R-Car Series, 3rd Generation

Power Management

Introduction

[Purpose]

Devices applied in automobiles require high-speed booting in response to the key being turned to start the engine and countermeasures for the generation of heat. We consider that the functions related to power control supported since the introduction of R-Car Series, 3rd Generation products effectively fulfill these two requirements. This document describes how to use the functions of these products that are related to power control and gives concrete examples of their application.

[Target Readers]

Readers of this document are assumed to have general knowledge in the fields and specific technologies listed below.

* Engineering, logic circuits, microcontrollers, and Linux.
* The functionality of the multiple processor cores of R-Car H3, R-Car M3-W, R-Car M3-W+, R-Car M3-N, and R-Car E3 products.
* The electrical specifications of the multiple processor cores of R-Car H3, R-Car M3-W, R-Car M3-W+, R-Car M3-N, and R-Car E3 products.
* The functions of the BSP drivers for R-Car H3, R-Car M3-W, R-Car M3-W+, R-Car M3-N, and R-Car E3 products.

[Note]

The separate specification with the filename “RENESAS\_RCH3M3M3NE3\_PowerManagement\_UME” gives further details on the functions related to power control, including an outline of operation, user interfaces, and parameters, so this document does not cover those points.

Statements in relation to operating systems in this document apply to Yocto v3.7.0 and later versions from Renesas.

Target Device

・R-Car H3

・R-Car M3-W

・R-Car M3-W+

・R-Car M3-N

・R-Car E3

Refer to RENESAS\_RCH3M3M3NE3\_PowerManagement\_UME for the correspondence of the power control related functions of each target device.

Contents

[1. Specifications of Software for Functions Related to Power Control 3](#_Toc536612045)

[2. Relationships between the Requirements and Functions Related to Power Control 4](#_Toc536612046)

[3. High-speed Booting in Response to the Key being Turned to Start the Engine 5](#_Toc536612047)

[3.1 Overview of the Requirements 5](#_Toc536612048)

[3.1.1 Background 5](#_Toc536612049)

[3.1.2 Realizing High-speed Booting 6](#_Toc536612050)

[3.2 Overview of the “System Suspend to RAM” Function 7](#_Toc536612051)

[3.3 Flow of Software for Using the “System Suspend to RAM” Function 8](#_Toc536612052)

[3.4 Control example for “System Suspend to RAM” 10](#_Toc536612053)

[4. Suppressing the Generation of Heat in On-board Terminals 11](#_Toc536612054)

[4.1 Overview of the Requirements 11](#_Toc536612055)

[4.1.1 Background 11](#_Toc536612056)

[4.1.2 Suppressing the Generation of Heat 12](#_Toc536612057)

[4.2 Overview of Thermal Management 13](#_Toc536612058)

[4.3 Examples of Parameters to be Set 14](#_Toc536612059)

[5. Setting example for power management functions 17](#_Toc536612060)

[5.1 Typical mode setting example 18](#_Toc536612061)

[5.2 High performance mode setting example 18](#_Toc536612062)

[5.3 Low power mode setting example 18](#_Toc536612063)

[Appendix 19](#_Toc536612064)

[A1. Overview 19](#_Toc536612065)

[A2. Times to Bring CPUs Online and Offline 20](#_Toc536612066)

[A3. Notes on Using Dynamic Voltage and Frequency Scaling (DVFS) or Dynamic Frequency Scaling (DFS) 21](#_Toc536612067)

[A3.1 Flow of Processing for DFS Control and Processing Times 22](#_Toc536612068)

[A3.2 Flow of Processing for DFVS Control and Processing Times 23](#_Toc536612069)

[A4. Resume times for System Suspend to RAM 24](#_Toc536612070)

# Specifications of Software for Functions Related to Power Control

Prior to a description of how to use the functions related to power control, this section describes software specifications for each function in outline.

Table 1‑1 Outline of Software for Functions Related to Power Control

|  |  |
| --- | --- |
| Function | Software Specification in Outline |
| CPU hotplug | The “CPU hotplug” function allows users to bring CPUs online and offline while a system is running. An offline CPU is not under the control of the kernel and its power supply is shut down. |
| CPU idle | The “CPU idle” function allows the kernel to dynamically have a CPU enter the sleep mode or core standby mode. The mode depends on the time until the next timer interrupt. If the time until the next timer interrupt is relatively long, the kernel has the CPU enter the core standby mode, which can save more power supply. |
| CPU freq | The “CPU freq” function changes the operating frequency and voltage of the CPU. |
| Runtime PM | The “runtime power management (runtime PM)” function controls the clock and power domains of the various devices as a whole, and turns clock or power domains on and off for the individual devices according to their usage. |
| System Suspend to RAM | The “System Suspend to RAM” function suspends the system while retaining the current state of operations. On release from suspension, the system is restored to its prior state. In addition, with the exception of the backup power supply, suspending the supply of power to all systems to which power is output from the power management IC (PMIC) can serve as a measure for reducing dark current. |
| Intelligent power allocation (IPA) | The IPA function monitors the junction temperature of the SoC, and applies the “CPU freq” function to adjust its performance to prevent the temperature of the SoC exceeding the maximum permissible range for tolerance of heat. |
| Emergency shutdown (EMS) | The EMS function monitors the junction temperature of the SoC, and applies the “CPU hotplug” function to shut its power supply down to prevent the temperature of the SoC exceeding the maximum permissible range for tolerance of heat. |
| Capacity Aware migration Strategy  (CAS) | The CAS performs the task placement that emphasizes performance in an environment where Cortex-A57 and Cortex-A53 operate simultaneously. |

# Relationships between the Requirements and Functions Related to Power Control

This section describes the relationships between the two key requirements (high-speed booting in response to the key being turned to start the engine and countermeasures for the generation of heat) and the functions related to power control. Sections 3 and 4 give details on fulfilling the requirements.

Functions related

to power control

**Power state coordination interface (PSCI)**

**CPU core frequency control**

SoC-dependent functions

**Suppressing the generation of heat in   
on-board terminals**

**High-speed booting in response to the key being turned to start   
the engine**

**Secure Layer**

**Kernel Layer**

**CPU freq**

**PMIC driver**

**Thermal control driver**

**CPU**

**Hotplug**

**Runtime**

**PM**

**CPU idle**

**System Suspend to RAM**

**Clock framework**

**CPU core power switch**

**IPA**

**Power domain control**

**EMS**

**Requirements**

Figure 2‑1 Relationships between the Requirements and Functions Related to Power Control

# High-speed Booting in Response to the Key being Turned to Start the Engine

## Overview of the Requirements

### Background

Current requirements for on-board information terminals in vehicles exceed the previous model of car navigation and audio systems. They are now required to provide various other items, such as a full range of entertainment and information communications. Demands for services that are stress-free in terms of greater efficiency, convenience, and comfort have also increased. For example, the navigation system and various button operations being enabled immediately after the key is turned to start the engine gives users a greater sense of comfort. In particular, users as a matter of course now require that the playback of music or video can soon be resumed from the state prior to suspension due to the key previously having been turned to stop the engine. For these reasons, high-speed booting after the key is turned to start the engine is considered to be essential.

### Realizing High-speed Booting

To realize high-speed booting after the key is turned to start the engine, the times for processing that takes relatively long when a system is booted up (transfer of data to DRAM, initialization of drivers and applications) need to be shortened. The “System Suspend to RAM” function is a way of shortening such processing times. This retains the state of operation of the system prior to the suspension at the time the key is turned to stop the engine and restores it when the key is turned to start the engine again, thus skipping time-consuming processing. Note that using the “System Suspend to RAM” function requires that the power supply to the DRAM be turned on since this is where the state prior to suspension is to be stored. If the vehicle’s battery goes flat while supplying power to the DRAM, restoring the system becomes impossible. Therefore, using this also requires a design in which power is only supplied to the minimum of systems to suppress current drawn from the battery (reducing dark current) during times when “System Suspend to RAM” is in use.

Renesas provides both the “System Suspend to RAM” function and countermeasures for dark current as standard items of support.

## Overview of the “System Suspend to RAM” Function

Renesas supports the “System Suspend to RAM” function, in which system operation is suspended while its state at that time is retained. On release from suspension, the system is restored to its prior state. Suspending the supply of power to all systems from the PMIC with the exception of the memory backup supply serves as a further measure for reducing dark current. Figure 3-1 is a schematic view of the state transitions before and after dark current is reduced. The flow of software for using the “System Suspend to RAM” function is depicted in section 3.3.

VD33, VD18, etc.

VD33, VD18, etc.

DVFS, VDD

DVFS, VDD

I2C\_SCL

I2C\_SDA

I2C\_SCL

I2C\_SDA

BKUP\_TRG

BKUP\_TRG

BKUP\_CTRL

BKUP\_CTRL

PRESETB

PRESETB

DDR0/

DDR1

DDR0/

DDR1

DDR0C/DDR1C

DDR0C/DDR1C

RSTB

RSTB

Off

On

Separate Devices

SW23

PMIC

DRAM

DDR IO interface

R-Car H3

Separate Devices

SW23

PMIC

DRAM

DDR IO interface

R-Car H3

Normal operation

System Suspend to RAM

Power is off.

Power is on.

Figure 3‑1 State of Power Supply to Systems when the “System Suspend to RAM” Function is in Use

## Flow of Software for Using the “System Suspend to RAM” Function

The flow of software for using the “System Suspend to RAM” function is given in figure 3-2.

ARM Trusted Firmware

ARM Trusted Firmware

App.

Drivers & kernel

Drivers & kernel

APP.

* + **Store playback point by App.**
  + **Stop data transfer and close Drivers by App..**
  + **Re-start playback point by App.**
  + **Initialize, re-detect devices and open Drivers.**

**Detect Suspend trigger**

Detect Resume trigger

**Call sysfs-IF\***

Framework start to work

All processes wakeup

→ Power Manager start.

Suspend

Resume

Suspend processing

- Backup registers by the ARM Trusted Firmware

Resume processing:

- Restore registers by the ARM Trusted Firmware.

All process freezing

→ power Manager stop.

Suspend processing：

- If needs, backup register by driver

- If needs, stop clock for device

Resume processing:

- If needs, restore register by driver

- If needs, supply clock for device

- If needs, initialize driver

Note: \* Before issuing a request for “System Suspend to RAM”, make the calls required to place the system in a state from which it can be suspended.

Figure 3‑2 Flow of Software for Using the “System Suspend to RAM” Function

In the “System Suspend to RAM” function, data are saved and devices suspended at the time of suspension, and data are restored and devices restarted at the time of resumption. This achieves suspension of the system with the state of operations retained, followed by resumption of operations in the state prior to suspension. This can be used to skip time-consuming processing (transfer of data to DRAM, initialization of drivers and applications) in a full system boot-up, so that users can handle on-board facilities after a quick resumption of system operations.

## Control example for “System Suspend to RAM”

In the case of use-case in which the running application continues to operate when "System Suspend to RAM" is used,

It is necessary to suspend the running application at Suspend and resume the application at Resume.

As an example, we will introduce a use-case in which "System Suspend to RAM" is executed during video playback operation.

As shown in Figure 3, the video playback application uses Gstreamer to perform video playback. Also, the system manager manages each application and executes "System Suspend to RAM".

At the time of Suspend, it keeps the state (playback position) currently being played. And then it resumes from the playing state which was held at Resume, so that video playback will continue to operate continuously.

(Refer to "RENESAS\_RCH3M3M3NE3\_GStreamer\_UsersManual\_UME" for details on video playback)

User

Linux

HW

Video playback Application

System Manager

GStreamer

Driver

Video playback

Video playback

Stop

During playback

Restart

Image update

Suspend

Image update

During playback

\*1

\*2

\*3

\*4

\*5

\*6

Display ON

Display ON

\*7

Figure 3‑3 “System Suspend to RAM” use-case in video playback

Table 3‑1 The detailed behavior for video playback

|  |  |  |
| --- | --- | --- |
| Notes | Overview | Details |
| \*1 | Video playback start request | Requests video playback from the video playback application to GStreamer. |
| \*2 | Application suspension request | Request the stop from the system manager to the video playback application.  (Hold playback position with video playback application) |
| \*3 | Video playback end request | Requests playback stop from the video playback application to GStreamer. |
| \*4 | System Suspend to RAM execution | Execute "System Suspend to RAM" and shift to the suspend state. |
| \*5 | Resume request | Return from Suspend. |
| \*6 | Application restart request | Request the restart from the system manager to the video playback application. |
| \*7 | Video playback restart request | Specify the playback position from the video playback application to GStreamer and request video playback. |

# Suppressing the Generation of Heat in On-board Terminals

## Overview of the Requirements

### Background

Current requirements for on-board information terminals in vehicles exceed the previous model of car navigation and audio systems. They are now required to provide various other items, such as a full range of entertainment and information communications. On the other hand, the increases in power consumption that accompany improved performance lead to an enormous generation of heat that may readily exceed the heat-tolerance temperature ratings of the SoCs and DRAM chips used in on-board information terminals. Countermeasures for heat and other design measures to cope with this have thus become more difficult. Also, since generated heat strongly affects the life cycles and failure rates of semiconductor products, countermeasures for heat, such as fans and radiators, are currently very expensive to vendors of finished products. This makes software-based measures to suppress the generation of heat very desirable in terms of reducing the contributions of fans, radiators, and so on to bills of materials.

### Suppressing the Generation of Heat

Reducing power consumption through software has been employed as a general method for suppressing the generation of heat. The kernel layer supports functions (thermal management) that automatically reduce power consumption in response to excessive rises in temperature of R-Car Series, 3rd Generation and later versions of SoC chips in this family. This subsection gives an overview of thermal management with examples.

## Overview of Thermal Management

* Thermal management can be used to control power consumption to suppress the generation of heat in high-temperature environments.  
  For more details, see the separate document with the filename “RENESAS\_RCH3M3M3NE3\_PowerManagement\_UME”.

Table 4‑1 Overview of Software for Thermal Management

|  |  |
| --- | --- |
| Function | Software Specification in Outline |
| IPA | The IPA function monitors the junction temperature of the SoC, and applies the “CPU freqˮ function to adjust its performance to prevent the temperature of the SoC exceeding the maximum permissible range for tolerance of heat. |
| EMS | The EMS function monitors the junction temperature of the SoC, and applies the “CPU hotplug” function to shut its power supply down to prevent the temperature of the SoC exceeding the maximum permissible range for tolerance of heat. |

* Power consumption is reduced by lowering the frequency of the operating clock, lowering the voltage, or cutting off the supply of power to some of the Cortex-A57 cores. These measures may greatly lower the performance of a product. Specifically, it may lead to poor response by running applications, lowering the sense of comfort for users. Accordingly, thermal management is just a backup plan to protect a product against the generation of heat beyond its heat-tolerance temperature rating. If you wish to give higher priority to the maintenance of the sense of comfort for users, we recommend that you apply countermeasures for reducing power consumption such as suspending unnecessary applications before the environment you are using approaches temperatures high enough to require further measures for thermal management.
* Using thermal management requires setting the parameters for use in control. The parameters depend on the environment. For details on how to set the parameters, see section 4.3.

## Examples of Parameters to be Set

* Parameters for thermal management need to be set to suit the environment in which you apply it.
* Table 4-2 lists the parameters Renesas sets for the Salvator-X system evaluation board and the concepts used to determine them.

Table 4‑2 Examples of Parameters to be Set

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Parameter | Concept | Setting by Renesas |
| <1> | Threshold temperature for starting and stopping IPA | Set a temperature lower than <2> and higher than Tj when your environment is in the idle state (low load). Setting a temperature lower than Tj in the idle state causes constant operation of the IPA function. | 90°C |
| <2> | Target temperature for IPA | Set a temperature lower than <3>. In setting the temperature, take into account the maximum performance that will be obtainable in the use case you envisage. | 100°C |
| <3> | Threshold temperature for starting EMS | Set a temperature lower than <5>. Note that this temperature will depend on the countermeasures for heat dissipation in your environment. Specifically, the junction temperature of the SoC may fall to <4> immediately after the EMS function has been started in some cases (when the number of online CPUs is reduced from 4 to 1). In such cases, the junction temperature repeatedly swings between <3> and <4> according to the state of the load. Therefore, take the effect of the countermeasures for heat dissipation in your environment into sufficient consideration when setting <3> and <4>. | 110°C |
| <4> | Threshold temperature for stopping EMS | Set a temperature lower than <3>. Note that this temperature will depend on the countermeasures for heat dissipation in your environment. Specifically, the junction temperature of the SoC may rise to <3> immediately after the EMS function has been stopped in some cases (when the number of online CPUs is increased from 1 to 4). In such cases, the junction temperature repeatedly swings between <3> and <4> according to the state of the load. Therefore, take the effect of the countermeasures for heat dissipation in your environment into sufficient consideration when setting <3> and <4>. | 95°C |
| <5> | Threshold temperature for shutting the system down | Set a temperature in accord with the requirement for the maximum heat-tolerance temperature of the SoC. | 120°C |

* Setting the parameters without considering the countermeasures for heat dissipation in your environment leads to EMS repeatedly starting and stopping, as shown in figure 4-1.

|  | Temperature Tj (°C) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  | EMS starts. |  |  |  | EMS starts. |  |  |  | EMS starts. |  |  |
|  | Threshold temperature  for starting EMS |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Threshold temperature  for stopping EMS |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  | **EMS stops.** |  |  | EMS stops. |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Number of cores online | 4 | | | | | | 1 | | 4 | | 1 | | 4 | | 1 | |  |
|  | EMS control | — | | | | | | EMS starts. | | EMS stops. | | EMS starts. | | EMS stops. | | EMS starts. | |  |
|  |  |  | | | |  | |  | |  |  |  |  |  |  |  |  |  |

Figure 4‑1 Temperature Transitions when Parameters are Inappropriate

* Figure 4-2 is a schematic view of temperature control by using the parameters specified by Renesas on the Salvator-X system evaluation board with the SoC under heavy and medium loads.

|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Temperature Tj (°C) | |  |  |  |  | *IPA operating period* |  |  |  |  | EMS operating period |  |  |  | *IPA operating period* |  |  |  |
|  | Threshold temperature  for shutting the system  down: 120°C | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Threshold temperature  for starting EMS: 110°C | |  |  |  |  |  | EMS starts. |  |  |  |  | Load is lowered (due to external factors). |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  | Tj reaches the threshold temperatures. |  |  |  |  |  |  |
|  | Target temperature  for IPA: 100°C | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  | Tj reaches the threshold  temperatures. |  |  |  |  |  |  |  |  |  |  |  |
|  | Threshold temperature  for stopping EMS: 95°C | |  |  |  |  |  |  |  |  |  |  |  | EMS stops. |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  | IPA starts. |  |  |  |  |  |  |  |  |  |  |  |  |
| Threshold temperature  for starting and  stopping IPA: 90°C |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Load | | | Heavy load | | Heavy load | | Heavy load | | | | Medium load | | Medium load | | | |  |  |
|  | Control of the generation of heat | IPA control | | Disabled | | Enabled | | Disabled | | | | Disabled | | Enabled | | | |  |  |
|  | EMS control | | Disabled | | Disabled | | Enabled | | | | Enabled | | Disabled | | | |  |  |
|  | CPU (CA57) | Number of cores online | | 4 | | *4* | | 1 | | | | 1 | | *4* | | | |  |  |
|  | Operating frequency | | 1.7 GHz | | *0.5 to 1.7GHz* | | 0.5 GHz | | | | 0.5 GHz | | *0.5 to 1.7GHz* | | | |  |  |
|  | GPU | Number of shading clusters | | 3 | | *3* | | 1 | | | | 1 | | *3* | | | |  |  |
|  | Operating frequency | | 0.6 GHz | | *0.6 GHz* | | 0.6 GHz | | | | 0.6 GHz | | *0.6 GHz* | | | |  |  |
|  | Note: “Heavy load” refers to a load of 90% to 100% on the CPU and GPU.  “Medium load” refers to a load of approximately 50% on the CPU and GPU. | | | | | | | | | | | | | | | | |  |  |
|  |  | |  | | | |  | |  | |  |  |  |  |  |  |  |  |  |

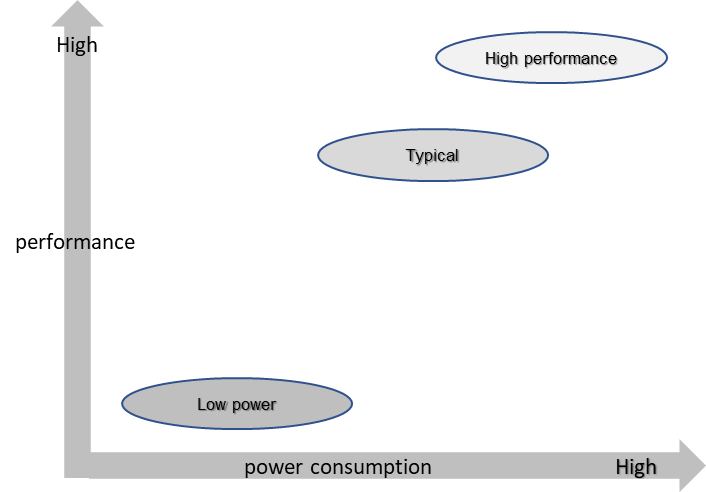
Figure 4‑2 Schematic View of Temperature Transitions   
with the SoC under Heavy and Medium Loads

# Setting example for power management functions

Since the setting of power management functions strongly depends on the customer's use-case, it is necessary for the customer to set it appropriately. The following are three modes shown in Table 5-1 as a reference case for setting of power management functions. The setting of each mode is an example when R-Car H3 is used.

Table 5‑1 The mode setting for power management

|  |  |
| --- | --- |
| mode | Overview |
| Typical mode | This mode is used when power control related function is enabled while maintaining processing performance of standard specifications.  (R-Car Series, 3rd Generation BSP default setting) |
| High performance mode | This mode is used to operate the entire system with maximum performance.  However, since power consumption / heat generation increases, attention is necessary. |
| Low power mode | This mode is used when operating the entire system with low power consumption.  However, care should be taken because the processing performance of the entire system will be lowered. |



**Figure 5‑1 Performance and power consumption by mode setting of power management functions**

## Typical mode setting example

**Table 5‑2 Typical mode setting example**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Function | Overview | command |
| 1 | CPU Freq | Specify frequency  (1.5GHz) | No setting required  (R-Car Series, 3rd Generation BSP default setting) |
| 2 | CPU Idle | Enable CPU Idle | No setting required  (R-Car Series, 3rd Generation BSP default setting) |
| 3 | CPU Hotplug | Maximize the number of CPUs | No setting required  (R-Car Series, 3rd Generation BSP default setting) |

## High performance mode setting example

**Table 5‑3 High performance mode setting example**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Function | Overview | command |
| 1 | CPU Freq | Specify frequency  (1.7GHz) | $ echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling\_governor  $ echo 1 > /sys/devices/system/cpu/cpufreq/boost  $ echo 1700000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling\_setspeed |
| 2 | CPU Idle | Disable CPU Idle | $ echo 1 > /sys/devices/system/cpu/cpu0/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu1/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu2/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu3/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu4/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu5/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu6/cpuidle/state1/disable  $ echo 1 > /sys/devices/system/cpu/cpu7/cpuidle/state1/disable |
| 3 | CPU Hotplug | Maximize the number of CPUs | No setting required  (R-Car Series, 3rd Generation BSP default setting) |

## Low power mode setting example

**Table 5‑4 Low power mode setting example**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Function | Overview | command |
| 1 | CPU Freq | Specify frequency  (500MHz) | $ echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling\_governor  $ echo 500000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling\_setspeed |
| 2 | CPU Idle | Enable CPU Idle | No setting required  (R-Car Series, 3rd Generation BSP default setting) |
| 3 | CPU Hotplug | Turn off the CPU 1-3 | $ echo 0 > /sys/devices/system/cpu/cpu1/online  $ echo 0 > /sys/devices/system/cpu/cpu2/online  $ echo 0 > /sys/devices/system/cpu/cpu3/online |

# Appendix

# Overview

The “CPU hotplug” function and “CPU freq” function are both functions related to power control to realize high-speed booting in response to the key being turned to start the engine and to suppress the generation of heat in on-vehicle terminals. This appendix gives notes, processing times, and so on regarding the use of these functions individually. Note that the values given here are for reference, so may differ with the environment and conditions of measurement.

# Times to Bring CPUs Online and Offline

The “CPU hotplug” function is used both in bringing some CPUs offline to lower power consumption or as a countermeasure for the generation of heat and in bringing offline CPUs back online when their performance is required. In the latter case, a time lag is incurred. Therefore, take this into account when using this function in use cases where a specified level of performance is essential. The times to handle the individual processing by the “CPU hotplug” function are listed below.

* **Processing times**
* **Bringing CPUs online: 36 ms**
* **Bringing CPUs offline: 53 ms**
* Measurement environment
* Software: Yocto v3.7.0 (BSP 3.6.2 kernel)
* Hardware: R-Car H3 Ver.3.0 (on a Salvator-XS board)
* Measurement conditions
* Measurement proceeds with the system in the idle state, i.e., booted up but with no applications running.
* The target CPU is any of the Cortex-A57 processors.
* Measurement points
* Processing time of cpu\_up and cpu\_down functions
* Measuring method
* Measure with ftrace

Note: ftrace time stamps are used to obtain measurements of time.

# Notes on Using Dynamic Voltage and Frequency Scaling (DVFS) or Dynamic Frequency Scaling (DFS)

The processing times for DVFS or DFS control differ with the size of the change in frequency. For a change in frequency that requires changing the multiplier setting for PLL0, the hardware specifications require that the CPU be placed in the low-speed mode for up to 50 µs. For this reason, if the timing of CPU operations is required to be precise to the µs scale (e.g., in access to an SD card) or responses are expected to be within 50 µs (e.g., for interrupts), be careful not to change the frequency such that a change to the multiplier setting for PLL0 is required. Table A3-1 shows the transitions which require changes to the multiplier setting for PLL0.

Table A3‑1 Table of Settings for DVFS or DFS

Boosted\*1

Normal\*2

Range of control by DVFS

Range of control by DFS

|  |  |  |  |
| --- | --- | --- | --- |
| Frequency of the Cortex-A57 | Voltage | CPU Divider | PLL0 |
| 1.7 GHz | 0.96 V | 32/32 | Times 100 |
| 1.6 GHz | 0.90 V | 32/32 | Times 96 |
| 1.5 GHz | 0.82 V | 32/32 | Times 90 |
| 1.0 GHz | 0.82 V | 21/32 | Times 90 |
| 0.5 GHz | 0.82 V | 11/32 | Times 90 |

Notes: The frequency table for the AVS0 signal in the R-Car H3 is used as an example.

1. The multiplier setting for PLL0 is changed in changes to the frequency   
 accompanying transitions between boosted and normal operations and while   
 boosting is in effect.

2. Since the multiplier setting for PLL0 is the same at all frequencies in normal   
 operation, these changes to the frequency do not involve low-speed operation   
 for up to 50 µs.

The following two operations correspond to the conditions shown in table A3-1.  
For more details, see the separate document with the filename “RENESAS\_RCH3M3M3NE3\_PowerManagement\_UME”.

* Setting to enable or disable boosting through sysfs
* Changes (static or dynamic) to the operating frequency when boosting is enabled

For the differences between processing for DVFS and DFS, see A3.1 and A3.2.

## Flow of Processing for DFS Control and Processing Times

An example of the flow of processing for DFS control is given in figure A3-1.

Clock framework

cpufreq framework

Regulator framework

i2c-dvfs

PMIC

Z clock

PLL0

Setting for change to the frequency

Changing the frequency division ratios

14 µs

33 µs

Software

Hardware

The DFS control only allows changes to the frequency division ratios. Therefore, the processing time is approximately 33 µs.

This result was obtained under the measurement conditions listed below. The values may differ with the environment and conditions of measurement.

<Measurement environment>

* Software: Yocto v3.7.0 (BSP 3.6.2 kernel)
* SoC: R-Car H3 Ver.3.0 (on a Salvator-XS board)
* PMIC: bd9571mwv
* Measurement conditions: Measurement proceeds with the system in the idle state, i.e., booted up but with no applications running.
* Measurement points: Processing time of clk\_set\_rate functions

Figure A3‑1 DFS Control Sequence (for the Change from 0.5 GHz to 1.5 GHz)

## Flow of Processing for DFVS Control and Processing Times

The flow of processing for DVFS control is given in figure A3-2.

123 µs

673µs

Setting for change to the frequency

Changing the frequency division ratios

Obtaining the PMIC information

Hardware

Reading twice

Changing the PMIC setting

Read–modify–write operation

Changing the   
multiplier setting

Setting for change to the voltage

813 µs

Software

Clock framework

cpufreq framework

Regulator

framework

i2c-dvfs

PMIC

Z clock

PLL0

In boosted operation, the PLL oscillation settling waiting time (up to 50 µs) is required within the period of 123 µs in the above figure since   
the multiplier setting for PLL0 is changed.

Interrupts are enabled over this period, so other processing can be handled.

Since the PLL0 output clock cannot be used during this period, the EXTAL clock is supplied to the Cortex-A57, so operation is at low speed.

This result was obtained under the measurement conditions listed below. The values may differ with the environment and conditions   
of measurement.

<Measurement environment>

* Software: Yocto v3.7.0 (BSP 3.6.2 kernel)
* SoC: R-Car H3 Ver3.0 (on a Salvator-XS board)
* PMIC: bd9571mwv
* Measurement conditions: Measurement proceeds with the system in the idle state, i.e., booted up but with no applications running.
* Measurement points: Processing time of \_set\_opp\_voltage and clk\_set\_rate functions

Figure A3‑2 DVFS Control Sequence (for the Change from 0.5 GHz to 1.7 GHz)

# Resume times for System Suspend to RAM

When System Suspend to RAM is executed, the resume times from suspend may be delayed depending on connection method of USB devices. Sample cases are shown in Table A4‑1.

Table A4‑1 Resume times for USB devices connection

|  |  |  |  |
| --- | --- | --- | --- |
| USB terminal | Connection method | Connected USB devices | Resume times (\*1)(\*2) |
| USB 2.0 | direct | USB mouse x 1 | 1391 ms |
| USB 2.0 | USB Hub | USB mouse x 1 | 2266 ms |
| USB 3.0 | direct | USB mouse x 1 | 1653 ms |
| USB 3.0 | USB Hub | USB mouse x 1 | 2362 ms |
| None | No connection | None | 1391 ms |

(\*1) This is one sample of measurement values on Yocto v3.15.0. It may be slightly different on customer environment.

(\*2) The resume period is measured from the beginning of ARM Trusted Firmware to the completion of kernel resume processing.

The delayed of resume times is caused by USB framework of Linux Kernel. Since this framework implements a reset sequence also supported other USB drivers, it causes the unnecessary wait processing is executed on using our USB drivers. However, our BSP policy is basically prohibiting to modify the common framework such as USB framework, therefore this implementation is kept.

This issue can be avoided by the following procedure.

* Execute unload / reload of USB 2.0 host control drivers before / after System Suspend to RAM

In order to be a countermeasure of the user application side, it is necessary for the customer side to deal with it. The detailed procedure is shown as below.

Step1 : Modularize USB 2.0 host control drivers

(1) Modify Linux Kernel Configuration

arch/arm64/configs/defconfig

CONFIG\_USB\_EHCI\_HCD=m

CONFIG\_USB\_OHCI\_HCD=m

CONFIG\_PHY\_RCAR\_GEN3\_USB2=m

(2) Re-build Linux Kernel and Module

$ make defconfig

$ make Image

$ make modules

(3) Install Linux Kernel and Module

Linux:

Overwrite with arch/arm64/Image

Module:

$ make modules\_install INSTALL\_MOD\_PATH=<rootfs\_dir>

Step2 : Unload / Reload USB 2.0 host control drivers

(1) Unload the target modules before System Suspend to RAM

$ rmmod -f ohci-platform

$ rmmod -f ohci-hcd

$ rmmod -f ehci-platform

$ rmmod -f ehci-hcd

$ rmmod -f phy\_rcar\_gen3\_usb2

(2) Reload the target modules after System Suspend to RAM

$ modprobe phy\_rcar\_gen3\_usb2

$ modprobe ehci-hcd

$ modprobe ehci-platform

$ modprobe ohci-hcd

$ modprobe ohci-platform

Note: This procedure can’t be applied if the either conditions as following are matched.

- The rmmod command can’t be executed before System Suspend to RAM.

- The target USB device is HID devices (mouse / keyboard)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Description | |
| Page | Summary |
| 1.01 | July, 2017 | — | First edition issued. |
| 1.02 | November, 2017 | 1 | Target device and Readers updated.(“R-Car M3-N”) |
| 1,15,21 | Reference document file name updated. |
| 1.03 | June, 2018 | 1 | Target device and Readers updated.(“R-Car E3”) |
| 1,14,22 | Reference document file name updated. |
| 2 | Table updated.(Table 1-1) |
| 10 | 3.4 “System suspend to RAM” control case added |
| 18 | 5.Setting example of Functions Related to Power Control added |
| 1.04 | February, 2019 | 24, 25 | A4 Resume times for System Suspend to RAM added |
| 1 | Target device and Readers updated.(“R-Car M3-W+”) |

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 VIL (Max.) and VIH (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 VIL (Max.) and VIH (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 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.

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(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.4.0-1 November 2017)

|  |  |  |
| --- | --- | --- |
| Corporate Headquarters |  |  |
| TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan  [www.renesas.com](https://www.renesas.com/) |  |  |
| Trademarks |  |  |
| Arm, Cortex are registered trademarks or trademarks of Arm Limited.  Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners. |  |  |



変更内容〔ルネサス内部向け〕

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | 発行日 | 改訂内容 | |
| ページ | ポイント |
| 1.01 |  | － | ・REV1.01の翻訳版をベースとしてRev1.01とする |
| 1.02 | 2017.11 | P1 | ・ターゲットデバイスとReadersにR-Car M3-Nを追加 |
| P1,P15,P21 | ・参照ドキュメントのファイル名を更新  (RENESAS\_RCH3M3M3N\_PowerManagement\_UME) |
| 1.03 | 2018.06 | P1 | ・ターゲットデバイスとReadersにE3を追加 |
| P1,P21,P22,  P23 | ・Yoctoのバージョンを3.7.0に修正 |
| P1,P14,P22 | ・参照ドキュメントのファイル名を更新  (RENESAS\_RCH3M3M3NE3\_PowerManagement\_UME) |
| P2 | ・表を更新(表1-1)  CASの説明追加 |
| P10 | ・3.4 System Suspend to RAMの制御事例を新規追加 |
| P18 | ・5. 電力制御関連機能の設定事例を新規追加 |
| P21,P22,P23 | ・処理時間、測定ポイント、測定方法を更新 |
| 1.04 | 2019.01 | P24,P25 | ・A.4 System Suspend to RAMの復帰時間の注意点を追加 |
| P1 | ・ターゲットデバイスとReadersにR-Car M3-W+の追加 |