Renesas RA Family

Third-Party Program Protection

Introduction

This application note provides information on how to protect the Third-Party Program (Software IP) contents of the RA0E1.

As used here, the term “third-party program” refers to valuable software IP supplied in formats such as libraries or binary format. The security functions of the RA0E1 can be employed to prevent unauthorized usage of such software IP.

Third-party program protection makes use of the following functions of the RA0E1.

- Memory Protection (Flash Read Protection) – Implements memory protection functionality to prevent unauthorized access to the software IP program code stored in flash memory. Guidelines are provided on configuring and utilizing the memory protection feature of the RA0E1 MCU to safeguard software IP stored in flash memory.
- Startup Control with Unique ID – Enables a startup control mechanism that verifies an embedded ID in flash memory during software IP launch. Instructions are provided on the implementation of startup control to verify a Unique ID during program launch, ensuring authenticity and integrity.

Contents

1. Overview

1.1 About This Application Note

This application note describes the third-party program (software IP) protection that can be implemented using the security functions of the RA0E1.

1.2 Third-Party Program (Software IP) Protection Requirement

Software IP is subject to issues such as the following:

- When software IP is supplied in binary format, it can be duplicated, and multiple copies can be used by unauthorized users.
- When the software IP has been programmed to an MCU supplied as a finished product, it can be read from the flash memory, analyzed, and re-used in unauthorized products.

1.3 Possible Protection Countermeasures

Two examples of countermeasures that can be implemented on the RA0E1 to protect software IP are described below.

- Memory Protection (Flash Read Protection)

  This approach implements memory protection functionality to prevent unauthorized access to the software IP program code stored in flash memory. This protection involves defining specific regions of the code flash as read-prohibited, enhancing the security of the IP.

  Refer to Section 2.1, Memory Protection (Flash Read Protection), for more information.

- Startup Control

  This approach checks an ID programmed in the flash memory when the software IP is launched. This approach is suitable for cases where the developer supplies the software IP in binary format.

  Refer to Section 2.2 Startup Control (Link to Unique ID) for more information.

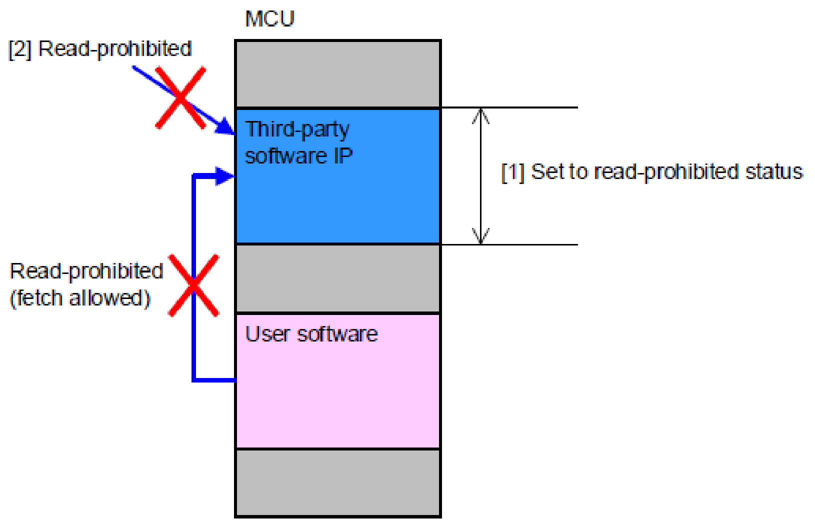
2. Security Functions of the RA0E1

The RA0E1 is provided with the security functions listed below. These functions can be used individually or in combination to protect the software IP from unauthorized use.

- Memory Protection (Flash Read Protection)
- Startup Control (Link to Unique ID)

2.1 Memory Protection (Flash Read Protection)

Flash read protection is a function that prevents specified areas of the code flash from being read by the CPU, DTC, and Debugger. This function can be used to prohibit the reading of software IP programmed in the code flash memory. Areas that are set to read-prohibited status can only be accessed with instruction fetches by the MCU. Note that programs running from a read-prohibited area cannot themselves read the read-prohibited data. Therefore, it is necessary that all data used by a program run from a read-prohibited area, which must be located in an area that is not protected.



[1] The blocks in which the software IP is stored are designated as a read-prohibited area.
[2] The software IP can be run, but user software cannot read the software IP program code.
It also cannot be read using an external flash writer or the like.

Figure 2.1. Flash Read Protection

Features: Prevents analysis or copying of the software IP set to read-prohibited status. To prevent unauthorized use or re-use of the software IP by the recipient, it is necessary to supply CPUs with the software IP pre-programmed in a read-prohibited area of the flash memory.

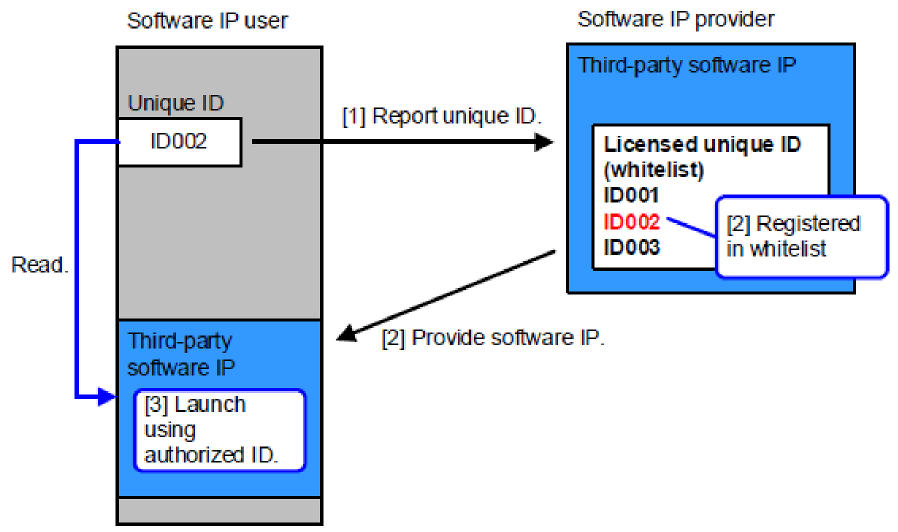
For information on how to set a memory area to read-prohibited status, refer to Section 4, **Memory Protection Settings Methods**.

2.2 Startup Control (Link to Unique ID)

The Unique ID is a value specific to the individual device that is stored in the flash memory at the time the MCU is manufactured.

By registering Unique IDs within a software program, it is possible to limit individuals (MCUs) authorized to run it.

In cases where the software license is dependent on the number of copies of the product, it is possible to maintain a list of licensed Unique IDs within the software, thereby ensuring that it can only be run on products with licensed Unique IDs.



[1] The software IP user reads the unique ID and reports it to the software IP provider.
[2] The software IP provider records the reported unique ID in a whitelist within the software IP and provides the software IP to the user.
[3] When the unique ID matches, the software IP runs.

Figure 2.2. Startup Control (Link to Unique ID)

Features: Prevents unauthorized use in cases where it is not possible to prohibit copying of software IP. If the software IP is copied to another MCU with a different and unauthorized Unique ID, it will not run.

The 128-bit Unique ID can be read directly with the FSP API **R_BSP_UniqueIdGet().**

```
typedef struct st_bsp_unique_id
{
    union
    {
        uint32_t unique_id_words[4];
        uint8_t  unique_id_bytes[16];
    };
} bsp_unique_id_t;

uint32_t mcu_id[4];
volatile const bsp_unique_id_t* unique_id = R_BSP_UniqueIdGet();
mcu_id[0] = unique_id->unique_id_words[0];
mcu_id[1] = unique_id->unique_id_words[1];
mcu_id[2] = unique_id->unique_id_words[2];
mcu_id[3] = unique_id->unique_id_words[3];
```

For more information on **R_BSP_UniqueIdGet()**, refer to FSP User's Manual (R11UM0155).

3. Use Cases for Third-Party Program (Software IP) Protection

The Unique ID is used as an example to demonstrate a use case for third-party program (software IP) protection. A second example is shown where access protection can be used to protect user application programs as well as third-party programs (software IP).

3.1 Third-Party Program Protection

Protects third-party software IP in cases where a third party undertakes the entire software development.

Prevents unauthorized use of software IP by general users and set makers.

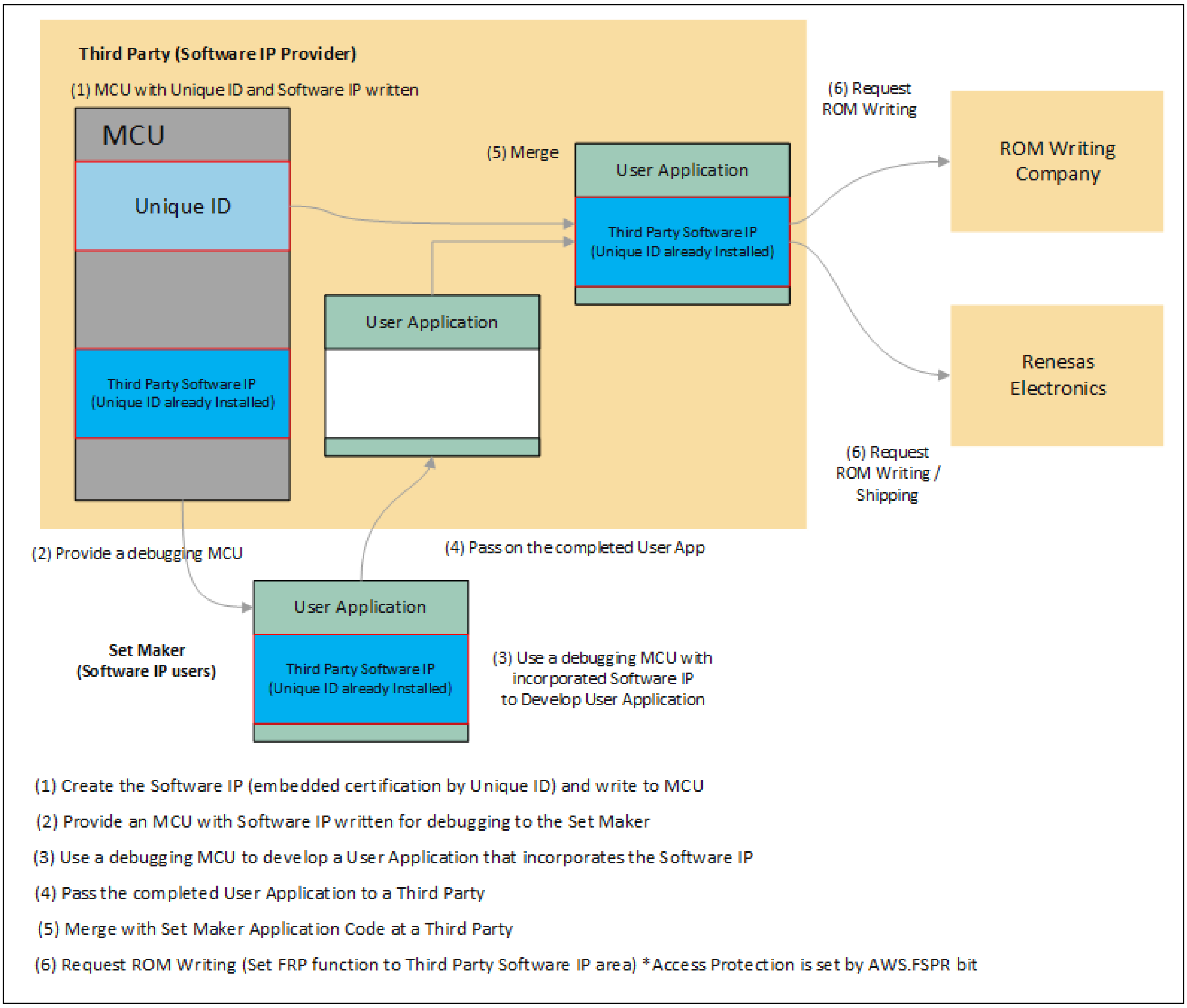


Figure 3.1. Third-Party Program Protection

3.2 User Application Protection

In the case where software IP is provided by a third party and software is developed by a set maker, the third-party software IP and the user application of the set maker are protected.

Prevents unauthorized use of third-party software IP and user application by end users.

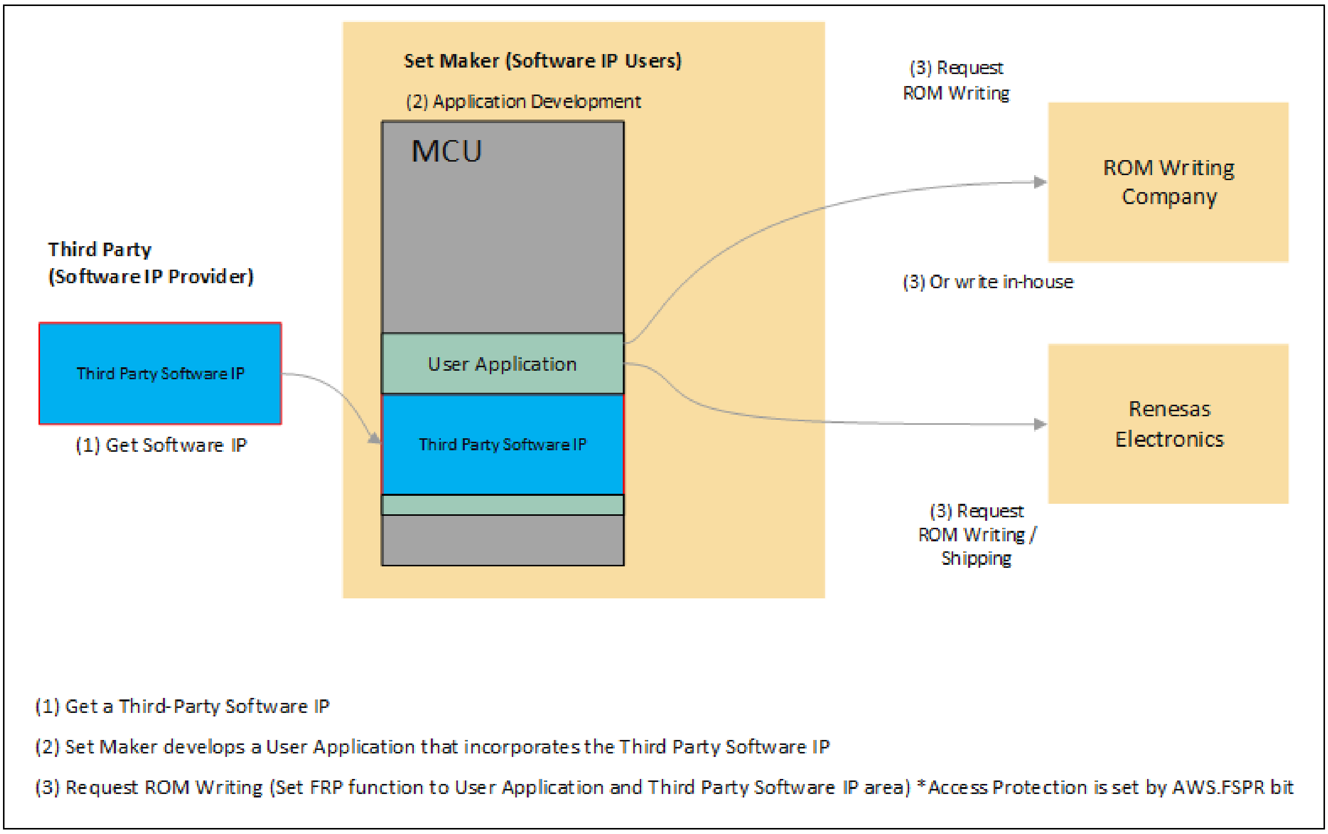


Figure 3.2. User Application Protection

4. Memory Protection Setting Methods

To set up Memory Protection on the RA0E1, it is important to understand the system architecture, which includes three mechanisms that provide access protection to the user application and software IP. The functions are Flash Read Protection (FRP), Access Window Setting (AWS), and ID Code Protection.

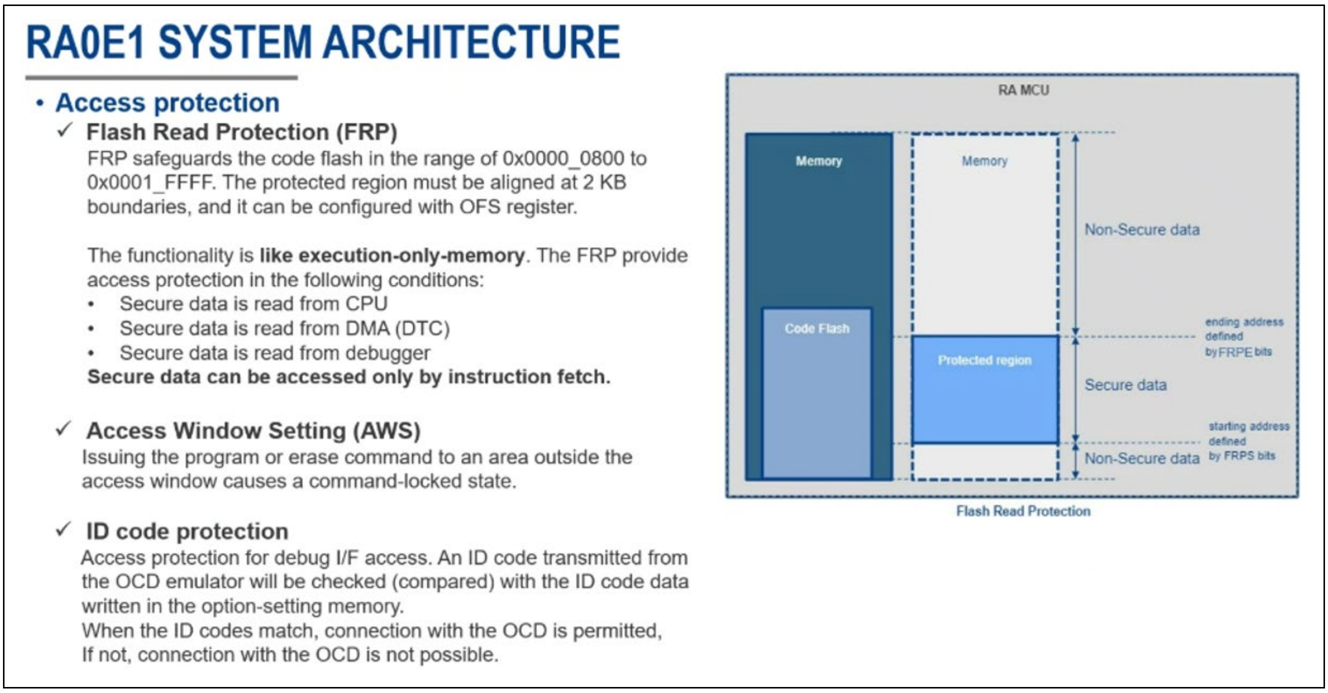


Figure 4.1. RA0E1 Access Protection

User application self-programming and debuggers use flash read/erase methods that must be considered in determining the required level of access protection for the user application and software IP. In the RA0E1, the Option-setting memory is allocated to the Configuration area access and the Program flash area, which are used to set up the access protection. Under certain conditions, the Configuration area can be reprogrammed by self-programming or a debugger. In this case, the Flash Read Protection can be disabled.

To provide the best level of access protection, use the Access Window Protection Flag FSPR=0, where the AWS.FSPR bit can be used to lock the AWS configuration permanently. When locked, the AWS register cannot be rewritten by any method, such as a flash programmer, debugger, or user application self-programming. With this feature, the Flash Read Protection is permanent.

Note: The AWS.FSPR bit cannot be changed to 1 once it is set to 0. At that time, the access window and startup area selection cannot be set again.

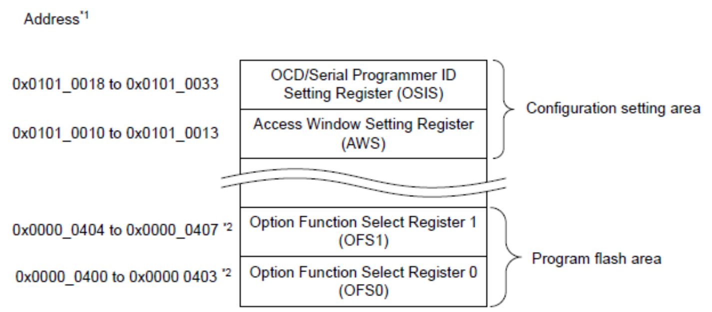
Issuing the program or erasing a command to an area outside the access window causes a command-locked state. The access window is only valid in the program flash area. The access window provides protection in self-programming mode, and on-chip debug mode.

With ID Code Protection, debug interface access is handled by an ID code transmitted from the OCD emulator that is compared to the ID code data written in the Option-Setting Memory. Connection with the OCD is only permitted when the ID code matches. Note with ID Code Protection, only the OSIS register can be re-written by self-programming.

For more information on Flash Read Protection, AWS, and ID Code Protection, refer to RA0E1 User's Manual (R01UH1040) Section 6 **Option-Setting Memory**.

Memory Protection is enabled by setting the Option-Setting Memory. Refer to Figure 4.2 below for the RA0E1 Option-Setting Memory Area register map and the RA0E1 User's Manual (R01UH1040) Section 6.2 **Register Descriptions** for more details on the OSIS, AWS, and Option Function Select (OFS) Registers.

The Option-Setting Memory determines the state of the MCU after a reset. The Option-Setting Memory is divided into the **Configuration setting area** and the **Program flash area** of the flash memory. The available methods of setting are different for each of the two areas.



Note 1. The option-setting memory must be allocated to the user area of the flash memory.
Note 2. The address of these registers will be changed when the boot swap is set. See section 6.2.1. OFS0 : Option Function Select Register 0 and section 6.2.2. OFS1 : Option Function Select Register 1 for details.

Figure 4.2. RA0E1 Option-Setting Memory Area

4.1 Setting the Option-Setting Memory

4.1.1 Allocation of Data in Option-Setting Memory

Programming data is allocated to the addresses in the option-setting memory, as shown in **Figure 4.2**. The option-setting memory must be allocated to the user area of the flash memory. The allocated data is used by tools such as flash programming software or an on-chip debugger.

Note: The programming formats vary depending on the compiler used. See the compiler manual for more details.

4.1.2 Setting Data for Programming the Option-Setting Memory

Allocating data according to the procedure described in Section 4.1.1 Allocation of Data in Option-Setting Memory alone does not write the data to the option-setting memory. You must follow one of the actions described below in this section.

1. Changing the Option-Setting Memory by Self-Programming

   Use the programming command to write data to the program flash area. Next, use the configuration setting command to write data to the option-setting memory in the configuration setting area for the OSIS and Access Window Setting Registers. In addition, use the startup area select function to safely update the boot program, which includes the option-setting memory. For more details on the programming command, the configuration setting command, and the startup area select function, see Section 28 **Flash Memory** of the RA0E1 User's Manual (R01UH1040).

2. Debugging through an OCD or Programming by a Flash Writer

   This procedure depends on the tool or utility in use. See the tool manual for more details.

   In general, the MCU provides two setting procedures:

   - Read the data allocated as described in Section 4.1.1 Allocation of Data in Option-Setting Memory from an object file or Motorola S-format file generated by the compiler and write the data to the MCU.
   - Use the GUI interface of the tool to program the MCU with the same data in Section 4.1.1 Allocation of Data in Option-Setting Memory.

Note: If you are using the e2 studio IDE, the OFS settings are configured in e2 studio and are set by writing the object after compilation.

Refer to the RA0E1 User's Manual (R01UH1040) Section 6.3 **Setting Option-Setting Memory** for more information and usage notes.

5. Related Application Notes

Application notes related to this application note are listed below.

1. FPB-RA0E1 Example Project Bundle (R20AN0745)
2. FPB-RA0E1 Tutorial (R01AN7315)
3. RA0 Quick Design Guide (R01AN7309)

6. Next Steps

Visit renesas.com/ra/fpb-ra0e1 for more information about the FPB-RA0E1 example kit, including its Quick Start Guide, design data, ordering information, and other useful application projects.

7. Reference Documents

RA0E1 Group FPB-RA0E1 User’s Manual (R20UT5378) FPB-RA0E1 – User's Manual

RA0E1 Group User's Manual: Hardware (R01UH1040) FPB-RA0E1 Documents

Flexible Software Package (FSP) User’s Manual: (R11UM0155) RA Flexible Software Package (FSP)

Website and Support

Visit the following vanity URLs to learn about key elements of the RA family, download components and related documentation, and get support.

RA Product Information	www.renesas.com/ra
RA Product Support Forum	www.renesas.com/ra/forum
RA Flexible Software Package	www.renesas.com/FSP
Renesas Support	www.renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul.31.24	—	First release document

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.

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/.