R01AN6929EJ0110
Rev.1.10

RH850

Multi-core LSI

Introduction

This application note describes how to use Renesas Electronics multi-core LSI. This note will focus on the key points when creating multi-core software and how to use the tool (Renesas Electronics CS+).

Target Device

RH850/U2Bx, RH850/C1x, RH850/C1M-Ax

Contents

Contents

1. Introduction

1.1 Overview

Multiple CPUs equipped in one LSI is called multi-core. Renesas Electronics have a wide selection of LSI equipped with RH850 G4MH/G3MH/G3M which are applicable for HEV/EV. Among them are RH850/U2Bx, RH850/C1M-Ax and RH850/C1M-x. For further information, please do not hesitate to contact us.

RH850/U2Bx, RH850/C1x and RH850/C1M-Ax are specialized in motor control. Previously, a single LSI can only control one motor, but multi-core system (e.g., one RH850/U2Bx) is able to control two motors, and hence reducing BOM cost.

1.2 Purpose

This application note describes how to use Renesas Electronics multi-core LSI. Generally, each CPU operates while sharing peripheral IP in one LSI, but software can be created based on the same idea as single-core LSI. This note will focus on the precautions when creating multi-core software and how to use the tool (Renesas Electronics CS+).

2. What is Multi-core?

2.1 The Concept of Multi-core System

The most frequently asked question is whether it would affect each CPU if there were two CPUs operating at the same time. IP is shared inside the LSI. Therefore, even there is waiting time when accessing the same IP simultaneously, CPU will still operate independently. For instance, up until now, one LSI can only control one motor. However, two motors can be controlled through one LSI if multi-core LSI (e.g. RH850/U2Bx) is used.

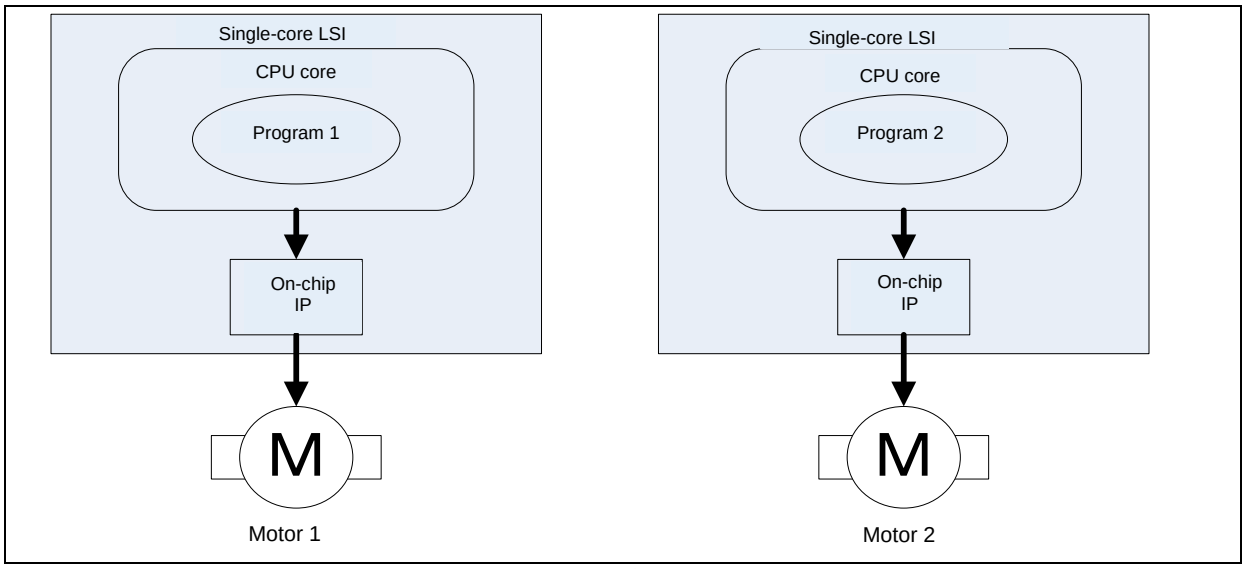

Fig 2-1. Schematic diagram of single-core LSI（one single-core LSI controls one motor）

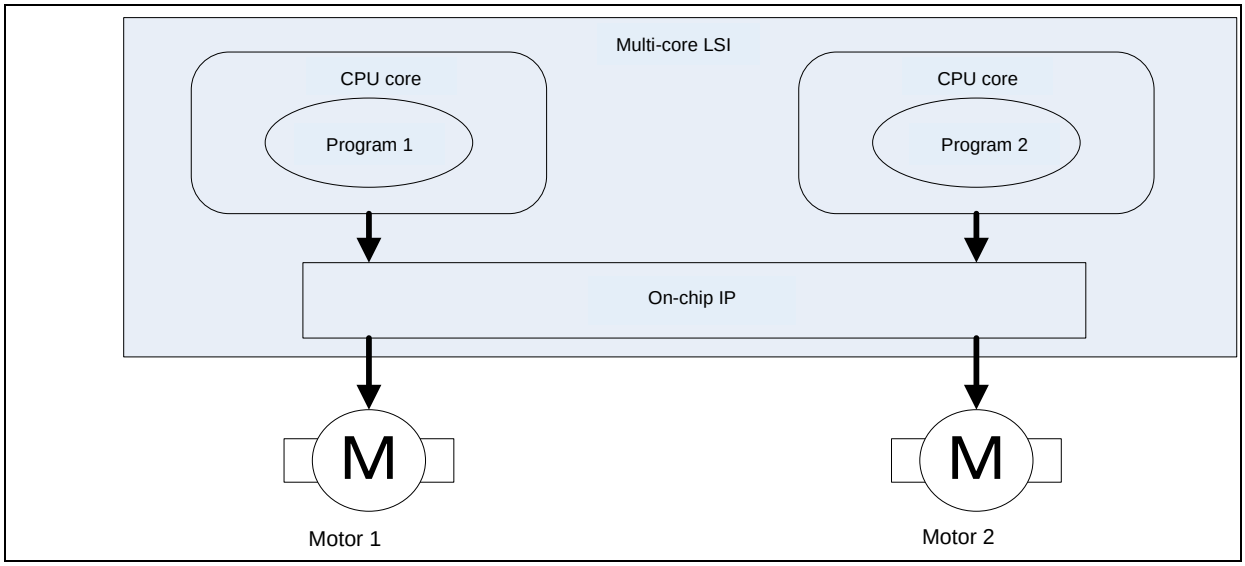

Fig 2-2. Schematic diagram of multi-core LSI（one multi-core LSI controls two motors）

Each CPU stores software independently since Code Flash area for storing programs can be provided separately for each CPU. Please refer to User's Manual of each LSI for more details.

2.2 Identifier

CPU and peripheral IP are connected by an internal bus. IP that has authority to access the internal bus is called “master” or “bus master”. This includes the CPUs, DMAs, and debuggers (on-chip debugging unit). The master has its own identifiers to allow internal bus to know which master is making the access. These identifiers include processor element identifier (PEID) and system protection identifier (SPID). PEID is used to identify which CPU core is performing a specific program. Meanwhile, SPID indicates the system protection number and access will be granted when there is a matching SPID.

The SPID register in U2Bx specifies the group which program is being executed, and thus enabling memory resources and peripheral devices to be assigned to the program. The value of SPID for CPU cores, DTS/sDMAC, DFP and MAU can be set individually in each configuration register. For the other masters, it can be set in SPID module. SPID module supervises all SPIDs except for some bus masters which have their own configuration. U2Bx also has a bus master identifier (BMID) to indicate the bus master.

Table 2-1 shows the bus master and identifier value of U2Bx which differs from C1x and C1M-Ax. Please refer to User’s Manual of each LSI for information on identifier for each master.

Table 2-1 Master and identifier of U2Bx

Bus Master	CPU0	CPU1	CPU2	CPU3	CPU4	CPU5	sDMAC, DTS, DFP, MAU	Others
PEID	B’00	B’01	B’10	B’11	B’100	B’101	Set individually in each configuration register	Set in SPID module
SPID	B’00	B’01	B’10	B’11	B’100	B’101		
BMID	B’00	B’01	B’10	B’11	B’100	B’101		

CAUTION

According to the value of the SPID, how operations are assigned to memory resources and peripheral devices is determined by the specifications of the product.

2.3 Self

One of the unique concepts of multi-core system is “self”, which reflects the resources of each CPU. “Self” is notated as “self area” in C1x and C1M-Ax User’s Manual. U2Bx also has INTC1 of SELF which reflects INTC1 of each CPU.

Fig 2-3 shows the conceptual diagram of U2Bx.

e.g. Accessing Local RAM (self) from CPU0: access Local RAM (CPU0)
Accessing Local RAM (self) from CPU1: access Local RAM (CPU1)
Accessing Local RAM (self) from CPU2: access Local RAM (CPU2)
Accessing Local RAM (self) from CPUn: access Local RAM (CPUn), (n: 3, 4,5)

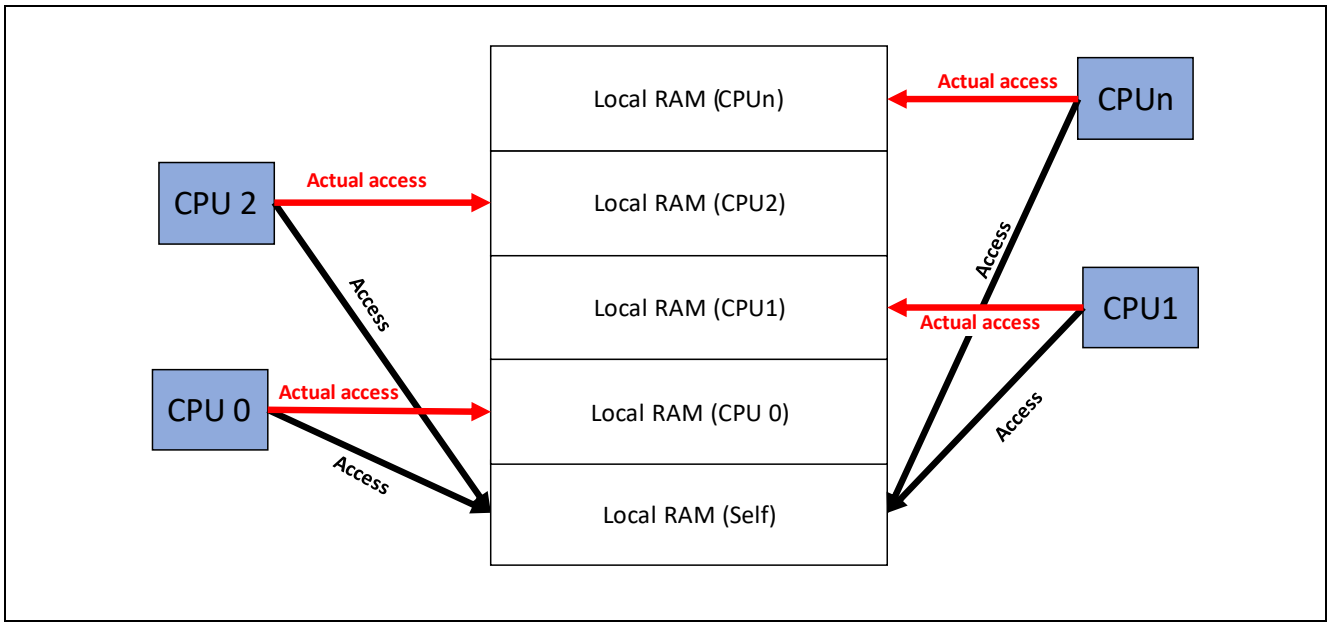

Fig 2-3. “Self” Conceptual Diagram (Local RAM)

By using self, it is possible to access only the Local RAM area of one’s own CPU without destroying the Local RAM area of the other CPU. Fig 2-4 shows the example on how to access self.

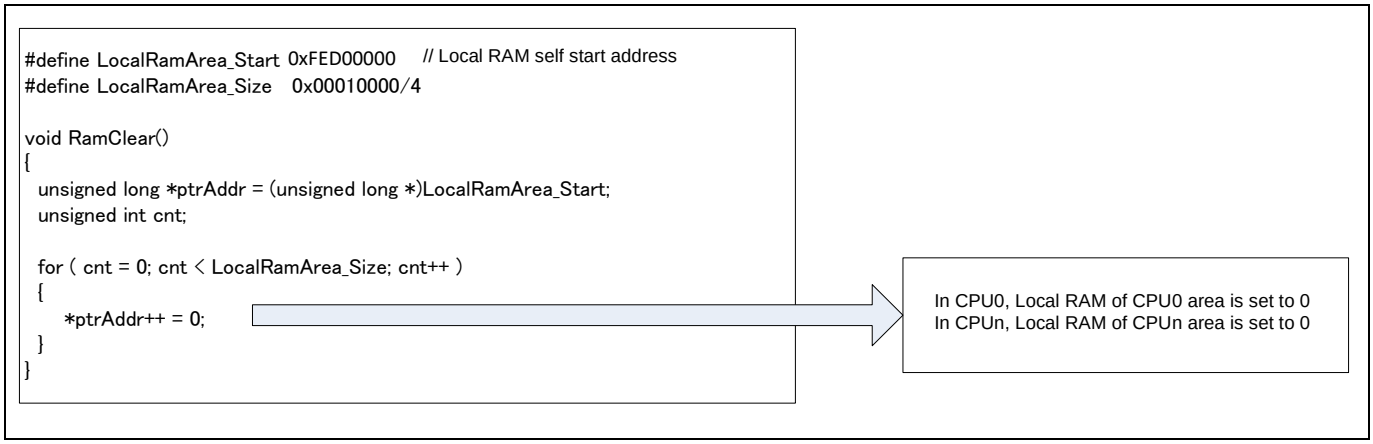

Fig 2-4. "Self” Access Example

Caution

Access to self is only possible from the CPU.
When accessing from sDMAc and DTS, specify the actual area (Local RAM CPU0 area etc.).

The on-chip I/O register (self) is the reflection of the internal I/O register of each CPU. Please refer to User’s Manual of each LSI for more details.

2.4 CPU Boot up

The procedure for CPU boot up can be divided into 3 types of methods.

- Method of branching from PEID (U2Bx, C1M-Ax, C1x)
- Method of boot configuration by each CPU (U2Bx)
- Method of boot for CPU0 from user boot area, and boot of other CPUs from user area (U2Bx)

Users are free to choose the best approach as required by their application.

2.4.1 Method of Branching from PEID

In normal mode, each CPU starts its operation according to its own RBASE (Reset Vector Base Address).

If there are multiple CPUs set with the same RBASE address, it is necessary to use PEID to determine the branching to each CPU during startup.

Example is as shown in Fig. 2-5.

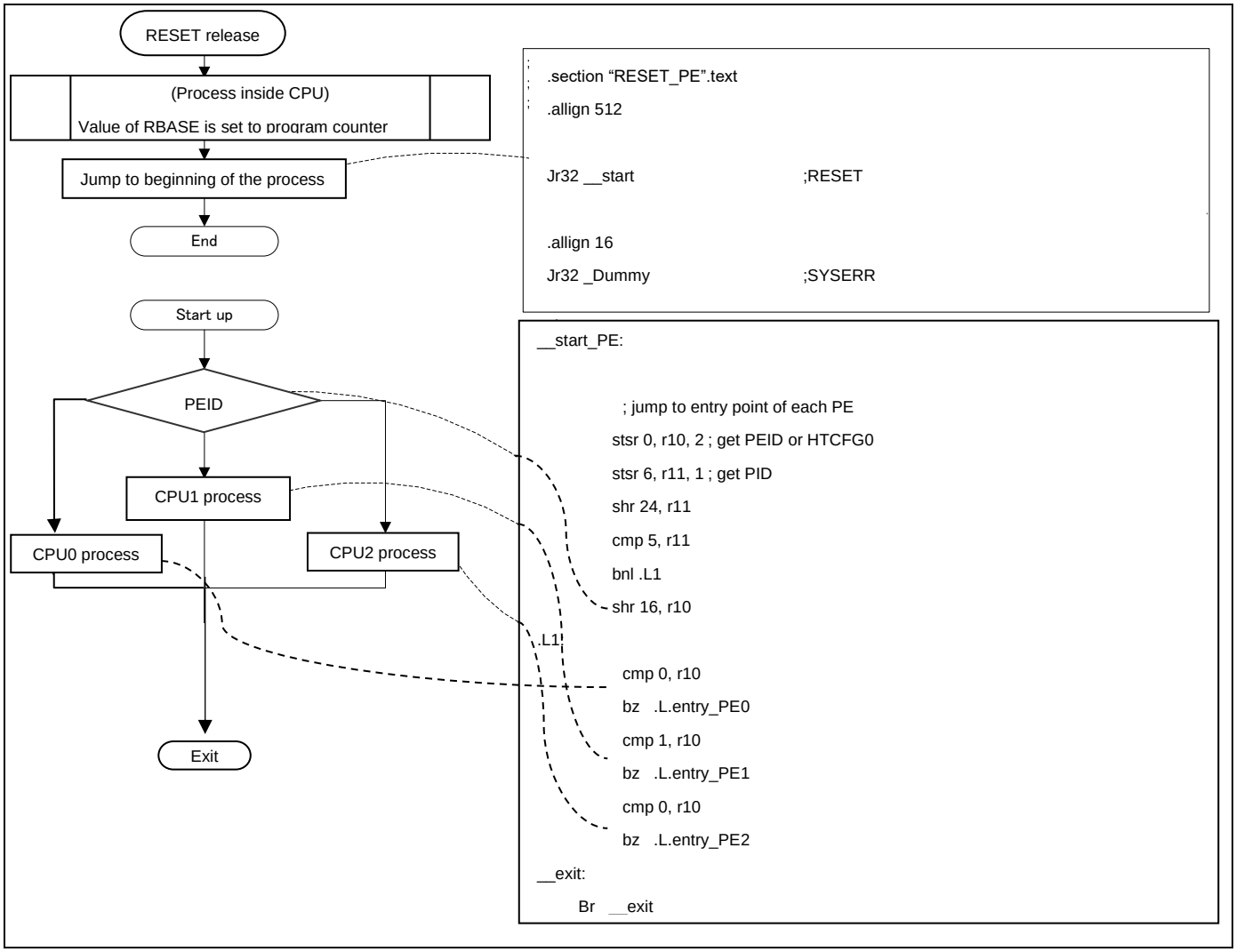

Fig 2-5. CPU Bootup (RH850/U2B6)

2.4.2 Method of Boot Configuration by Each CPU

In normal mode, each CPU starts its operation according to its own RBASE (Reset Vector Base Address).

The RBASE for each CPU can be set individually in U2Bx. In this case, the program of each CPU is separately stored in the memory and operates independently. Therefore, there is no need of PEID usage for branching.Example is as shown below.

The RBASE for each CPU is set to Reset Vector (PEn) of configuration setting area. RBASE cannot be set individually for each CPU in C1x and C1M-Ax, so this method cannot be used.

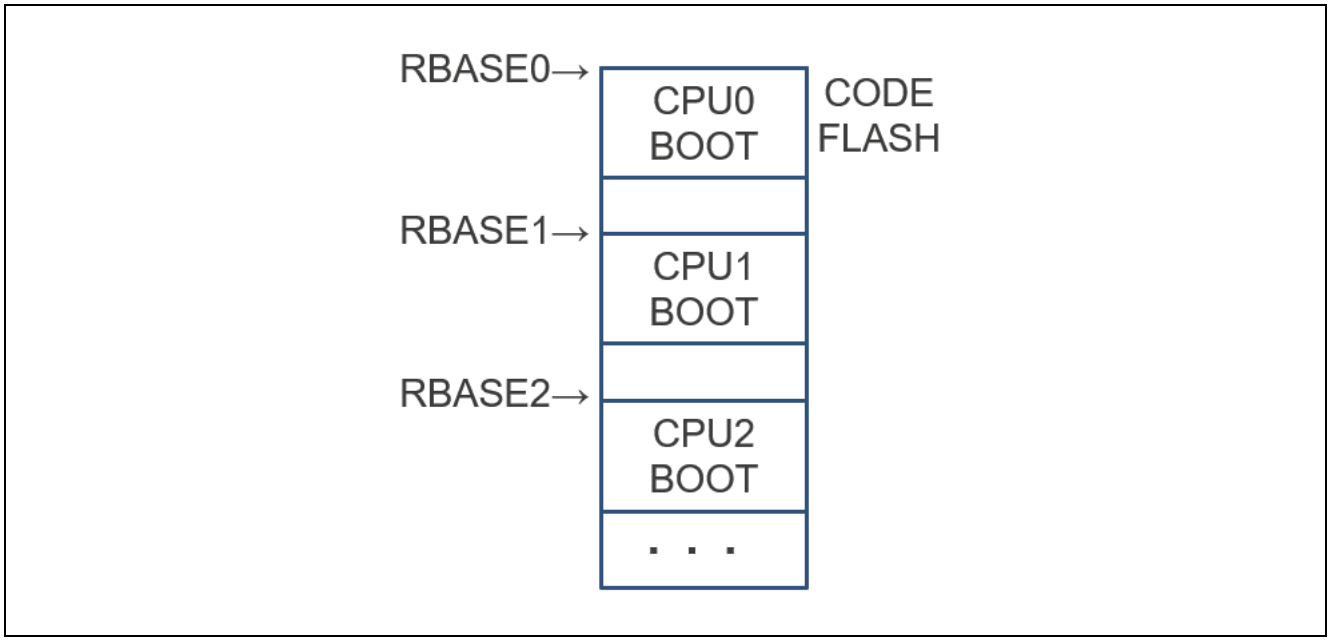

Fig 2-6. Boot Configuration by each CPU

2.4.3 Method of Boot for CPU0 From User Boot Area, and Boot of Other CPUs From User Area (U2Bx)

When booting in user boot mode, each CPU will boot up from the following area as shown in the table below.

Table 2-2. Boot Area

LSI	CPU	Boot up area
U2Bx	CPU0	User boot area
	CPU(1-5)	User area (same as Normal mode)
C1x, C1M-Ax	All CPUs	User boot area

In case of U2Bx, the program for CPU0 will start from a different area than the other CPUs. Programs other than CPU0 are started as described in 2.4.1 and 2.4.2.

2.4.4 CPU Boot Up Other Than CPU1 (Troubleshooting)

In RH850/U2Bx, startup of CPUs can be started selectively by asserting the corresponding bit in Boot Control register (BOOTCTRL).

In RH850/C1H, CPU1/CPU2 will be started simultaneously after reset. However, depending on the type of compiler, there are also cases where some CPUs except CPU1 may not operate unless CPU1 issues a start request. (CC-Cube, etc.)

In RH850/C1M-Ax, the startup mode for CPU1 and CPU2 can be selected from STARTUPPE bit of option byte 6 and BOOTCTRL register. Regardless of the startup mode of the CPUs, the SubCPU (PE3) in EMU3 will not be operating after release from a reset. It is started by software running on CPU1 (PE1) or CPU2 (PE2).

It is important to note that depending on the setting of the debugger, all CPUs will start running when the debugger is connected. CPU will be started up based on the order of the CPU. If the next CPU start request process is not included in the previous CPU process, it will end up in a situation where it works on the debugger, but it does not work on its own. Fig 2-7 shows the CPU startup order for U2Bx.

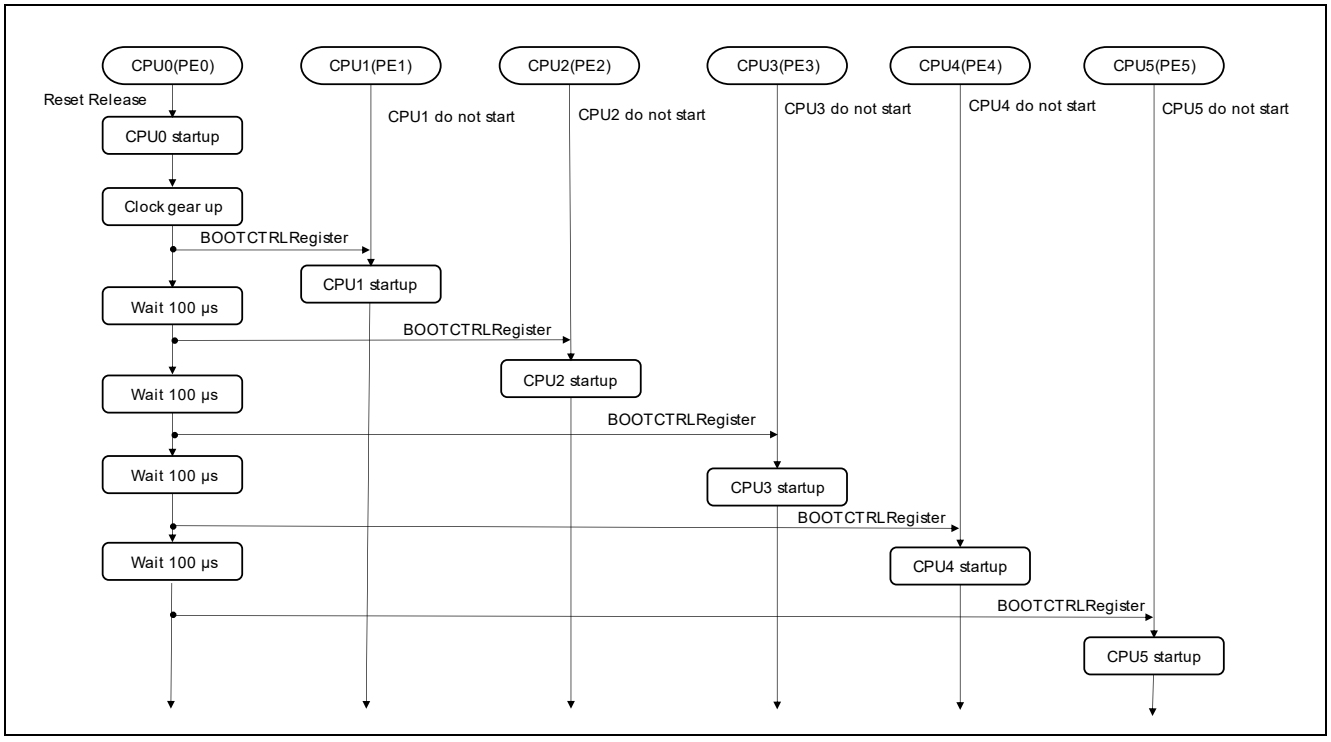

Fig 2-7. CPU Start Up Order

Please refer to the User's Manual of each LSI for the method of booting up other CPUs than CPU0 (U2Bx) and CPU1 (C1x&C1M-Ax).

In U2Bx, it is necessary to pay attention to the CPU startup order and startup interval to prevent excessive current flow during startup.

3. Tools Manual

This chapter describes how to create and debug multi-core projects using Renesas Electronics integrated development environment (CS+).

Please contact us for further information.

3.1 Creating a Multi-core Project

Renesas Electronics integrated development environment (CS+) supports multi-core and making it easier to create independent software for each CPU.

A multi-core project is consisted of “bootloader project” and “application project”. The application project is created for each CPU and then linked by the bootloader project as a single project. The idea of this project is as shown in the figure below.

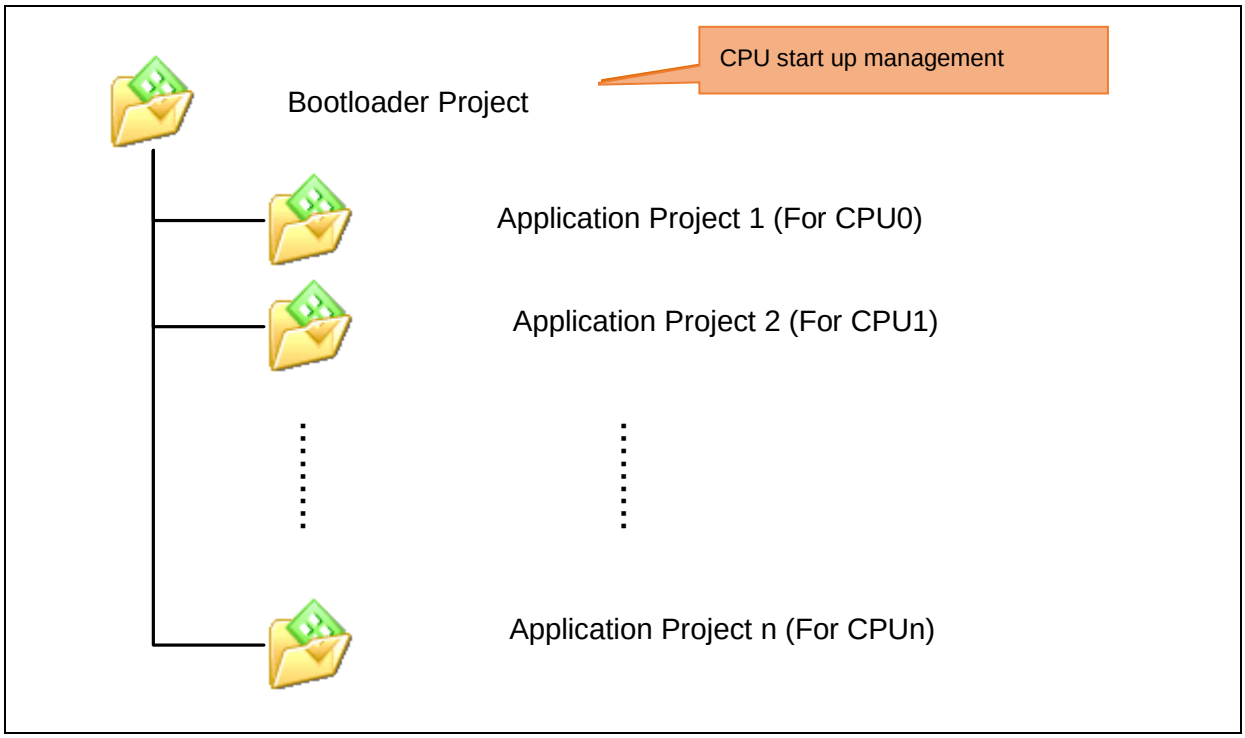

Fig 3-1 Multi-core Project Structure

The structure of the project in CS+ is as shown below.

- Main project：Bootloader project
- Subproject：Application projects（Number of CPU cores）

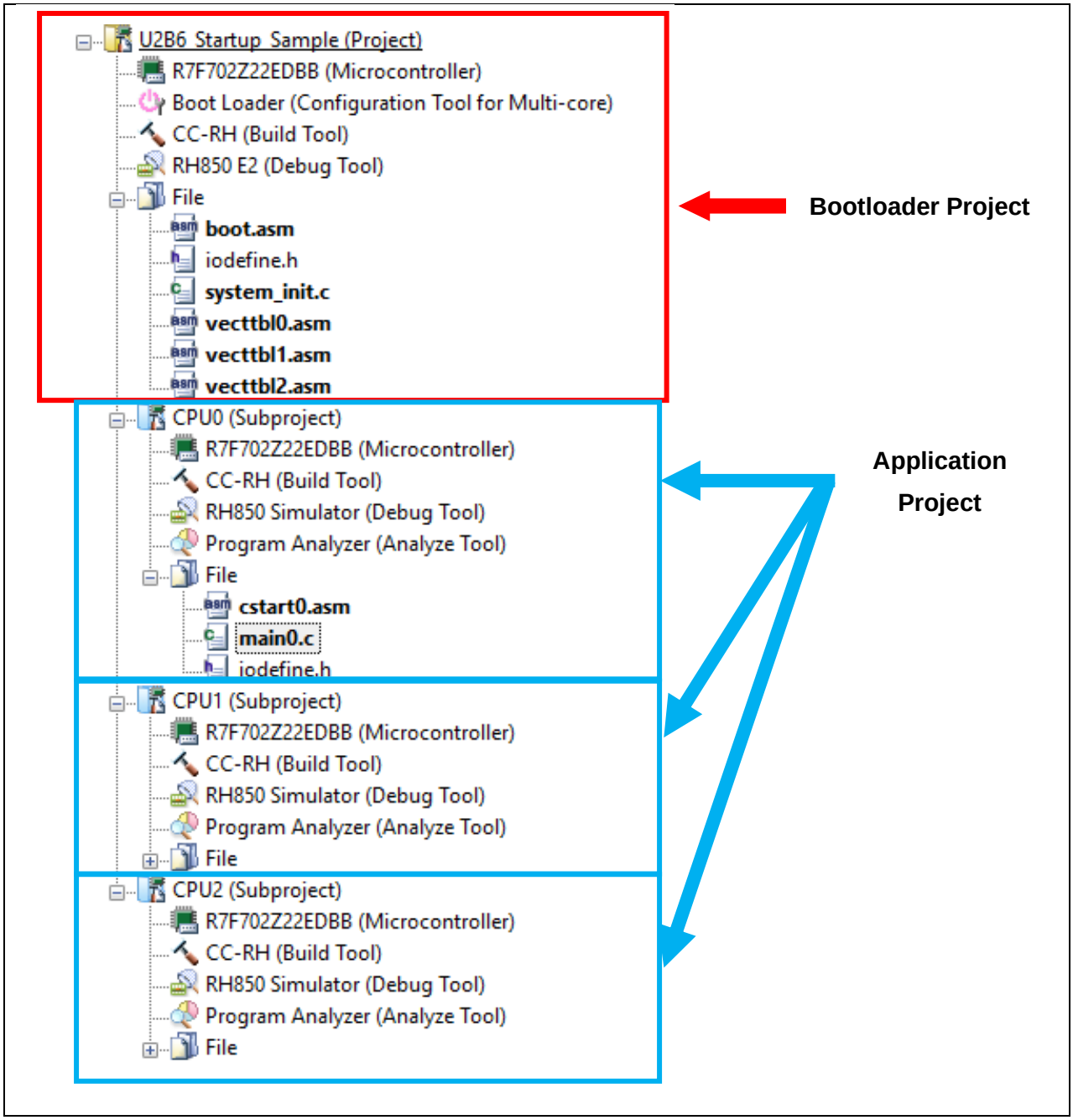

Fig 3-2. CS+ Project Tree

Please refer to CS+ User's Manual for information on how to combine a bootloader project with multiple application projects.

Each application project will be compiled (assembled) and linked, so it is created as a completely independent project. It can be imagined as building one library in one application project.

Therefore, even if a variable with the same name label exists in application project of different CPUs, it will not cause a link error. The software creator of each CPU can create software without worrying about double definitions in the other CPUs.

To be exact, it does not mean that multiple application projects are linked to the bootloader project, but it is a merging of generated executable files that are linked by multiple application projects. Hence, the address of section definitions should not overlap in each application project.

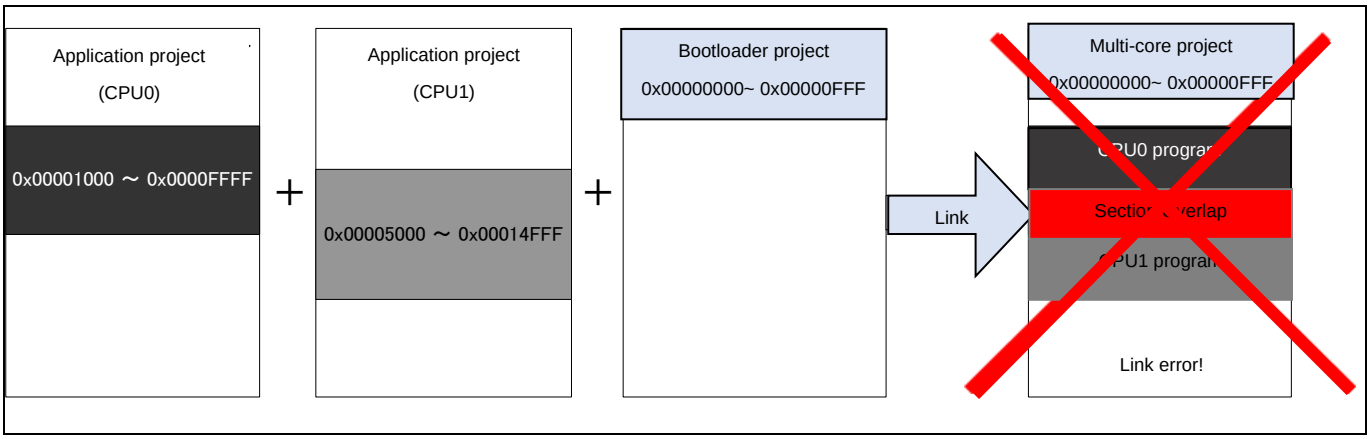

Fig 3-3. Precaution on Creating Multi-core Project (Link Error Occurred)

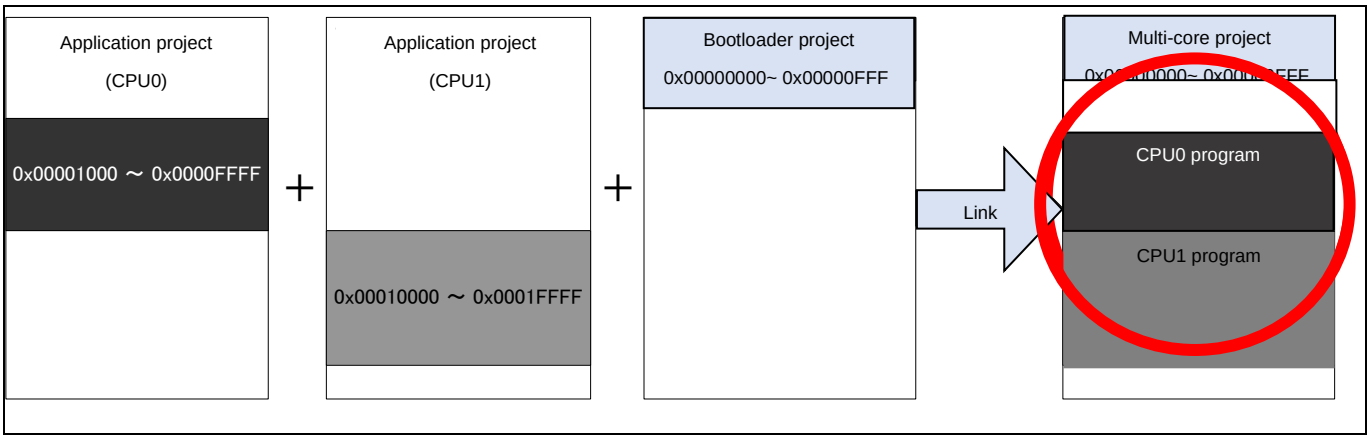

Fig 3-4. Precaution on Creating Multi-core Project (Successfully Completed)

In Fig 3-3, the section area of CPU0 and CPU1 application projects are overlapping at 0x00005000~0x0000FFFF, causing a link error. It is necessary to define the section so that the address of the bootloader project and each application project do not overlap, as shown in Fig 3-4.

Section definitions can be set in CS+ through CC-RH(Build Tool), in「Link Options」tab and under [section] – [Section start address] of bootloader project and each application projects.

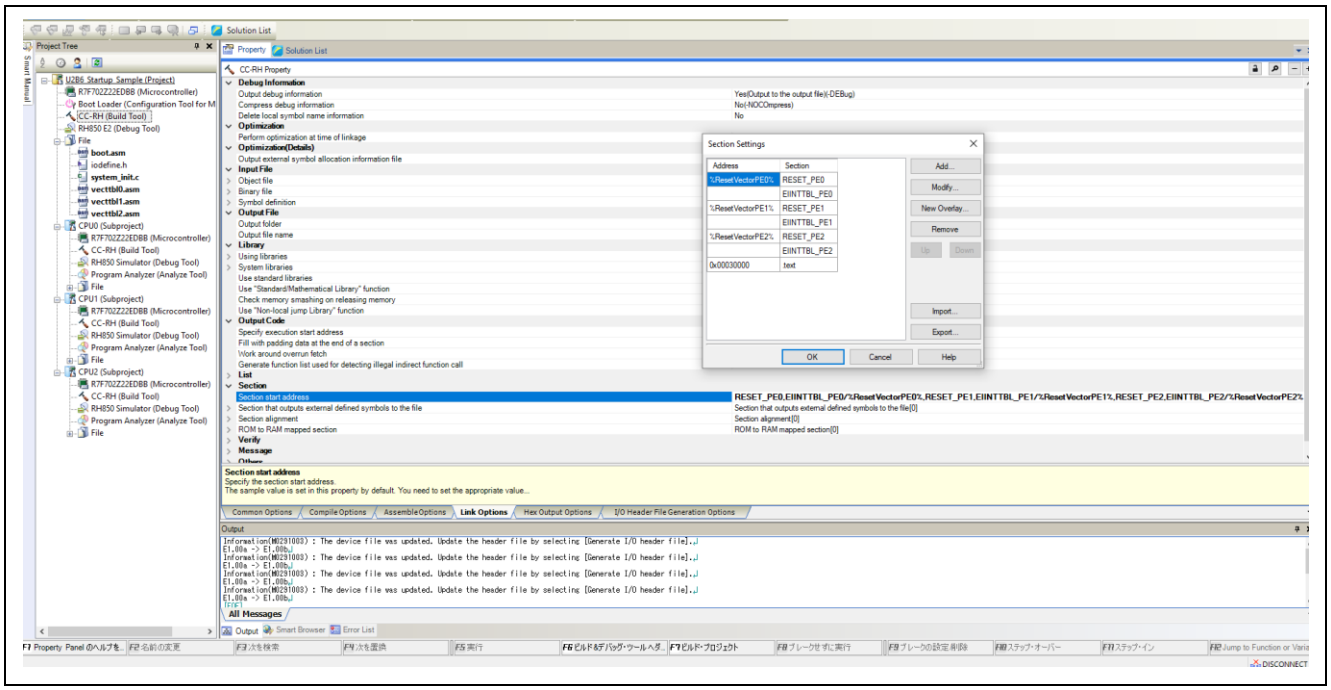

Fig 3-5. Section Definition

MEMO

It is also possible to create “shared functions” that can be called from multiple application projects and “shared variables” that can be accessed from multiple application projects.
Please refer to the CS+ User’s Manual for more details.

3.2 Debugging

In CS+, it is possible to switch which CPU to debug through CPU switching GUI. Open the debug manager window or use the CPU switching GUI at the bottom of the CS+ screen to switch the CPU.

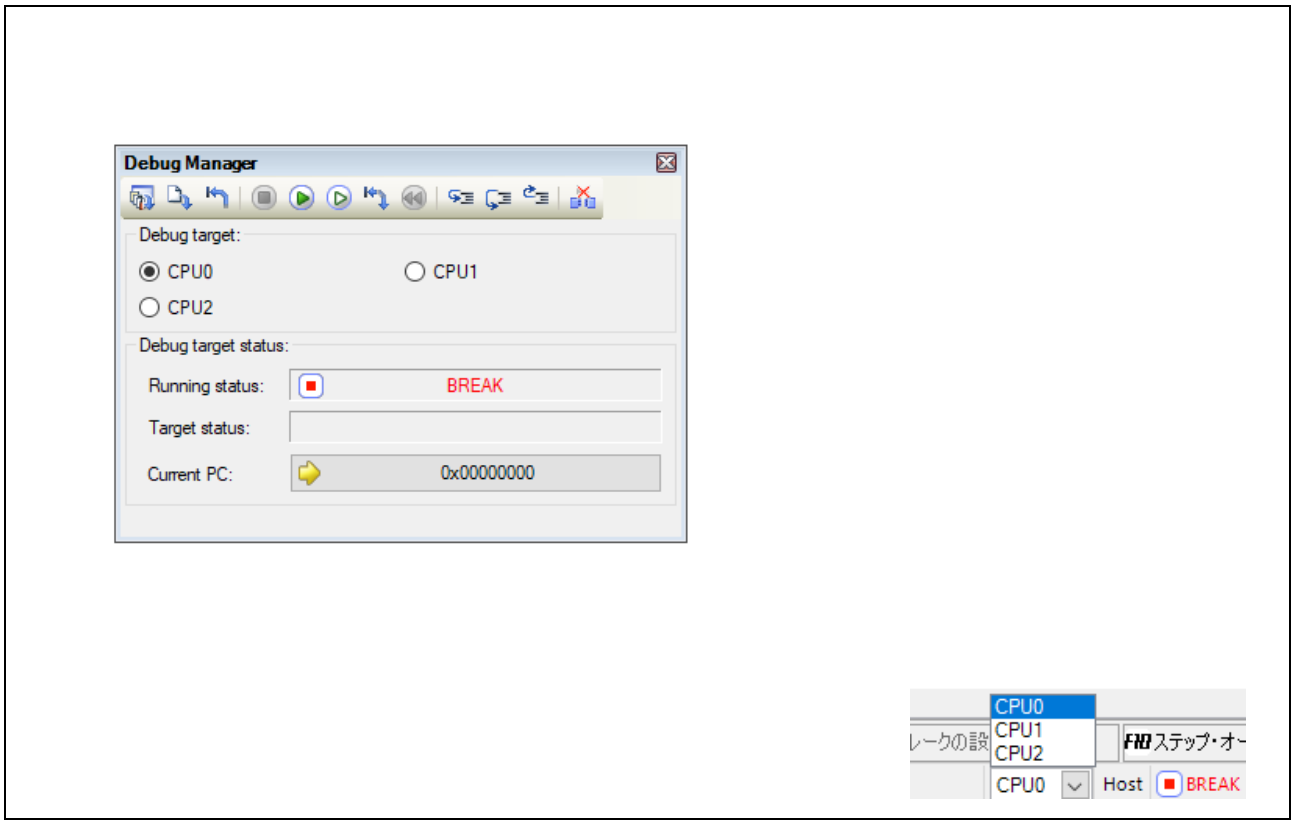

Fig 3-6. CPU Debug Switching

Please switch to “CPU1” when debugging CPU1 software, and switch to “CPU2” when debugging CPU2 software.

Caution

Breakpoints are intentional stopping set for address.
If a breakpoint set during CPU0 debugging is not removed, the breakpoint will be applied when CPU0 executes the relevant address during other CPUs debugging.
It is important to note on things such as removing the breakpoints set in other CPUs before debugging the targeted CPU.

4. Access to Peripheral IP

This chapter explains the precautions related to the access to peripheral IP in multi-core system. As long as these precautions are followed, the rest is basically the same as for single-core LSIs.

4.1 Inter-CPU Communication

Communication protocol such as serial I/F is used to exchange information between LSIs when using a system with two LSIs. As for RH850/U2Bx, C1M-Ax and C1x, information can be exchanged between CPUs using the stated methods below. In U2Bx, inter-CPU Communication is consisted of Inter-Processor Interrupt (IPIR), Barrier-Synchronization (BARR), and Time Protection Timer (TPTM). On the other hand, C1x and C1M-Ax only possess the Inter-Processor Interrupt (IPIR), and the exclusive control register(MEV) which does not exist in U2Bx. In addition, data exchange is made in Global RAM for C1x and C1M-Ax, and Cluster RAM for U2Bx. The detail of each register is as shown below.

4.1.1 Inter-processor Interrupts (IPIR)

CPU has the IPIR register as its own peripheral device. Setting the IPIR register enables an EI level interrupt request from a CPU to another CPU. For instance, interrupt request can be issued from CPU0 to CPU1 or CPU1 to CPU0. This IPIR register supports the inter-PE interrupt function of 4 channels.

U2Bx has more registers under the IPIR module which includes registers in self region. The registers in the self region do not physically exist but are there as virtual registers. When PE accesses the self register, PE can access the corresponding register for each PE. For example, when PE1 accesses IPI0REQS(IPIR0 Inter-PE interrupt request self-register), PE1 can also access IPI0REQ1(IPIR0 Inter-PE interrupt request register).

In C1x and C1M-Ax, the interrupt for specific PEs (including own PE) can be requested by manipulating bits corresponding to respective PEs in IPIR_CHn register. However, in case of C1x, inter-CPU functions can only be found in C1H.

Please refer to each LSI User’s manual for more details on IPIR.

4.1.2 Barrier-Synchronization (BARR)

Barrier-synchronization registers are used to set up a barrier that prevent the next process from starting before the previous process is completed. This control is required in parallel processing where there may be a need to make the cores wait until the data required for the next processing is ready, at which point the processing can proceed to the next processing.

4.1.3 Time Protection Timer (TPTM)

Time Protection Timer (TPTM) is sets of timers dedicated for CPU. It enables CPU to achieve high performance and gives timing protection. Each CPU will be assigned with one timer set which consisted of 2 interval timers, 1 Free-run timer and 2 Up timers.

4.1.4 List of Interrupt Requests for Inter-CPU Communication in U2Bx

The interrupt requests in U2Bx are listed in the following table. BARR has no interrupt request. “-“means the value depends on the specification of product.

Table 4-1. Interrupt Request (1/2)

Interrupt Symbol Name	Unit Interrupt Signal	Description	Interrupt Number	sDMA Trigger Number	DTS Trigger Number
INTIPIRO	ipir_int_ch0	IPIR CH0 interrupt	EIINT0	-	-
INTIPIR1	ipir_int_ch1	IPIR CH1 interrupt	EIINT1	-	-
INTIPIR2	ipir_int_ch2	IPIR CH2 interrupt	EIINT2	-	-
INTIPIR3	ipir_int_ch3	IPIR CH3 interrupt	EIINT3	-	-
FEINT or INTTPTM	TPTM_IRQ	TPTM interval interrupt	FEINT or EIINT31	-	-
INTTPTMU00	INTTPMU0	TPTM up timer interrupt for PE0 with comparison value 0	EIINT413	-	Group 1-116
INTTPTMU01	INTTPMU0	TPTM up timer interrupt for PE0 with comparison value 1	EIINT414	-	Group 1-117
INTTPTMU02	INTTPMU0	TPTM up timer interrupt for PE0 with comparison value 2	EIINT415	-	-
INTTPTMU03	INTTPMU0	TPTM up timer interrupt for PE0 with comparison value 3	EIINT416	-	-
INTTPTMU10	INTTPMU1	TPTM up timer interrupt for PE1 with comparison value 0	EIINT417	-	Group 1-118
INTTPTMU11	INTTPMU1	TPTM up timer interrupt for PE1 with comparison value 1	EIINT418	-	Group 1-119
INTTPTMU12	INTTPMU1	TPTM up timer interrupt for PE1 with comparison value 2	EIINT419	-	-
INTTPTMU13	INTTPMU1	TPTM up timer interrupt for PE1 with comparison value 3	EIINT820	-	-
INTTPTMU20	INTTPMU2	TPTM up timer interrupt for PE2 with comparison value 0	EIINT843	-	Group 1-120
INTTPTMU21	INTTPMU2	TPTM up timer interrupt for PE2 with comparison value 1	EIINT844	-	Group 1-121
INTTPTMU22	INTTPMUm2[2]	TPTM up timer interrupt for PE2 with comparison value 2	EIINT845	-	-
INTTPTMU23	INTTPMUm2[3]	v TPTM up timer interrupt for PE2 with comparison value 3	EIINT846	-	-
INTTPTMU30	INTTPMUm3[0]	TPTM up timer interrupt for PE3 with comparison value 0	EIINT847	-	Group 1-122

Table 4-2. Interrupt Request (2/2)

Interrupt Symbol Name	Unit Interrupt Signal	Description	Interrupt Number	sDMA Trigger Number	DTS Trigger Number
INTTPTMU31	INTTPMUm3[1]	TPTM up timer interrupt for PE3 with comparison value 1	EIINT848	-	Group 1-123
INTTPTMU32	INTTPMUm3[2]	TPTM up timer interrupt for PE3 with comparison value 2	EIINT849	-	-
INTTPTMU33	INTTPMUm3[3]	TPTM up timer interrupt for PE3 with comparison value 3	EIINT850	-	-
INTTPTMU40	INTTPMUm4[0]	TPTM up timer interrupt for PE4 with comparison value 0	EIINT973	-	Group 1-124
INTTPTMU41	INTTPMUm4[1]	TPTM up timer interrupt for PE4 with comparison value 1	EIINT974	-	Group 1-125
INTTPTMU42	INTTPMUm4[2]	TPTM up timer interrupt for PE4 with comparison value 2	EIINT975	-	-
INTTPTMU43	INTTPMUm4[3]	TPTM up timer interrupt for PE4 with comparison value 3	EIINT976	-	-
INTTPTMU50	INTTPMUm5[0]	TPTM up timer interrupt for PE5 with comparison value 0	EIINT977	-	Group 1-126
INTTPTMU51	INTTPMUm5[1]	TPTM up timer interrupt for PE5 with comparison value 1	EIINT978	-	Group 1-127
INTTPTMU52	INTTPMUm5[2]	TPTM up timer interrupt for PE5 with comparison value 2	EIINT979	-	-
INTTPTMU53	INTTPMUm5[3]	TPTM up timer interrupt for PE5 with comparison value 3	EIINT980	-	-

4.1.5 Cluster RAM (U2Bx), Global RAM、MEV (C1x & C1M-Ax),

MEV is not included in U2BX. Global RAM (GRAM) in C1x and C1M-Ax is equivalent to Cluster RAM (CRAM) in U2Bx.

The local RAM, global RAM, and exclusive control register (MEV) are available as a resource for exclusive control in C1x and C1M-Ax. Inter-CPU communication can be made without accessing RAM through the usage of MEV register. The exclusive control register (MEV) can be accessed without waiting because they have an independent path from each CPU. However, there is waiting time when the same MEV register is used. Please use this register for small data transfer as 32-bits register provided 32 lines. The boot program also uses the MEV register to synchronize CPU1/CPU2. For large data exchange, please use global RAM instead.

Data exchanged between CPU can be made through Cluster RAM and global RAM as they are memories shared by all CPUs.

4.2 Interrupt Setting

Interrupt Controller (INTC) determines priority of interrupt sources and controls interrupt requests to the CPU. INTC consists of INTC1 and INTC2. All CPUs have their own INTC1 which control high-speed interrupts. Meanwhile, INTC2 is a common interrupt controller that shared by all CPUs and in charge of controlling low-speed interrupts. The overview of the interrupt is as shown in Fig. 4-1.

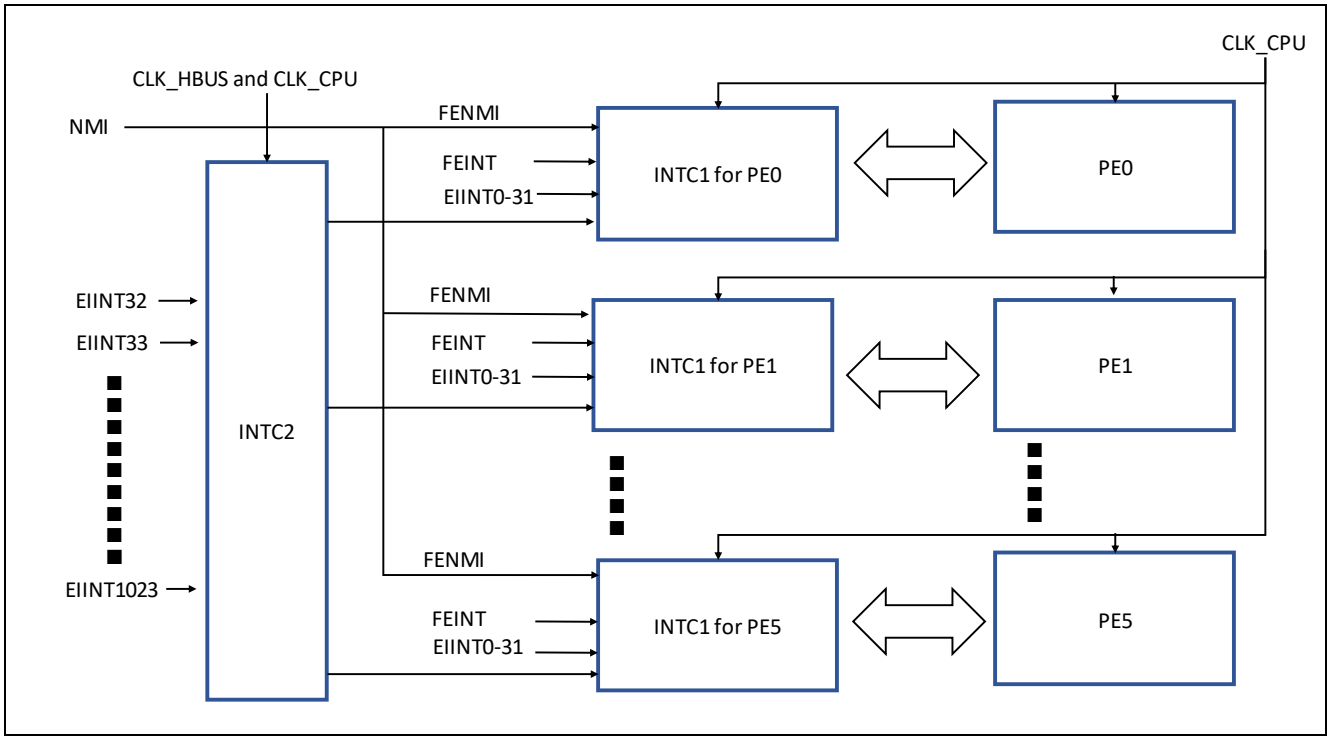

Fig 4-1. Block Diagram of Interrupt Units

4.2.1 Binding setting

In a multi-core system, peripheral IP interrupts are bound to the CPU. Each interrupt can only be generated on the bound CPU. It is necessary to specify which CPU to be notified when using INTC. The selection of which CPU is to be notified can be made by setting EI level Interrupt bind register (EIBDn). Just like as shown in Fig.4-1, EIINT0-31 are used to set the binding setting for INTC1, and EIINT 32-1023 are used for INTC2. Differ from U2Bx, only INTC2 has interrupt binding function in C1x and C1M-Ax.

Although this register contains several interrupt sources, the interrupt sources connected to INTC1 cannot be changed.

Caution

In C1x and C1M-Ax ,interrupt-related registers (EICn, IMRn, EIBDn, etc.) can only be written by the CPU bound to EIBDn register. Changing the corresponding EIBDn register during the processing of an EIINT request is prohibited.

4.2.2 Exception Handler Address

In a multi-core system, exception handler is set for each CPU.

There are two methods to determine the exception handler address in resets, exceptions, and interrupts, which are the direct vector method and table reference method. In the direct vector method, the exception handler address is determined by adding the offset address to the base address indicated by the RBASE or EBASE register. If the table reference method is selected, execution can branch to the address indicated by the exception handler table allocated in the memory. The selection of these methods is as shown in Fig 4-2.

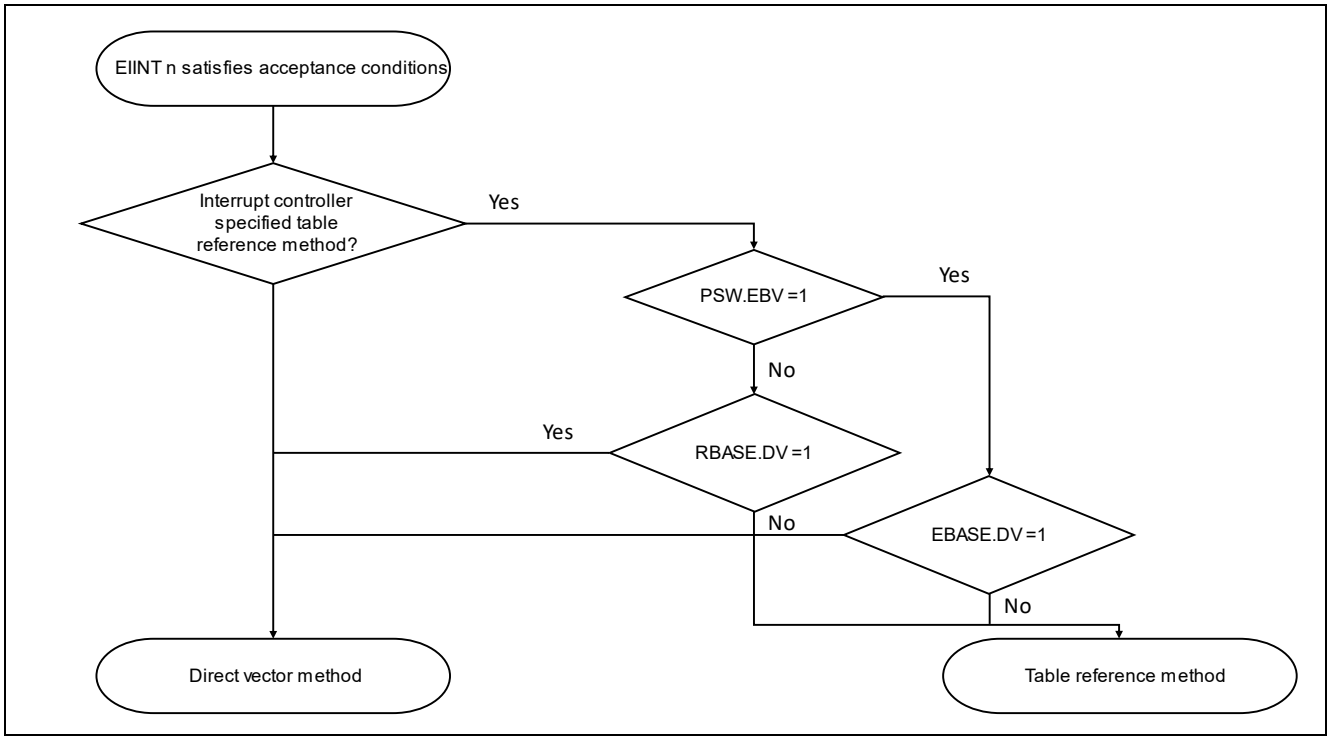

Fig 4-2. Selection of Exception Handler Address Method

4.2.3 Broadcast Notification Interrupt (U2Bx only)

INTC2 in U2Bx has a function of broadcast notification interrupt. In this function, EIINTn interrupt request is forwarded to EIINT4-7 of each CPU's INTC1 as a broadcast notification 0-3 without priority judgement. Interrupt detection type of EIINTn must be edge detection when using this function. This function does not exist in C1x and C1M-Ax.

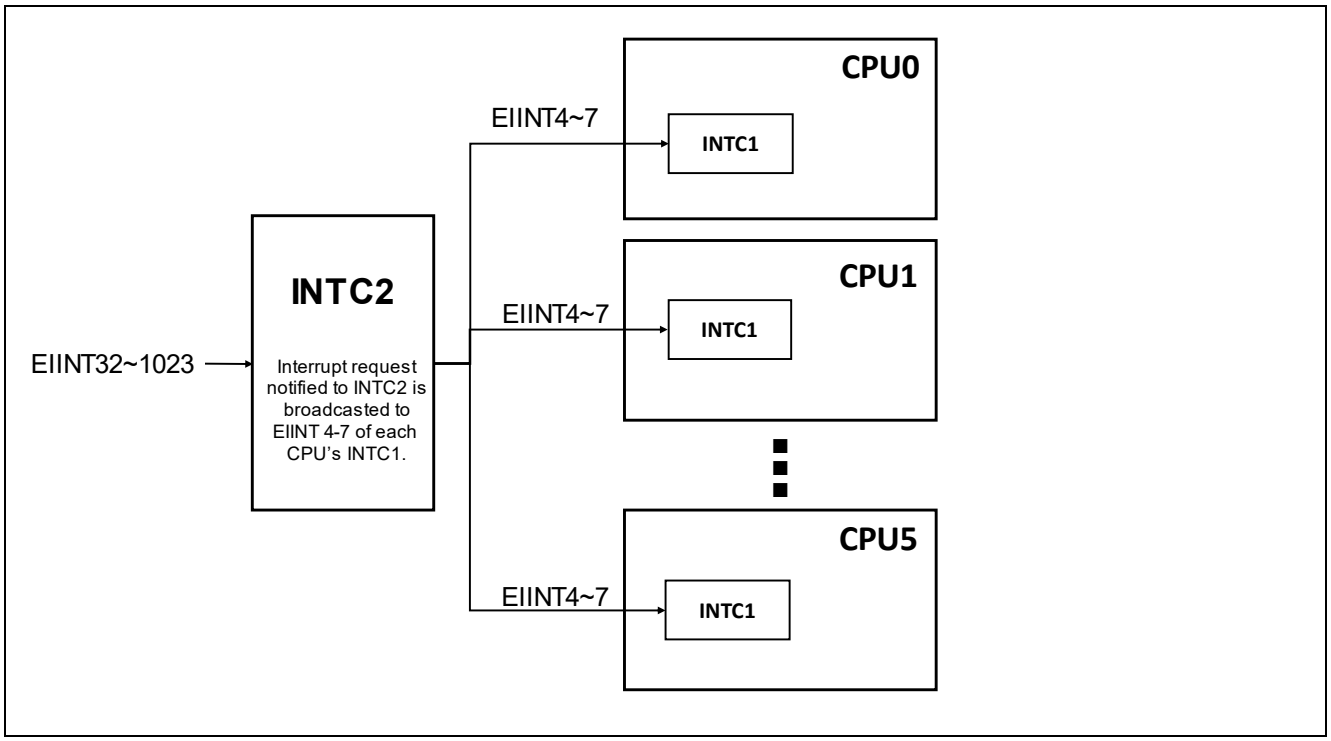

Fig 4-3. Schematic Diagram of Notification Interrupt

4.3 Reliability Functions / Memory Protection

As mentioned before, in multi-core system, internal peripheral IP is shared by multiple CPUs. Therefore, the possibility that the value set in peripheral IP is overwritten by the other CPU due to factor such as software failure is higher compared with single-core LSI with multiple systems.

In order to protect the peripheral IP from such events, several reliability functions and memory protection functions had been provided. The overview of memory protection architecture is as shown in the figure below.

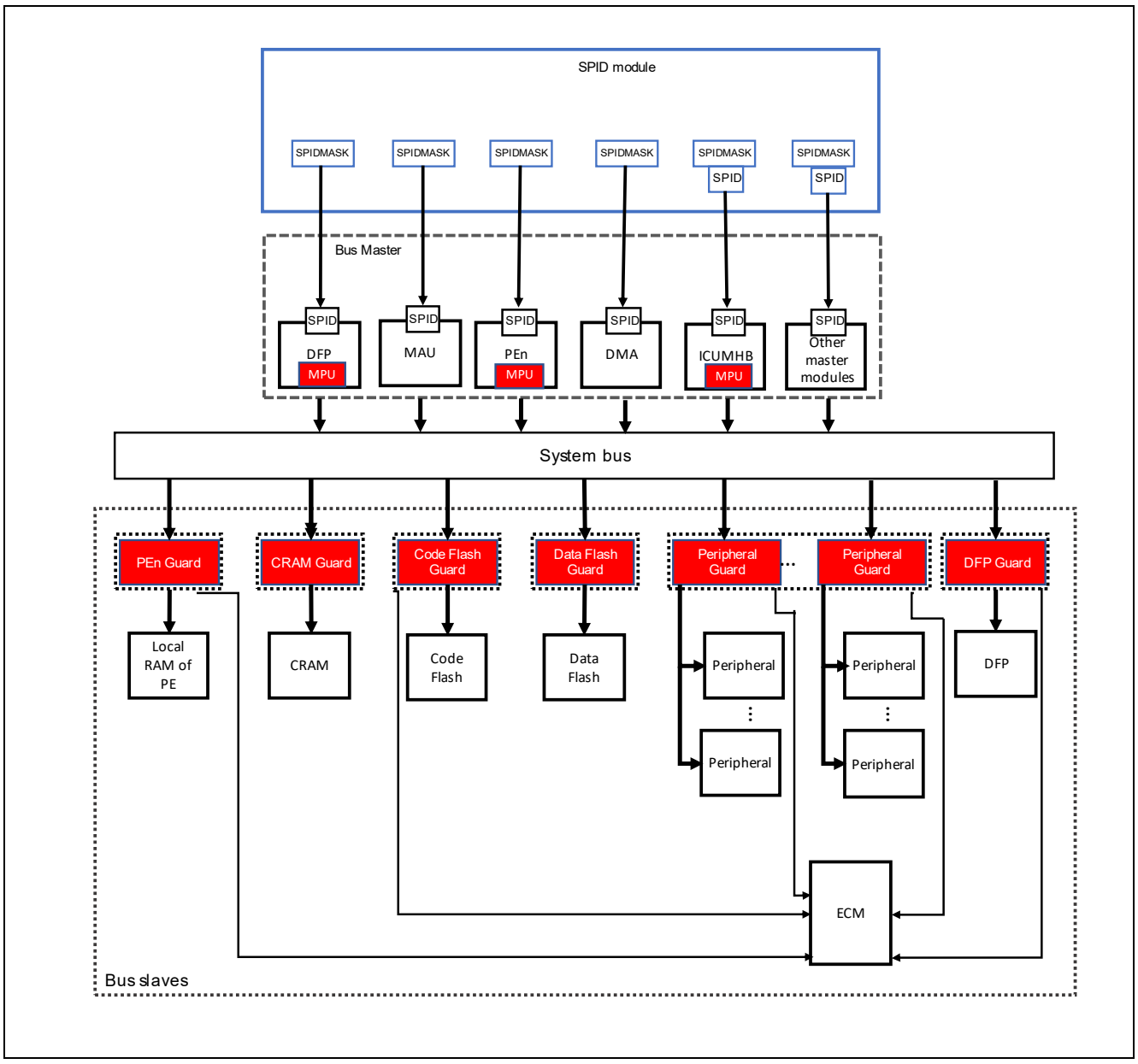

Fig 4-4. Memory Protection Architecture

4.3.1 PE Guard Function (PEG)

The PEG is a constituent of the Slave Guard function and blocks unauthorized accesses to the resources in the PE (local RAM) from the external bus master.

It is not recommended but if it is necessary to access the Local RAM of the other CPUs for some reason, the protection by PEG need to be removed.

Caution

PEG protection is also applied when accessing from DMA/sDMAC/DTS.
It is necessary to remove PEG protection according to SPID setting value of the DMA/sDMAC/DTS

4.3.2 Peripheral Bus Guard（PBG）

The control registers in the peripheral circuits and memories are protected against illegal accesses.

IPs that are protected by PBG function protect the register contents by denying access from anyone other than the master that has authority to access them.

In the post-reset state, some IPs are prohibited to be accessed except from CPU0(U2Bx) / CPU1(c1x&C1M-Ax). Hence, it is necessary to set the PBG function when accessing from other CPUs.

Please refer to the User's Manual of each LSI for the information on IPs covered by PBG functions.

4.3.3 CRG and GRG

Shared memory by all CPUs is known as Cluster RAM (CRAM) in U2Bx and global RAM (GRAM) in C1x and C1M-Ax. Cluster RAM guard (CRG) and global RAM guard (GRG) each protect the CRAM and GRAM from unpermitted access. Only bus master with permission is allows to access the content of CRAM and GRAM.

4.3.4 Other reliability functions

There are other reliability functions which is not directly related to multi-core system such as INTC2 Guard, H-bus Guard (HBG), etc. Please refer to User's Manual of each LSI for more details.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	2023.6.13		First edition
1.10	2023.11.30		Format correction

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.