R-Car Series, 3rd Generation

Processor Affinity

Introduction

[Background]

Third-generation R-Car series products have two types of CPU: Cortex-A57 cores (CA57s) with high processing performance and Cortex-A53 cores (CA53s) with low power consumption but also lower processing performance. In-vehicle systems require several applications, such as navigation, the playback of music or video, and display on meters, to run at the same time. A multi-core environment, which includes multiple CPUs, is generally more suitable for such cases. Accordingly, third-generation R-Car series products are suitable in such cases.

On the other hand, booting up the CA57s and CA53s with their different performances at the same time creates problems that must be taken into account. Namely, since the Linux scheduler assigns processes to CPUs such that the load on each CPU is equal in the environment, the CPU on which a process runs may switch between a CA57 and a CA53. This leads to a problem of applications behaving unstably, including changes of operating speed on the user’s side. “Processor affinity” can serve as one solution to this problem. Processor affinity refers to the binding of processes to and unbinding them from specific CPUs. With processor affinity, a process can be executed solely on either a CA57 or a CA53 without switching.

Accordingly, applying processor affinity is essential to obtain stable maximum performance in an environment where the CA57s and CA53s are booted up at the same time.

[Purpose]

To give users a greater sense of comfort in handling on-board facilities in on-board information systems, in which several applications run at the same time, Renesas suggests that multi-core solutions should utilize all CPUs. This document describes how to apply processor affinity in an environment where the CA57s and CA53s are booted up at the same time.

[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 and R-Car M3-W+ products.
* The electrical specifications of the multiple processor cores of R-Car H3, R-Car M3-W and R-Car M3-W+ products.
* The functions of the BSP drivers for R-Car H3, R-Car M3-W and R-Car M3-W+ products.

[Notes]

* Settings for processor affinity depend on the environment of the system in which you are to apply it. This document is for reference, so for actual applications, make settings that are appropriate for your system.
* Statements in relation to operating systems in this document apply to Yocto v2.12.0 (for the R-Car H3, R-Car M3-W) from Renesas, and may be subject to change, including those in the procedure for configuring the environment, with later versions of Yocto.

Target Device

　・R-Car H3

　・R-Car M3-W/R-Car M3-W+

Contents

[1. Realizing Processor Affinity 3](#_Toc475972841)

[1.1 Environments where the CA57s and CA53s are Booted up at the Same Time:   
Problems and Solutions 3](#_Toc475972842)

[1.1.1 Problems 3](#_Toc475972843)

[1.1.2 Solutions 4](#_Toc475972844)

[1.2 Overview of Processor Affinity 5](#_Toc475972845)

[1.2.1 Processor Affinity 5](#_Toc475972846)

[1.3 Overview of Cgroup 6](#_Toc475972847)

[1.3.1 Cgroup 6](#_Toc475972848)

[2. Grouping Applications 7](#_Toc475972849)

[2.1 Example of the Classification of Applications 7](#_Toc475972850)

[2.2 Using Cgroup to Control Groups 8](#_Toc475972851)

[3. Procedure for Setting Cgroup 9](#_Toc475972852)

[3.1 Design Procedure for Cgroup in Outline 9](#_Toc475972853)

[3.2 Preparation in Advance of Using Cgroup 10](#_Toc475972854)

[3.3 Creating Groups 11](#_Toc475972855)

[3.4 Assigning Applications 12](#_Toc475972856)

[Appendix 13](#_Toc475972857)

[A1. Booting the CA57s and CA53s up at the Same Time 13](#_Toc475972858)

[A2. Master Boot CPU 14](#_Toc475972859)

[A2.1 Setting the Master Boot CPU 14](#_Toc475972860)

# Realizing Processor Affinity

## Environments where the CA57s and CA53s are Booted up at the Same Time: Problems and Solutions

### Problems

Several applications, such as navigation, the playback of music or video, and display on meters, will generally be running at the same time in an on-board information system. Handling the applications in parallel by using multiple processor cores leads to smooth operation. Third-generation R-Car series products have two types of CPU, the CA57 and CA53. Accordingly, all cores need to be used to provide an environment that is suitable for on-board information systems.

On the other hand, when the CA57s and CA53s with their different processing performances are booted up at the same time, the Linux scheduler assigns processes (applications) to CPUs so as to equalize the loads on each of the CPUs. Defining the type of CPU on which a given application will run as a CA57 or CA53 is thus not possible. This may create problems in some cases. For example, an application that imposes a heavier load than the processing performance of a CA53 may in fact be assigned to a CA53, and the result will be failure to fulfill the performance requirements of the application (the left side of figure 1-1).

### Solutions

To solve the problems described on the previous page, heavy-load applications need to be run on a CA57. This document describes how to apply processor affinity, which allows users to bind an application to a specified CPU or a specified range of CPUs for execution, as a solution (the right side of figure 1-1).

The heavy-load application is exclusively assigned to the CA57 by applying processor affinity.

App. 2

App. 2

App. 3

App. 3

App. 1

App. 1

Processing performance

A heavy-load application is assigned to the CA53, which results in overly long times for processing.

Environment where processor affinity is applied

Applications

Processing performance

Linux environment

CA53

CA57

CA53

CA57

Figure 1‑1 Assignment of Applications in Outline

## Overview of Processor Affinity

### Processor Affinity

Processor affinity is the method of allowing users to bind a specified application to a particular CPU or range of CPUs for running the application. Applying processor affinity in an environment where the CA57s and CA53s are booted up at the same time allows users to bind an application to the CA57s or CA53s to prevent a CA53 from unexpectedly having to handle a heavy load of processing.

There are several methods of realizing processor affinity under Linux. They are listed and described in table 1-1. We recommend control group (cgroup) from the viewpoint of controlling processes in groups. This document describes how to apply cgroup.

Table 1‑1 Realizing Processor Affinity

|  |  |
| --- | --- |
| Method of Realizing Processor Affinity | Outline |
| Cgroup | Cgroup is a Linux standard feature, where processes are classified into groups for control of the assignment of resources such as CPUs and memory. Cgroup can be handled through sysfs. |
| taskset | The taskset command can be used to realize processor affinity by specifying process IDs (PIDs) and the CPUs to run the processes from the command line. With affinity through the taskset command, CPUs must be assigned per process. |
| sched\_setaffinity | Calling the sched\_setaffinity function in a program enables the execution of a process on a specified CPU. With affinity through the sched\_setaffinity function, CPUs must be assigned per process. |

## Overview of Cgroup

### Cgroup

Cgroup is a standard Linux feature in which processes are classified into groups for control of the assignment of resources such as CPUs and memory. Multiple groups can be created. Processes classified in the same group run by using specified resources. Applying cgroup in an environment where the CA57s and CA53s are booted up at the same time allows the classification of applications for execution on a CA57 or a CA53, so the applications only run on a specified CPU resource as shown in figure 1-2.

Applying processor

affinity through

cgroup

App. 1

App. 2

App. 3

Group classified for the CA53

CPU resources

Applications

Group classified   
for the CA53

Processing   
performance

Group classified for the CA57

CPU resources

Application

Group classified for the CA57

App. 1

Heavy-load application

CA53

CA57

App. 3

App. 2

CA53

CA53

App. 1

CA57

CA57

CA57

CA57

CA53

CA53

App. 3

App. 2

Figure 1‑2 Example of Assignment of Applications through Cgroup

# Grouping Applications

## Example of the Classification of Applications

This section describes how to classify applications in an environment where the CA57s and CA53s are booted up at the same time.

Cgroup controls the assignment of resources such as CPUs and memory to each of the groups. Accordingly, first of all, classify applications in terms of the performance required of CPU resources. An example of a system of classification is given in table 2-1.

Table 2‑1 Example of Classifications for Applications

|  |  |  |  |
| --- | --- | --- | --- |
| Classification of Applications | Descriptions and Examples of Applications | Required CPU Resource | Group |
| Applications which require a specified level of performance from the CPU | An application that, if running on the CA53 is attempted, the performance required for its operation (in terms of, e.g., the frame rate) may not be achieved  Example: A navigation system | CA57 | Big group |
| Applications which require higher priority for realtime control | An application which requires the quickest possible response and is suited to running on the CA57 due to its high processing performance  Example: An instrument cluster | CA57 | Big group |
| Applications which do not require a particularly high level of performance from the CPU | An application which does not require a particularly high level of performance from the CPU and can run in low power consumption mode throughout its operation  Example: The playback of music, radio, etc. | CA53 | Little group |
| Applications other than the above | An application with operation that switching between the CA57 and CA53 does not affect.  Example: Background processes, kernel threads, etc. | CA57 or CA53 | Root group |

## Using Cgroup to Control Groups

This subsection describes how to use cgroup to control the groups of applications classified in section 2.1. Use cgroup to group all applications (processes). Cgroup binds all CPU resources to a group by default. In this document, the group created by default is referred to as the root group. After that, create the new big and little groups and bind the CA57 and CA53 CPU resources to the respective groups. Finally, from among the applications in the initial root group place those to be exclusively run on a CA57 in the big group and those to be exclusively run on a CA53 in the little group. This ends the assignment of the applications to the respective groups.

A big-group task

Little-group tasks

Root-group tasks

Root group

App. 4

App. 3

App. 2

Little group

Big group

App. 6

App. 5

App. 1

Little-group CPUs

Root-group CPUs

CA53

CA53

CA57

CA57

CA53

CA53

CA57

CA53

CA57

Big-group CPUs

CA57

CA57

CA53

Figure 2‑1 Using Cgroup to Group Applications

# Procedure for Setting Cgroup

## Design Procedure for Cgroup in Outline

This subsection describes how to build the system described in section 2 in a Yocto environment from Renesas.

1. Preparation in Advance of Using Cgroup

The cpuset filesystem is not usable by default in the Yocto environment from Renesas. Accordingly, the method of making the cpuset filesystem usable is described in section 3.2.

2. Creating groups

The method of creating groups with cgroup is described in section 3.3. As the cpuset group\* is automatically created at start-up, this only covers how to create the big and little groups.

3. Assigning Applications

The method of assigning applications to the created groups is described in section 3.4.

Note: \* The cpuset group is here referred to as the root group.

## Preparation in Advance of Using Cgroup

Step 1: Booting up the CA57s and CA53s at the same time  
Rebuild the ARM trusted firmware so that the CA57s and CA53s are booted up at the same time by following the procedure in appendix 1.

Step 2: Enabling the cpuset filesystem  
To use the cpuset filesystem, rebuild the Linux kernel with the configuration item CONFIG\_CPUSETS (disabled by default) enabled.  
For details on how to build the kernel, refer to the following URL on the official Yocto site.

http://www.yoctoproject.org/docs/2.0.2/dev-manual/dev-manual.html#using-menuconfig

→ General setup

→ [\*] Control Group support

→ [\*] Cpuset controller

After starting up the board, confirm that the /sys/fs/cgroup/cpuset directory has been created.

## Creating Groups

Step 1: Confirming the CPUs that have been booted up  
Execute the following command to confirm the CPU number and type (CA57 or CA53).

# cat /proc/cpuinfo

/\* The types of CPU can be confirmed from the CPU part in the result of execution as follows. CA57 -> 0xd07 and CA53 -> 0xd03. \*/

Step 2: Creating the big and little groups  
Execute the following commands to create the big and little groups. As the cpuset group has automatically been created, you do not have to newly create it.

# mkdir /sys/fs/cgroup/cpuset/big

# mkdir /sys/fs/cgroup/cpuset/little

Step 3: Setting the memory node  
The memory node is a parameter for use in setting the assignment of memory to the cgroup. If you do not wish to specify the assignment of memory, set it to 0 as in our example.

# echo 0 > /sys/fs/cgroup/cpuset/big/cpuset.mems

# echo 0 > /sys/fs/cgroup/cpuset/little/cpuset.mems

Step 4: Assigning CPU resources to the groups  
Register the CPU numbers for the CA57 and CA53 processors confirmed in step 1 with the big and little groups, respectively. In this case, the CA57s have been registered with the CPU numbers 0 to 3 and the CA53s with 4 to 7.

# echo 0-3 > /sys/fs/cgroup/cpuset/big/cpuset.cpus

# echo 4-7 > /sys/fs/cgroup/cpuset/little/cpuset.cpus

## Assigning Applications

Step 1: Confirming the PIDs of applications  
Execute the following command to confirm the PIDs for applications to be assigned with cgroup.

# ps

/\* Confirm the PID lines in the result of executing the command. \*/

Step 2: Assigning the applications  
Execute the following commands to assign applications to the big and little groups. Replace [PID] in the respective commands with the PIDs for the applications to be assigned before execution.  
• For assignment to the big group

# echo [PID] > /sys/fs/cgroup/cpuset/big/tasks

• For assignment to the little group

# echo [PID] > /sys/fs/cgroup/cpuset/little/tasks

Assigning an application to the cpuset group involves releasing the setting which had been made for its assignment to the big or little group.  
• Release from assignment to the big or little group

# echo [PID] > /sys/fs/cgroup/cpuset/tasks

# Appendix

# Booting the CA57s and CA53s up at the Same Time

By default, only the CA57s are initially booted up in the Yocto environment from Renesas. Booting up both the CA57s and CA53s at the same time requires a change to the value of the PSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOT flag in the ARM Trusted Firmware (ATF) and rebuilding the ATF.

To boot both the CA57s and CA53s up at the same time, set the PSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOT flag to 0 and rebuild the ATF. Which of the individual CPUs will be booted up and not booted up in response to the settings of the PSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOT flag is listed in table A1-1. The table applies in the case of the R-Car H3.

Note: For details on building the ATF, see sections 4.3, Option Setting, and 4.4, How to, in the separate document with the filename “RENESAS\_RCH3M3\_SecurityBSP\_UME\_v1.0.3.pdf”.

When the CA57s and CA53s are booted up at the same time, the ID numbers of the CPUs controlled by Linux can be confirmed in /proc/cpuinfo.

# cat /proc/cpuinfo

/\* The types of CPU can be confirmed from the CPU part in the result of execution as follows. CA57 -> 0xd07 and CA53 -> 0xd03. \*/

Table A1‑1 CPUs to be Booted up

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Setting of the PSCI\_DISABLE\_ BIGLITTLE\_IN\_ CA57BOOT flag | CA57 | | | | CA53 | | | |
| CPU0  Master Boot CPU | CPU1 | CPU2 | CPU3 | CPU4 | CPU5 | CPU6 | CPU7 |
| 1 (default) | ON | ON | ON | ON | **OFF** | **OFF** | **OFF** | **OFF** |
| 0 (booting the CA57s and CA53s up at the same time) | ON | ON | ON | ON | **ON** | **ON** | **ON** | **ON** |

# Master Boot CPU

### Setting the Master Boot CPU

When both the CA57s and CA53s are to be booted up at the same time, the CPU to serve as the master boot CPU is selectable. On the Salvator-X system evaluation board, the settings of DIP switches (SW10) shown in table A2-1 determine the master boot CPU. Table A2-2 shows examples of the CPU configurations (for the R-Car H3) when the master boot CPU is changed. Note that changing the master boot CPU does not require changes to software (the IPL, u-boot, kernel, etc.).

Table A2‑1 DIP Switch Settings

|  |  |  |
| --- | --- | --- |
| MD7 (Pin 1 of SW10) | MD6 (Pin 2 of SW10) | Description |
| 0 | 0 | A CA57 serves as the master boot CPU. |
| 0 | 1 | A CA53 serves as the master boot CPU. |

Table A2‑2 CPU Configuration

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CPU0  Master Boot  CPU | CPU1 | CPU2 | CPU3 | CPU4 | CPU5 | CPU6 | CPU7 |
| Booting up from a CA57 | **CA57** | CA57 | CA57 | CA57 | CA53 | CA53 | CA53 | CA53 |
| Booting up from a CA53 | **CA53** | CA57 | CA57 | CA57 | CA57 | CA53 | CA53 | CA53 |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Description | |
| Page | Summary |
| 1.00 | May, 2017 | — | First edition issued. |
| 1.01 | March, 2019 | 1 | Target Readers, Target Device 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** |  |  |
| 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 |
| 1.00 | 2017.05 | － | ・Rev0.50（更新版）の翻訳版をベースとしてRev1.00とする  ・ファイル名変更  ・ターゲットデバイス名を記載  ・ドキュメント管理番号、発行年月を記載  ・ご注意書きのページ追加  ・TableA1-1 「Master boot CPU」の”b”をTableA2-2に合わせて大文字”B”に変更 |
|  |  |  |  |