R-Car Series, 3rd Generation

Processor Affinity

要旨

[Background]

R-Car Series, 3rd Generationでは処理性能が高いCortex-A57(CA57)と処理性能は低いが低消費電力動作可能なCortex-A53(CA53)の2種類のCPUを搭載しております。車載システムのような、ナビゲーション、音楽再生、動画再生、メーター表示等、複数のアプリケーションを同時に動作させる場合には、CPUの個数が多いマルチコア環境の方が快適に動作することが一般的であり、本利用シーンにはR-Car Series, 3rd Generationの活用が適しています。

一方で、処理性能が異なるCA57とCA53を同時起動する場合は考慮すべき弊害が生じます。Linuxのスケジューラーは各CPUに対して処理負荷が均等になるようにプロセスを割り付けるため、本環境下で動作するプロセスはCA57とCA53の間を切り替わりながら動作する場合があります。ユーザ視点では、アプリケーションの挙動が急に早くなったり遅くなったり不安定になるため、アプリケーションが正常に動作しないなどの課題があります。本課題へのソリューションの１つとして、Processor affinityを使う方法があります。 Processor affinityはプロセスが特定のCPUと関連付けられた制御を行うもので、プロセスがCA57とCA53の間を切り替わらないように制御することが可能です。

従って、 CA57/CA53同時起動環境において、安定した最大パフォーマンスを得るためには Processor affinityも合わせて使用することが必要不可欠となります。

[Purpose]

複数アプリケーションを同時に動作する車載情報システムで快適な操作性をユーザーへ提供することを目的として、弊社では全CPUを活用したマルチコアソリューションを提案致します。本ドキュメントではCA57/CA53同時起動環境下におけるProcessor affinityの設定方法を説明します。

[Readers]

本ドキュメント読者は、下記の一般的な知識を持っていることを想定しています。

* 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.

[Note]

* Processor affinityはお客様のシステム環境に合わせて設定が必要なため、本ドキュメントを参考にお客様毎に適用をお願いします。
* 本ドキュメントは弊社が提供するYocto v2.12.0(R-Car H3/M3-W)を対象に作成しております。今後のリリースバージョンに応じて環境構築手順等に変更が発生する場合があります。

ターゲットデバイス

・R-Car H3

・R-Car M3-W/R-Car M3-W+

目次

[1. Processor Affinityの実現 3](#_Toc478135476)

[1.1 CA57/CA53同時起動環境の課題と解決策 3](#_Toc478135477)

[1.1.1 課題 3](#_Toc478135478)

[1.1.2 解決策 4](#_Toc478135479)

[1.2 Processor Affinityの概要 5](#_Toc478135480)

[1.2.1 Processor Affinity 5](#_Toc478135481)

[1.3 Cgroupの概要 6](#_Toc478135482)

[1.3.1 Cgroup 6](#_Toc478135483)

[2. アプリケーションのグループ化 7](#_Toc478135484)

[2.1 アプリケーションの分類事例 7](#_Toc478135485)

[2.2 Cgroupによるグループ管理 8](#_Toc478135486)

[3. Cgroupの設定手順 9](#_Toc478135487)

[3.1 Cgroupの設計手順概要 9](#_Toc478135488)

[3.2 Cgroupの事前準備 10](#_Toc478135489)

[3.3 グループの作成 11](#_Toc478135490)

[3.4 アプリケーションの割り付け 12](#_Toc478135491)

[Appendix 13](#_Toc478135492)

[A1. CA57/CA53同時起動方法 13](#_Toc478135493)

[A2. Master boot CPU 14](#_Toc478135494)

[A2.1 Master boot CPUの設定方法 14](#_Toc478135495)

# Processor Affinityの実現

## CA57/CA53同時起動環境の課題と解決策

### 課題

車載情報システムでは、ナビゲーション、音楽再生、動画再生、メーター表示等、複数のアプリケーションを同時に動作させる場合には、マルチコア環境でアプリケーションを並列に処理させた方が快適に動作します。R-Car Series, 3rd GenerationではCA57とCA53の2種類のCPUが搭載されており、全てのコアを使用し車載情報システムに適した環境を提供する必要があります。

しかし、処理性能が異なるCA57とCA53を同時起動した場合、Linuxのスケジューラーは各CPUに対して処理負荷が均等になるようにプロセス(アプリケーション)を割り付けるため、動作するCPUがCA57となるかCA53となるか確定しません。そのため、動作するCPUによっては、CA53の処理性能を超えた高負荷アプリケーションがCA53に割り付けられて、アプリケーションが要求する性能が出ないなどの課題があります。(図1-1左側)

### 解決策

上記課題を解決するためには、高負荷アプリケーションをCA57固定で動作させる必要があります。本ドキュメントでは、ユーザがアプリケーションを指定したCPU又はCPUグループに固定で割り付け動作させるProcessor affinityを使用した解決策について説明します(図1-1右側)。

App. 3

App. 2

App. 1

App. 3

App. 1

App. 2

Linux OS環境

Processor affinity適用環境

アプリケーション

高負荷アプリケーションをProcessor affinityでCA57に固定で割り付ける

高負荷アプリケーションが

CA53に割り当たり処理時間が長期化する

処理性能

CA53

CA57

処理性能

CA53

CA57

図 1‑1　アプリケーションの割り付け概要

## Processor Affinityの概要

### Processor Affinity

Processor affinityは、ユーザが指定したアプリケーションをCPU又はCPUグループに固定で割り付け動作させる手法です。CA57とCA53が同時起動した環境では、ユーザがアプリケーションをCA57又はCA53に固定的に割り付けることで、高負荷プロセスが不用意にCA53で動作することがなくなります。

Processor affinityを実現するための手法は表1-1に示したようにいくつかありますが、プロセスをグループで管理できる観点から、Cgroupを推奨します。そのため、本ドキュメントではCgroupを使用した方法について説明します。

表 1‑1　Processor Affinityの実現手法

|  |  |
| --- | --- |
| Processor affinity | 概要 |
| Cgroup | CgroupはLinuxの標準機能で、プロセスをグループ化して、CPUやメモリなどのリソースをグループ単位に割り当て管理することができます。Cgroupはsysfsからコントロールすることができます。 |
| taskset | tasksetは、コマンドラインからプロセスID(PID)と動作させるCPUを指定し実行することで、Processor affinityを実現することができます。tasksetではプロセス単位でCPUを割り当てる必要があります。 |
| sched\_setaffinity | sched\_setaffinityはプログラム内でコールすることで、プロセスを特定のCPUで実行させることが可能です。sched\_setaffinityではプロセス単位でCPUを割り当てる必要があります。 |

## Cgroupの概要

### Cgroup

CgroupはLinuxの標準機能で、プロセスのグループを作成して、CPUやメモリなどのリソースをグループ単位に割り当て管理することができる機能です。グループは複数作成することができ、グループに属するプロセスは割り当てられたリソース内で動作します。CA57とCA53を同時起動させた環境の場合は、図1-2に示したように、CA57又はCA53のみで動作させるアプリケーションごとにグループ化し、各CPUリソースを割り当てることで、アプリケーションをCA57又はCA53に固定で割り付けることができます。

App. 3

App. 2

App. 1

App. 2

App. 3

App. 1

CA53 group

CA57 group

処理性能

CA53

CA57

高負荷

アプリケーション

Cgroupによる

Processor affinity

CA53 group

CA57 group

App. 3

アプリケーション

App. 2

CPUリソース

CA53

CA53

アプリケーション

App. 1

CPUリソース

CA57

CA57

CA57

CA57

CA53

CA53

図 1‑2　Cgroupを用いたアプリケーションの割り付け例

# アプリケーションのグループ化

## アプリケーションの分類事例

本章では、CA57とCA53を同時起動させた環境下でのアプリケーションの分類方法について説明します。

CgroupはCPUやメモリなどのリソースをグループ単位に割り当て管理するため、まずはじめにCPUリソースの使い方の観点からアプリケーションを分類しました。その事例を表2-1に示します。

表 2‑1　アプリケーションの分類例

|  |  |  |  |
| --- | --- | --- | --- |
| アプリケーションの分類 | アプリケーションの例 | 必要とする  CPUリソース | グループ |
| CPU性能を要求するアプリケーション | CA53で動作した場合に要求性能(例：フレームレート等)を満たすことができない可能性があるアプリケーション  例：ナビゲーション | CA57 | Big group |
| リアルタイム性が必要なアプリケーション | 可能な限り早いレスポンスが求められ処理性能が高いCA57で動作させることが望ましいアプリケーション  例：メータークラスター | CA57 | Big group |
| CPU性能を必要としないアプリケーション | 常時CPU性能を必要とせず、低消費電力で動作させるアプリケーション  例：音楽再生、ラジオ、etc | CA53 | Little group |
| 上記以外のアプリケーション | 動作CPUがCA57とCA53間で切り替わっても動作に影響がないアプリケーション  例：バックグラウンドプロセス、kernelスレッド、etc | CA57 or CA53 | Root group |

## Cgroupによるグループ管理

2.1章で分類したアプリケーショングループをCgroupで管理する方法について説明します。Cgroupは全てのアプリケーション(プロセス)をグループ化し、全CPUリソースが割り当たっているグループがデフォルトで作成されます。本ドキュメントでは、デフォルトで作成されるグループをroot groupとします。big groupとlittle groupに関しては新規で作成し、CA57とCA53のCPUリソースをそれぞれ割り当てます。ユーザはroot groupに属しているアプリケーションの内、CA57で固定して動作させたいものをbig groupに、CA53で固定して動作させたいものをlittle groupにセットすることでアプリケーションを各グループに分類します。

Little-group tasks

A big-group task

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

Big-group CPUs

CA53

CA53

CA57

CA57

CA53

CA53

CA57

CA53

CA57

CA57

CA57

CA53

図 2‑1　Cgroupのグループ分類

# Cgroupの設定手順

## Cgroupの設計手順概要

本章では、2章で説明したシステムを弊社のYocto環境で構築する方法について説明します。

1. Cgroupの事前準備

弊社のYocto環境ではデフォルトcpusetが使用できません。そのためcpusetを使用する方法について説明します。

2. グループの作成

Cgroupのグループ作成方法について説明します。cpuset group(\*)は起動時に自動で作成されるため、ここではbig groupとlittle groupの作成方法について説明します。

3. アプリケーションの割り付け

アプリケーションを作成したgroupに割り付ける方法について説明します。

(\*)：本章ではcpusetをroot groupとして説明します。

## Cgroupの事前準備

Step 1：CA57/CA53同時起動

AppendixのCA57/CA53同時起動の手順に従いビルドしてください。

Step 2：cpusetの有効

cpusetを利用するためにLinuxのCONFIG\_CPUSET(デフォルト無効)を有効にしてビルドしてください。  
build方法に関してはYoctoの公式HPを参照してください。  
http://www.yoctoproject.org/docs/2.0.2/dev-manual/dev-manual.html#using-menuconfig

→ General setup

→ [\*] Control Group support

→ [\*] Cpuset controller  
ボード起動後、/sys/fs/cgroup/cpuset が作成されていることを確認してください。

## グループの作成

Step 1：起動CPUの確認

下記のコマンドを実行し、CPU番号とCPUの種類(CA57 or CA53)を確認してください。

# cat /proc/cpuinfo

/\* CPUの種類は、実行結果のCPU partで確認できます。　CA57→0xd07、CA53→0xd03 \*/

Step 2：big groupとlittle groupの作成

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

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

下記のコマンドでbig/little groupを作成します。cpuset groupは自動で作成されているので新たに作成する必要はありません。

Step 3：memory nodeの設定

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

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

memory nodeは、Cgroupのメモリの割り当てを設定するパラメータです。メモリの割り当てを指定しない場合は"0"を設定してください。

Step 4：各グループへのCPUリソースの割り当て

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

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

STEP1で確認したCA57のCPU番号をbig group、CA53のCPU番号をlittle groupに登録します。 ここではCA57のCPU番号は0～3、CA53のCPU番号は4～7として説明します。

## アプリケーションの割り付け

Step 1：アプリケーションのPIDの確認

# ps

/\* 実行結果のPID列を確認してください \*/

下記のコマンドでCgroupで割り付けるアプリケーションのPIDを確認してください。

Step 2：アプリケーションの割り付け

下記のコマンドでアプリケーションをbig/little groupに割り付けます。コマンドの [PID] は割り付けるアプリケーションのPIDに置き換えて実行してください。

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

* big groupに割り付け
* little groupに割り付け

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

また、cpuset groupにアプリケーションを割り付けることで、big/little groupの割り付けを解除することができます。

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

* big/little group割り付けの解除

# Appendix

# CA57/CA53同時起動方法

弊社Yocto環境では、デフォルトでCA57のみ起動します。CA57とCA53を同時に起動させるには、ARM Trusted Firmware(ATF)のPSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOTフラグの値を変更し、ATFを再度buildする必要があります。

CA57とCA53を同時に起動させる場合は、PSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOTに0をセットし、ATFを再度buildしてください。PSCI\_DISABLE\_BIGLITTLE\_IN\_CA57BOOTのフラグで起動するCPUを表A1-1に示します。表はR-Car H3の場合を示しています。

(\*) buildの詳細は「RENESAS\_RCH3M3\_SecurityBSP\_UME\_v1.0.3.pdf」 の4.3 Option Settingと4.4 How toを参照してください。

CA57とCA53を同時に起動させた場合、Linuxで管理されるCPU番号は /proc/cpuinfo で確認することができます。

# cat /proc/cpuinfo

/\* CPUの種類は、実行結果のCPU partで確認できます。　CA57→0xd07、CA53→0xd03 \*/

表 A1‑1　起動CPU一覧

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PSCI\_DISABLE\_ BIGLITTLE\_IN\_ CA57BOOT | CA57 | | | | CA53 | | | |
| CPU0  Master  Boot  CPU | CPU1 | CPU2 | CPU3 | CPU4 | CPU5 | CPU6 | CPU7 |
| 1(デフォルト値) | ON | ON | ON | ON | **OFF** | **OFF** | **OFF** | **OFF** |
| 0(CA57とCA53同時起動) | ON | ON | ON | ON | **ON** | **ON** | **ON** | **ON** |

# Master boot CPU

### Master boot CPUの設定方法

CA57とCA53を同時に起動させた場合、Master boot CPUをCA57とCA53から選択することができます。Salvator-Xでは表A2-1に示すDip switch(SW10)の設定でMaster boot CPUを切り替えることができます。例として、Master boot CPUを変更した場合のCPU構成(R-Car H3)を表A2-2に示します。なお、Master boot CPUの切り替えには、ソフトウェア(IPL/u-boot/Kernel等)の変更は不要です。

表 A2‑1　Dip switch設定

|  |  |  |
| --- | --- | --- |
| MD7(SW10-1) | MD6(SW10-2) | Description |
| 0 | 0 | Master boot CA57 |
| 0 | 1 | Master boot CA53 |

表 A2‑2　CPU構成

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CPU0  Master Boot  CPU | CPU1 | CPU2 | CPU3 | CPU4 | CPU5 | CPU6 | CPU7 |
| boot CA57 | **CA57** | CA57 | CA57 | CA57 | CA53 | CA53 | CA53 | CA53 |
| boot CA53 | **CA53** | CA57 | CA57 | CA57 | CA57 | CA53 | CA53 | CA53 |

改訂記録

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | 発行日 | 改訂内容 | |
| ページ | ポイント |
| 1.00 | 2017.03 | ― | 新規発行 |
| 1.01 | 2019.03 | P1 | ・Target Readers, Target DeviceにR-Car M3-W+を追加 |
| P1,P3 | ・R-Car Gen3をR-Car Series, 3rd Generationへ表記変更 |



製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

CMOS製品の取り扱いの際は静電気防止を心がけてください。CMOS製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS製品の入力がノイズなどに起因して、VIL（Max.）からVIH（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、VIL（Max.）からVIH（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。

2. 当社製品、本資料に記載された製品デ－タ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。

3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。

4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。

5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、家電、工作機械、パーソナル機器、産業用ロボット等

高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

当社製品は、データシート等により高信頼性、Harsh environment向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。

6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。

7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。

8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。

9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。

10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。

11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1 2017.11)

|  |  |  |
| --- | --- | --- |
| 本社所在地 |  |  |
| 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）  [www.renesas.com](https://www.renesas.com/) |  |  |
| 商標について |  |  |
| ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。 |  |  |

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 |
| 0.5 | Oct. 27th, 2016 |  | ・カラー図をモノクロに変更  - パワポP.7, 9, 12  ・文言の変更  - パワポP.8「下記の表」→「表1-1」  - パワポP.9「下図」→「図1-2」  - パワポP.11「下記」→「表2-1」  - パワポP.12「2.1.」→「2.1章」  - パワポP.19「下記の表.」→「表A1-1」  - パワポP.20 2行目「下記の表.」→「表A2-1」  - パワポP.20 4行目「下記の表.」→「表A2-2」  ・注記マークを (\*) のスタイルに統一  - パワポP.14 ※1  - パワポP.19 ※  ・全角から半角に変更  - パワポP.16, 17 STEP１の"1" |
| 1.00 | 2017.03 | －  －  P1  －  P25  P3-P8,P13-P14  P1,P3,P5  P1  P18  P4,P6,P8  P7  P8 | ・Rev0.50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のCOPYRIGHTを2017年版に変更  ・REV0.50での更新点（赤字）を黒字に変更  ・英語版での次の修正を日本語版に反映  ‐タイトルで使用されているProcessor affinity のaを小文字から大文字に変更  ‐Target Readersの記載を英語版の記載に変更  ‐タイトルで使用されている Master boot CPUのbを小文字から大文字に変更  ‐App\*をApp.\*に変更(\*は数字)  ‐Table2-1内のbig group, little group, root groupのb,l,rを小文字から大文字に変更  ‐Figure2-1内のLittle group,Root group, Big groupの単語の間にハイフンを追加 |
|  |  |  |  |