turn off system when it's predicting excess than limit temperature.



Depending on each platform, the support of power management functions is different. Below table shows more detail about support of power management functions on RZ/G2L Group, RZ/V2L Group, RZ/Five and RZ/G3E Group platform:

**Table 1-2 Detailed support of power management functions**

Platform Functions	RZ/G2L	RZ/G2LC	RZ/G2UL	RZ/V2L Group	RZ/Five	RZ/G3E
CPU Hotplug	yes	yes	no	yes	no	yes
CPU Freq	yes	yes	yes	yes	yes	yes
Runtime PM	yes	yes	yes	yes	yes	yes
System shutdown	yes	yes	yes	yes	yes	yes
Module Standby Mode*	yes	yes	yes	yes	no	no
Suspend to Idle	yes	yes	yes	yes	no	yes
Suspend to Ram	no	no	no	no	no	yes

Note (\*): Module Standby Mode is only supported for Linux kernel v5.10

## 1.3 Reference

### 1.3.1 Standard

There is no reference document on standards.

### 1.3.2 Related documents

**Table 1-3 Related documents**

Number	Issue	Title	Edition	Data
-	-	-	-	-

## 1.4 Restrictions

There is no restriction.

## 1.5 Notice

There is no notice.

## 2. Terminology

The following table shows the terminology related to this module.

**Table 2-1 Terminology**

<b>Terms</b>	<b>Explanation</b>
DFS	<b>D</b> ynamic <b>F</b> requency <b>S</b> caling
PSCI	<b>P</b> ower <b>S</b> tate <b>C</b> oordinate <b>I</b> nterface
CA55	<b>C</b> ortex <b>A55</b> for AP-System Core
WFI	<b>W</b> ait <b>F</b> or <b>I</b> nterrupt
SMC	<b>S</b> ecure <b>M</b> onitor <b>C</b> all
CPG	<b>C</b> lock <b>P</b> ulse <b>G</b> enerator
SYSC	<b>S</b> ystem <b>C</b> ontroller

## 3. Operating Environment

### 3.1 Hardware Environment

The following table lists the hardware needed to use this module.

**Table 3-1 Hardware specification (RZ/G2L Group, RZ/V2L Group and RZ/Five)**

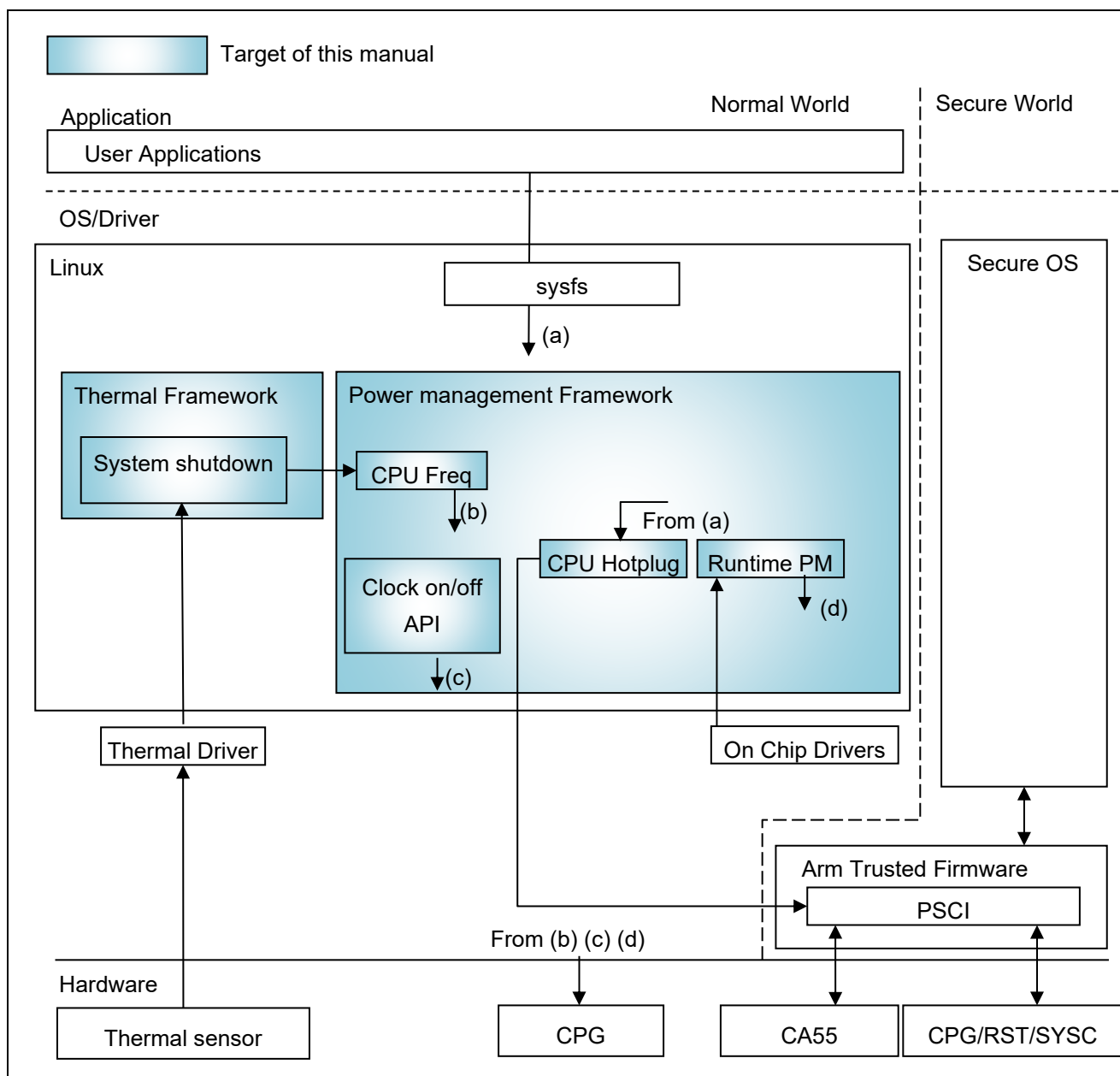
Name	Version	Manufacturer
RZ/G2L Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/G2LC Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/G2UL Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/V2L Evaluation Board Kits	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/Five Evaluation Board Kits	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH

**Table 3-2 Hardware Environment (RZ/G3E)**

Name	Product number
RZ/G3E SMARC Evaluation Board Kit	RTK9947E57S01000BE

## 3.2 Software configuration

The following figure shows the software configuration of this module.



**Figure 3–1 Block diagram of power management**

PSCI defines an Arm standard interface. It supports communication between Normal world and Secure world. Then, it allows Arm Trusted Firmware to arbitrate power management requests from secure software and Linux.

The following table shows the PSCI function that power management uses.

**Table 3-3 PSCI Function**

Function	Description
CPU_OFF	Power down the calling core. This call is intended for use in hotplug. The core that is powered down by CPU_OFF can only be powered up again in response by CPU_ON.
CPU_ON	Power up a core. This call is intended for dynamic addition of cores, for use in secondary boot, hotplug.

## 4. Function

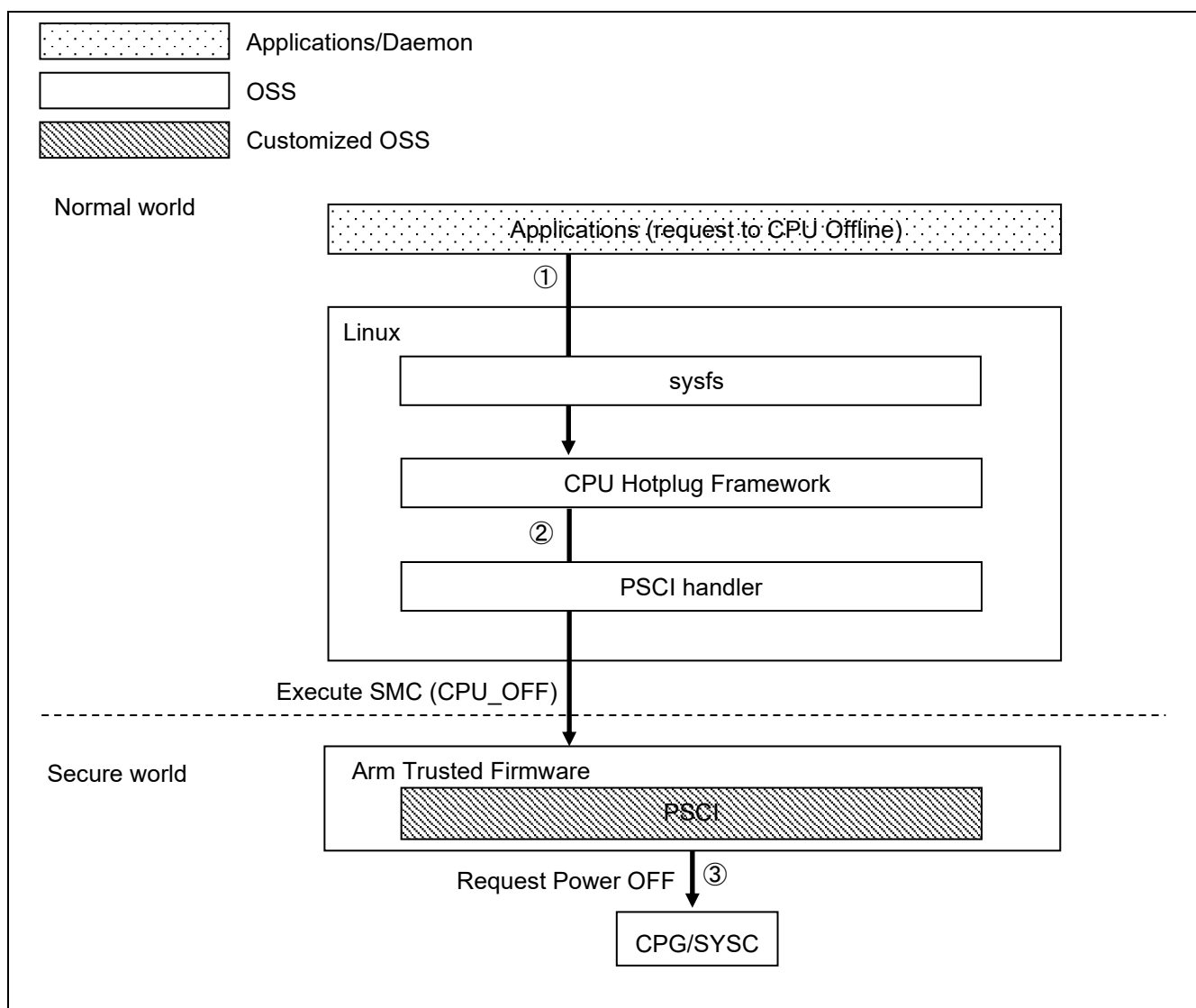
### 4.1 CPU Hotplug

CPU Hotplug is Linux function to perform online and offline on multi-core system:

- CPU Online: plug each CPU into the system
- CPU Offline: unplug each CPU from the system

#### 4.1.1 CPU Offline

The following figure shows the processing flow of CPU Offline.

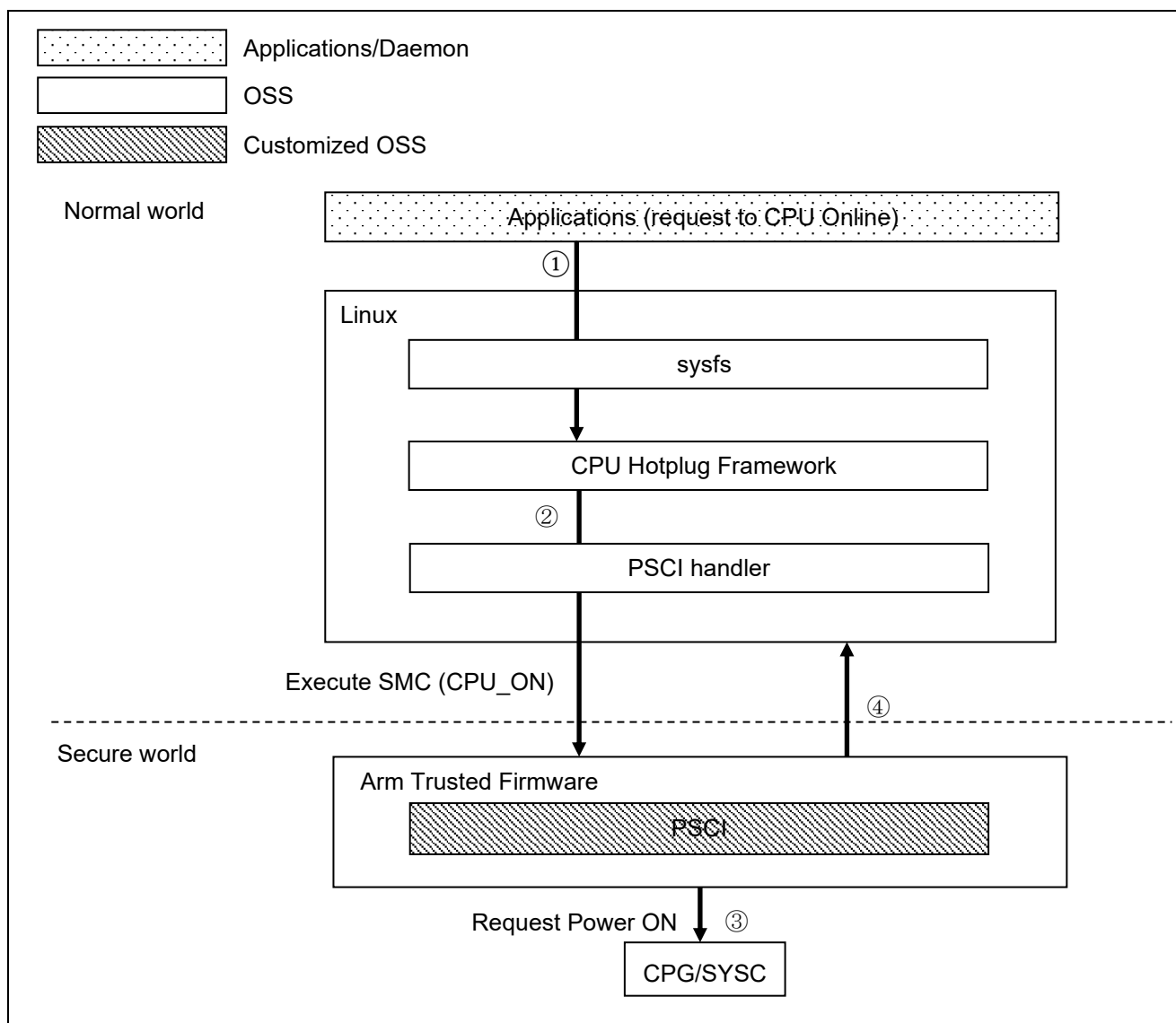


**Figure 4-1 Processing flow of CPU Offline**

- ① Applications request CPU Offline via sysfs.  
echo 0 > /sys/devices/system/cpu/cpu1/online
- ② CPU Hotplug Framework requests to take CPU down to PSCI handler. Then, the PSCI handler issues CPU\_OFF request to Arm Trusted Firmware via SMC instruction.
- ③ CPU which requested CPU\_OFF is turned off in secure world.

### 4.1.2 CPU Online

The following figure shows the processing flow of CPU Online.



**Figure 4-2 Processing flow of CPU Online**

- ① Applications to work on Master CPU request CPU Online via sysfs.  
echo 1 > /sys/devices/system/cpu/cpu1/online
- ② CPU Hotplug Framework requests to take CPU up to PSCI handler. Then, the PSCI handler issues CPU\_ON request to Arm Trusted Firmware via SMC instruction.
- ③ Slave CPU is turned on in secure world.
- ④ Slave CPU jumps to entry point and returns Linux.



## 4.2 CPU Freq

CPU Freq (DFS) is Linux function to dynamically change frequency of CPU.

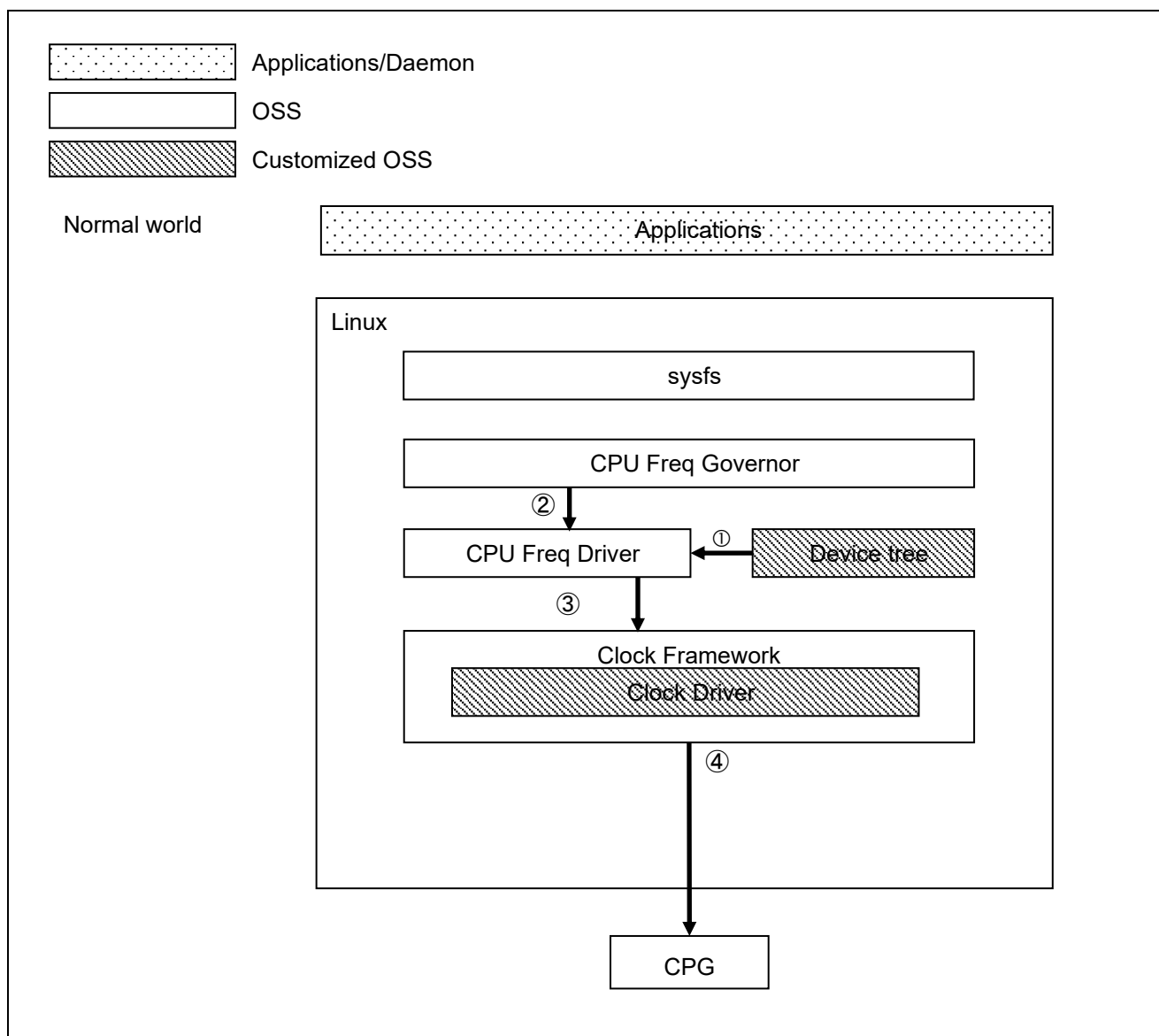
### 4.2.1 DFS

CPU Freq has 6 operating modes which are called CPU Freq Governor. The voltage and frequency of CPU are changed according to category of CPU Freq Governor.

The following table shows the CPU Freq Governor operating modes current BSP.

**Table 4-1 CPU Freq Governor operation mode**

Governor	Description
Performance	The frequency is set max. This is default operation mode.
Powersave	The frequency is set min.
Ondemand	If CPU load is bigger than 95%, the frequency is set max. If CPU load is equal to or less than 95%, the frequency is set based on CPU load.
Conservative	If CPU load is bigger than 80%, the frequency is set one level higher than current frequency. If CPU load is equal to or less than 20%, the frequency is set one level lower than current frequency.
Userspace	It sets frequency which is defined by user.
Schedutil	Schedutil governor is driven by scheduler. It uses scheduler-provided CPU utilization information as input for making its decisions.



**Figure 4-3 Processing flow of DFS by CPU Freq**

- ① When Linux boots up (initialize), CPU Freq Driver gets setting values of frequency which defined in device tree. Also, it sends them to CPU Freq Governor.
- ② CPU Freq Governor chooses the setting value of frequency of CPU according to Governor operating mode (\*1). Also, it notifies the requested value of voltage and frequency to CPU Freq Driver.
 

(\*1) Governor requests the setting value of frequency in accordance with the highest CPU load in CPU cluster.
- ③ CPU Freq Driver notifies the requested value of frequency to Clock Framework, notifies the requested voltage to Regulator Framework.
- ④ Clock Driver changes the requested frequency by CPG control.

## 4.2.2 Frequency

**Table 4-2 CA55 frequency table of RZ/G2L, RZ/G2LC and RZ/V2L**

Operating mode	CA55 Freq	Voltage
Normal mode	1.2 GHz	Fixed value
	600 MHz	
	300 MHz	
	150 MHz	

**Table 4-3 CA55 frequency table of RZ/G2UL**

Operating mode	CA55 Freq	Voltage
Normal mode	1.0 GHz	Fixed value
	500 MHz	
	250 MHz	
	125 MHz	

**Table 4-4 AX45MP frequency table of RZ/Five**

Operating mode	AX45MP Freq	Voltage
Normal mode	1.0 GHz	Fixed value
	500 MHz	
	250 MHz	
	125 MHz	

**Table 4-5 CA55 frequency table of RZ/G3E**

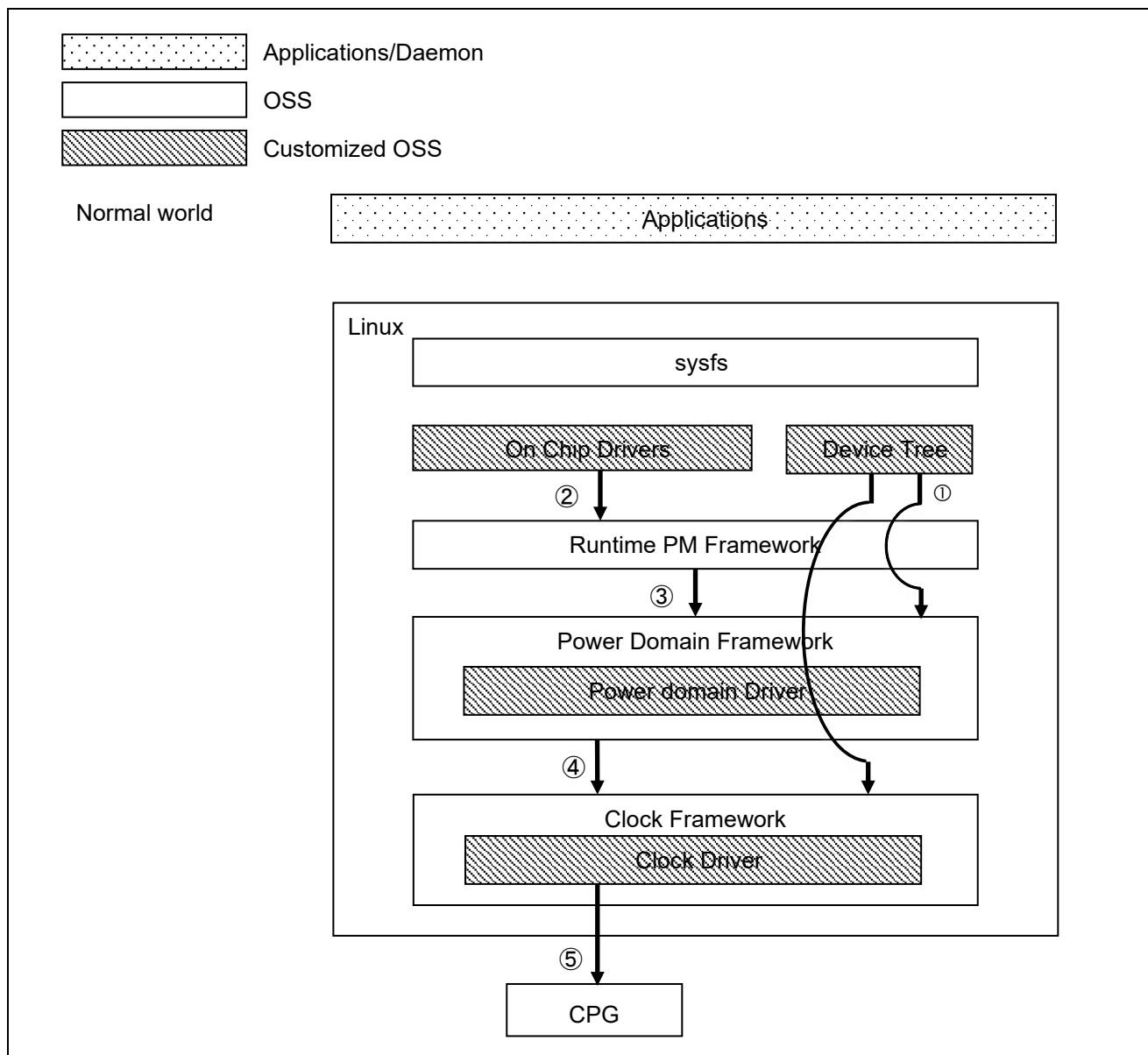
Operating mode	CA55 Freq	Voltage
Normal mode	1.7 GHz	Fixed value
	850 MHz	
	425 MHz	
	212,5 MHz	

### 4.3 Runtime PM

Runtime PM is Linux function to control clock supply and power domain for IP modules. Also, it is controlled in consideration of the configuration of the power domain.

Currently, we just control clock supply in RZ/G2L Group, RZ/V2L, RZ/G3E and RZ/Five.

The following figure shows the processing flow of Runtime PM.



**Figure 4-4 Processing flow of Runtime PM**

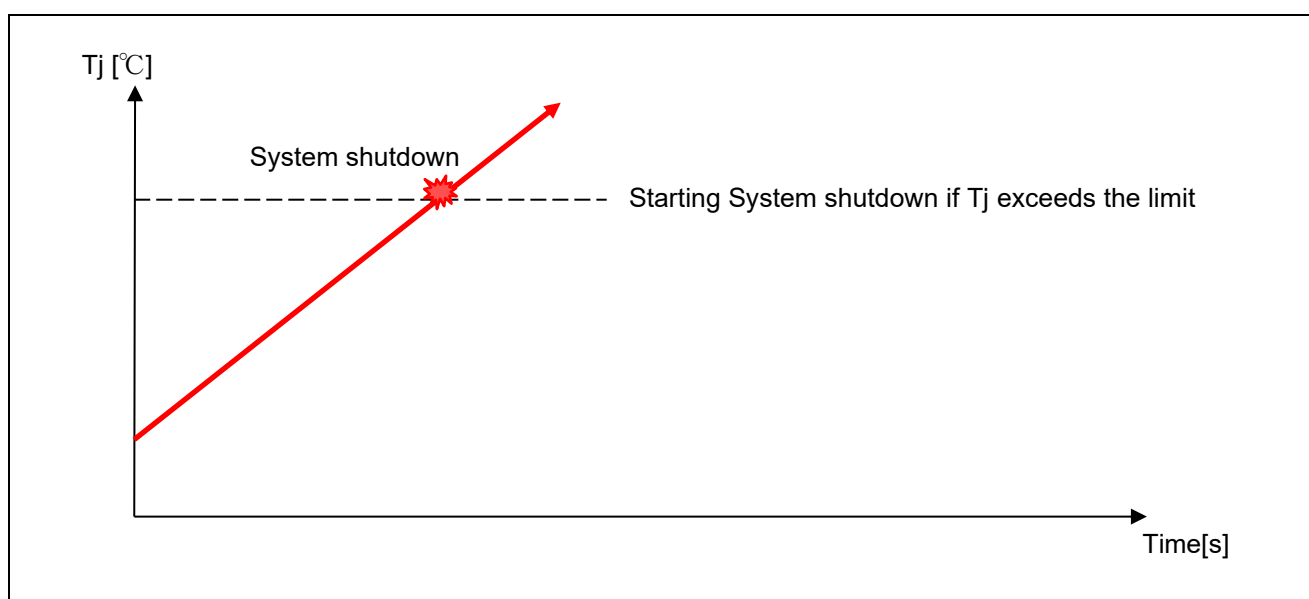
- ① When Linux boots up (initialize), Clock Framework gets settings of clock which defined in device tree. Also, Power Domain Framework gets settings of power domain which are defined in device tree.
- ② On Chip Drivers notify status of module which controls its own to Runtime PM Framework via Runtime PM API.
- ③ Runtime PM Framework notifies status of module to Power Domain Framework.
- ④ Power domain Framework requests the clock supply or stop of the module to the Clock Framework.
- ⑤ Clock Framework enables or stops the clock of module by CPG control via Clock Driver.

## 4.4 Thermal Management

Thermal management is to control SoC temperature using power management function. Thermal management has the following three functions.

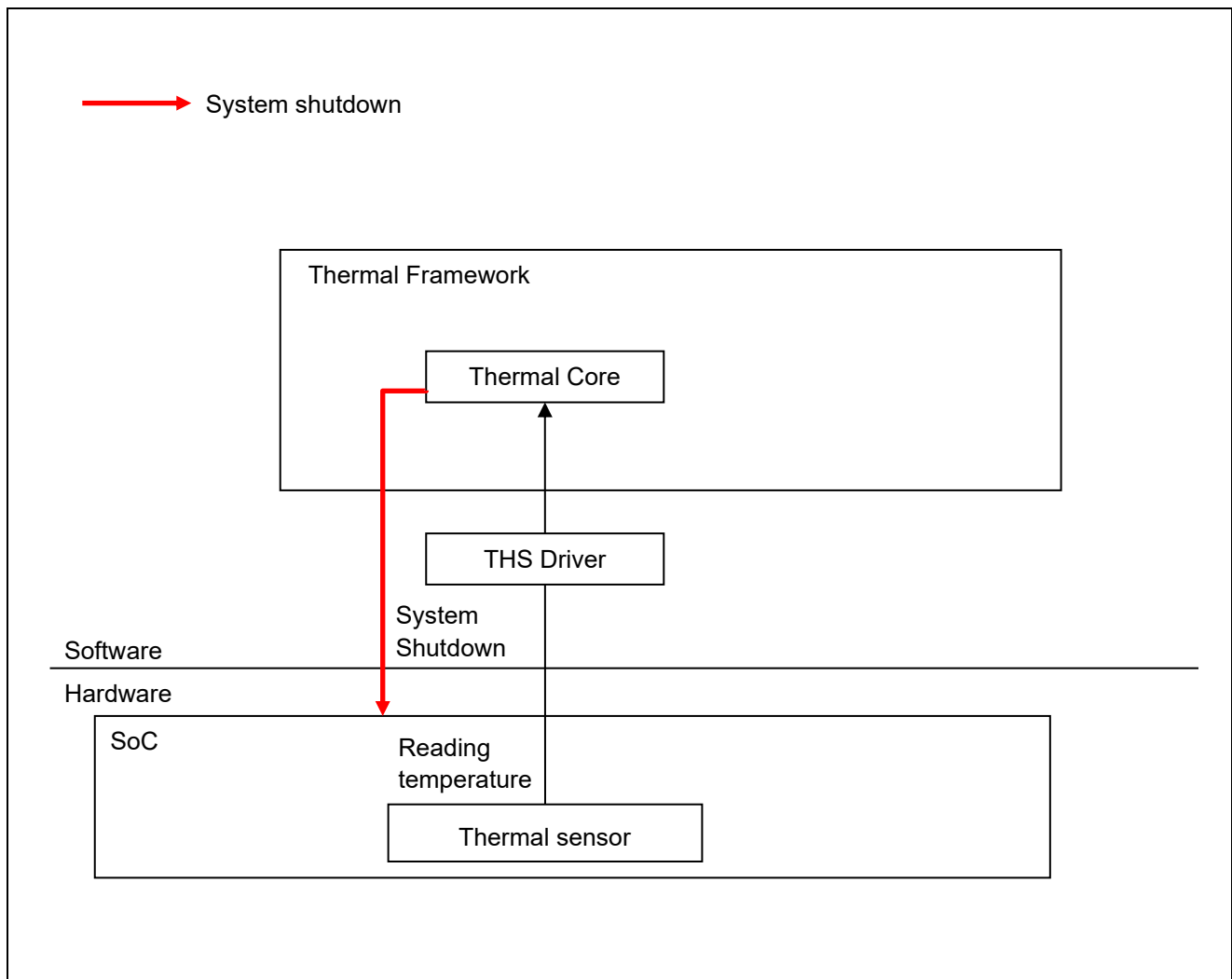
**Table 4-6 Thermal management functions**

Function	Description
System shutdown	System shutdown is a function to turn off system before SoC temperature reaches a limit.



**Figure 4-5 Summary of thermal management starting**

The following figure shows the software configuration of thermal management system.



**Figure 4–6 Block diagram of thermal management**

**Processing flow of System shutdown:**

- ① THS Driver reads  $T_j$  from Thermal Sensor. Then, they are input to Thermal Core.
- ② If  $T_j$  from Thermal Sensor is over 125 degree Celsius, Thermal Core shuts down the system.

## 4.5 Module Standby Mode

The Module Standby Mode is a mode that requests the clock stop of the module specified by the master.

This mode is implemented by stopping the clock supply by the CPG register and switching the MSTOP signal of the bus through CPG register settings.

**Table 4-7 Module Standby Mode related registers**

Register name	Abbreviation
Clock Control Register	CPG_CLKON_***
MSTOP Register	CPG_BUS_***_MSTOP CPG_MHU_STOP
GPU Lowpower Sequence Control Register (*)	SYS_LP_GPU_CTL

(\*) Currently, GPU Lowpower Sequence Control is not support in our BSP environment.

The Module Standby Mode can be applied to the modules by:

- When entering the Module Standby Mode:
  - Set the MSTOP Register bit of the target module to 1.
  - Set the Clock Control Register bit of the target module to 0.
- When exiting the Module Standby Mode:
  - Set the MSTOP Register bit of the target module to 0.
  - Set the Clock Control Register bit of the target module to 1.

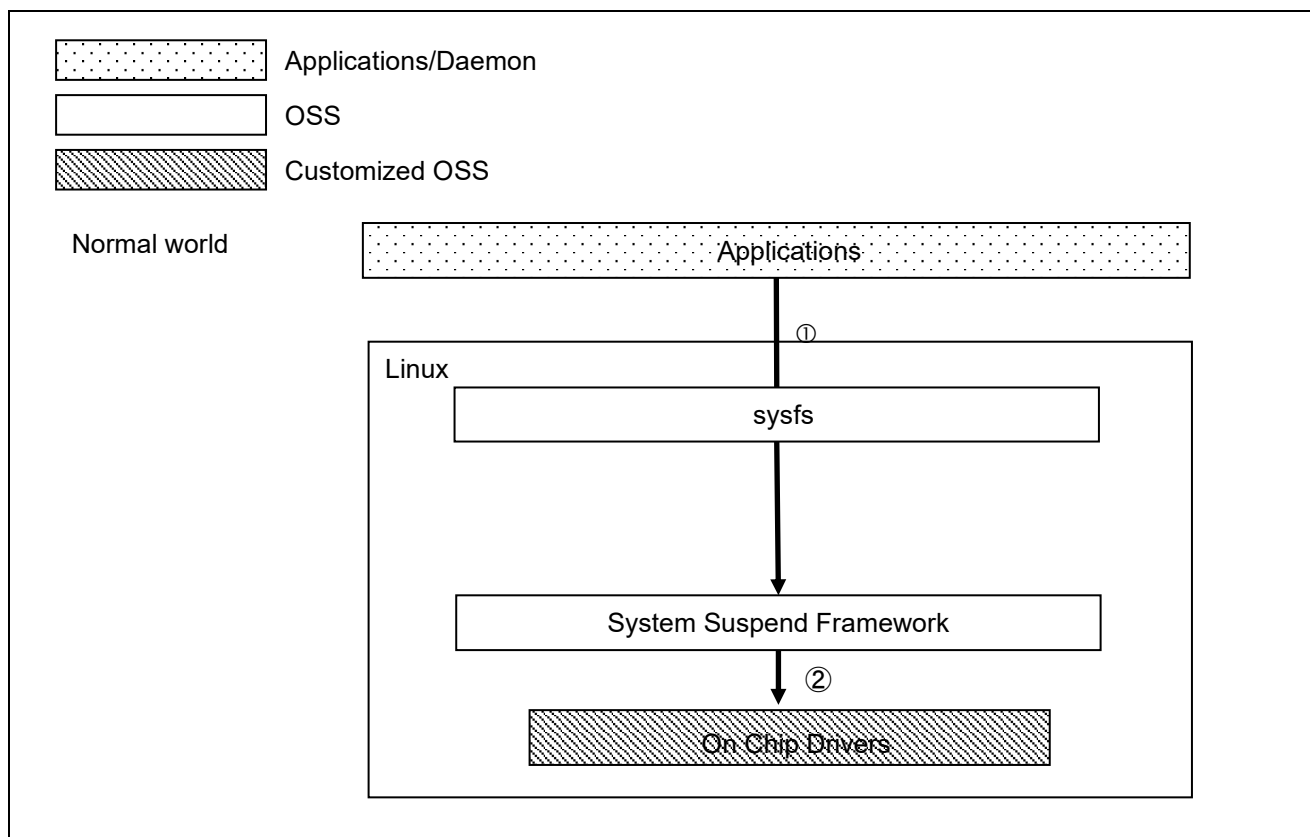
About support target module for Module Standby Mode, they are described in Hardware manual with chapter “**Low Power Mode**” and subchapter “**Setting of Clock control and MSTOP register**”.

## 4.6 Suspend to Idle

System suspend to idle is a sleep state that is supported in Linux framework.

It allows more energy to be saved relative to runtime idle by freezing user space, suspending the timekeeping and putting all I/O devices into low-power states.

The following figure shows the processing flow of Suspend to Idle.



**Figure 4–7 Processing flow of Suspend to Idle**

- ① Application requests System Suspend to Idle via sysfs by choosing one of the following runtime commands:

```
$ echo freeze > /sys/power/state
```

or

```
$ echo s2idle > /sys/power/mem_sleep  
$ echo mem > /sys/power/state
```

- ② System Suspend Framework notifies device driver (On Chip Drivers) of moving to System Suspend via suspend/resume API Linux Power Management Framework.



## 4.7 Suspend to Ram

The Suspend to RAM mode is one of the power management modes in a Linux framework, where the system suspends its operation and saves its state into RAM, then powers down most hardware components to conserve energy. When a user action or a wake-up event occurs, the system is "woken up" and resumes operation from its previous state.

The following step will shows the processing flow of Suspend to Ram.

- ① Application requests System Suspend to Ram via sysfs by following runtime commands:

```
$ echo 0 > /sys/module/printk/parameters/console_suspend
$ echo mem > /sys/power/state
```

- ② System Suspend Framework notifies device driver (On Chip Drivers) of moving to System Suspend to Ram state via suspend/resume API Linux Power Management Framework.
- ③ Create a wake-up event to resume board  
Ex: press Power button to wake up board from Suspend to Ram state

Example: Suspend to Ram on RZ/G3E SMARC Evaluation Board Kit

Run command:

```
cat /sys/power/mem_sleep
```

If you can see [deep] then the board has supported Suspend to Ram.

Example on RZ/G3E SMARC Evaluation Board Kit:

```
root@smarc-rzg3e:~# cat /sys/power/mem_sleep
s2idle [deep]
```

Requests System Suspend to Ram via sysfs

```
root@smarc-rzg3e:~# echo 0 > /sys/module/printk/parameters/console_suspend
root@smarc-rzg3e:~# echo mem > /sys/power/state
[ 70.295787] PM: suspend entry (deep)
[ 70.312054] Filesystems sync: 0.016 seconds
[ 70.314475] Freezing user space processes
[ 70.316727] Freezing user space processes completed (elapsed 0.002 seconds)
[ 70.316737] OOM killer disabled.
[ 70.316740] Freezing remaining freezable tasks
[ 70.318366] Freezing remaining freezable tasks completed (elapsed 0.001 seconds)
[ 70.485749] dwc-eth-dwmac 15c30000.ethernet eth0: Link is Down
[ 70.511502] Disabling non-boot CPUs ...
[ 70.570527] psci: CPU1 killed (polled 0 ms)
[ 70.606267] psci: CPU2 killed (polled 0 ms)
[ 70.630156] psci: CPU3 killed (polled 0 ms)
```

Long press Power button (RED button) to wake up board from Suspend to Ram state

```
NOTICE: BL2: v2.10.5(release):2.10.5/rzg3e_1.2.0_rc1-dirty
NOTICE: BL2: Built : 09:57:57, May 16 2025
NOTICE: BL2: SYS_LSI_MODE: 0x3e06
NOTICE: BL2: SYS_LSI_DEVID: 0x8679447
NOTICE: BL2: SYS_LSI_PRR: 0x0
NOTICE: BL2: Booting BL31
[ 70.636127] Enabling non-boot CPUs ...
```

```
[ 70.636318] Detected VIPT I-cache on CPU1
[ 70.636378] GICv3: CPU1: found redistributor 100 region 0:0x0000000014960000
[ 70.636415] CPU1: Booted secondary processor 0x0000000100 [0x412fd050]
[ 70.639697] CPU1 is up
[ 70.639788] Detected VIPT I-cache on CPU2
[ 70.639816] GICv3: CPU2: found redistributor 200 region 0:0x0000000014980000
[ 70.639833] CPU2: Booted secondary processor 0x0000000200 [0x412fd050]
[ 70.640190] CPU2 is up
[ 70.640280] Detected VIPT I-cache on CPU3
[ 70.640308] GICv3: CPU3: found redistributor 300 region 0:0x00000000149a0000
[ 70.640325] CPU3: Booted secondary processor 0x0000000300 [0x412fd050]
[ 70.640760] CPU3 is up
[ 72.067132] rzv2h-pcie 13400000.pcie: resume PCIe link down
[ 72.067140] rzv2h-pcie 13400000.pcie: PM: dpm_run_callback():
genpd_resume_noirq+0x0/0xe0 returns -110
[ 72.067158] rzv2h-pcie 13400000.pcie: PM: failed to resume noirq: error -110
[ 72.168963] dwmac4: Master AXI performs any burst length
[ 72.169006] dwc-eth-dwmac 15c30000.ethernet eth0: No Safety Features support found
[ 72.169050] dwc-eth-dwmac 15c30000.ethernet eth0: IEEE 1588-2008 Advanced Timestamp
supported
[ 72.169185] dwc-eth-dwmac 15c30000.ethernet eth0: configuring for phy/rgmii-id link
mode
[ 72.244913] dwmac4: Master AXI performs any burst length
[ 72.244951] dwc-eth-dwmac 15c40000.ethernet end1: No Safety Features support found
[ 72.244991] dwc-eth-dwmac 15c40000.ethernet end1: IEEE 1588-2008 Advanced Timestamp
supported
[ 72.245116] dwc-eth-dwmac 15c40000.ethernet end1: configuring for phy/rgmii-id link
mode
[ 72.248784] usb usb2: root hub lost power or was reset
[ 72.248794] usb usb1: root hub lost power or was reset
[ 72.248815] usb usb4: root hub lost power or was reset
[ 72.337326] usb usb5: root hub lost power or was reset
[ 72.337502] usb usb6: root hub lost power or was reset
[ 72.425351] usb usb3: root hub lost power or was reset
[ 72.719503] usb 1-1: reset high-speed USB device number 2 using ehci-platform
root@smarc-rzg3e[ 72.987236] OOM killer enabled.
[ 72.987241] Restarting tasks ... done.
:~# [ 72.989879] random: crng reseeded on system resumption
[ 72.990596] PM: suspend exit
[ 75.035761] dwc-eth-dwmac 15c30000.ethernet eth0: Link is Up - 100Mbps/Full - flow
control off

root@smarc-rzg3e:~#
```

Note: Suspend to Ram is only support on RZ/G3E Smarc Evaluation Board Kit

## 5. External Interface

The following table shows system information of power management.

**Table 5-1 System information of power management**

Function	Function Information
CPU Hotplug	/sys/devices/system/cpuX/online
	/sys/devices/system/cpuX/offline
	/sys/devices/system/cpu/cpuX/online
CPU Freq	/sys/devices/system/cpu/cpuX/cpufreq/*
Suspend to Idle	/sys/power/state /sys/power/mem_sleep
Suspend to Ram	/sys/power/state /sys/power/mem_sleep

**Table 5-2 Linux document of power management**

Function	Document Information
CPU Hotplug	Documentation/cputopology.txt
	Documentation/cpu-hotplug.txt
CPU Idle	Documentation/cpuidle/sysfs.txt
CPU Freq	Documentation/cpu-freq/user-guide.txt
System Suspend to Idle	<a href="https://www.kernel.org/doc/html/v5.10/admin-guide/pm/sleep-states.html">https://www.kernel.org/doc/html/v5.10/admin-guide/pm/sleep-states.html</a>

## 5.1 CPU Hotplug

### 5.1.1 CPU Hotplug definition

CPU Hotplug doesn't need a special definition.

### 5.1.2 CPU Hotplug operation

CPU Hotplug can be operated via sysfs (/sys/devices/system/cpu/cpuY/online).

The following figure shows example of operation for CPU Hotplug.

```
/* cpu1 to offline */  
  
$ echo 0 > /sys/devices/system/cpu/cpu1/online  
  
/* cpu1 to online */  
  
$ echo 1 > /sys/devices/system/cpu/cpu1/online
```

**Figure 5–1 Example of operation for CPU Hotplug**

## 5.2 CPU Freq

### 5.2.1 CPU Freq definition

CPU Freq function can be used by defining operating-points in the device tree. The following figure shows example of definition for operating-points.

The following figure shows example of definition for CPU Freq of RZ/G2L (r9a07g044.dtsi):

```
cluster0_opp: opp-table-0 {
    compatible = "operating-points-v2";
    opp-shared;

    opp-1500000000 {
        opp-hz = /bits/ 64 <1500000000>;
        opp-microvolt = <1100000>;
        clock-latency-ns = <300000>;
    };
    opp-3000000000 {
        opp-hz = /bits/ 64 <3000000000>;
        opp-microvolt = <1100000>;
        clock-latency-ns = <300000>;
    };
    opp-6000000000 {
        opp-hz = /bits/ 64 <6000000000>;
        opp-microvolt = <1100000>;
        clock-latency-ns = <300000>;
    };
    opp-12000000000 {
        opp-hz = /bits/ 64 <12000000000>;
        opp-microvolt = <1100000>;
        clock-latency-ns = <300000>;
        opp-suspend;
    };
};

cpus {
    #address-cells = <1>;
    #size-cells = <0>;

    ...

    cpu0: cpu@0 {
        compatible = "arm,cortex-a55";
        reg = <0>;
        device_type = "cpu";
        #cooling-cells = <2>;
        next-level-cache = <&L3_CA55>;
        enable-method = "psci";
        clocks = <&cpg CPG_CORE R9A07G044_CLK_I>;
        operating-points-v2 = <&cluster0_opp>;
    };

    cpu1: cpu@100 {
        compatible = "arm,cortex-a55";
        reg = <0x100>;
        device_type = "cpu";
        next-level-cache = <&L3_CA55>;
        enable-method = "psci";
        clocks = <&cpg CPG_CORE R9A07G044_CLK_I>;
        operating-points-v2 = <&cluster0_opp>;
    };

    ...
};
```

Figure 5–2 Example of RZ/G2L CPUFreq definition on device tree for operating-points

### 5.2.2 CPU Freq operation

The confirmation of parameters and operation for CPU Freq can be done via sysfs (/sys/devices/system/cpu/cpuX/cpufreq/\*).

The following figure shows example of operation for CPU Freq.

```
/* Checking Governor which is currently used */  
  
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor  
  
/* Switching to Ondemand Governor */  
  
$ echo ondemand > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor  
  
/* Checking current frequency of CPU */  
  
$ cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq  
  
/* Checking the available frequency */  
  
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies  
  
/* Change frequency of CPU to new frequency */  
  
$ echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor  
  
$ echo new_frequency > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed  
  
/* with new_frequency is available_frequencies */
```

**Figure 5–3 Example of operation for CPU Freq**

Note: The CPU Freq configuration is shared between the same kind of CPUs in current BSP. Therefore, if the frequency in CPU0 is changed, the frequency of other CPUs on same cluster (CA55) are also changed as same frequency.

## 5.3 Runtime PM

### 5.3.1 Runtime PM definition

Runtime PM APIs can be used by defining the following items in device node of consumer driver.

The purpose of the definition is for power consumer device to probe directly with compatible driver and to bind to clock and power domain providers that it uses.

**Table 5-3 Definition of clock and power domain in device node**

Definition	Explanation
<code>compatible = "&lt;vendor&gt;, &lt;platform-module name&gt;";</code>	This property ensures for device can be probed directly with compatible driver.
<code>clocks = &lt;&amp;cpg CPG_MOD clock_id&gt;;</code>	<p>The 'clocks' property contains module stop clock that device uses for its operation.</p> <p><i>Note: 'clock_id' is identified number of dedicated module clock.</i></p> <p><i>They are defined in header files:</i></p> <p><i>include/dt-bindings/clock/r9a07g044-cpg.h (for RZ/G2L and RZ/G2LC)</i></p> <p><i>include/dt-bindings/clock/r9a07g043-cpg.h (for RZ/G2UL and RZ/Five)</i></p> <p><i>include/dt-bindings/clock/r9a07g054-cpg.h (for RZ/V2L)</i></p>
<code>power-domains = &lt;&amp;cpg&gt;;</code>	<p>This property contains the pm runtime handler for this device.</p> <p><i>Note: 'clock_id' is identified number of module clock.</i></p>

The following figure shows example of clock and power domain control properties of a device node in the device tree of RZ/G2L.

```

scif0: serial@1004b800 {
    ...
    compatible = "renesas,scif-r9a07g044";
    clocks = <&cpg CPG_MOD R9A07G044_SCIF0_CLK_PCK>;
    clock-names = "fck";
    power-domains = <&cpg>;
    ...
};

```

**Figure 5-4 Device node with clock and power domain example**

### 5.3.2 Runtime PM operation

#### Using Runtime PM APIs in device driver code

After device is successfully probed with driver, the Runtime PM state of the device is disabled. The clock and power domain of the device are also turned off. The following basic steps show how to add Runtime APIs to the driver source code:

##### **Step 1:** Enable Runtime PM for the device

Before using other Runtime PM APIs, it must be ensured that the device has been enabled.

The `pm_runtime_enable()` is usually called in *probe()* or *init()* function of the driver.

##### **Step 2:** Resume the device

Before using hardware (e.g. initialize setting, transfer data ...), the device must be resumed.

The `pm_runtime_get_sync()/pm_runtime_get()` (in synchronous/asynchronous context respectively) is usually called before hardware is set, to turn on the power domain and module clock of the device.

##### **Step 3:** Suspend the device

After the device has been completed operating, the device should be suspended to save power.

The `pm_runtime_put_sync()/pm_runtime_put()` (in synchronous/asynchronous context respectively) is usually called after the device completes its operation to turn off module clock and power domain to save power.

##### **Step 4:** Disable the device before removing it from system

Before removing the device from system, it must be disabled to prevent subsystem-level Runtime PM callbacks from being run for the device.

The `pm_runtime_disable()` must be called before device removal or error occurring case (symmetric with `pm_runtime_enable()`)



The following figure shows basic example of using Runtime PM APIs, the foo driver will resume its device each time start transferring data and suspend device when transferring complete.

```
static int foo_probe(struct platform_device *pdev)
{ ...
    pm_runtime_enable(dev);
    ...
}

static int start_transfer (struct foo_dev *dev, int *buf, int length)
{ ...
    pm_runtime_get_sync(dev);
    //init channel and set data to transfer to register
    ...
}

static int end_transfer(struct foo_dev *dev)
{ ...
    //finish transfer, setting register before stopping channel
    pm_runtime_put_sync(dev);
    ...
}

static int foo_remove(struct platform_device *pdev)
{ ...
    pm_runtime_disable(dev);
}
```

**Figure 5–5 Runtime PM APIs basic usage flow.**

## 5.4 System Shutdown

System shutdown can be used by defining Thermal sensor in the device tree. The following example shows definition of IPA in RZ/G2L (r9a07g044.dtsi)

```
thermal-zones {
    emergency {
        polling-delay = <1000>;
        on-temperature = <110000>;
        off-temperature = <95000>;
        target_cpus = <&cpu0>;
        status = "disabled";
    };

    cpu-thermal {
        polling-delay-passive = <250>;
        polling-delay = <1000>;
        thermal-sensors = <&tsu 0>;
        sustainable-power = <717>;

        cooling-maps {
            map0 {
                trip = <&target>;
                cooling-device = <&cpu0 0 2>;
                contribution = <1024>;
            };
        };

        trips {
            sensor_crit: sensor-crit {
                temperature = <125000>;
                hysteresis = <1000>;
                type = "critical";
            };

            target: trip-point {
                temperature = <100000>;
                hysteresis = <1000>;
                type = "passive";
            };
        };
    };
};
```

**Figure 5–6 Example of definition on device tree of Thermal Sensor for System Shutdown**

### temperature = 125000:

This value indicates the temperature to execute the System shutdown.

It's set lower than limit temperature of Board.

After thermal sensor is successfully initialized and registered to thermal core, the core will continuously read temperature from the sensor.

The system will be shut down completely if System Shutdown trip point is reached.

## 5.5 Module Standby Mode

Module standby mode is described in clock driver of each platform via MSTOP macros.

- drivers/clk/renesas/rzg2l-cpg.h

```
/* MSTOP related registers offset */
#define CPG_BUS_ACPU_MSTOP      (0xB60)
#define CPG_BUS_MCPU1_MSTOP    (0xB64)
#define CPG_BUS_MCPU2_MSTOP    (0xB68)
#define CPG_BUS_PERI_COM_MSTOP (0xB6C)
#define CPG_BUS_PERI_CPU_MSTOP (0xB70)
#define CPG_BUS_PERI_DDR_MSTOP (0xB74)
#define CPG_BUS_REG0_MSTOP     (0xB7C)
#define CPG_BUS_REG1_MSTOP     (0xB80)
#define CPG_BUS_TZCDDR_MSTOP   (0xB84)
#define CPG_MHU_MSTOP          (0xB88)
#define CPG_BUS_MCPU3_MSTOP    (0xB90)
#define CPG_BUS_PERI_CPU2_MSTOP (0xB94)

#define DEF_PD(_name, _id, _mstop_conf, _flags)
{
    .name = (_name),
    .id = (_id),
    .conf = {
        .mstop = (_mstop_conf),
    },
    .flags = (_flags),
}
```

- Example of RZ/G2L power-domains definition (arch/arm64/boot/dts/renesas/r9a07g044.dtsi)

```
cpg: clock-controller@11010000 {
    compatible = "renesas,r9a07g044-cpg";
    reg = <0 0x11010000 0 0x10000>;

    clocks = <&extal_clk>;
    clock-names = "extal";
    #clock-cells = <2>;
    #reset-cells = <1>;
    #power-domain-cells = <1>;
};

dmac: dma-controller@11820000 {
    compatible = "renesas,r9a07g044-dmac", "renesas,rz-dmac";
    .....
    power-domains = <&cpg R9A07G044_PD_DMACH>;
    .....
};
```

Example for RZ/G2L clock driver definition (r9a07g044-cpg.c):

```

DEF_PD("always-on",          R9A07G044_PD_ALWAYS_ON,
      DEF_REG_CONF(0, 0),
      GENPD_FLAG_ALWAYS_ON),

DEF_PD("gic",                R9A07G044_PD_GIC,
      DEF_REG_CONF(CPG_BUS_REG1_MSTOP, BIT(7)),
      GENPD_FLAG_ALWAYS_ON),

DEF_PD("ia55",               R9A07G044_PD_IA55,
      DEF_REG_CONF(CPG_BUS_PERI_CPU_MSTOP, BIT(13)),
      GENPD_FLAG_ALWAYS_ON),

DEF_PD("dmac",               R9A07G044_PD_DMACH,
      DEF_REG_CONF(CPG_BUS_REG1_MSTOP,
      GENMASK(3, 0)),
      GENPD_FLAG_ALWAYS_ON),

```

When clock has MSTOP definition is turned off, this module will be transit to Module Standby State.

## 5.6 Suspend to Idle

Suspend to Idle is always supported if the below kernel configuration option is set.

```

/* Config to use suspend/resume PM API */
CONFIG_PM_SLEEP=y

/* Config to support Suspend to Idle */
CONFIG_SUSPEND=y

```

## 5.7 Suspend to Ram

Suspend to Ram is always supported on RZ/G3E SMARC Evaluation Board Kit and can check on user by run command:

```
cat /sys/power/mem_sleep
```

If you can see [deep] then the board has supported Suspend to Ram.

Example on RZ/G3E SMARC Evaluation Board Kit:

```

root@smarc-rzg3e:~# cat /sys/power/mem_sleep
s2idle [deep]

```

## 6. Integration

The power management directory configuration is shown below.

```
/* Devicetree definition for RZ/G2L Group RZ/V2L and RZ/G3E */
arch/arm64/boot/dts/renesas/

/* Devicetree definition for RZ/Five */
arch/riscv/boot/dts/renesas/

/* Thermal sensor driver source */
drivers/thermal/rzg2l_thermal.c

/* Clock definition headers */
include/dt-bindings/clock/
    └─ r9a07g043-cpg.h
    └─ r9a07g044-cpg.h
    └─ r9a07g054-cpg.h
    └─ renesas,r9a09g047-cpg.h
```

Revision History	Linux Interface Specification Power Management User's Manual: Software
------------------	---

Rev.	Date	Description	
		Page	Summary
1.7	Dec. 15, 2022	—	First Edition issued and keep revision to keep consistent with other documents
1.8	Mar. 31, 2025	21, 24-26	Correct definition of clock and power domain in device node, add new handle MSTOP via CPG power domains.
1.9	May 30, 2025	1	Add MPU information support for both kernel versions v5.10 and v6.1. Add note for Module Standby Mode.
1.10	Jun 30, 2025	—	Update RZ/Five information
1.11	Jul. 22, 2025	—	Update RZ/G3E information and Suspend to Ram function

---

Linux Interface Specification Power Management  
User's Manual: Software

Publication Date: Rev. 1.11 Jul. 22, 2025

Published by: Renesas Electronics Corporation

---

RZ/G2L Group, RZ/V2L Group, RZ/G3E Group  
and RZ/Five



Renesas Electronics Corporation