# Renesas RX Family

## Porting from CS+ (V850) to e² studio (RX) (with the Use of an Emulator)

### Introduction

This application note describes the differences among the methods for setting and operating the E2 emulator Lite, E2 emulator, and E20 emulator for e² studio when porting a project from CS+ for the V850 family to e² studio for the RX family.

### Target Device

RX family

### Operating Environment

e² studio version: 2024-01

CS+ for CA, CX version: V4.08.00

Refer to the Quick Start Guide for e² studio, which explains how to use the tool.

- e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family (R20UT5293)

For the debugging functions of the on-chip emulators, refer to the following document.

- On-chip Debuggers Performance Property (R20UT0616)

This application note has been created on the basis of the case where an RX65N is in use with an E2 Lite. However, the basic methods for setting and operating the emulators are common.

Contents

1. Introduction ..... 4

2. Differences in Usability of On-chip Emulators ..... 5

3. Differences in Functionality of On-chip Emulators ..... 5

4. Where are the Settings for Connection to an Emulator Made? ..... 6
4.1 Points where the Settings for Connection to the Emulator are Made ..... 6
4.2 Comparing the Settings for Connection to the Emulator ..... 7
4.2.1 Main Differences of Setting Points ..... 7
4.2.2 ID Code ..... 8
4.2.3 Setting for Overwriting the Internal Flash Memory ..... 9
4.2.4 Setting for Downloading a File to the External Flash Memory ..... 10

5. Launching the Debugger (E2, E2 Lite, or E20 Emulator) ..... 11
5.1 Launching the Debugger from e² studio ..... 11
5.2 Setting a File for Downloading and Command Processing before and after Downloading ..... 11
5.3 Disconnecting e² studio from the Debugger ..... 12
5.4 Hot Plug-in Connection ..... 13

6. Debugging Features ..... 14
6.1 Basic Debugging Features ..... 14
6.2 Breakpoints ..... 14
6.2.1 Setting Program Counter (PC) Breakpoints ..... 14
6.2.2 Generating a Break in Response to Access to a Variable or to a Specified Address ..... 17
6.3 Tracing ..... 19
6.3.1 Setting Tracing ..... 19
6.3.2 Setting Conditions for Starting and Ending Tracing ..... 20
6.3.3 Setting Conditions for Data Access Tracing ..... 21
6.3.4 Trace Eventpoints ..... 21
6.4 Performance (Timer Measurement) Features ..... 21
6.5 Manipulating Memory ..... 22
6.5.1 Filling Memory ..... 22
6.5.2 Saving Contents of Memory ..... 23
6.6 Manipulating Registers ..... 23
6.7 Automatically Updating Memory or Variables during the Execution of Programs ..... 24
6.7.1 Automatic Updating of the Memory View ..... 24
6.7.2 Automatic Updating of the Expressions View ..... 25

7. Using the Flash Writer Function ..... 26

8. FAQs ..... 27

Revision History ..... 28

# 1. Introduction

The help system of e² studio includes tutorials on the use of the debugger.

When you proceed through the methods from creation to the debugging of a program according to the tutorial indicated in the screen shot below, you can easily experience e² studio. Start by trying the tutorial.

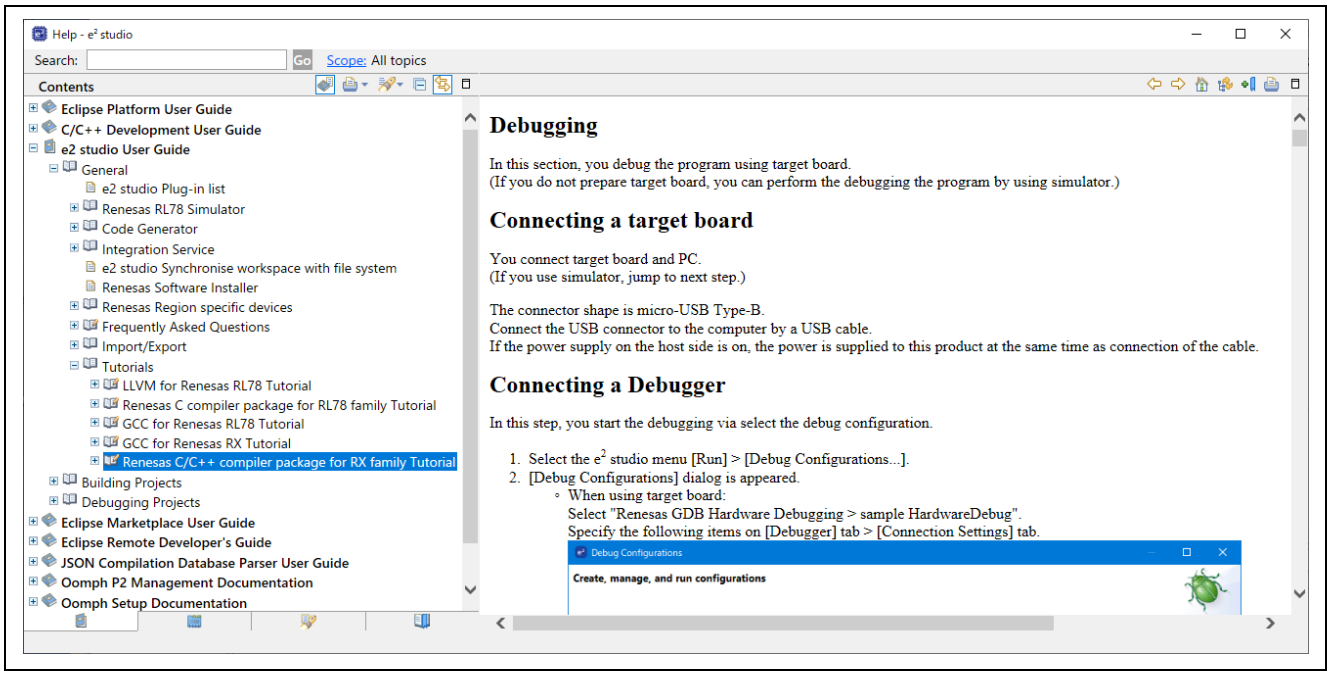


**Figure 1-1 Tutorial**

## 2. Differences in Usability of On-chip Emulators

The following indicates the on-chip emulators which can be used with the CS+ environment for the V850 family and e² studio for the RX family.

CS+ for the V850 family: E1, E20, and MINICUBE2 (QB-MINI2)

e² studio for the RX family: E1, E20, E2, and E2 Lite

Thus, if you already have an E1 or E20, the emulator is also usable with the RX family.

Note, however, that only the E2 or E2 Lite works with the new microcontrollers of the RX family. Refer to On-chip Debuggers Performance Property beforehand to see if the emulator will work with the device you will be using.

In general, if you do not already have an emulator, purchase an E2 or E2 Lite if you will be using devices of the RX family.

## 3. Differences in Functionality of On-chip Emulators

Refer to Table 3-1 for the differences between the main debugging functions when you are using an emulator which suits the family to which the microcontroller you are using belongs.

**Table 3-1 Comparison between Debugging Functions**

| | Usable Emulator | Hardware Break Function | Software Break Function | Tracing Function | Performance Measurement Function | Hot Plug-in Function |
|---|---|---|---|---|---|---|
| RX family | E2, E2 Lite, E1, E20 | Yes | Yes | Yes | Yes | Yes |
| V850 family | E1, MINICUBE2 (QB-MINI2) | Yes | Yes | No | No | Yes |

However, the debugging functions may differ from microcontroller to microcontroller, even if they are in the same family. For details, refer to On-chip Debuggers Performance Property.

For the connection between the microcontroller and the emulator, refer to E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RX Devices).

Note that using an on-chip emulator does not occupy a user resource (memory) of an RX device.

# 4. Where are the Settings for Connection to an Emulator Made?

## 4.1 Points where the Settings for Connection to the Emulator are Made

When you set the connection to the emulator in e² studio, click on [Run] -> [Debug Configurations…] to open the [Debug Configurations] window. After that, click on the name of debug configuration under [Renesas GDB Hardware Debugging] and make the settings on the [Debugger] tabbed page.

In the simulator, click on the name of a debug configuration under [Renesas Simulator Debugging (RX,RL78)]. Note that the positions where settings are made differ between the emulator and the simulator.

When you wish to change the emulator type, select the emulator from the [Debug hardware] drop-down list.

When you wish to change the microcontroller to be connected, you can select the new one from the [Target Device] drop-down list. However, since the microcontroller setting for the project and that for connecting the microcontroller to the emulator are dependent on each other in e² studio, the new setting will not be reflected on that for the target microcontroller of the project.

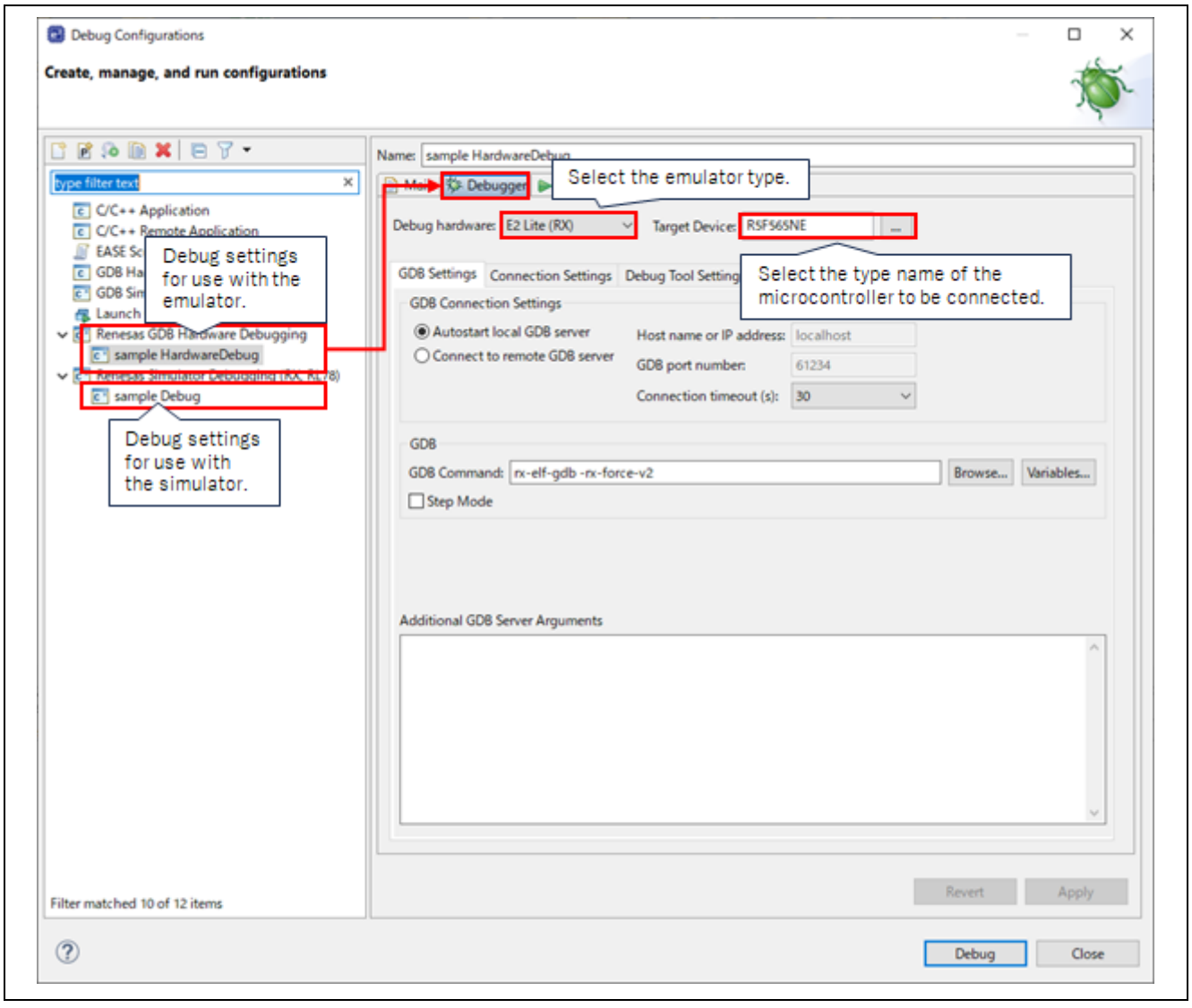


**Figure 4-1 Points where the Settings for Connection to the Emulator are Made**

## 4.2 Comparing the Settings for Connection to the Emulator

Make the settings for connection to the emulator on the [Debugger] tabbed page of e² studio.

### 4.2.1 Main Differences of Setting Points

Figure 4-2 shows a comparison of the setting points which are common to CS+ and e² studio.

If there are no items of correspondence for the setting points, the setting points are different. Details will be given in the subsequent sections.

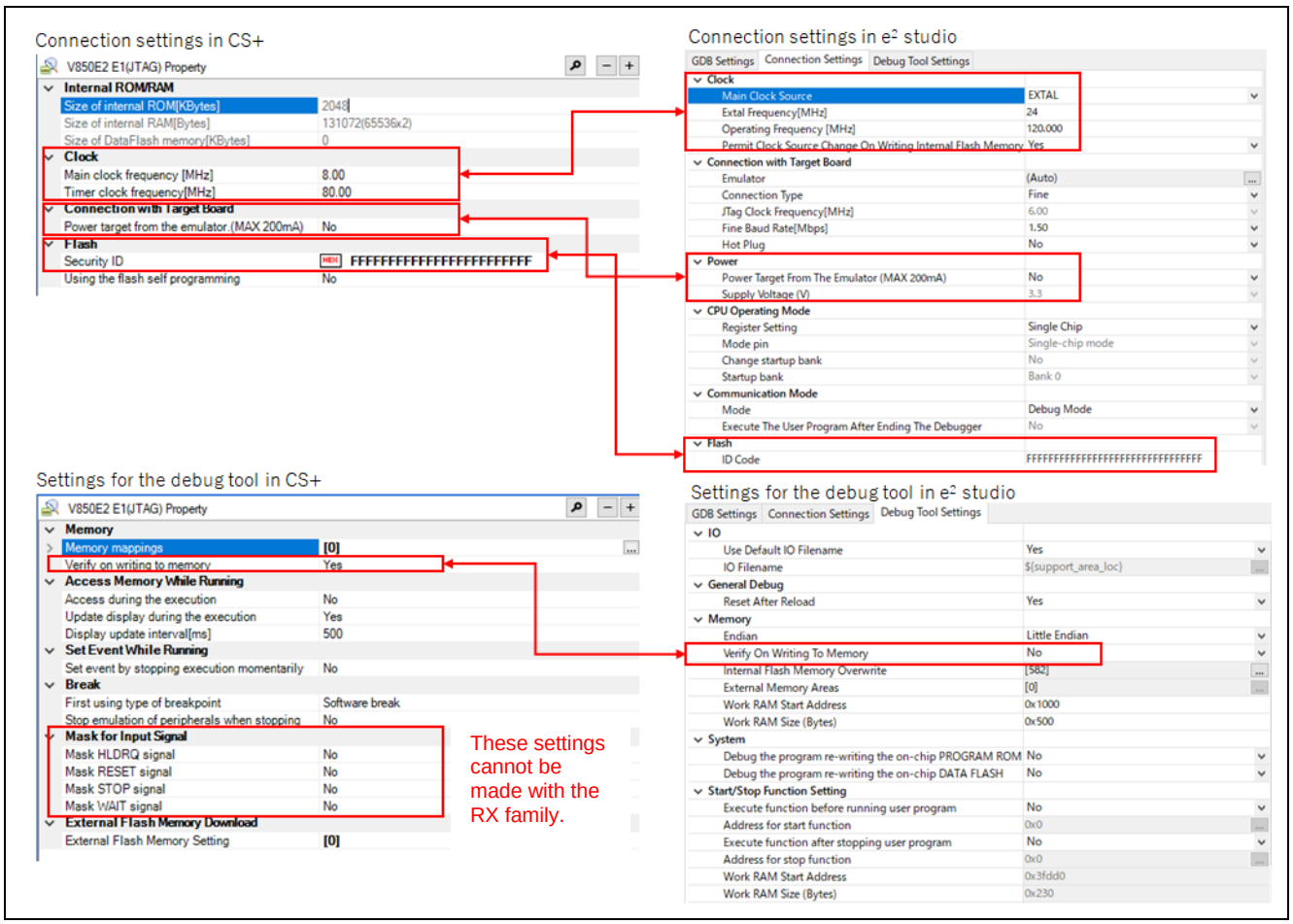


**Figure 4-2 Comparison of Setting Points between CS+ and e² studio (Example when an RX65N is in Use with the E2 Lite)**

### 4.2.2 ID Code

The specifications of ID code differ between the V850 and RX families.

For details, refer to Table 4-1.

**Table 4-1 Differences between the Specifications of ID Code**

| | Number of Digits | Address | Method of Setting | Method of Checking | Operation if the ID Code Does not Match | Applicability with the On-board Writer |
|---|---|---|---|---|---|---|
| RX family | 32 | Written in the user's manual of the microcontroller. | Set in the user program. | Set the value to be checked in the debugger. | Cannot be started.<br>For details, refer to the user's manual of the microcontroller. | ID authentication is required.<br>It is also required when the on-board writer is started. |
| V850 family | 20 (V850E1/ES)<br>24 (V850E2) | Flash option area | Set by the programmer. | Set the value to be checked in the debugger. | Cannot be started.<br>Reset and reconnect with the matching ID code. | ID authentication is not required.<br>It is only applicable for debugging. |

### 4.2.3 Setting for Overwriting the Internal Flash Memory

In overwriting of the internal flash memory by CS+, whether the original data are to be erased or retained when downloading a file depends on the “speed priority” mode; that is, the previous value in the free space before the first data and after the final data is retained and the free space between the first data and the final data is filled with FFH. With e² studio, this setting is made in block units.

In the dialog box which is opened by clicking on […] at the right of the [Internal Flash Memory Overwrite] item on the [Debug Tool Settings] sub-tabbed page, you can set whether the internal flash memory is to be overwritten without or after having been erased in block units during downloading.

The selected blocks are overwritten without being erased and the non-selected blocks are overwritten after being erased.

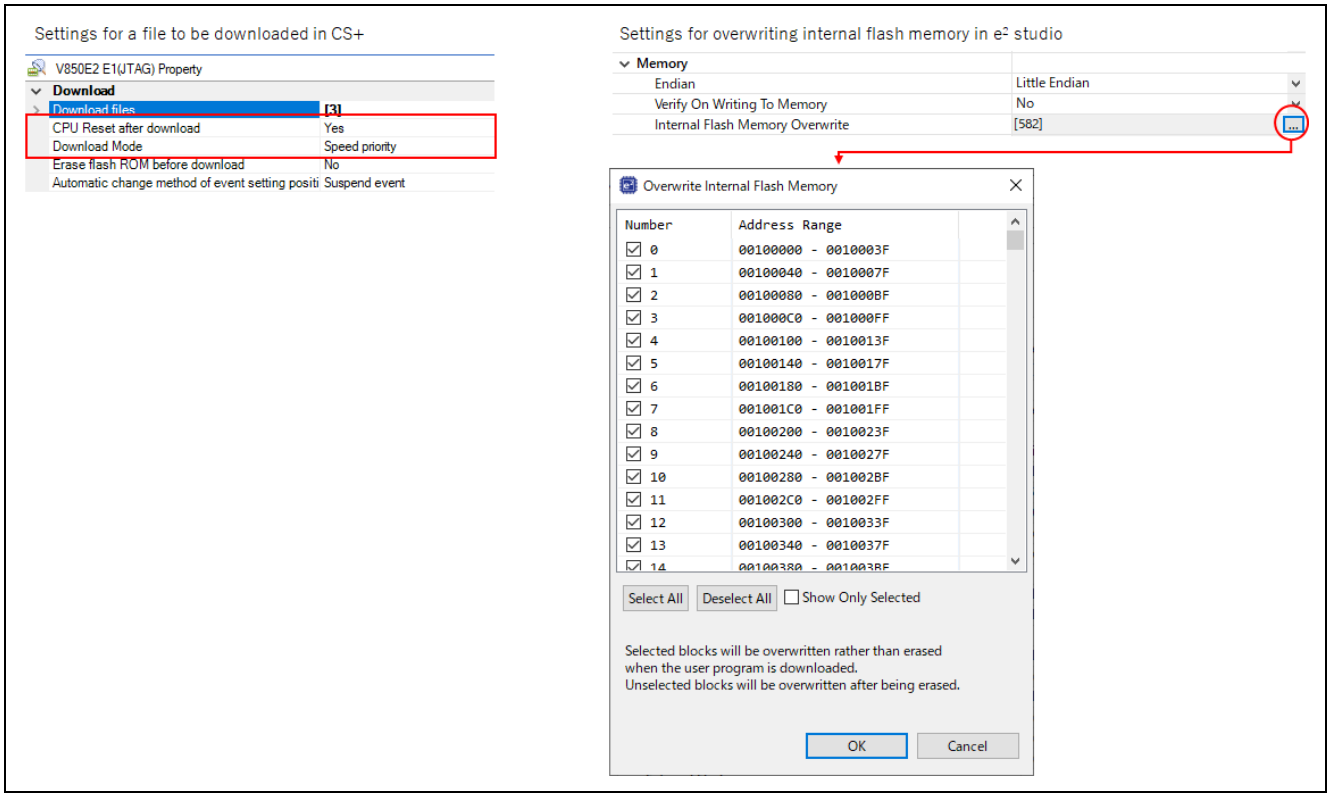


**Figure 4-3 Settings for Overwriting Internal Flash Memory**

### 4.2.4 Setting for Downloading a File to the External Flash Memory

In general, both CS+ and e² studio enable writing to the external flash memory.

With e² studio, however, writing to external parallel flash memory is only supported with the use of another tool, the External Flash Definition Editor.

The external flash memory information file (*.fdb) that has been in use with CS+ is not available.

After having set [Download Enabled] under the [External Flash] item on the [Debug Tool Settings] sub-tabbed page to [Yes], specify the USD file or set the erasure of the external flash memory before downloading in the dialog box which is opened by clicking on […] at the right of the [External Flash Definition] item.

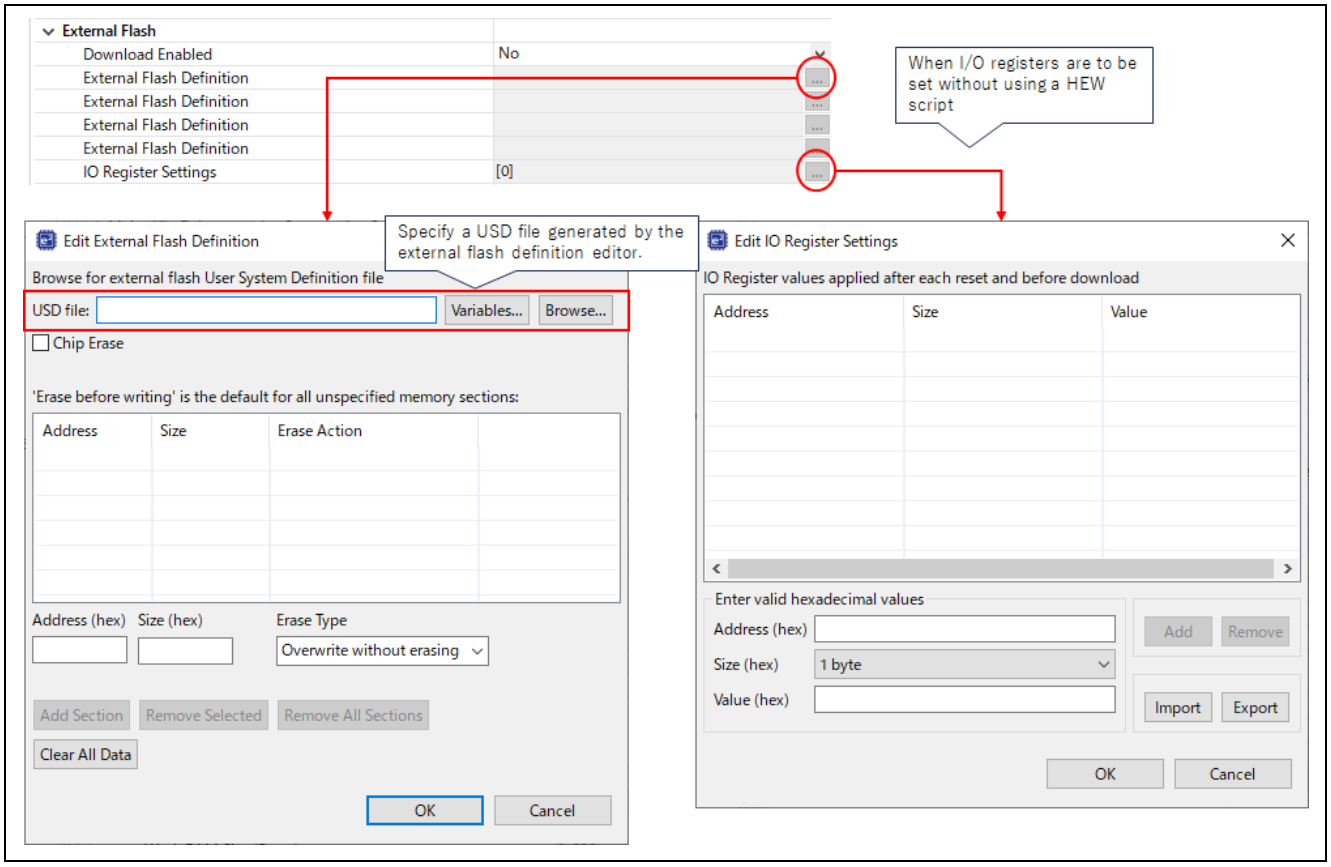


**Figure 4-4 Setting for Writing to the External Flash Memory**

When you write to the external flash memory, you need to change the operating mode of the microcontroller to the expansion mode before writing by using a HEW script or the [IO Register Setting] item.

For details on USD files and HEW commands, refer to External Flash Definition Editor (EFE) User's Manual.

# 5. Launching the Debugger (E2, E2 Lite, or E20 Emulator)

## 5.1 Launching the Debugger from e² studio

For details on the method for launching the debugger from e² studio, refer to the following FAQ.

How can I launch debugger in e² studio?

## 5.2 Setting a File for Downloading and Command Processing before and after Downloading

With the default setting, downloading proceeds after the debugger is launched in the state where the debugger had not currently been launched in CS+ or e² studio.

In e² studio, the [Startup] tabbed page in the [Debug Configurations] window is used to add the file for downloading or change the settings for the file.

However, the [Debug Tool Settings] sub-tabbed page is used to reset the debugger after downloading.

Note that the names of the load module files are of the form "*.x" for CC-RX and "*.elf" for GCC in e² studio although they take the form "*.lmf" in CS+. The [Image] of [Load Type] is equivalent to [Object] in CS+.

In e² studio, the [Startup] tabbed page in the [Debug Configurations] window is also used to operate the target in the debugger when the debugger is launched or before or after downloading.

This operation corresponds to a hook transaction in CS+.

Figure 5-1 shows a comparison of the points in e² studio where items that are common to those for CS+ are set.

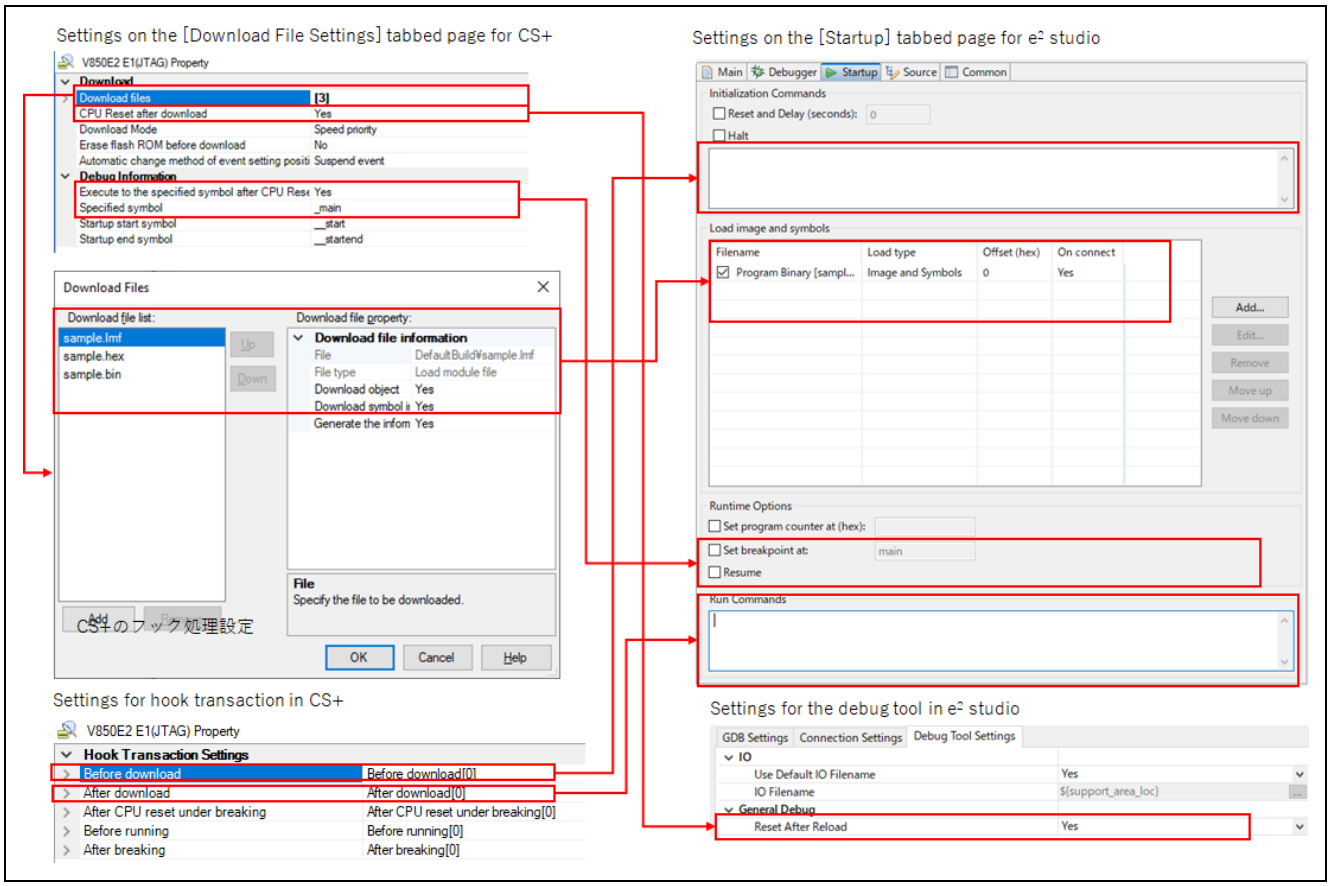


**Figure 5-1 File to be Downloaded and Command Processing**

Note that the command format differs between CS+ and e² studio; GDB commands are used in e² studio.

The following is an example of the statements for hook transactions having been converted to GDB commands.

```
SYSTEM.ROMWT 0x02     →    set {char} 0x0008101C = 0x02
SYSTEM.PRCR 0xa503    →    set {short}0x000803FE = 0xa503
```

## 5.3 Disconnecting e² studio from the Debugger

When e² studio is to be disconnected from the debugger, select [Terminate] or [Disconnect] from the [Run] menu or click on the  or button on the toolbar.

## 5.4 Hot Plug-in Connection

With the hot plug-in function, you can connect the emulator to the target board during the execution of a program and debug the program while it is running.

This section describes points for caution when using the hot plug-in function to connect an emulator in e² studio.

CS+ has a selection for making a hot plug-in connection in the menu. In e² studio, start by selecting [Run] -> [Debug Configurations…] to open the [Debug Configurations] window and specify [Yes] for [Hot Plug] on the [Connection Settings] sub-tabbed page on the [Debugger] tabbed page. After that, launch the debugger through the normal launching method.

Note that you should select [Symbol only] for [Load Type] of the file registered with [Load image and symbols] on the [Startup] tabbed page before launching the debugger.

In addition, deselect the checkbox for the setting of [Breakpoint Setting Destination]. Selection of this checkbox may lead to an error during connection processing.

When a debugger is launched, a message will appear in the dialog box. When you connect the emulator to an active target board and click on the [OK] button, the debugger will be connected to the microcontroller through hot plug-in.

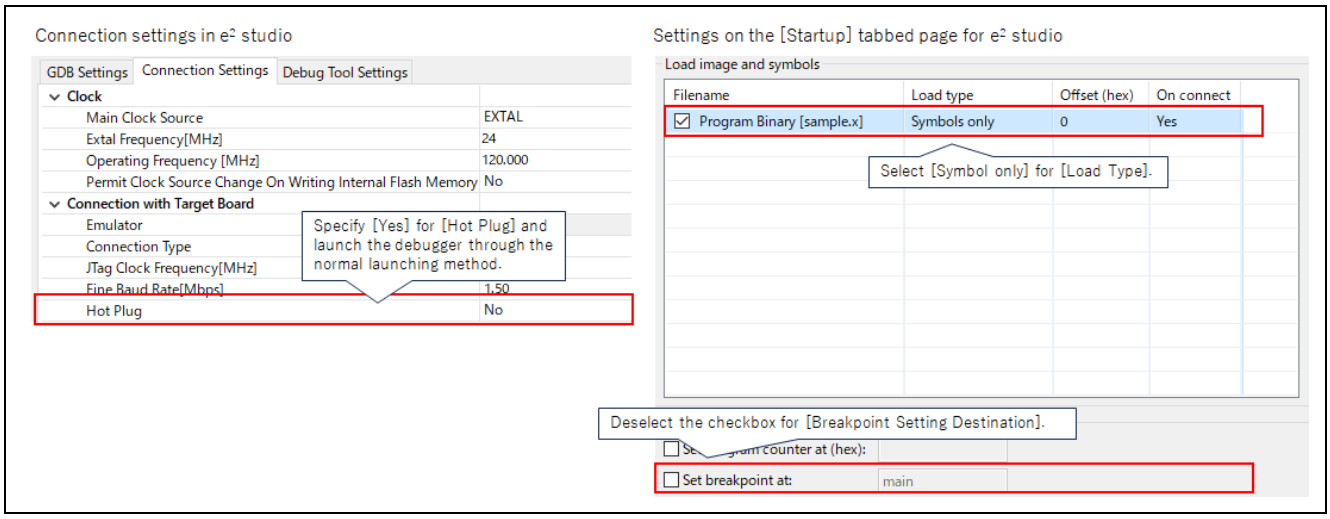


**Figure 5-2 Settings for a Hot Plug-in Connection**

# 6. Debugging Features

## 6.1 Basic Debugging Features

With e² studio, programs can be executed, stopped, reset by the CPU, downloaded, or step-executed from the menu or toolbar, in much the same ways as with CS+. For the main features, refer to section 5.4, Basic Debugging Features, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

## 6.2 Breakpoints

In e² studio, breakpoint features are usable in the same ways as with CS+. Note that breakpoints are managed in the [Breakpoints] view and [Eventpoints] view.

### 6.2.1 Setting Program Counter (PC) Breakpoints

In e² studio, breakpoints can be set in the left margin of the editor window (the space indicated by a red frame in the figure below) in the same way as in CS+.

Double-clicking on the line showing an address will set a breakpoint there. When the address specified by the PC reaches the breakpoint, a break will occur.

Double-clicking on a line where a breakpoint has been set removes the breakpoint. Note that a breakpoint is removed by single-clicking in CS+ but this requires double-clicking in e² studio.

A breakpoint set by this method is registered in the [Breakpoints] view ( ), which is opened by clicking on [Window] -> [Show View] -> [Breakpoints].

In the [Breakpoints] view, you can specify or modify the priority of a hardware breakpoint or software breakpoint among the registered breakpoints or refer to the state of the setting of breakpoints.

For details, refer to section 5.4.1, Breakpoints View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

Note that a breakpoint specified by this method leads to execution stopping before execution of the specified line (a “before PC” break).

Note also that if a setting for [Ignore count] is made, a break will occur after the breakpoint condition is satisfied the specified number of times but the execution of the user program will no longer be in real-time.

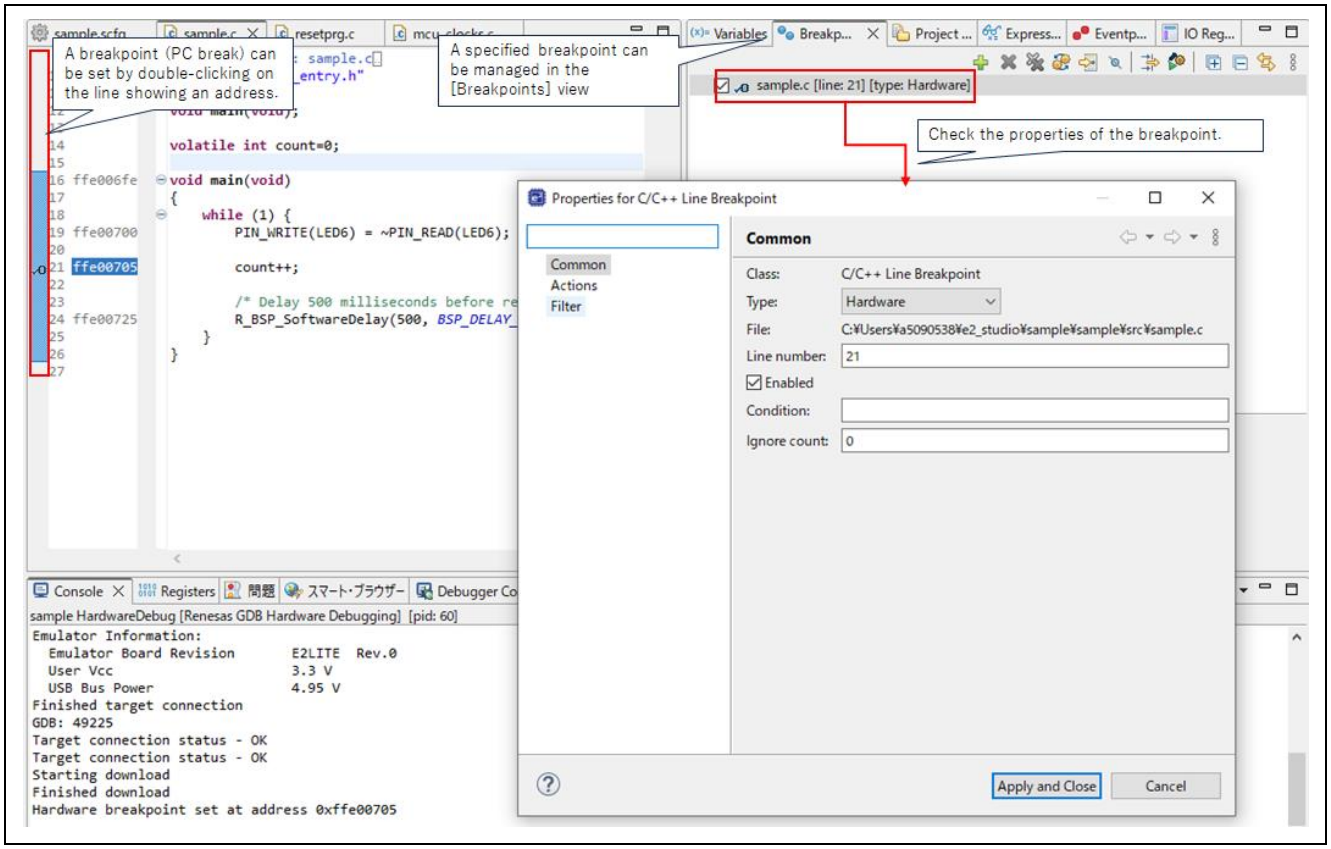


**Figure 6-1 Setting a PC Breakpoint (1)**

Clicking on [Renesas Views] -> [Debug] -> [Eventpoints] from the menu opens the [Eventpoints] view.

When you click on [Add] in the dialog box which is opened by clicking on [Event Break] and specify [Execution Address] for [Eventpoint Type] in the [Add Eventpoint] dialog box, a break of the same type as a PC break is set (although it is not displayed in the editor).

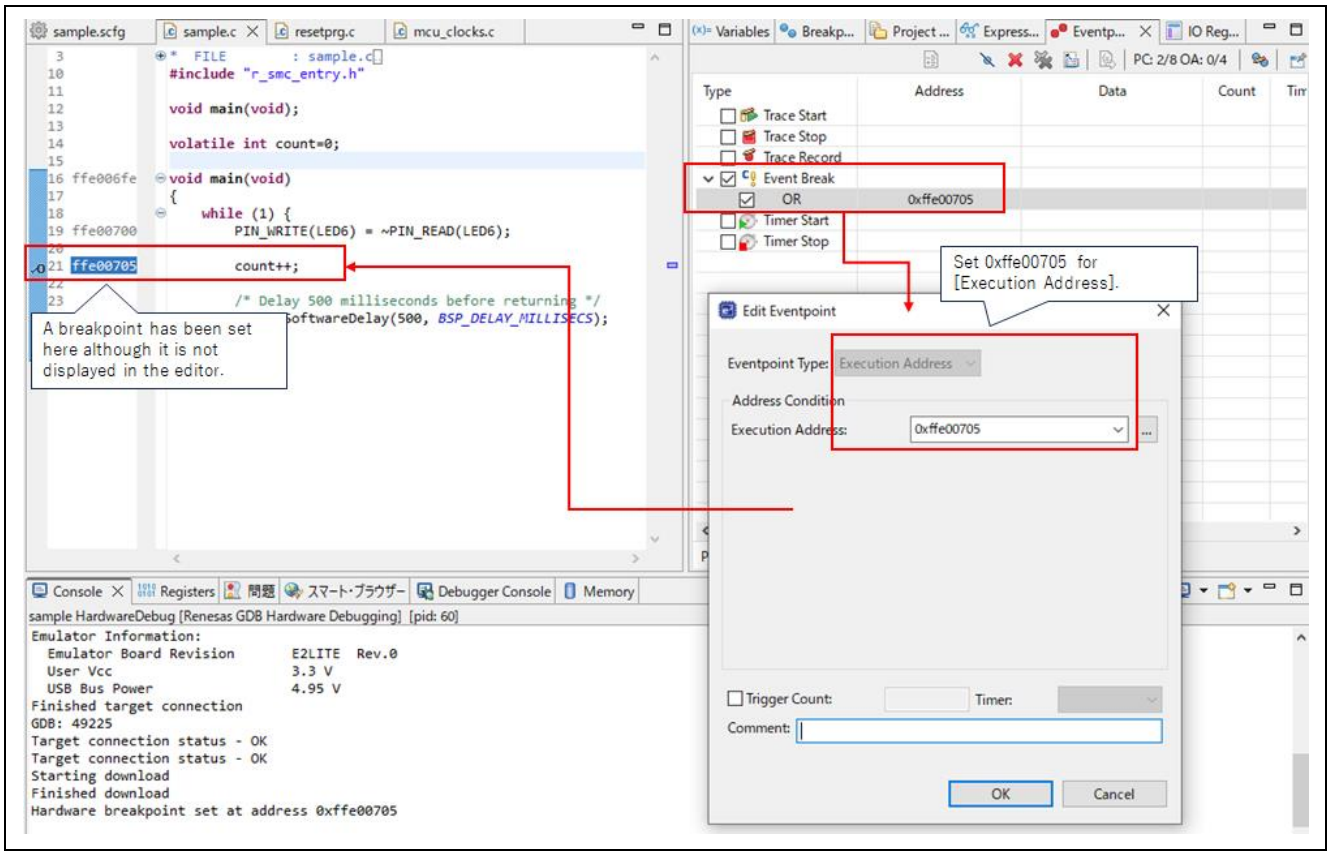


**Figure 6-2 Setting a PC Breakpoint (2)**

When [Pass count] is selected and a value is specified, a break occurs when the condition has been satisfied. In this case, real-time performance is maintained.

However, in the case of a breakpoint set by using this method, execution will be stopped after the instruction at the specified address is executed (a post-execution break). Thus, the breakpoint may lead to stopping at a location several instructions after the address where the breakpoint was set.

Hardware breaks set in the editor and breaks set for [Execution Address] share the same resources.

Therefore, we recommend that you set the PC break at the source line in the editor and confirm it in the [Breakpoints] view.

### 6.2.2 Generating a Break in Response to Access to a Variable or to a Specified Address

In e² studio, in much the same way as in CS+, if you want a break to be generated in response to access to a variable and at a specified address, you can set the break by setting reference to the variable in the [Expressions] view.

Click on [Window] -> [Show View] -> [Expressions] ( ) to open the [Expressions] view.

You can set an eventpoint for a data access break by registering the variable to which reference is to cause a break, selecting the variable, and right-clicking on [Renesas Eventpoint] -> [Add Event Break].

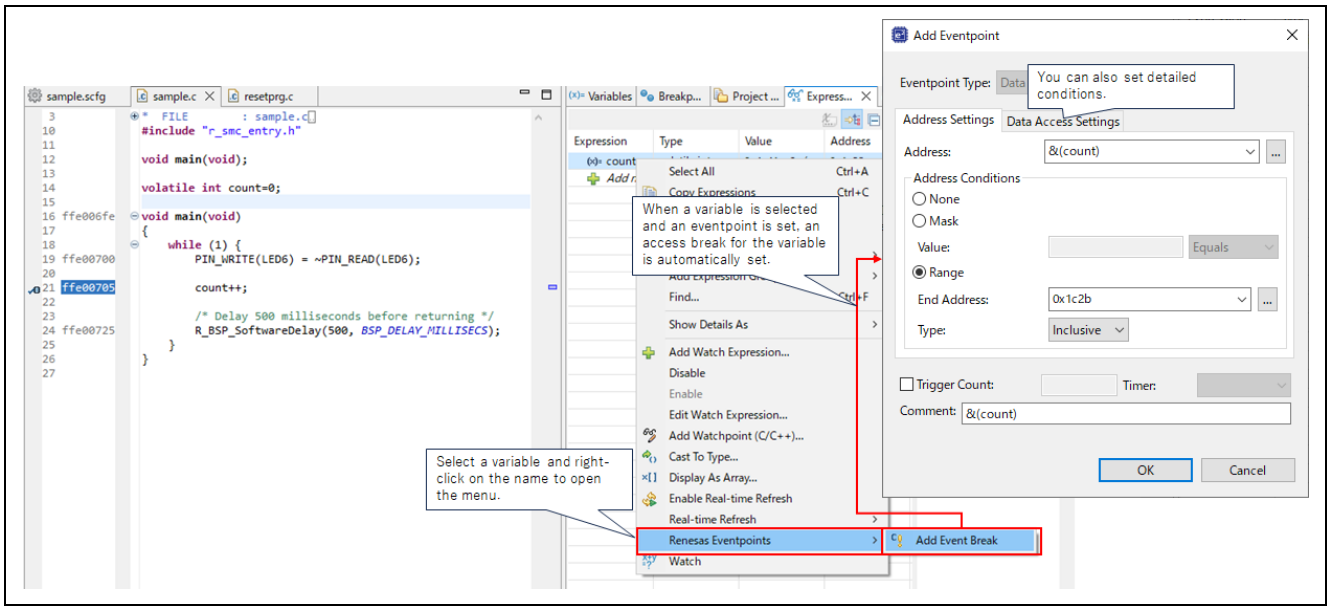


**Figure 6-3 Setting a Data Access Break (1)**

For details on the [Expressions] view, refer to section 5.4.2, Expressions View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

You can confirm the set event in the [Eventpoints] view, which is opened by clicking on [Renesas Views] -> [Debug] -> [Eventpoints] from the menu.

You can directly set a data access break by clicking on [Add] in the dialog box which is opened by clicking on [Event Break] in the [Eventpoints] view and specifying [Data Access] for [Eventpoint Type] in the [Add Eventpoint] dialog box. Use this method to directly specify an address value.

Note that a break specified by this method also leads to stopping after the condition has been satisfied. In such cases, the program may only be stopped after several further instructions have been executed.

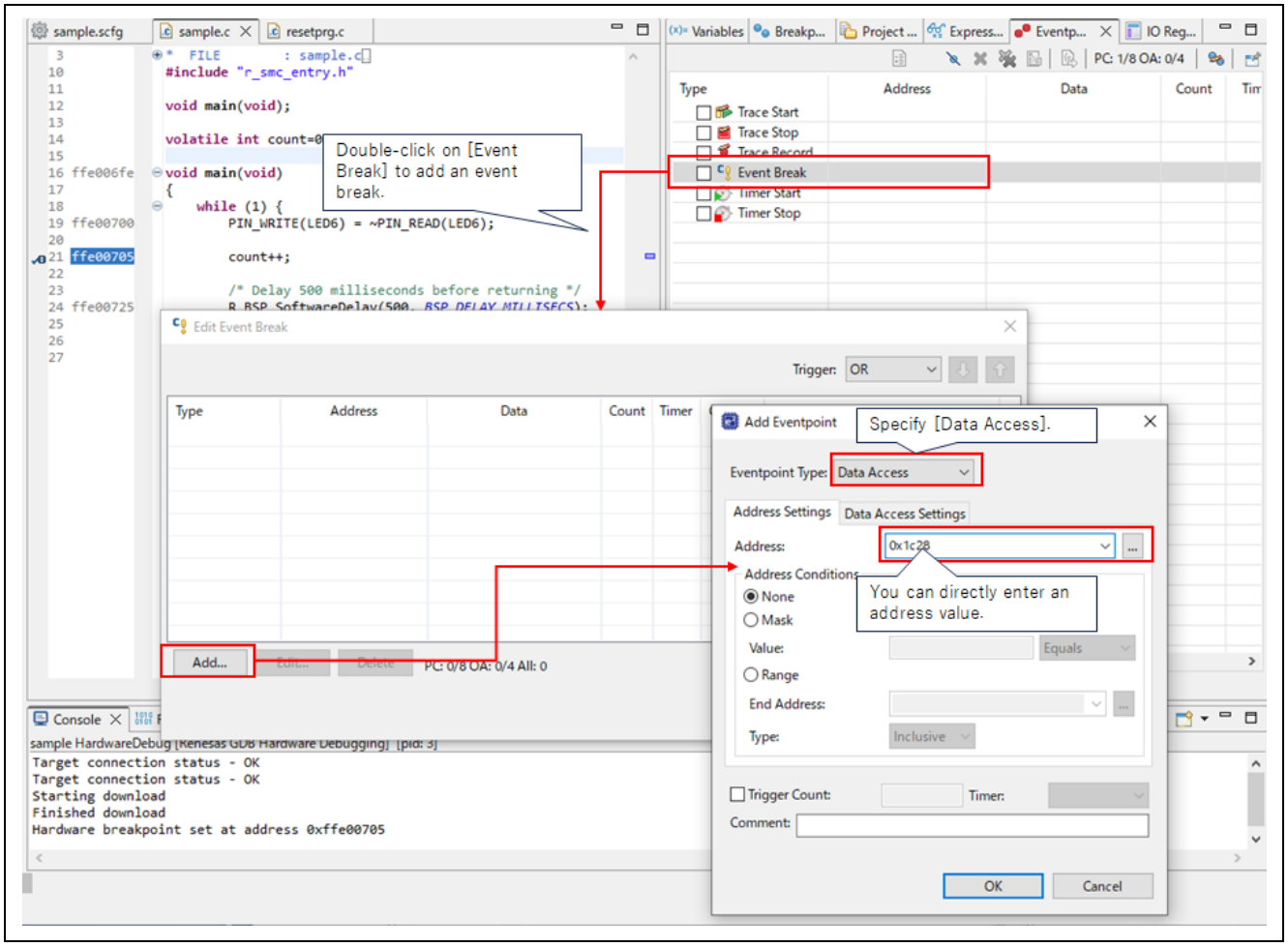


**Figure 6-4 Setting a Data Access Break (2)**

For details on the [Eventpoints] view, refer to section 5.4.7, Eventpoints View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

## 6.3 Tracing

e² studio (RX family) allows the use of trace features.

### 6.3.1 Setting Tracing

In e² studio, open the [Trace] view by clicking on [Renesas Views] -> [Debug] -> [Trace] from the menu. Click on the button in the [Trace] view to enable collecting trace information, then set tracing in the [Trace Acquisition] dialog box, which is displayed by clicking on the button.

The large-capacity trace feature involves the use of dedicated tracing pins and is used to set the external output of tracing. Note that this is only available with the E20 emulator.

Other features that are grayed-out in the figure below are not available for the RX family.

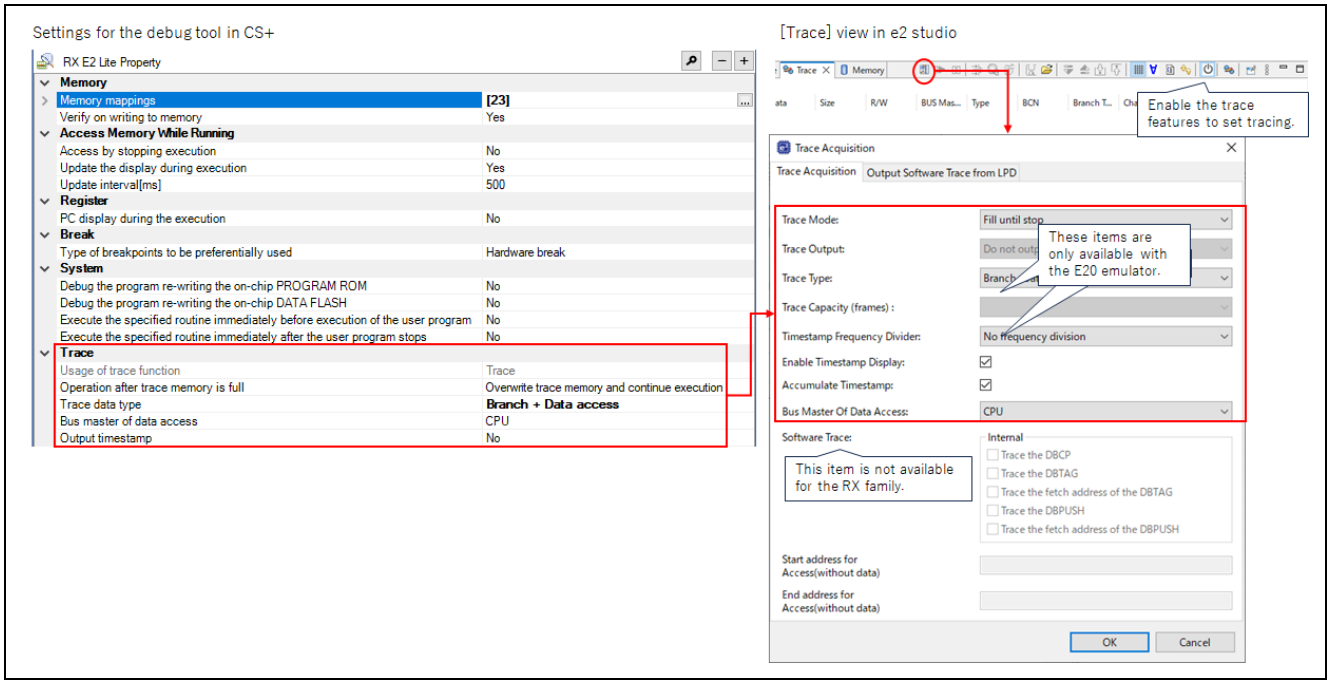


**Figure 6-5 Setting Trace Features**

### 6.3.2 Setting Conditions for Starting and Ending Tracing

If you do not want to start the acquisition of trace information in response to executing the user program but want to start and end acquisition in response to the execution of instructions at specific addresses, select [Renesas Views] -> [Debug] -> [Eventpoints] from the menu, open the [Eventpoints] view, and double-click on [Trace Start] and [Trace Stop] to open the relevant dialog boxes. In each dialog box, click on [Add] to set the condition for starting or stopping trace acquisition.

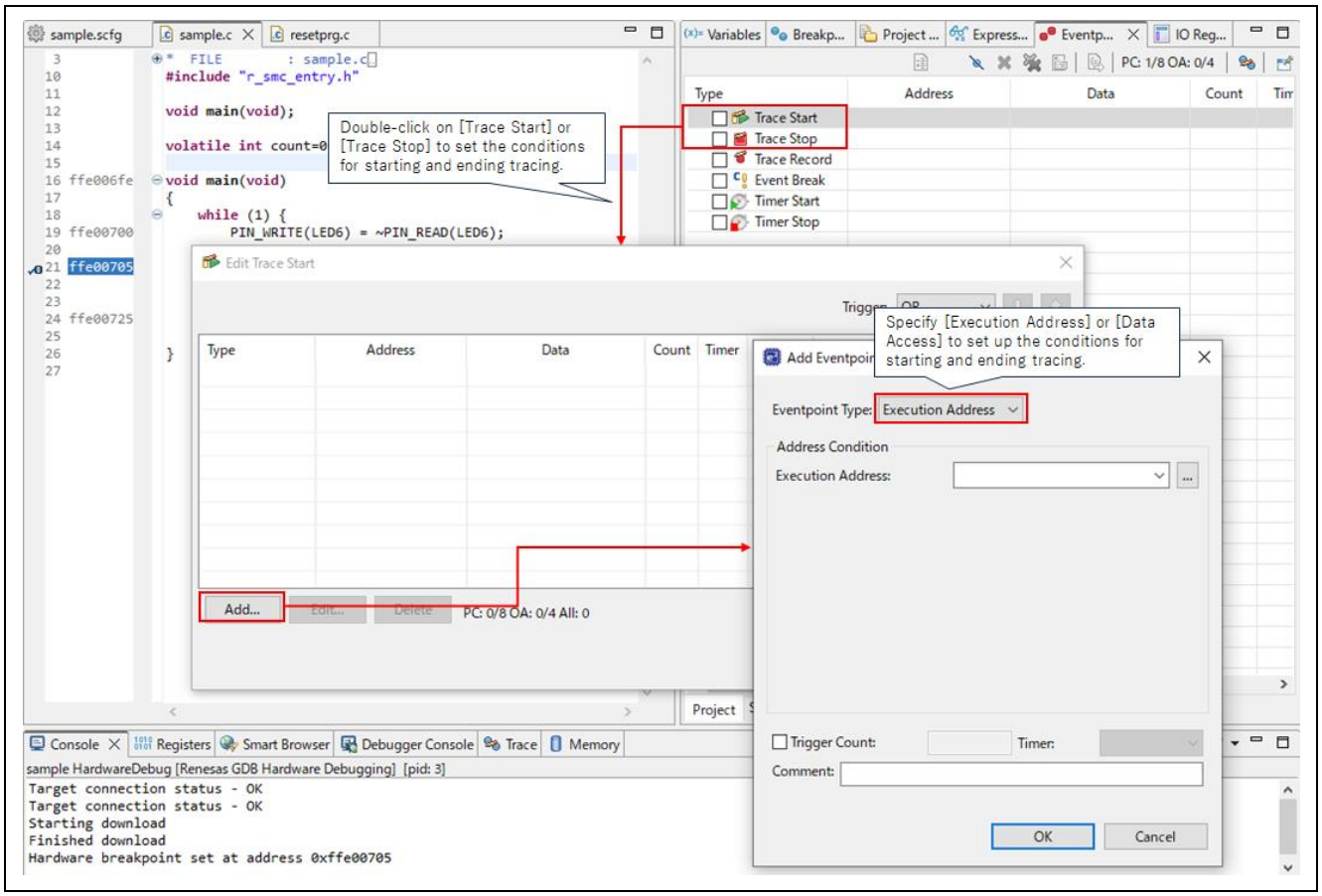


**Figure 6-6 Setting the Conditions for Starting and Ending Tracing**

### 6.3.3 Setting Conditions for Data Access Tracing

If you do not want to acquire all the data access to the user program but want to trace only data access to a specific address, double-click on [Trace Record] in the [Eventpoints] view and click on [Add] to set an event.

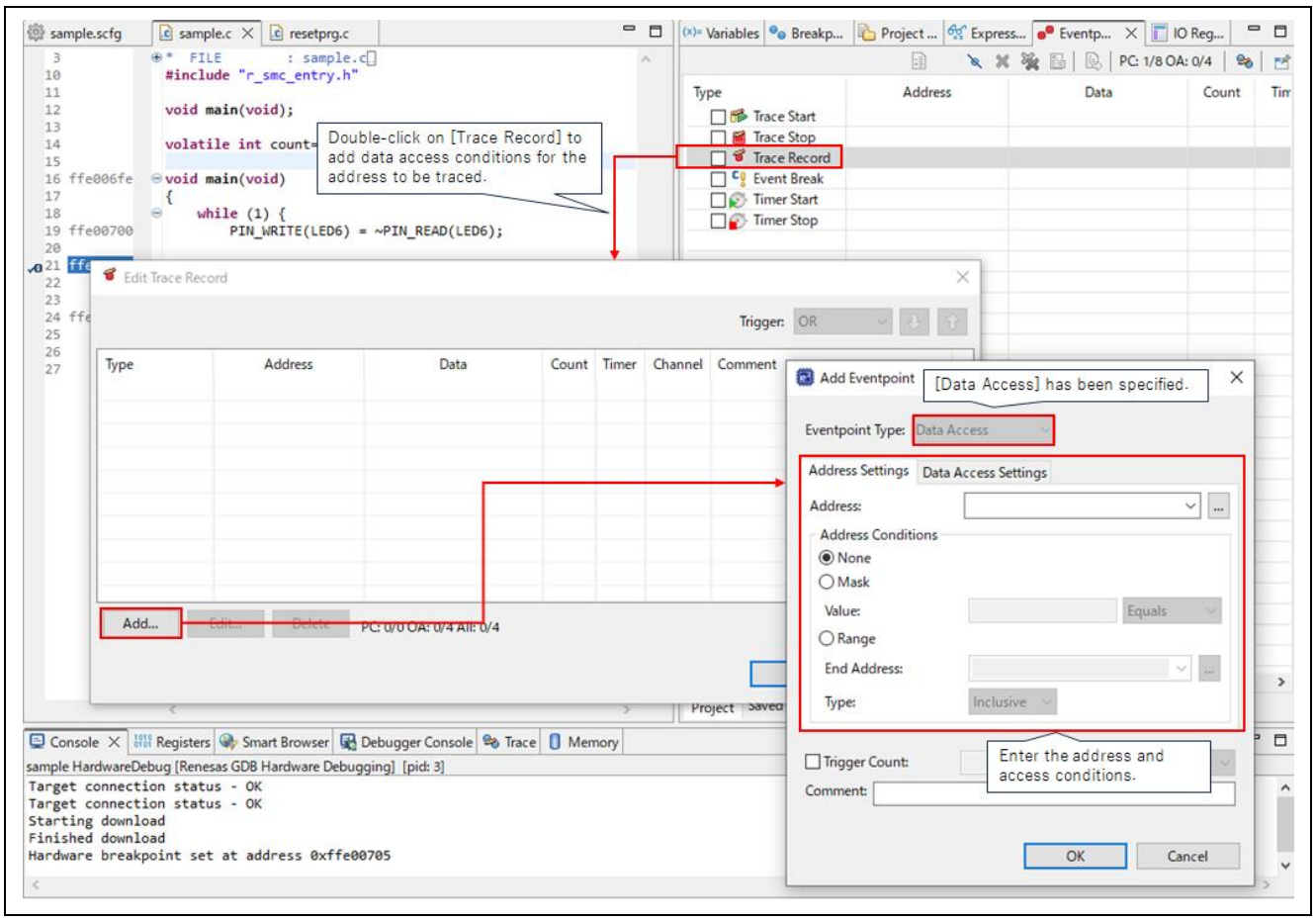


**Figure 6-7 Setting Data Access Tracing**

### 6.3.4 Trace Eventpoints

You can set trace events in the [Trace Eventpoints] dialog box, which is opened by clicking on the button in the [Trace] view.

For details, refer to section 5.4.9, Trace View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

## 6.4 Performance (Timer Measurement) Features

Performance measurement (measurement of the time for a specific range in the program) by using the timer measurement features is available in e² studio (RX family).

For the method of setting this feature and confirming the result of measurement, refer to the following FAQ.

How can I measure an execution time for the specified interval?

Note that RX600/700 has two timer channels enabling the measurement of the execution time for two ranges in the program. However, if the two timers are configured to operate as a single 64-bit timer, only a single range is measurable and only Timer 1 is displayed.

## 6.5 Manipulating Memory

For details on opening the [Memory] view of e² studio and on its basic features, refer to section 5.4.4, Memory View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

The following sections describe ways of manipulating memory which are frequently used in the debugger.

### 6.5.1 Filling Memory

Batched changing of all values in a range of memory (filling memory) in e² studio is enabled by selecting [Fill memory] in the dialog box which is opened by selecting [Find/Replace/Fill] from the menu of the Memory view.

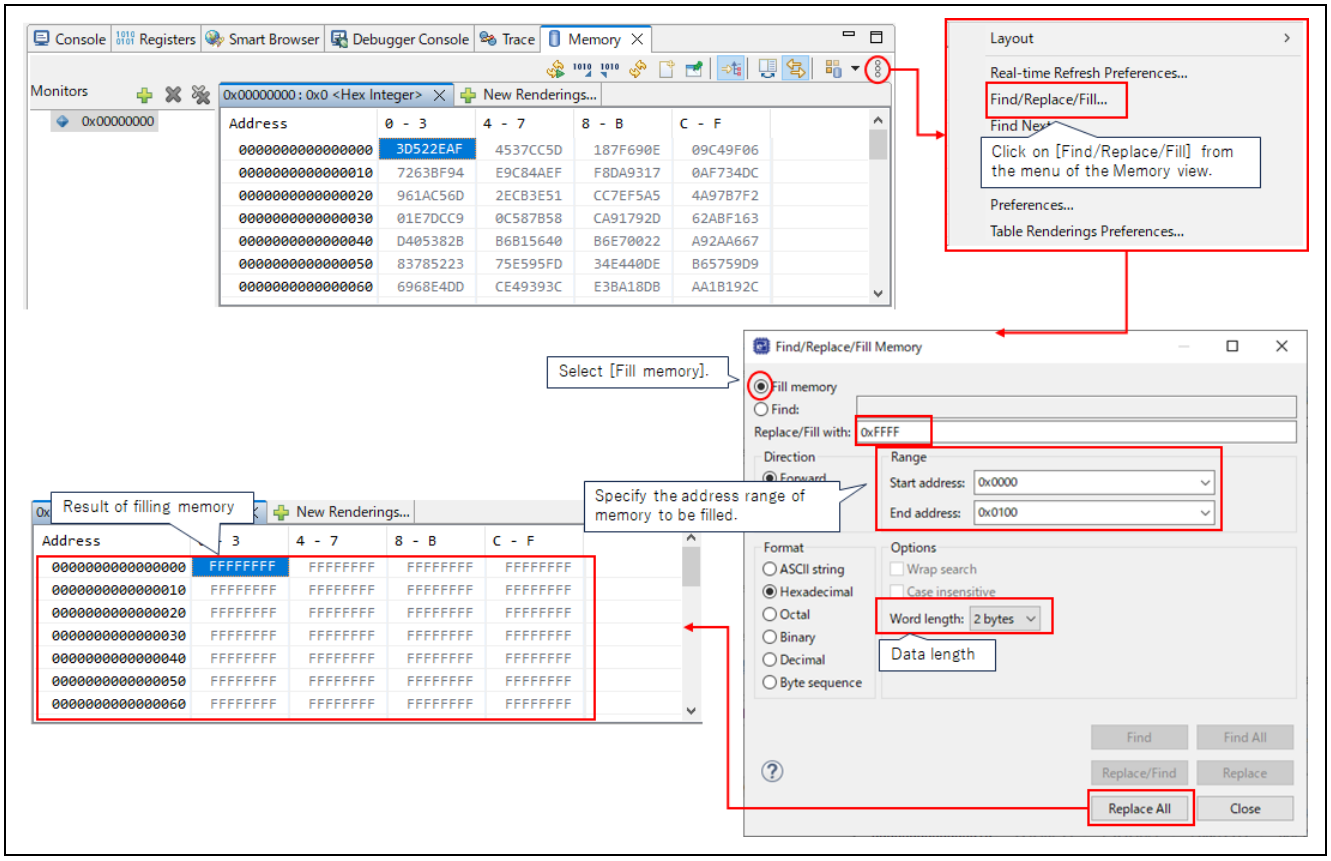


**Figure 6-8 Filling Memory**

### 6.5.2 Saving Contents of Memory

To save contents of memory in e² studio, click on the Export icon ( ) in the Memory view.

When the dialog box is opened, specify the type, address range, and name of the file where the contents of memory are to be saved and click on the [OK] button.

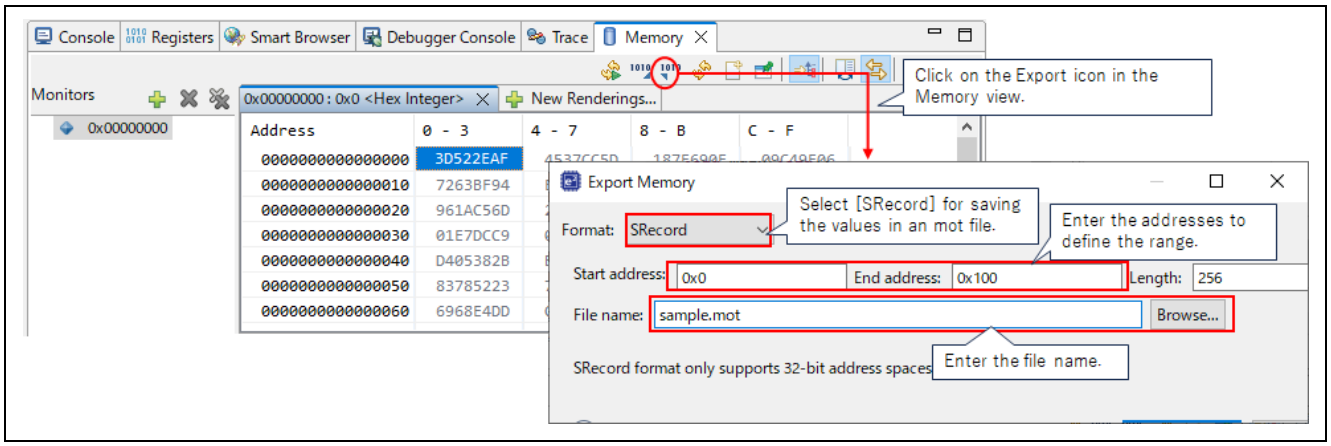


**Figure 6-9 Saving Memory**

## 6.6 Manipulating Registers

e² studio has views for reference to and modification of general-purpose registers and I/O registers, respectively.

For details on opening the views and their basic features, refer to section 5.4.3, Registers View, and section 5.4.8, IO Registers View, in the e² studio Quick Start Guide for RX/RL78/RH850/RISC-V MCU Family.

For the method of saving the values of I/O registers in a file, refer to the following FAQ.

How to save I/O register values to a file

## 6.7 Automatically Updating Memory or Variables during the Execution of Programs

In e² studio, automatic updating is available in the Memory view and Expressions view.

### 6.7.1 Automatic Updating of the Memory View

In e² studio, clicking on the real-time refresh button ( ) in the Memory view leads to updating of the memory values which are shown during the execution of programs to the latest values at a set refresh interval. Updated values are indicated in red.

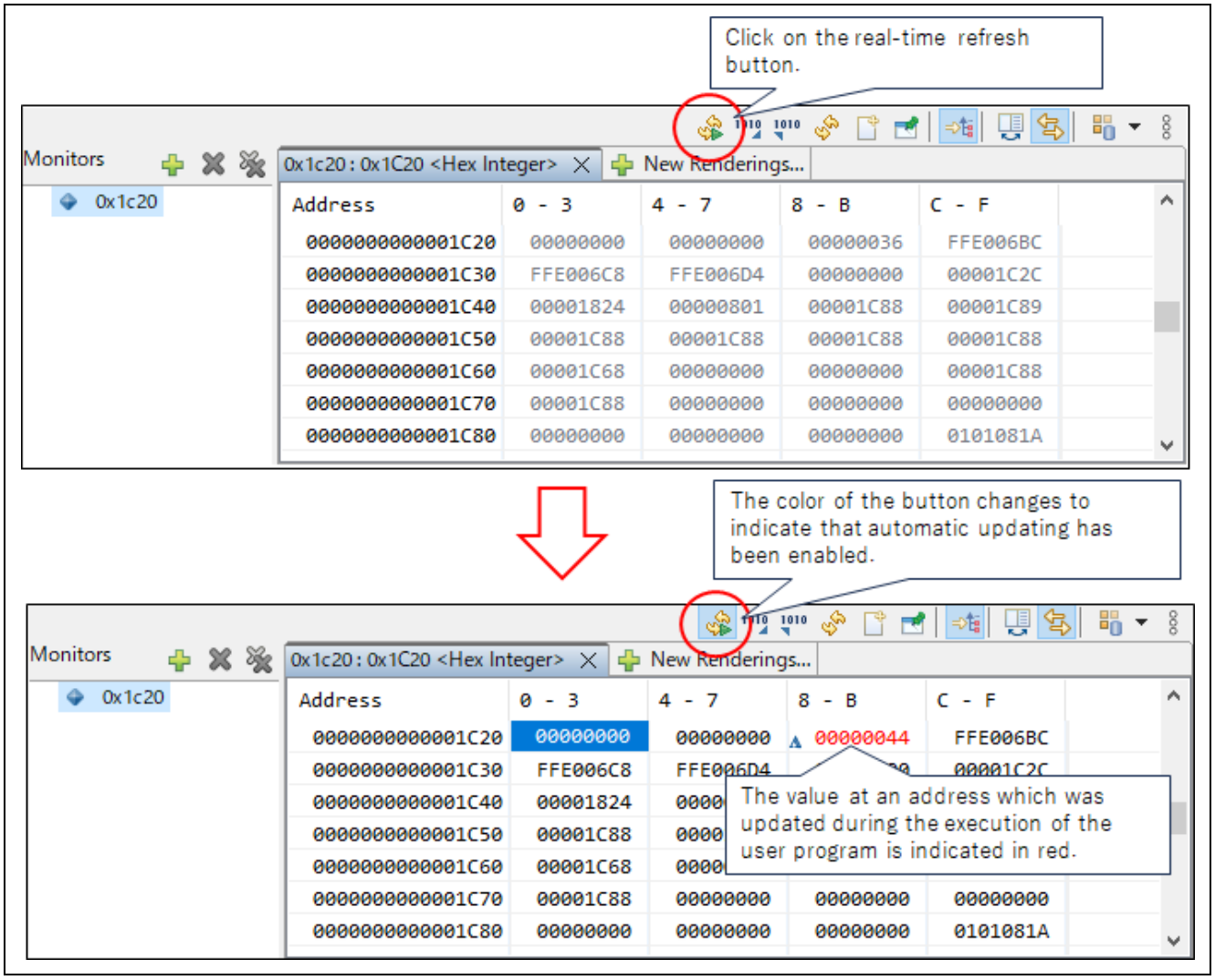


**Figure 6-10 Automatic Updating of the Memory View**

### 6.7.2 Automatic Updating of the Expressions View

In e² studio, enabling or disabling of automatic updating can be specified for the respective variables registered in the Expressions view.

When you select a variable, click on the right mouse button to open the pop-up menu and select [Enable Real-time Refresh]. The icon is shown to the left of a variable which is to be automatically updated during the execution of a program.

In addition, clicking on the button in the Expressions view disables automatic updating of all variables.

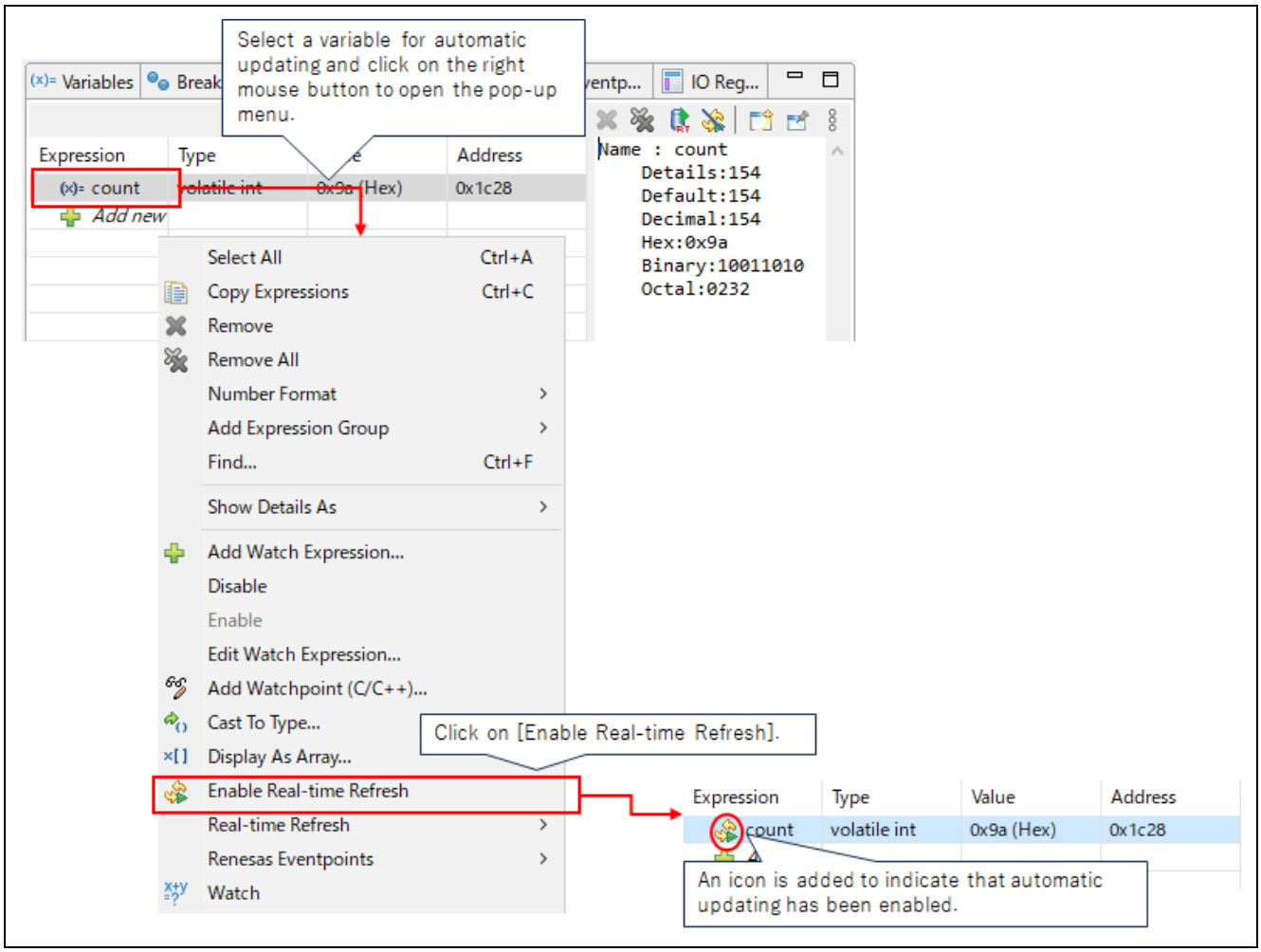


**Figure 6-11 Automatic Updating of the Expressions View**

## 7. Using the Flash Writer Function

e² studio has a mode for only writing to the on-chip flash memory. CS+ is not equipped with this.

To start the debugger, change [Mode] under [Communication Mode] on the [Connection Settings] sub-tabbed page from [Debug Mode] to [Write On Chip Flash Memory]. After that, on the [Startup] tabbed page, specify [No] for [On connect] for the file to be downloaded.

After the debugger has been started, manual downloading proceeds with writing to the flash memory.

If you download the file in [Debug Mode], the debugger will overwrite all bytes of the ID code with 0xFF and rewrite the endian select register in response to the settings of the debugger. However, if you download the file in [Write On Chip Flash Memory] mode, the results will be the same as in the case of using the flash writer to write to the flash memory.

Use this mode when you are confirming the operation of a board as a stand-alone unit.

In [Write On Chip Flash Memory] mode, the user program can be executed without disconnecting the emulator from the board after the debugger session has been terminated.

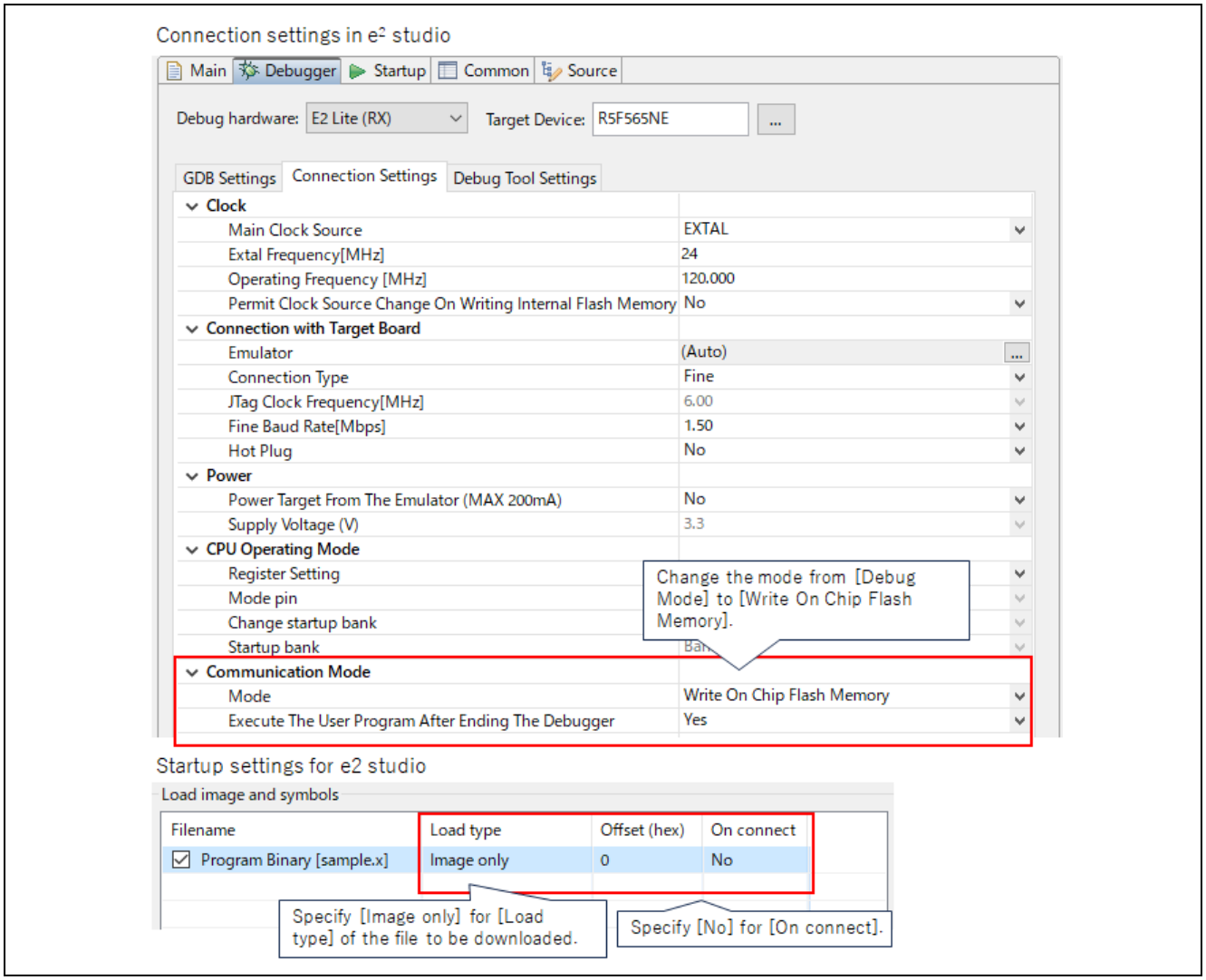


**Figure 7-1 Settings for Only Writing to the On-chip Flash Memory**

For details, also refer to the following FAQ.

[RX program runs via E1/E20 but doesn't work without emulator]

## 8. FAQs

Also refer to the following FAQs, which will be helpful for using e² studio or emulators.

e² studio FAQs

List of know-how when using E1/E20 emulator in CS+ environment (CS+, E2, E2 Lite, E1, E20)

The FAQs on using the E1/E20 emulator in the CS+ environment (CS+, E2, E2 Lite, E1, E20)

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Apr.05.24 | - | First Edition issued |
| | | | |

# 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 1Discharge (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/.

© 2024 Renesas Electronics Corporation. All rights reserved.