

To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: <http://www.renesas.com>

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (<http://www.renesas.com>)

Send any inquiries to <http://www.renesas.com/inquiry>.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of 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. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may 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.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

"Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or system 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

SuperH™ Family E10A-USB Emulator

Additional Document for User's Manual
Supplementary Information on Using
the SH-4A_custom_SoC

Renesas Microcomputer Development
Environment System

SuperH™ Family

E10A-USB for SH-4A_custom_SoC HS0778KCU01HE

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (<http://www.renesas.com>).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.

Contents





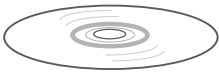
Section 1	Connecting the Emulator with the User System	1
1.1	Components of the Emulator	1
1.2	Connecting the Emulator with the User System	2
1.3	Installing the H-UDI Port Connector on the User System	2
1.4	Pin Assignments of the H-UDI Port Connector	2
1.5	Recommended Circuit between the H-UDI Port Connector and the MPU	2
Section 2	Software Specifications when Using the SH-4A_custom_SoC.....	3
2.1	Differences between the SH-4A_custom_SoC and the Emulator	3
2.2	Specific Functions for the Emulator when Using the SH-4A_custom_SoC	7
2.2.1	Event Condition Functions	7
2.2.2	Trace Functions.....	14
2.2.3	Notes on Using the JTAG (H-UDI) Clock (TCK)	21
2.2.4	Notes on Setting the [Breakpoint] Dialog Box	21
2.2.5	Notes on Setting the [Event Condition] Dialog Box and the BREAKCONDITION_SET Command	22
2.2.6	Note on Setting the UBC_MODE Command	23
2.2.7	Note on Setting the PPC_MODE Command	23
2.2.8	Performance Measurement Function	23

Section 1 Connecting the Emulator with the User System

1.1 Components of the Emulator

The emulator supports the SH-4A_custom_SoC. Table 1.1 lists the components of the emulator.

Table 1.1 Components of the Emulator

Classification	Component	Appearance	Quantity	Remarks
Hardware	Emulator box		1	HS0005KCU01H: Depth: 65.0 mm, Width: 97.0 mm, Height: 20.0 mm, Mass: 72.9 g or HS0005KCU02H: Depth: 65.0 mm, Width: 97.0 mm, Height: 20.0 mm, Mass: 73.7 g
	User system interface cable		1	14-pin type: Length: 20 cm, Mass: 33.1 g
	User system interface cable		1	36-pin type: Length: 20 cm, Mass: 49.2 g (only for HS0005KCU02H)
	USB cable		1	Length: 150 cm, Mass: 50.6 g
Software	E10A-USB emulator setup program, SuperH™ Family E10A-USB Emulator User's Manual, Supplementary Information on Using the SH-4A_custom_SoC*, and Test program manual for HS0005KCU01H and HS0005KCU02H		1	HS0005KCU01SR, HS0005KCU01HJ, HS0005KCU01HE, HS0778KCU01HJ, HS0778KCU01HE, HS0005TM01HJ, and HS0005TM01HE (provided on a CD-R)

Note: Additional document for the MPUs supported by the emulator is included. Check the target MPU and refer to its additional document.

1.2 Connecting the Emulator with the User System

To connect the E10A-USB emulator (hereinafter referred to as the emulator), the H-UDI port connector must be installed on the user system to connect the user system interface cable. When designing the user system, refer to an example of recommended connection between the connector and the MPU shown in this manual. In addition, read the E10A-USB emulator user's manual and hardware manual of the related MPU.

1.3 Installing the H-UDI Port Connector on the User System

Table 1.2 shows the recommended H-UDI port connector for the emulator.

Table 1.2 Recommended H-UDI Port Connector

Connector	Type Number	Manufacturer	Specifications
14-pin connector	2514-6002	Minnesota Mining & Manufacturing Ltd.	14-pin straight type

Note: When designing the 14-pin connector layout on the user board, do not place any components within 3 mm of the H-UDI port connector.

1.4 Pin Assignments of the H-UDI Port Connector

For the pin assignments of the 14-pin H-UDI port connector, ask Renesas Technology Corp. via the sales office.

1.5 Recommended Circuit between the H-UDI Port Connector and the MPU

For a recommended circuit for connection between the H-UDI port connector (14 pins) and the MPU when the emulator is in use, ask Renesas Technology Corp. via the sales office.

Section 2 Software Specifications when Using the SH-4A_custom_SoC

2.1 Differences between the SH-4A_custom_SoC and the Emulator

1. When the emulator system is initiated, it initializes the general registers and part of the control registers as shown in table 2.1. The initial values of the MPU are undefined when it is not connected to the emulator. When the emulator is initiated from the workspace, a value to be entered is saved in a session.

Table 2.1 Register Initial Values at Emulator Link Up

Register	Emulator at Link Up
R0 to R14	H'00000000
R15 (SP)	H'A0000000
R0_BANK to R7_BANK	H'00000000
PC	H'A0000000
SR	H'700000F0
GBR	H'00000000
VBR	H'00000000
MACH	H'00000000
MACL	H'00000000
PR	H'00000000
DBR	H'00000000
SGR	H'00000000
SPC	H'00000000
SSR	H'000000F0
FPUL	H'00000000
FPSCR	H'00040001
FR0 to FR15	H'00000000
XF0 to XF15	H'00000000

2. The emulator uses the H-UDI; do not access the H-UDI.

3. Low-Power States (Sleep and DDR-SDRAM Power-supply Backup)

For low-power consumption, the SH-4A_custom_SoC has sleep and DDR-SDRAM power-supply backup states.

The sleep state is switched using the SLEEP instruction. When the emulator is used, the sleep state can be cleared with either the normal clearing function or with the [STOP] button, and a break will occur.

The emulator does not support the DDR-SDRAM power-supply backup state.

Note: The memory must not be accessed or modified in sleep state.

4. Reset Signals

The MPU reset signals are only valid during emulation started with clicking the GO or STEP-type button. If these signals are enabled on the user system in command input wait state, they are not sent to the MPU.

Note: Do not break the user program when the reset, bus-release request, and wait control signals are being active. A TIMEOUT error will occur. If the wait control or bus-release request signal is fixed to active during break, a TIMEOUT error will occur at memory access.

5. Direct Memory Access Controller (DMAC)

The DMAC operates even when the emulator is used. When a data transfer request is generated, the DMAC executes DMA transfer.

6. Memory Access during User Program Execution

When a memory is accessed from the memory window, etc. during user program execution, the user program is resumed after it has stopped in the emulator to access the memory.

Therefore, realtime emulation cannot be performed.

The stopping time of the user program is as follows:

Environment:

Host computer: 800 MHz (Pentium® III)

JTAG clock: 20 MHz (TCK clock)

When a one-byte memory is read from the command-line window, the stopping time will be about 40 ms. Since the above values are for reference, they differ according to the MPU in use.

7. Memory Access during User Program Break

The emulator can download the program for the flash memory area (for details, refer to section 6.22, Download Function to the Flash Memory Area, in the SuperH™ Family E10A-USB Emulator User's Manual). Other memory write operations are enabled for the RAM area. Therefore, an operation such as memory write or BREAKPOINT should be set only for the RAM area.

8. Cache Operation during User Program Break

When cache is enabled, the emulator accesses the memory by the following methods:

- At memory write: Writes through the cache, then issues a single write to outside. The LRU is not updated.
- At memory read: Reads memory from the cache. The LRU is not updated.

Therefore, when memory read or write is performed during user program break, the cache state does not change.

- At breakpoint set: Disables the instruction cache.

9. UBC

When [User] is specified in the [UBC mode] list box in the [Configuration] dialog box, the UBC can be used in the user program.

Do not use the UBC in the user program as it is used by the emulator when [EML] is specified in the [UBC mode] list box in the [Configuration] dialog box.

10. Memory Access during Break

In the enabled MMU, when a memory is accessed and a TLB error occurs during break, it can be selected whether the TLB exception is controlled or the program jumps to the user exception handler in [TLB Mode] in the [Configuration] dialog box. When [TLB miss exception is enable] is selected, a "Communication Timeout error" will occur if the TLB exception handler does not operate correctly. When [TLB miss exception is disable] is selected, the program does not jump to the TLB exception handler even if a TLB exception occurs. Therefore, if the TLB exception handler does not operate correctly, a "Communication Timeout error" will not occur but the memory contents may not be correctly displayed.

11. Loading Sessions

Information in [JTAG clock] of the [Configuration] dialog box cannot be recovered by loading sessions. Thus the TCK value will be 5 MHz.

12. [IO] Window

- Display and modification

Do not change values of the User Break Controller because it is used by the emulator.

For each watchdog timer register, there are two registers to be separately used for write and read operations.

Table 2.2 Watchdog Timer Register

Register Name	Usage	Register
WTCSR(W)	Write	Watchdog timer control/status register
WTCNT(W)	Write	Watchdog timer counter
WTCSR(R)	Read	Watchdog timer control/status register
WTCNT(R)	Read	Watchdog timer counter

The watchdog timer operates only when the user program is executed. Do not change the value of the frequency change register in the [IO] window or [Memory] window.

The internal I/O registers can be accessed from the [IO] window. When the SH-4A_custom_SoC is used, the I/O-register definition file does not describe all modules. As required, change the I/O-register definition file according to the description in the hardware manual. The I/O-register definition file can be customized depending on its format. Note that, however, the emulator does not support the bit-field function.

- Verify

In the [IO] window, the verify function of the input value is disabled.

13. Illegal Instructions

If illegal instructions are executed by STEP-type commands, the emulator cannot go to the next program counter.

2.2 Specific Functions for the Emulator when Using the SH-4A_custom_SoC

The SH-4A_custom_SoC does not support the following function:

- AUD trace function

2.2.1 Event Condition Functions

The emulator is used to set 12 event conditions (Ch1 to Ch12) and the software trace. Table 2.3 lists the conditions of Event Condition.

Table 2.3 Types of Event Conditions

Event Condition Type	Description
Address bus condition (Address)	Breaks when the MPU address bus value or the program counter value matches the specified value.
Data bus condition (Data)	Breaks when the MPU data bus value matches the specified value. Byte, word, or longword can be specified as the access data size.
Bus state condition (Bus State)	There are two bus state condition settings: Bus state condition: Breaks or acquires a trace when the data bus or the X-Bus or Y-Bus address bus of the MPU is matched. Read/Write condition: Breaks or acquires a trace when the specified read/write condition is matched.
Window address condition	Breaks or acquires a trace when the data in the specified memory range is accessed.
System bus	Breaks or acquires a trace when the address or data on the system bus is matched.
LDTLB instruction event condition	Breaks when the MPU executes the LDTLB instruction.
Count	Breaks when the conditions set are satisfied the specified number of times.
Branch trace condition (Branch trace)	Breaks or acquires a trace when a branch occurs with the condition specified by the MPU. (By default, trace acquisition is enabled).
Software trace	Selects whether or not the software trace is acquired.
Action	Selects the operation when a condition, such as setting a break, trace, or performance start or end, is matched.

Table 2.4 lists the combinations of conditions that can be set under Ch1 to Ch12 and the software trace.

Table 2.4 Dialog Boxes for Setting Event Conditions

Dialog Box	Function										
	Address Bus Condition (Address)	Data Bus Condition (Data)	ASID Condition (ASID)	Bus State Condition (Bus Status)	Window Address Condition (Window address)	System Bus	LDTLB Instruction Break	Count Condition (Count)	Branch Condition (Branch Trace)	Software Trace	Action
[Event Condition 1] dialog box	O	X	O	O	X	X	X	X	X	X	O (B and P)
[Event Condition 2] dialog box	O	O	O	O	X	X	X	O	X	X	O (B and P)
[Event Condition 3] dialog box	O	X	O	X	X	X	X	X	X	X	O (B and P)
[Event Condition 4] dialog box	O	X	O	X	X	X	X	X	X	X	O (B and P)
[Event Condition 5] dialog box	X	X	O	O	O	X	X	X	X	X	O (B, T, and P)
[Event Condition 6] dialog box	X	X	O	O	O	X	X	X	X	X	O (B, T, and P)
[Event Condition 7] dialog box	X	X	X	X	X	X	O	X	X	X	Break fixed
[Event Condition 8] dialog box	O	X	X	X	X	O	X	X	X	X	O (B, T, and P)

Table 2.4 Dialog Boxes for Setting Event Conditions (cont)

Dialog Box	Function											Action
	Address Bus Condition (Address)	Data Bus Condition (Data)	ASID Condition (ASID)	Bus State Condition (Bus Status)	Window Address Condition (Window address)	System Bus	LDTLB Instruction Break	Count Condition (Count)	Branch Condition (Branch Trace)	Software Trace		
[Event Condition 9] dialog box	O	X	X	X	X	O	X	X	X	X	X	O (B, T, and P)
[Event Condition 10] dialog box	O	X	O	O	X	X	X	X	X	X	X	O (B and P)
[Event Condition 11] dialog box	O	O	O	O	X	X	X	O	X	X	X	O (B and P)
[Event Condition 12] dialog box	X	X	X	X	X	X	X	X	O	X	X	O (B, T, and P)
[Software trace] dialog box	X	X	X	X	X	X	X	X	X	X	O	Trace fixed

- Notes:
1. O: Can be set in the dialog box.
X: Cannot be set in the dialog box.
 2. For the Action item,
B: Setting a break is enabled.
T: Setting a trace is enabled.
P: Setting a performance start or end condition is enabled.

Sequential Setting: In the emulator, sequential setting of an Event Condition is enabled.

Table 2.5 Sequential Event Conditions

	Type	Event Condition	Description
[CPU Sequential Event] Page	2 Channel Sequential	Ch2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 2, 1. An event condition must be set for Ch2 and Ch1.
		Ch4 -> 3	Halts a program when a condition is satisfied in the order of Event Condition 4, 3. An event condition must be set for Ch4 and Ch3.
		Ch6 -> 5	Halts a program when a condition is satisfied in the order of Event Condition 6, 5. An event condition must be set for Ch6 and Ch5.
		Ch11 -> 10	Halts a program when a condition is satisfied in the order of Event Condition 11, 10. An event condition must be set for Ch11 and Ch10.
Many Channel Sequential		Ch3 -> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 3, 2, 1. An event condition must be set for Ch3, Ch2, and Ch1.
		Ch4 -> 3-> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 4, 3, 2, 1. An event condition must be set for Ch4, Ch3, Ch2, and Ch1.
		Ch5 -> 4 -> 3-> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 5, 4, 3, 2, 1. An event condition must be set for Ch5, Ch4, Ch3, Ch2, and Ch1.
		Ch6 -> 5 -> 4 -> 3-> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 6, 5, 4, 3, 2, 1. An event condition must be set for Ch6, Ch5, Ch4, Ch3, Ch2, and Ch1.
		Ch10 -> 6 -> 5 -> 4 -> 3-> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 10, 6, 5, 4, 3, 2, 1. An event condition must be set for Ch10, Ch6, Ch5, Ch4, Ch3, Ch2, and Ch1.
		Ch11 -> 10 -> 6 -> 5 -> 4 -> 3-> 2 -> 1	Halts a program when a condition is satisfied in the order of Event Condition 11, 10, 6, 5, 4, 3, 2, 1. An event condition must be set for Ch11, Ch10, Ch6, Ch5, Ch4, Ch3, Ch2, and Ch1.

Table 2.5 Sequential Event Conditions (cont)

	Type	Event Condition	Description
[CPU Sequential Event] Page (cont)	CPU Extend		Expands the [CPU Sequential Extend] page. The sequential setting is enabled with any combination. For details, refer to section 2.2.1, Sequential Break Extension Setting, in this manual.
[SystemBus Sequential Event] Page	SystemBus Sequential Event	Ch9 -> 8	Halts a program when a condition is satisfied for Event Condition 9, 8. An event condition must be set for Ch9 and Ch8.
		Ch8 -> 9	Halts a program when a condition is satisfied for Event Condition 8, 9. An event condition must be set for Ch8 and Ch9.
	SystemBus Extend		Expands the [SystemBus Sequential Extend] page. The sequential setting is enabled with any combination. For details, refer to section 2.2.1, Sequential Break Extension Setting, in this manual.

Sequential Break Extension Setting:

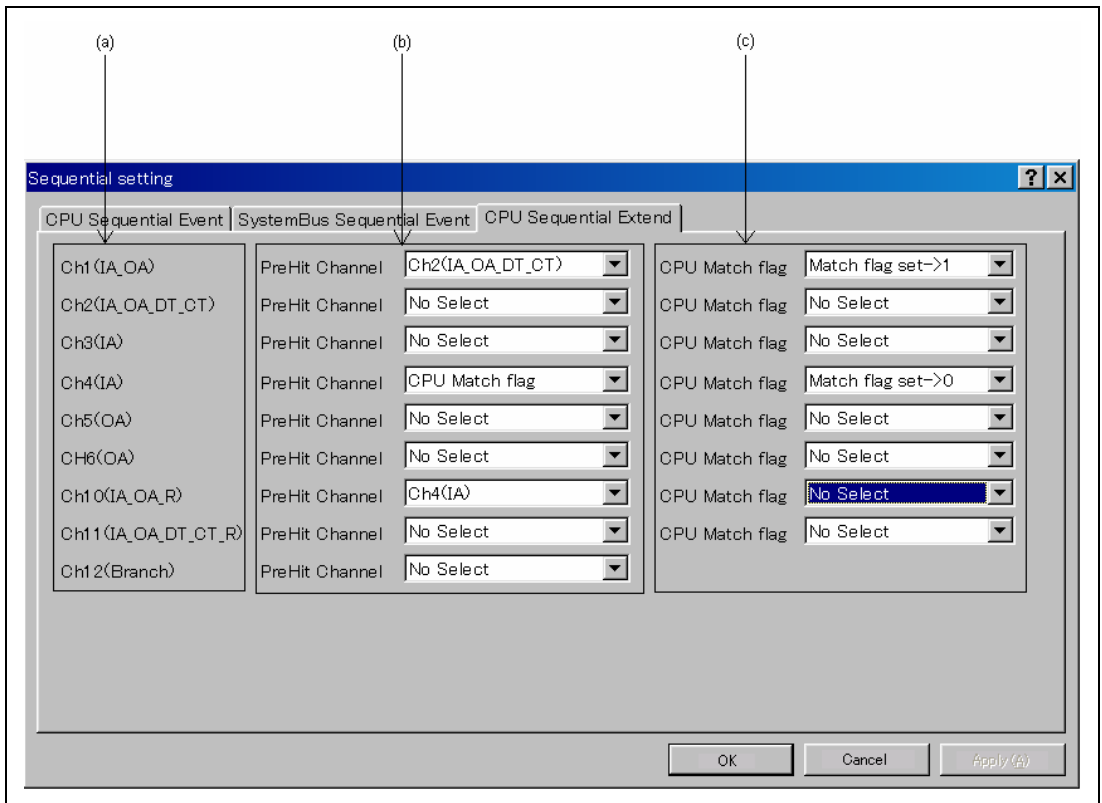


Figure 2.1 [CPU Sequential Extend] Page

- (a) Indicates the channel name for setting conditions.
- (b) Selects a condition that is satisfied before the channel which sets up conditions.
When a channel name is selected, it is required that the condition of the channel selected here must have already been satisfied.
When [CPU Match flag] is selected, the CPU match flag must be set.
When a condition is selected by the channel selected here, no break will occur.
- (c) When a condition is satisfied, the CPU match flag is set or cleared.
When a program breaks, the CPU match flag is initialized.

Set the event condition for each channel in the [Event Condition] dialog box; this also applies to the [System Bus Sequential Extend] page.

Usage Example of Sequential Break Extension Setting: A tutorial program provided for the product is used as an example. For the tutorial program, refer to section 6, Tutorial, in the SuperH™ Family E10A-USB Emulator User's Manual.

The conditions of Event Condition are set as follows:

1. Ch1
Breaks address H'00001068 when the condition [Prefetch address break after executing] is satisfied.
2. Ch2
Breaks address H'00001058 when the condition [Prefetch address break after executing] is satisfied.
3. Ch4
Breaks address H'0000107a when the condition [Prefetch address break after executing] is satisfied.
4. Ch10
Breaks address H'00001086 when the condition [Prefetch address break after executing] is satisfied.
Note: Do not set other channels.
5. Set the [CPU Sequential Extend] page as shown in figure 2.1.

Then, set the program counter and stack pointer (PC = H'00000800, R15 = H'00010000) in the [Registers] window and click the [Go] button. If this does not execute normally, issue a reset and execute the above procedures.

The program is executed up to the condition of Ch10 and halted. Here, the condition is satisfied in the order of Ch2 -> 1 -> 4 -> 10.

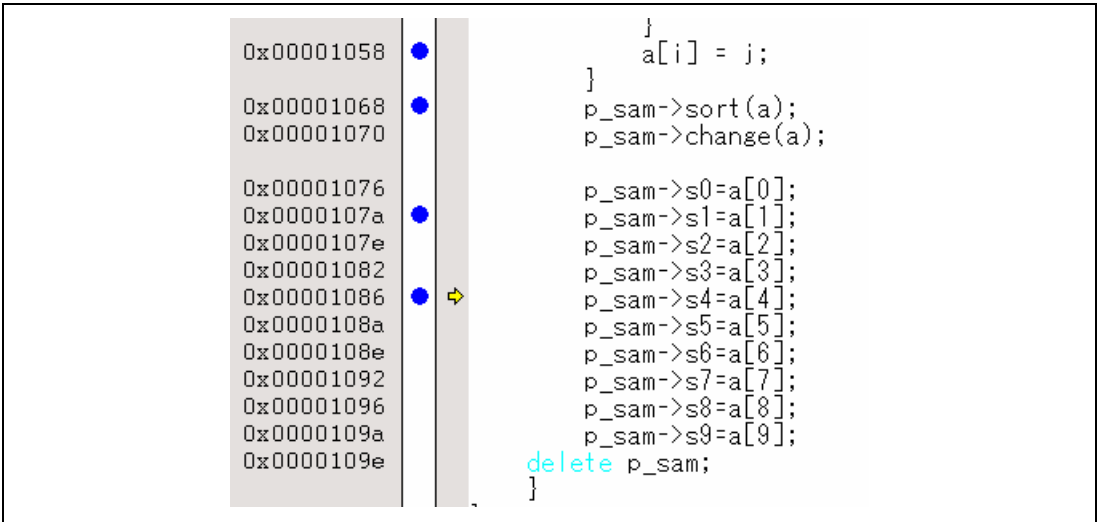


Figure 2.2 [Source] Window at Execution Halted (Sequential Break)

2.2.2 Trace Functions

The emulator supports the trace functions listed in table 2.6.

Table 2.6 Trace Functions

Function	Internal Trace	Memory Output Trace
Branch trace	Supported (eight branches)	Supported
Range memory access trace	Supported (eight events)	Supported
Software trace	Supported (eight events)	Supported

Branch Trace Functions: The branch source and destination addresses, their source lines, branch types, and types of accessed bus masters are displayed.

[Setting Method]

Select the check box in the [Branch] group box in the [Branch trace] page of the [Branch trace] dialog box that opens by double-clicking on the Ch12 (Branch) column of the [Eventpoint] window. The branch condition to be acquired can be set.

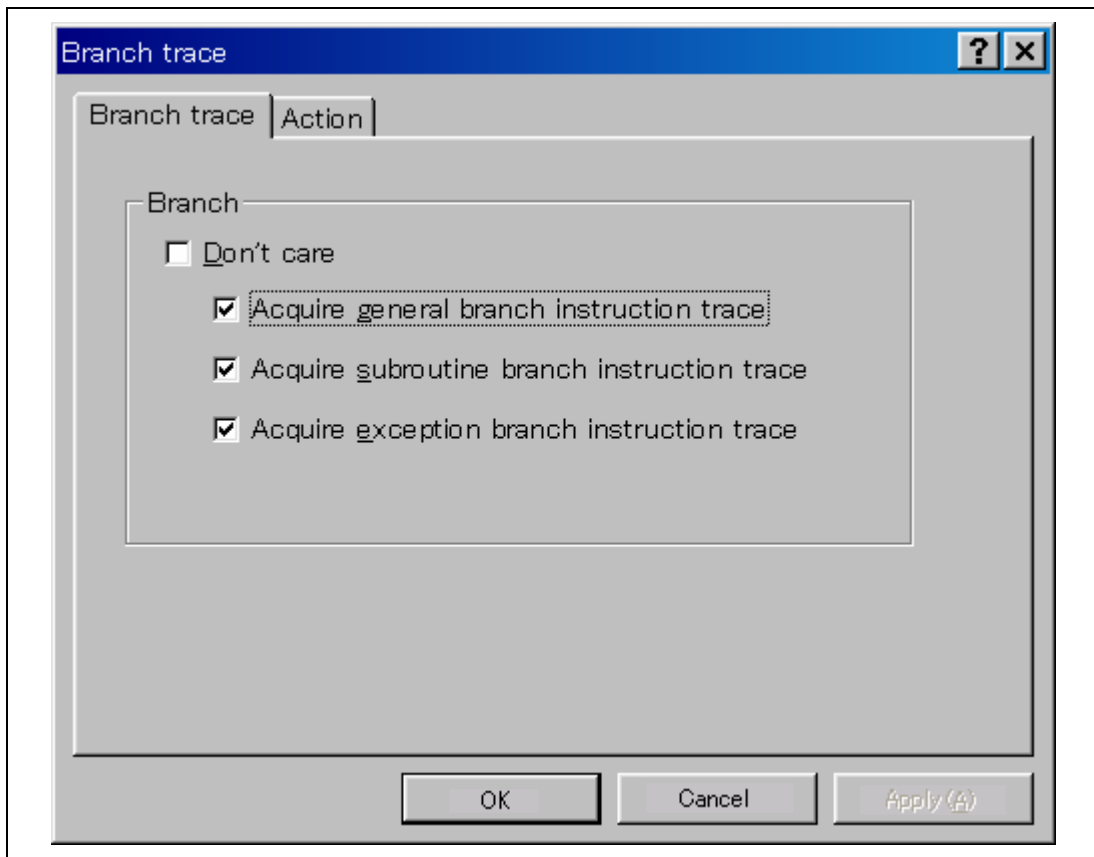


Figure 2.3 [Branch trace] Dialog Box

A branch trace can be acquired by selecting the [Acquire trace] check box of the [Action] page.

Note: To cancel settings, select [Delete] from the popup menu that is opened by clicking on the Ch12 (Branch) column with the right-mouse button.

Range Memory Access Trace Functions: The memory access within the specified range is acquired by a trace. The read cycle, write cycle, or read/write cycle can be selected as the bus type, ASID value, or bus cycle for trace acquisition.

[Setting Method]

- (i) To open the [Event condition 5] or [Event condition 6] dialog box, double-click on the Ch5 (OA) or Ch6 (OA) column of the [Eventpoint] window.
- (ii) Remove the check mark of the [Don't care] check box in the [Window address] page and enter the memory range to be set.

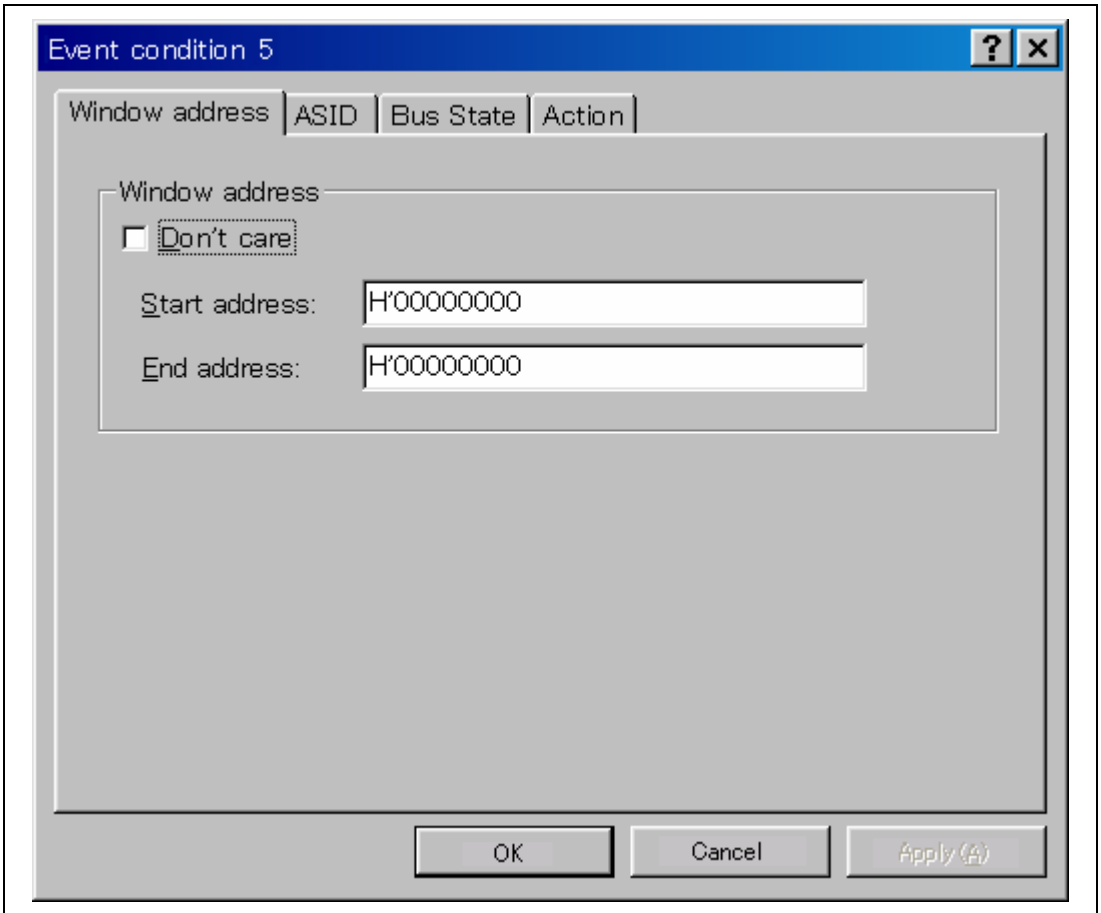


Figure 2.4 [Window address] Page

- (iii) Open the [ASID] page, remove the check mark of the [Don't care] check box, and enter the ASID value to be set.
When the ASID value is not set as a condition, do not remove the check mark of the [Don't care] check box.
- (iv) Open the [Bus state] page and specify the bus type and bus cycle that are to be set.

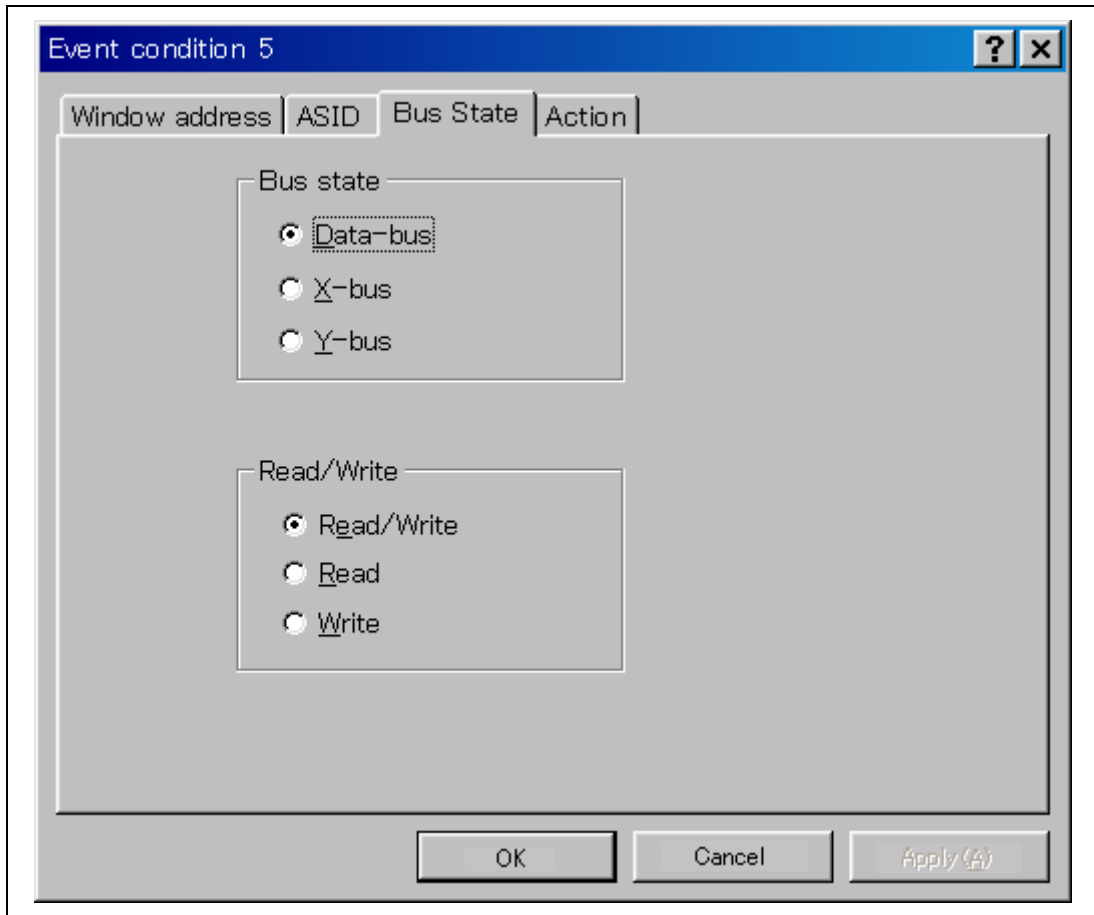


Figure 2.5 [Bus State] Page

- (v) Selecting the [Acquire trace] check box in the [Action] page enables acquiring memory access within the range.

Note: To cancel settings, select the popup menu that is opened by clicking on the Ch5 (OA) or Ch6 (OA) column with the right-mouse button.

Software Trace Function:

Note: This function can be supported with SHC/C++ compiler (manufactured by Renesas Technology Corp.; including OEM and bundle products) V6.0 or later.

However, SHC/C++ compiler (including OEM and bundle products) V8.0 or later is needed when instructions other than those compatible with SH4 are output.

When a specific instruction is executed, the PC value at execution and the contents of one general register are acquired by trace. Describe the Trace(x) function (x is a variable name) to be compiled and linked beforehand. For details, refer to the SuperH™ RISC engine C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual.

When the load module is downloaded on the emulator and is executed while a software trace function is valid, the PC value that has executed the Trace(x) function, the general register value for x, and the source lines are displayed.

To activate the software trace function, select the [Acquire Software trace] radio button in the [Software trace] dialog box that is opened by double-clicking on the software Trace column of the [Eventpoint] window.

Note: To cancel settings, select the [Don't care] radio button in the [Software trace] dialog box or select [Delete] from the popup menu that is opened by clicking on the software Trace column with the right-mouse button.

Internal Trace Function: This function is activated by selecting the [Internal trace] radio button in the [Trace type] group box of the [Trace mode] page. Set the trace condition to be used.

Notes: 1. If an interrupt is generated at the program execution start or end, including a step operation, the emulator address may be acquired. In such a case, the following message will be displayed. Ignore this address because it is not a user program address.

*** EML ***

2. If a completion-type exception occurs during exception branch acquisition, the next address to the address in which an exception occurs is acquired.
3. Trace information cannot be acquired for the following branch instructions:
 - The BF and BT instructions whose displacement value is 0
 - Branch to H'A0000000 by reset

Memory Output Trace Function: This function is activated by selecting the [Use Memory trace] radio button in the [Trace type] group box of the [Trace mode] page.

In this function, write the trace data in the specified user memory range.

Specify the start address to output a trace for the [Start] edit box in the [User memory area] group box, and the end address for the [End Address] edit box. Set the trace condition to be used.

Table 2.7 shows the memory-output trace acquisition mode that can be set in each trace function.

Table 2.7 Memory-Output Trace Acquisition Mode

Type	Mode	Description
Continuous trace occurs	Realtime trace	When the next branch occurs while the trace information is being output, all the information may not be output. The user program can be executed in realtime, but some trace information will be lost.
	Non realtime trace	When the next branch occurs while the trace information is being output, the CPU stops operations until the information is output. The user program is not executed in realtime.
Trace buffer full	Trace continue	This function overwrites the oldest trace information to store the latest trace information.
	Trace stop	After the trace buffer becomes full, the trace information is no longer acquired. The user program is continuously executed.

To set the memory-output trace acquisition mode, click the [Trace] window with the right mouse button and select [Setting] from the pop-up menu to display the [Acquisition] dialog box. The memory-output trace acquisition mode can be set in the [Trace mode1] or [Trace mode2] group box in the [Trace mode] page of the [Acquisition] dialog box.

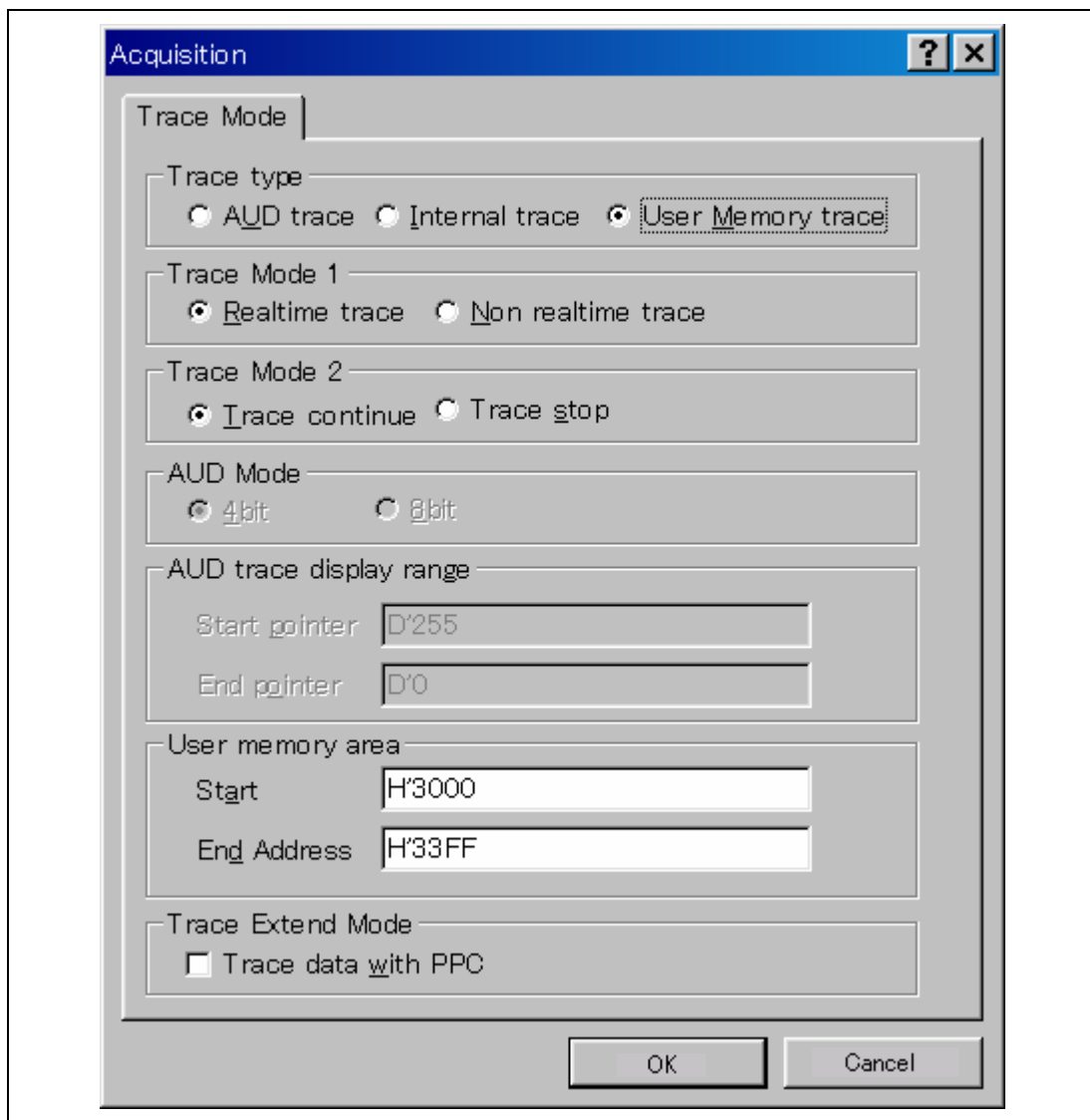


Figure 2.6 [Trace Mode] Page

- Notes:
1. The memory range for which trace is output is the address on the system bus and not supported for the MMU or cache.
 2. In the memory range for output, do not specify the ranges that the user program has been downloaded or the user program accesses.
 3. Do not specify the internal RAM area for the output range.
 4. The range for trace output must be 1 MB or less.

2.2.3 Notes on Using the JTAG (H-UDI) Clock (TCK)

1. Set the JTAG clock (TCK) frequency to lower than the frequency of the SH-4A_custom_SoC peripheral module clock (CKP).
2. The set value of the JTAG clock (TCK) is initialized by executing [Reset CPU] or [Reset Go]. Thus the TCK value will be 5 MHz.

2.2.4 Notes on Setting the [Breakpoint] Dialog Box

1. When an odd address is set, the next lowest even address is used.
2. A BREAKPOINT is accomplished by replacing instructions of the specified address. Accordingly, it can be set only to the RAM areas in CS0 to CS6 and the internal RAM areas. A BREAKPOINT cannot be set to the following addresses:
 - ROM areas in CS0 to CS6
 - Areas other than CS0 to CS6 except for the internal RAM
 - A slot instruction of a delayed branch instruction
 - An area that can be only read by MMU
3. During step operation, BREAKPOINTS are disabled.
4. When execution resumes from the address where a BREAKPOINT is specified, single-step operation is performed at the address before execution resumes. Therefore, realtime operation cannot be performed.
5. When a BREAKPOINT is set to the slot instruction of a delayed branch instruction, the PC value becomes an illegal value. Accordingly, do not set a BREAKPOINT to the slot instruction of a delayed branch instruction.
6. When the [Normal] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog box, a BREAKPOINT is set to a physical address or a virtual address according to the MPU's MMU status during command input when the VPMAP_SET command setting is disabled. The ASID value of the MPU's PTEH register during command input is used. When VPMAP_SET command setting is enabled, a BREAKPOINT is set to a physical address into which address translation is made according to the VP_MAP table. However, for addresses out of the range of the VP_MAP table, the address to which a BREAKPOINT is set depends on the MPU's MMU status during command input. Even when

the VP_MAP table is modified after BREAKPOINT setting, the address translated when the BREAKPOINT is set valid.

7. When the [Physical] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog box, a BREAKPOINT is set to a physical address. A BREAKPOINT is set after disabling the MPU's MMU upon program execution. After setting, the MMU is returned to the original state. When a break occurs at the corresponding virtual address, the cause of termination displayed in the status bar and the [Output] window is ILLEGAL INSTRUCTION, not BREAKPOINT.
8. When the [Virtual] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog box, a BREAKPOINT is set to a virtual address. A BREAKPOINT is set after enabling the MPU's MMU upon program execution. After setting, the MMU is returned to the original state. When an ASID value is specified, the BREAKPOINT is set to the virtual address corresponding to the ASID value. The emulator sets the BREAKPOINT after rewriting the ASID value to the specified value, and returns the ASID value to its original value after setting. When no ASID value is specified, the BREAKPOINT is set to a virtual address corresponding to the ASID value at command input.
9. An address (physical address) to which a BREAKPOINT is set is determined when the BREAKPOINT is set. Accordingly, even if the VP_MAP table is modified after BREAKPOINT setting, the BREAKPOINT address remains unchanged. When a BREAKPOINT is satisfied with the modified address in the VP_MAP table, the cause of termination displayed in the status bar and the [Output] window is ILLEGAL INSTRUCTION, not BREAKPOINT.
10. If an address of a BREAKPOINT cannot be correctly set in the ROM or flash memory area, a mark ● will be displayed in the [BP] area of the address on the [Source] or [Disassembly] window by refreshing the [Memory] window, etc. after Go execution. However, no break will occur at this address. When the program halts with the event condition, the mark ● disappears.

2.2.5 Notes on Setting the [Event Condition] Dialog Box and the BREAKCONDITION_SET Command

1. When [Go to cursor], [Step In], [Step Over], or [Step Out] is selected, the settings of Event Condition 3 are disabled.
2. When an Event Condition is satisfied, emulation may stop after two or more instructions have been executed.
3. If a PC break address condition is set to the slot instruction after a delayed branch instruction, user program execution cannot be terminated before the slot instruction execution; execution stops before the branch destination instruction.

2.2.6 Note on Setting the UBC_MODE Command

In the [Configuration] dialog box, if [User] is set while the [UBC mode] list box has been set, Ch10 (IA_OA_R) and Ch11 (OA_OA_CT_R) of Event Condition cannot be used.

2.2.7 Note on Setting the PPC_MODE Command

In the [Configuration] dialog box, if [User] is set while the [PPC mode] list box has been set, Ch1 and Ch2 of the performance analysis function and options 1 and 2 of the profile function cannot be used.

2.2.8 Performance Measurement Function

The emulator supports the performance measurement function.

1. Setting the performance measurement conditions

To set the performance measurement conditions, use the [Performance Analysis] dialog box and the PERFORMANCE_SET command. When a channel line on the [Performance Analysis] window is clicked with the right mouse button, the popup menu is displayed and the [Performance Analysis] dialog box is displayed by selecting [Setting].

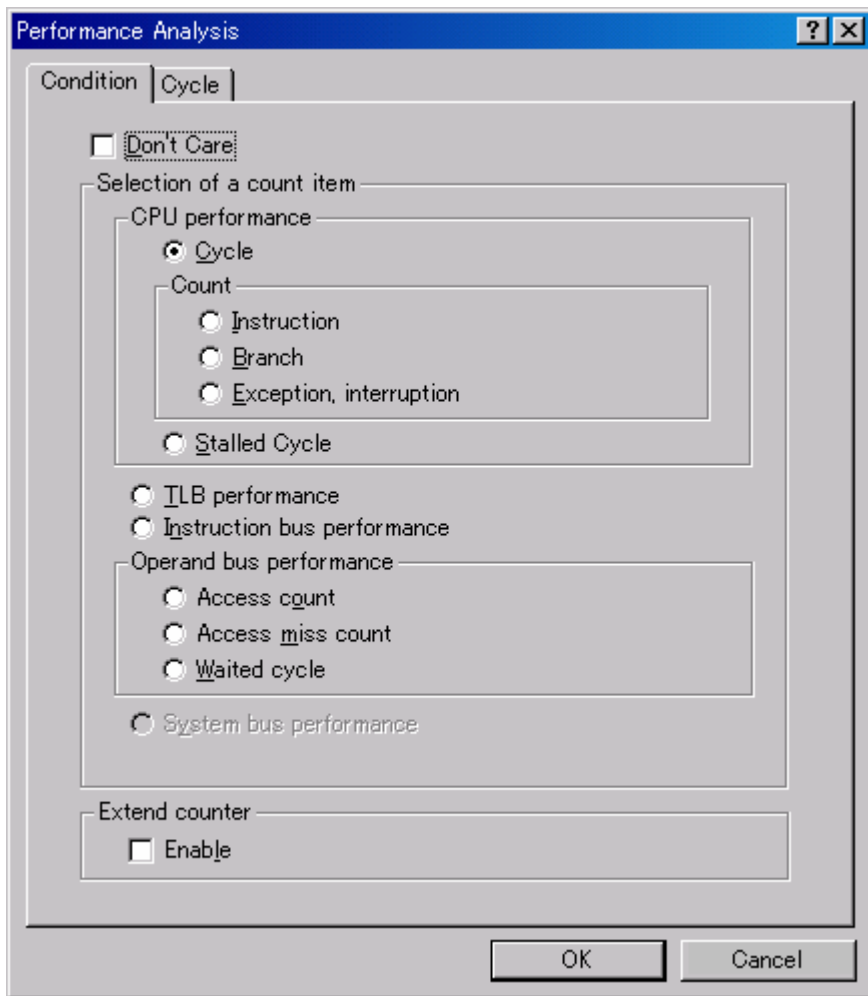


Figure 2.7 [Performance Analysis] Dialog Box

Note: For the command line syntax, refer to the online help.

(a) Specifying the measurement start/end conditions

Set the performance measurement conditions in the [Action] page after conditions have been set in the [Event Condition] dialog box that is opened by double-clicking Ch1 to Ch6 and Ch8 to Ch12 on the [Event Condition] sheet of the [Eventpoint] window.

- Notes:
1. When no measurement start/end conditions are specified, measurement is started by executing a program and ended when an event condition is satisfied.
 2. When only the measurement start or end condition is specified, performance cannot be measured. Be sure to specify both of the measurement start and end conditions.
 3. When the measurement start/end conditions are specified, step operation cannot be performed.

Table 2.8 Conditions Specified in the [Action] Page

Item		Description
PA1	pa1_start_point	Specifies the conditions of Event Condition that has been set as the measurement start condition of performance channel 1.
	pa1_end_point	Specifies the conditions of Event Condition that has been set as the measurement end condition of performance channel 1.
PA2	pa2_start_point	Specifies the conditions of Event Condition that has been set as the measurement start condition of performance channel 2.
	pa2_end_point	Specifies the conditions of Event Condition that has been set as the measurement end condition of performance channel 2.
PA3	pa3_start_point	Specifies the conditions of Event Condition that has been set as the measurement start condition of performance channel 3.
	pa3_end_point	Specifies the conditions of Event Condition that has been set as the measurement end condition of performance channel 3.
PA4	pa4_start_point	Specifies the conditions of Event Condition that has been set as the measurement start condition of performance channel 4.
	pa4_end_point	Specifies the conditions of Event Condition that has been set as the measurement end condition of performance channel 4.

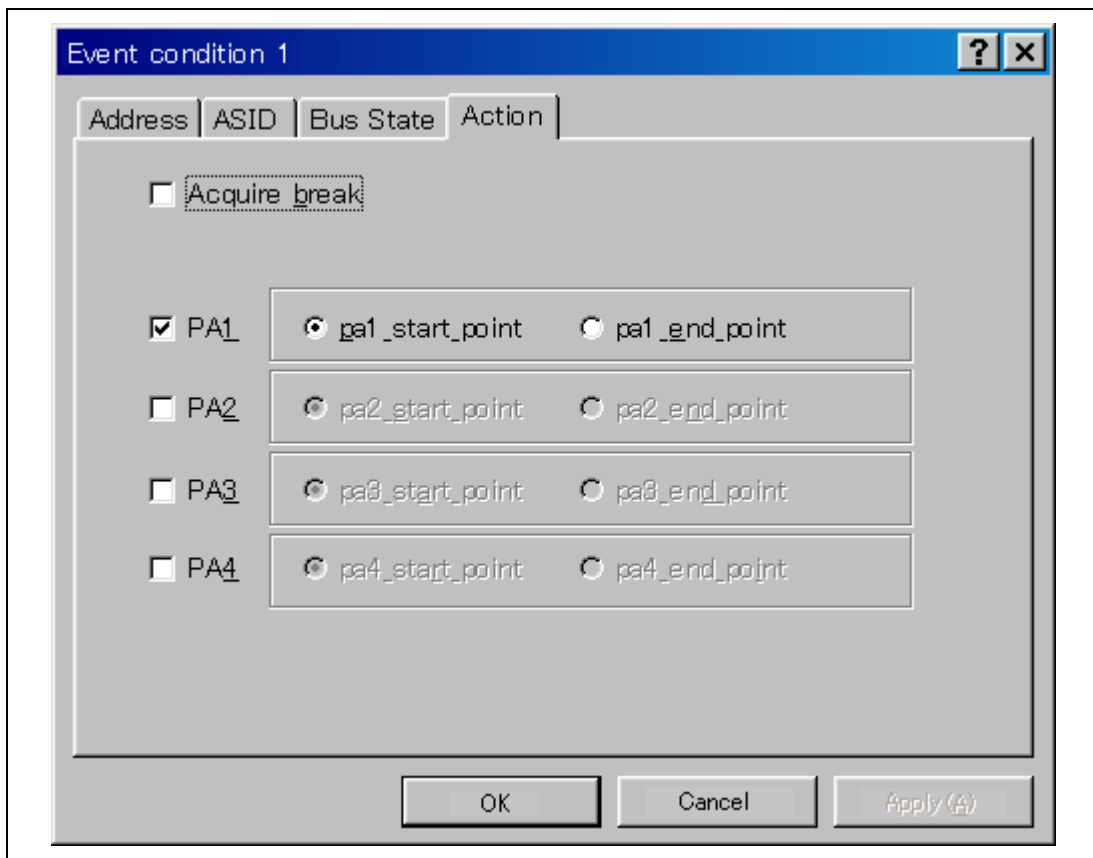


Figure 2.8 [Action] Page

Note: PA1 or PA2 cannot be set for Ch8 and Ch9.

(b) Measurement tolerance

- The measured value includes tolerance.
- Tolerance will be generated before or after a break.

For details, see table 2.11.

(c) Measurement items

Items are measured in the [Performance Analysis] dialog box for each channel from Ch1 to Ch4. A maximum of four conditions can be specified at the same time. Table 2.9 shows the measurement items. (Options in table 2.9 are parameters for <mode> of the PERFORMANCE_SET command. They are displayed in CONDITION of the [Performance Analysis] window.)

Table 2.9 Measurement Items

Classification	Type	Measurement Item	Option	Note
Disabled			None	Not measured.
CPU performance	Cycle	Elapsed cycles	AC	Except for power-on period; counted by the CPU clock.
		Cycles executed in privileged mode	PM	The number of privileged-mode cycles among the number of elapsed cycles.
		Cycles for asserting the SR.BL bit	BL	The number of cycles when the SR.BL bit = 1 among the number of elapsed cycles.
	Instruction	Number of effective instructions issued	I	The number of execution instructions = number of valid instructions issued + number of cases of simultaneous execution of two instructions. The number of valid instructions means the number of completed instructions.
		Number of 2 instruction executed simultaneously	2I	The number of times that two instructions are executed simultaneously among the valid instructions issued.
	Branch	Number of unconditional branch	BT	The number of unconditional branches other than branches occurring after an exception. However, RTE is counted.
	Exception, interruption	Number of exceptions accepted	EA	Interrupts are included.
		Number of interrupts accepted	INT	NMI is included.
		Number of UBC channel hit	UBC	Performs OR to count the number of channel-hits in the CPU.

Table 2.9 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
CPU performance (cont)	Stalled cycle	Cycles stalled in full-trace mode (with multi-counts)	SFM	All items are counted independently.
		Cycles stalled in full-trace mode (without multi-counts)	SF	This item is not counted if the stall cycle is generated simultaneously with a stall cycle that has occurred due to instruction execution.
TLB performance	TLB	Number of UTLB miss for instruction fetch	UMI	The number of TLB-miss exceptions generated by an instruction fetch (number of EXPEVT sets).
		Number of UTLB miss for operand fetch	UMO	The number of TLB-miss exceptions generated by an operand access (number of EXPEVT sets).
		Number of ITLB miss	IM	The number of ITLB misses for valid accesses (does not include UTLB hits or misses).
Instruction bus performance	Instruction	Number of memory accesses for instruction fetch	MIF	The number of memory accesses by an instruction fetch. Accesses canceled by an instruction-fetch bus are not counted. Instruction fetches, which have been fetched in anticipation of a branch but not actually executed, are counted. Accesses by the PREFI instruction are included.
		Number of instruction cache access	IC	The number of accesses for an instruction cache during memory access of the opcode.

Table 2.9 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
Instruction bus performance (cont)	Instruction (cont)	Number of instruction cache miss	ICM	The number of cache misses by an instruction cache access (the number of accesses to the outside of the CPU core due to a cache miss).
		Number of internal-RAM access for instruction fetch (XY-RAM or L memory)	XL	The number of accesses for the XY or L memory in the MPU during memory accesses of the opcode.
Operand bus performance	Access count	Number of memory access for operand fetch (READ)	MR	The number of memory accesses by an operand read (equal to loading on the operand bus). Accesses by the PREF instruction or canceled accesses are not included.
		Number of memory access for operand fetch (WRITE)	MW	The number of memory accesses by an operand write (equal to storing memory on the operand bus). Canceled accesses are not included.
		Number of operand cache access (READ)	CR	The number of operand-cache reads during memory access (read) of an operand.
		Number of operand cache access (WRITE)	CW	The number of operand-cache reads during memory access (write) of an operand.
		Number of internal-RAM access for operand fetch (READ) (XY-RAM or L memory)	XLR	The number of accesses to XY or L memory in the MPU during memory access (read) of an operand. (Accesses via the XY bus and the operand bus are included. When MOVX and MOVY are executed simultaneously, it increments one count regardless of the read or write.)

Table 2.9 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
Operand bus performance (cont)	Access count (cont)	Number of internal-RAM access for operand fetch (WRITE) (XY-RAM or L memory)	XLW	The number of accesses to XY or L memory in the MPU during memory access (write) of an operand. (Accesses via the XY bus and the operand bus are included. When MOVX and MOVS are executed simultaneously, it increments one count regardless of the read or write.)
	Access miss count	Number of operand cache miss (READ)	CMR	The number of cache misses by an operand cache access (read) (number of accesses to the outside of the CPU core due to a cache miss). Cache misses are not counted by the PREF instruction.
		Number of operand cache miss (WRITE)	CMW	The number of cache misses by an operand cache access (write) (number of accesses to the outside of the CPU core due to a cache miss). Write-through accesses are not counted. Cache misses are not counted by the PREF instruction.
	Waited cycle	Waited cycles for operand fetch (READ)	WOR	The number of wait cycles by a memory access (read) of an operand.
		Waited cycles for operand fetch (WRITE)	WOW	The number of wait cycles by a memory access (write) of an operand.
		Waited cycles for operand cache miss (READ)	WCMR	The number of wait cycles by an operand cache miss (read) (however, the number of wait cycles of cache FIII is included due to contention).
		Waited cycles for operand cache miss (WRITE)	WCMW	The number of wait cycles by an operand cache miss (write).

Table 2.9 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
System bus performance (only available for Ch3 and Ch4)	System bus	Number of requests	RQ	The number of valid bus cycles (cells) is counted by the system bus clock.
		Number of responses	RS	The number of valid bus cycles (cells) is counted by the system bus clock.
		Waited cycles for request	WRQ	The cycles for an issued request (req), that no acceptance signal (gnt) is issued to, are counted by the system bus clock. Even if the waits are issued simultaneously for multiple requests, they are counted as 1.
		Waited cycles for response	WRS	The cycles for an issued response (r_req), that no acceptance signal (r_gnt) is issued to, are counted by the system bus clock. Even if the waits are issued simultaneously for multiple requests, they are counted as 1.

Table 2.10 shows the measurement items and methods that are mainly used.

Table 2.10 Main Measurement Items

Main Measurement Item	Measurement Method
Elapsed time	Number of elapsed cycles x CPU clock cycles
Number of execution instructions	Number of valid instructions issued + number of cases of simultaneous execution of two instructions
Number of interrupts accepted	Number of exceptions accepted
Number of instruction fetches (for both cache and non-cache)	Number of memory accesses in an opcode
Instruction-cache hit ratio	(Number of instruction-cache accesses – instruction-cache miss counts)/instruction-cache access counts
Number of operand accesses (for both cache and non-cache)	Number of memory accesses in an operand (read) + number of memory accesses in an operand (write)
Operand-cache hit ratio (read)	(Number of operand-cache accesses (read) – number of operand-cache misses (read))/number of operand-cache accesses (read)
Operand-cache hit ratio (write)	(Number of operand-cache accesses (write) – number of operand-cache misses (write))/ number of operand-cache accesses (write)
Operand-cache hit ratio	(Number of operand-cache accesses (read) + number of operand-cache accesses (write) – number of operand-cache misses (read) – number of operand-cache misses (write))/(number of operand-cache accesses (read) + number of operand-cache accesses (write))
System bus: occupied rate of request bus	(The equivalent CPU clock value of the number of requests)/number of elapsed cycles
System bus: occupied rate of response bus	(The equivalent CPU clock value of the number of responses)/number of elapsed cycles

Each measurement condition is also counted when conditions in table 2.11 are generated.

Table 2.11 Performance Measurement Conditions to be Counted

Measurement Condition	Notes
No caching due to the settings of TLB cacheable bit	Counted for accessing the cacheable area.
Cache-on counting	Accessing the non-cacheable area is counted less than the actual number of cycles and counts. Accessing the cacheable, X/Y-RAM, and U-RAM areas is counted more than the actual number of cycles and counts.
Branch count	The counter value is incremented by 2. This means that two cycles are valid for one branch.

- Notes:
1. In the non-realtime trace mode of the memory output trace, normal counting cannot be performed because the generation state of the stall or the execution cycle is changed.
 2. Since the clock source of the counter is the CPU clock, counting also stops when the clock halts in the sleep mode.

(d) Extension setting of the performance-result storing counter

The 32-bit counter stores the result of performance, and two counters can be used as a 64-bit counter.

To set a 64-bit counter, check the [Enable] check box in the [Extend counter] group box of the [Performance Analysis] dialog box for Ch1 and Ch3.

2. Displaying the result of performance

The result of performance is displayed in the [Performance Analysis] window or the PERFORMANCE_ANALYSIS command in hexadecimal (32 bits).

However, when the extension counter is enabled, it is displayed in hexadecimal (64 bits).

Note: If a performance counter overflows as a result of measurement, “*” will be displayed for upper bits.

3. Initializing the measured result

To initialize the measured result, select [Initialize] from the popup menu in the [Performance Analysis] window or specify INIT with the PERFORMANCE_ANALYSIS command.

**SuperH™ Family E10A-USB Emulator
Additional Document for User's Manual
Supplementary Information on Using the SH-4A_custom_SoC**

Publication Date: Rev.1.00, September 14, 2006
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Customer Support Department
Global Strategic Communication Div.
Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan



RENESAS SALES OFFICES

<http://www.renesas.com>

Refer to "<http://www.renesas.com/en/network>" for the latest and detailed information.

Renesas Technology America, Inc.

450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.

Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.

7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.

10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.

1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.

Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd

Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

**SuperH™ Family E10A-USB Emulator
Additional Document for User's Manual
Supplementary Information on Using the
SH-4A_custom_SoC**



Renesas Electronics Corporation

1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ10J1464-0100