Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Cautions

Hitachi Microcomputer Development Environment System

SH7750 E10A Emulator

User's Manual

RENESAS

ADE-702-196A

Rev. 2.0
01/15/01
Hitachi, Ltd.

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.

2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.

3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.

4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.

5. This product is not designed to be radiation resistant.

6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.

7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

IMPORTANT INFORMATION

READ FIRST

• **READ this user's manual before using this emulator product.**

• <u>**KEEP the user's manual handy for future reference.**</u>

Do not attempt to use the emulator product until you fully understand its mechanism.

Emulator Product:

Throughout this document, the term "emulator product" shall be defined as the following products produced only by Hitachi, Ltd. excluding all subsidiary products.

• Emulator

• User system interface cable

The user system or a host computer is not included in this definition.

Purpose of the Emulator Product:

This emulator product is a software and hardware development tool for systems employing the Hitachi microcomputer. This emulator product must only be used for the above purpose.

Limited Applications:

This emulator product is not authorized for use in MEDICAL, atomic energy, aeronautical or space technology applications without consent of the appropriate officer of a Hitachi sales company. Such use includes, but is not limited to, use in life support systems. Buyers of this emulator product must notify the relevant Hitachi sales offices before planning to use the product in such applications.

Improvement Policy:

Hitachi, Ltd. (including its subsidiaries, hereafter collectively referred to as Hitachi) pursues a policy of continuing improvement in design, performance, and safety of the emulator product. Hitachi reserves the right to change, wholly or partially, the specifications, design, user's manual, and other documentation at any time without notice.

Target User of the Emulator Product:

This emulator product should only be used by those who have carefully read and thoroughly understood the information and restrictions contained in the user's manual. Do not attempt to use the emulator product until you fully understand its mechanism.

It is highly recommended that first-time users be instructed by users that are well versed in the operation of the emulator product.

RENESAS

LIMITED WARRANTY

Hitachi warrants its emulator products to be manufactured in accordance with published specifications and free from defects in material and/or workmanship. Hitachi, at its option, will replace any emulator products returned intact to the factory, transportation charges prepaid, which Hitachi, upon inspection, shall determine to be defective in material and/or workmanship. The foregoing shall constitute the sole remedy for any breach of Hitachi's warranty. See the Hitachi warranty booklet for details on the warranty period. This warranty extends only to you, the original Purchaser. It is not transferable to anyone who subsequently purchases the emulator product from you. Hitachi is not liable for any claim made by a third party or made by you for a third party.

DISCLAIMER

HITACHI MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, ORAL OR WRITTEN, EXCEPT AS PROVIDED HEREIN, INCLUDING WITHOUT LIMITATION THEREOF, WARRANTIES AS TO MARKETABILITY, MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR USE, OR AGAINST INFRINGEMENT OF ANY PATENT. IN NO EVENT SHALL HITACHI BE LIABLE FOR ANY DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES OF ANY NATURE, OR LOSSES OR EXPENSES RESULTING FROM ANY DEFECTIVE EMULATOR PRODUCT, THE USE OF ANY EMULATOR PRODUCT, OR ITS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCEPT AS EXPRESSLY STATED OTHERWISE IN THIS WARRANTY, THIS EMULATOR PRODUCT IS SOLD "AS IS ", AND YOU MUST ASSUME ALL RISK FOR THE USE AND RESULTS OBTAINED FROM THE EMULATOR PRODUCT.

State Law:

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may have other rights which may vary from state to state.

The Warranty is Void in the Following Cases:

Hitachi shall have no liability or legal responsibility for any problems caused by misuse, abuse, misapplication, neglect, improper handling, installation, repair or modifications of the emulator product without Hitachi's prior written consent or any problems caused by the user system.

All Rights Reserved:

This user's manual and emulator product are copyrighted and all rights are reserved by Hitachi. No part of this user's manual, all or part, may be reproduced or duplicated in any form, in hard-copy or machine-readable form, by any means available without Hitachi's prior written consent.

Other Important Things to Keep in Mind:

1. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.

2. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi.

Figures:

Some figures in this user's manual may show items different from your actual system.

Device names:

Sections 1 to 5 in this user's manual use SHxxxx as an example of the device names.

Limited Anticipation of Danger:

Hitachi cannot anticipate every possible circumstance that might involve a potential hazard. The warnings in this user's manual and on the emulator product are therefore not all inclusive. Therefore, you must use the emulator product safely at your own risk.

SAFETY PAGE

READ FIRST

• **READ this user's manual before using this emulator product.**

• **KEEP the user's manual handy for future reference.**

Do not attempt to use the emulator product until you fully understand its mechanism.

DEFINITION OF SIGNAL WORDS

⚠ **This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.**

⚠ DANGER   **DANGER** indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ WARNING   **WARNING** indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.

⚠ CAUTION   **CAUTION** indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury.

CAUTION   **CAUTION** used without the safety alert symbol indicates a potentially hazardous situation which, if not avoided, may result in property damage.

NOTE emphasizes essential information.

RENESAS

⚠ WARNING

Observe the precautions listed below. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

1. **Do not repair or remodel the emulator product by yourself for electric shock prevention and quality assurance.**

2. **Always switch OFF the host computer and user system before connecting or disconnecting any CABLES or PARTS.**

3. **Connect the connectors in the user system and in the user interface cable by confirming the correct direction.**

4. **If the PCI interface board for the E6000 or E8000 emulator (HS6000EIC01H) and the E10A emulator PCI card are mounted on the same host computer, the connectors may be illegally connected.**

RENESAS

Warnings on Emulator Usage

Be sure to read and understand the warnings below before using this emulator. Note that these are the main warnings, not the complete list.

⚠ WARNING

Always switch OFF the host computer and user system before connecting or disconnecting any CABLES or PARTS.
Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

CAUTION

Place the host computer and user system so that no cable is bent or twisted. A bent or twisted cable will impose stress on the user interface leading to connection or contact failure.
Make sure that the host computer and the user system are placed in a secure position so that they do not move during use nor impose stress on the user interface.

RENESAS

Preface

Thank you for purchasing the E10A emulator.

CAUTION

READ section 2, Preparation before Use, of this User's Manual before using the emulator product. Incorrect operation will damage the user system and the emulator product.

This emulator is an efficient development tool for software and hardware of user systems based on Hitachi's original microprocessor. The emulator operates using the Hitachi debugging interface (hereafter referred to as the HDI), which is the interface program that runs on Microsoft® Windows® 95, Microsoft® Windows® 98, or Microsoft® Windows NT® operating system.

This manual describes the functions and operating procedures of the E10A emulator. Sections 1 to 5 describe common features of all types of E10A emulators. Section 6 describes supplements to the E10A emulator. Read section 1.1, Warnings, carefully before using the emulator.

This manual consists of six sections. The information contained in each section is summarized below:

- Section 1, Overview, gives the emulator overview.
- Section 2, Preparation before Use, gives instructions for first-time users, such as preparation before use and system connection.
- Section 3, Tutorial, describes HDI operating examples.
- Section 4, Descriptions of Windows, describes HDI windows for operating the emulator.
- Section 5, Command-line Functions describes how to input HDI commands and command types.
- Section 6, SHxxxx E10A Emulator Specifications describes the features of the E10A emulator for each MCU. Section 7 describes the important information of the E10A emulator according to emulator products. Read these sections before using the E10A emulator.

The HDI installation disks are provided by the CD-R. Refer to the descriptions in the manuals of the host computer or operating system.

RENESAS

Related Manuals:

- SH Series Cross Assembler User's Manual
- H Series Linkage Editor User's Manual
- H Series Librarian User's Manual
- SuperH RISC Engine C/C++ Compiler User's Manual
- Hitachi Debugging Interface User's Manual
- Hardware Manual for each MCU
- Programming Manual for each MCU

Notes: 1. IBM PC is a registered trademark of International Business Machines Corporation.

2. Microsoft®, Windows®, and Windows NT® are registered trademarks of Microsoft Corporation in the United States and/or other countries.

Microsoft® Windows® 95 operating system is referred to as Windows® 95 in this user's manual.

Microsoft® Windows® 98 operating system is referred to as Windows® 98 in this user's manual.

RENESAS

Contents

RENESAS

RENESAS

Figures

RENESAS

Tables

RENESAS

RENESAS

Section 1   Overview

The E10A emulator (hereafter referred to as the emulator) is a software and hardware development support tool for application systems using the microprocessor developed by Hitachi, Ltd.

The PCMCIA card emulator or PCI card emulator (hereafter referred to as the card emulator), which is the main unit of the emulator, is connected, through the Hitachi-UDI (user debug interface) port*, to the user system. The user system can be debugged under the conditions similar to the actual application conditions. The emulator enables debugging anywhere indoors or out. The host computer for controlling the emulator must be an IBM PC compatible machine with a PCMCIA type II or PCI slot.

Figures 1.1 and 1.2 show the system configuration using the emulator.

Note: The Hitachi-UDI is an interface compatible with the Joint Test Action Group (JTAG) specifications.
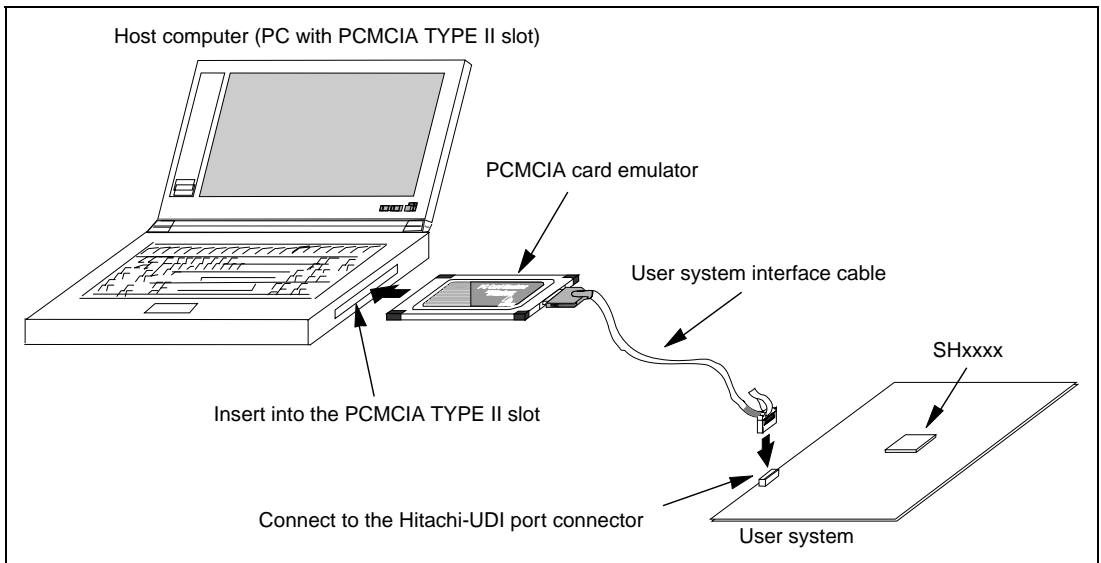
Figure 1.1   System Configuration with the Emulator (PCMCIA Card Emulator Used)
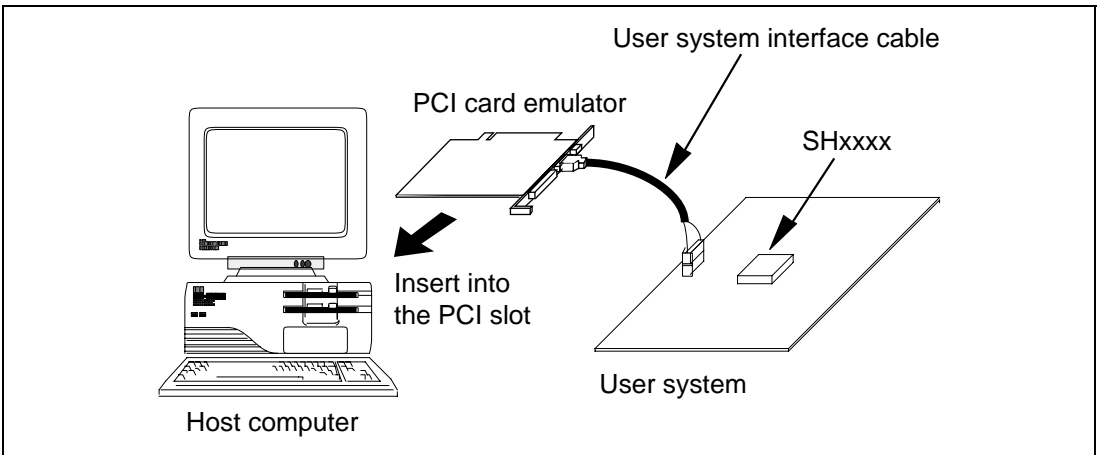
Figure 1.2   System Configuration with the Emulator (PCI Card Emulator Used)

The emulator provides the following features:

- Excellent cost-performance card emulator

  Compactness and low price are implemented using the PCMCIA interface or the PCI interface.

- Realtime emulation

  Realtime emulation of the user system is enabled at the maximum operating frequency of the CPU.

- Excellent operability

  Using the Hitachi Debugging Interface (HDI) on the Microsoft® Windows® 95, Microsoft® Windows® 98, Microsoft® Windows® 2000, and Microsoft® Windows NT® operating systems enables user program debugging using a pointing device such as a mouse. The HDI enables high-speed downloading of load module files.

- Various debugging functions

  Various break and trace functions enable efficient debugging. Breakpoints and break conditions can be set by the specific window, trace information can be displayed on a window, and command-line functions can be used.

- Memory access during emulation

  During emulation, the memory contents can be read and modified.

- Debugging of the user system in the final development stage

  The user system can be debugged under conditions similar to the actual application conditions.

- Compact debugging environment

  When the card emulator specific to the PCMCIA interface is used, a laptop computer can be used as a host computer, creating a debugging environment in any place.

- AUD trace function*

  The AUD trace function enables realtime trace.

RENESAS

Note: The AUD is an abbreviation of the Advanced User Debugger. Support for the AUD varies
with the product.

1.1   Warnings

<div style="border:1px solid black">

CAUTION

**READ the following warnings before using the emulator   product. Incorrect operation will
damage the user system and the emulator product. The USER PROGRAM will be LOST.**

</div>

1. Check all components against the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Protect the emulator from excessive impacts and stresses. For details, refer to section 1.2,
    Environmental Conditions.
4. Do not insert the emulator into any slot (PCMCIA TYPE II slot or PCI slot) other than the
    specified one.
5. When moving the host computer or user system, take care not to vibrate or damage it.
6. After connecting the cable, check that it is connected correctly. For details, refer to section 2,
    Preparation before Use.
7. Supply power to the connected equipment after connecting all cables. Cables must not be
    connected or removed while the power is on.

RENESAS

1.2 Environmental Conditions

```
CAUTION

Observe the conditions listed in tables 1.1 and 1.2 when using the emulator.  Failure to do
so will cause illegal operation in the user system, the emulator product, and the user
program.
```

Table 1.1   Environmental Conditions

Item	Specifications
Temperature	Operating: +10°C to +35°C
	Storage: −10°C to +50°C
Humidity	Operating: 35% RH to 80% RH, no condensation
	Storage: 35% RH to 80% RH, no condensation
Vibration	Operating: 2.45 m/s^2 max.
	Storage: 4.9 m/s^2 max.
	Transportation: 14.7 m/s^2 max.
Ambient gases	No corrosive gases may be present

Table 1.2 lists the acceptable operating environments.

RENESAS

Table 1.2   Operating Environments

Item	Description
Host computer	Built-in Pentium or higher-performance CPU (200 MHz or higher recommended); IBM PC or compatible machine with the PCMCIA TYPE II slot or the PCI slot.
OS	Windows® 95, Windows® 98, Windows® 2000, or Windows NT®
Minimum memory capacity	32 Mbytes or more (double of the load module size recommended)
Hard-disk capacity	Installation disk capacity: 10 Mbytes or more. (Prepare an area at least double the memory capacity (four-times or more recommended) as the swap area.)
Pointing device such as mouse	Connectable to the host computer; compatible with Windows® 95, Windows® 98, Windows® 2000, and Windows NT®.
Power voltage	5.0 ± 0.25 V
Current consumption	HSxxxxKCM01H: 110 mA (max) HSxxxxKCM02H: 230 mA (max) HSxxxxKCI01H: 340 mA (max) HSxxxxKCI02H: 600 mA (max)
CD-ROM drive	Required to install the HDI for the emulator or refer to the emulator user's manual.

RENESAS

1.3   Components

Check all the components unpacking. For details on the E10A emulator components, refer to section 6.1, Components of the Emulator. If the components are not complete, contact a Hitachi sales agency.

RENESAS

Section 2   Preparation before Use

2.1   Emulator Preparation

⚠ **WARNING**

READ the reference sections shaded in figure 2.1 before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

Unpack the emulator and prepare it for use as follows:

	Reference	
Unpack the emulator	Component list	When the emulator is used first.
Check the components against the component list		
Set up the emulator - Install the HDI - Insert the card emulator	Section 2	
Start the HDI	Section 3	When the emulator is used for second time or later.
Turn on the user system		
Input the user system reset signal		
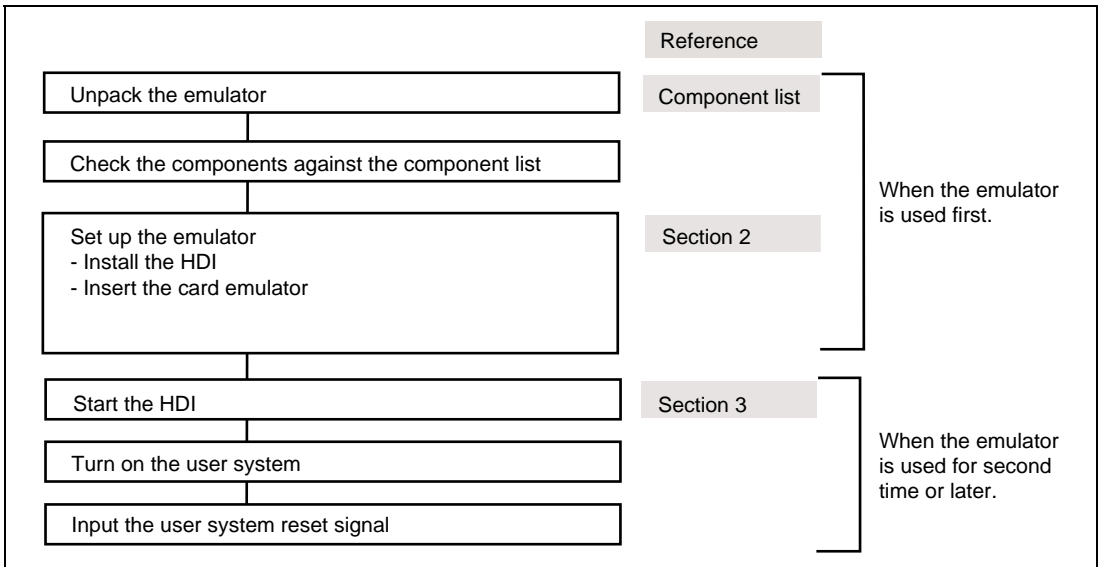
Figure 2.1   Emulator Preparation Flow Chart

RENESAS

2.2 HDI Installation

When the CD-R is inserted in the host computer's CD-ROM drive, the HDI installation wizard is automatically activated (holding the Shift key down while the CD-R is inserted cancels this automatic activation). To run the installation wizard when it has not been automatically activated, execute Setup.exe from the root directory of the CD-R.

Follow the cues given by the installation wizard to install the HDI.

Since hardware settings are also made during installation, the installation procedure differs according to the operating system or interface (PCI or PCMCIA) being used. Follow the installation steps carefully according to the environment you are using.

2.2.1 Installing under Windows®95 and Windows®98 Operating Systems

(1) When the emulator is a PCI card:
1. Install the HDI (when the component type has to be selected during installation, be sure to select [PCI Card Driver]).
2. Shut the operating system down and turn off the power to the host computer.
3. Insert the PCI-card emulator in a slot on the host computer. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
4. Restart the host computer. The hardware is now recognized and the driver is automatically installed.*

(2) When the emulator is a PCMCIA card:
1. Install the HDI (when the component type has to be selected during installation, be sure to select [PC Card Driver (PCMCIA)]).
2. Insert the PCMCIA-card emulator in the host computer's slot. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
3. The hardware is now recognized and the driver is automatically installed.*

Note: When [Add New Hardware Wizard] is displayed, select the [Search for the best driver for your device. (Recommended)] radio button and then the [Specify a location] check box to select the path to be searched for drivers. The location must be specified according to the emulator type, as indicated below:
When using the PCI-card emulator: <Drive>:\DRIVERS\PCI\95
When using the PCMCIA-card emulator: <Drive>:\DRIVERS\PCMCIA\95
(<Drive> is the CD-ROM drive name.)

RENESAS

2.2.2 Installing under Windows NT®4.0 Operating System

(1) When the emulator is a PCI card:

1. Shut the operating system down and turn off the power to the host computer.
2. Insert the PCI-card emulator in a slot on the host computer. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
3. Start the host computer and log-on with an administrator-level user name.
4. Install the HDI. (For a component, be sure to select [PCI Card Driver]. There is a check box for selecting the type name of the product under the [PCI Card Driver] component. Select the appropriate type name. If the correct name is not selected, the correct driver will not be installed, and the emulator will not operate.)
5. Restart the host computer.

(2) When the emulator is a PCMCIA card:

1. Shut the operating system down and turn off the power to the host computer.
2. Insert the PCMCIA-card emulator in the host computer's slot. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
3. Start the host computer and log-on with an administrator-level user name.
4. During HDI installation, the setting value should be checked beforehand because inquiries are made about the resource used by the PCMCIA-card emulator.

   Start the [Start] menu -> [Programs] -> [Administrative Tools (Common)] -> [Windows NT Diagnostics], check the status of the IRQ, I/O port, and memory from the resource panel, and determine the setting values that do not conflict with other devices. (The following resources are used: IRQ: one channel, I/O port: H'F byte, and memory: H'4000 byte.)
5. Install the HDI. (For a component, be sure to select [PC Card Driver (PCMCIA)]. There is a check box for selecting the type name of each product under the [PC Card Driver (PCMCIA)] component. Select the appropriate type name. If the correct name is not selected, the correct driver will not be installed and the emulator will not operate.)
6. Restart the host computer.

Notes: 1. For the SH7729, SH7729R, and SH7622 E10A emulators, there is a check box for selecting the MODEL name that appears on the component list. Select the correct type name.

2. The driver that has been selected in the [Drivers] component starts after the host computer is initiated. If the host computer is initiated with the card disconnected or with the incorrect driver installed, the driver cannot initiate and the service control manager informs the system of an error. This, however, is not a problem.

RENESAS

2.2.3 Installing under Windows®2000 Operating System

(1) When the emulator is a PCI card:
1. Log-on with an administrator-level user name.
2. Install the HDI. (When a component is selected, be sure to select [PCI Card Driver].)
3. Shut the operating system down and turn off the power to the host computer.
4. Insert the PCI-card emulator in a slot on the host computer. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
5. Restart the host computer and log-on with an administrator-level user name. The hardware is now recognized and the driver is automatically installed.*

(2) When the emulator is a PCMCIA card:
1. Log-on with an administrator-level user name.
2. Install the HDI. (When a component is selected, be sure to select [PC Card Driver (PCMCIA)].)
3. Insert the PCMCIA-card emulator in the host computer's slot. Refer to section 2.3, Connecting the Card Emulator to the Host Computer.
4. The hardware is now recognized and the driver is automatically installed.*

Note: When [Found New Hardware Wizard] is displayed, select the [Search for a suitable driver for my device (recommended).] radio button and then the [Specify a location] check box to select the path to be searched for drivers. The location must be specified according to the emulator type, as indicated below:
When using the PCI-card emulator: <Drive>:\DRIVERS\PCI\2000
When using the PCMCIA-card emulator: <Drive>:\DRIVERS\PCMCIA\2000
(<Drive> is the CD-ROM drive name.)

RENESAS

2.3 Connecting the Card Emulator to the Host Computer

Insert the card emulator, according to its type, in a PCMCIA TYPE II slot or PCI slot on the host computer (figures 2.2 and 2.3).

Note: **When using Windows®95, Windows®98, or Windows®2000, be sure to install the HDI before putting the card emulator in place.**

Host computer (PC with the PCMCIA TYPE II slot)

PCMCIA card emulator

Insert into the PCMCIA TYPE II slot
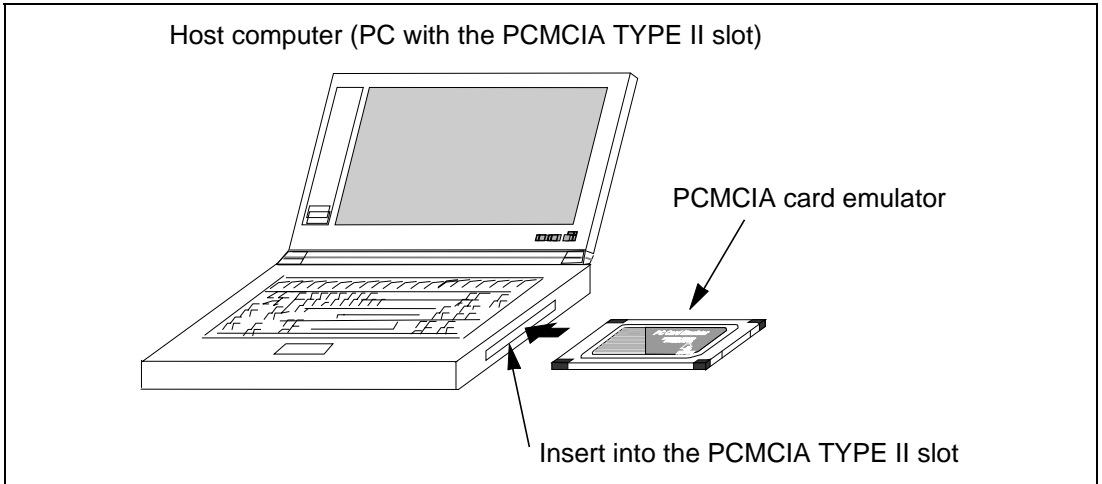
Figure 2.2   Inserting the PCMCIA Card Emulator in the Host Computer's Slot

PCI card emulator

Insert into the PCI slot
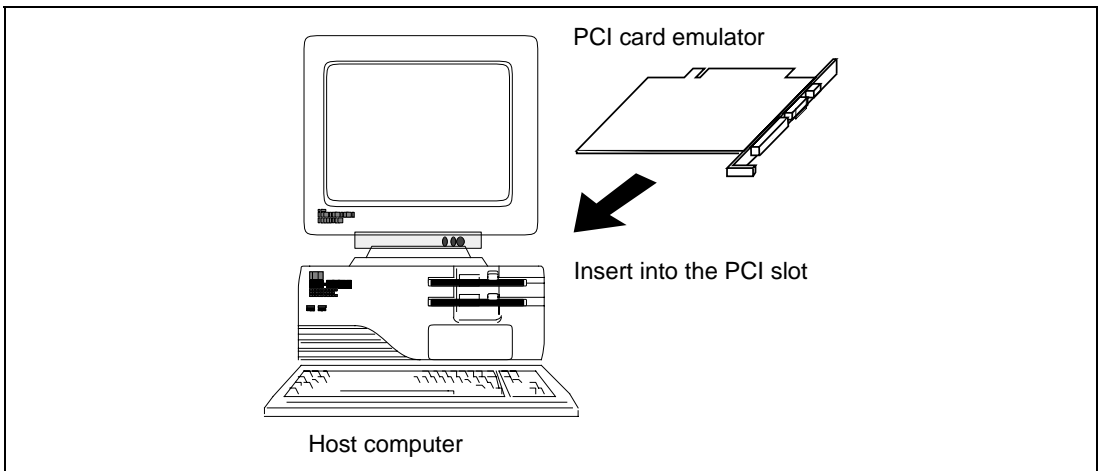
Host computer

Figure 2.3   Inserting the PCI Card Emulator in the Host Computer's Slot

Use the procedure, described in section 2.4, to connect the emulator to the user system with the user system interface cable, or to disconnect them when moving the emulator or the user system.

⚠ **WARNING**

When inserting the PCI-card emulator, note the following. Failure to do so will damage the host computer.

1. Turn off the host computer.

2. Insert the emulator into the PCI slot in parallel.

3. Screw in the emulator after checking the connector and cable positions.

2.4 Connecting the Card Emulator to the User System

(1) The Hitachi-UDI port connector must be installed to the user system. Table 2.1 shows the recommended Hitachi-UDI port connector for the emulator.

Table 2.1 Recommended Hitachi-UDI Port Connector

Connector	Type Number	Manufacturer	Specifications
14-pin connector	2514-6002	Minnesota Mining & Manufacturing Ltd.	14-pin straight type
36-pin connector	DX10M-36S	Hirose Electric Co., Ltd.	Screw type
	DX10M-36SE, DX10GM-36SE		Lock-pin type

Note: When the 14-pin connector is used, do not install any components within 3 mm of the Hitachi-UDI port connector.
When the 36-pin connector is used, do not connect other signal lines to the Hitachi-UDI port connector.

(2) Note that the TDO signal of the user system interface cable connector must be connected to the TDI pin of the Hitachi-UDI port connector and the TDI signal of the user system interface cable connector must be connected to the TDO pin of the Hitachi-UDI port connector. Section 6.2 shows the pin arrangement of the Hitachi-UDI port connector.

(3) Figure 2.4 shows how to connect the user system interface cable to the user system when the 14-pin straight type connector is used. Connect the ground line of the cable to the user system ground. The end of the ground line has a hole having a diameter of 3 mm, and therefore, when the ground line is screwed to the user system, the screw diameter must be 3 mm.

RENESAS
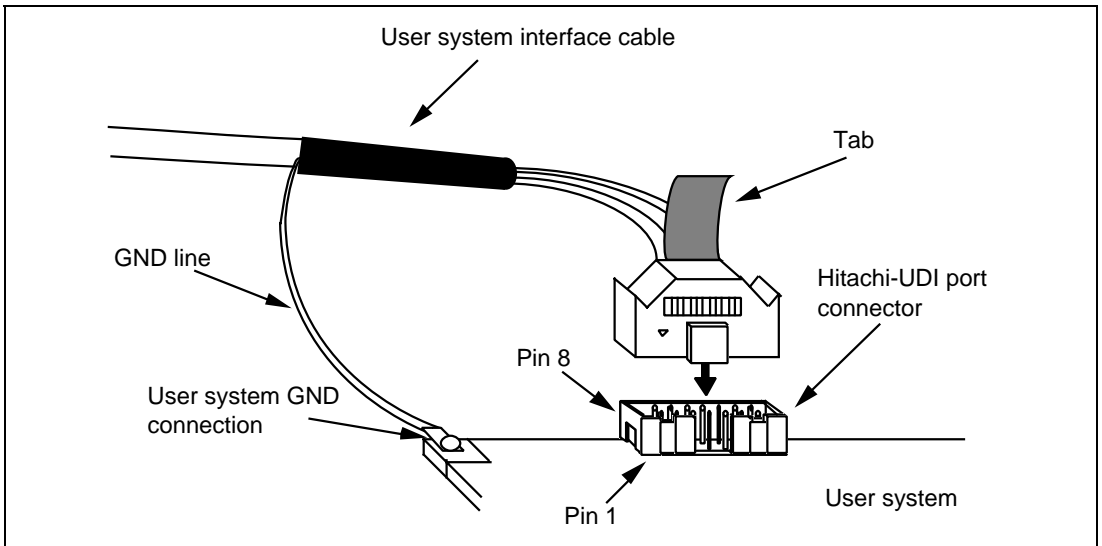
**Figure 2.4   Connecting the User System Interface Cable to the User System
when the 14-pin Straight Type Connector is Used**

Notes: 1. To connect the signals output from the Hitachi-UDI port connector, refer to the device pin alignment.

2. To remove the user system interface cable from the user system, pull the tab on the connector upward.

3. The range of frequencies that the Hitachi-UDI operates at is different according to the devices used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).

4. Connect the Hitachi-UDI signals from the Hitachi-UDI port connector directly to the device.

5. When developing user systems, do not connect the TDI and TDO signals of the device to the boundary scan loop, or separate them by using a switch (figure 2.5).
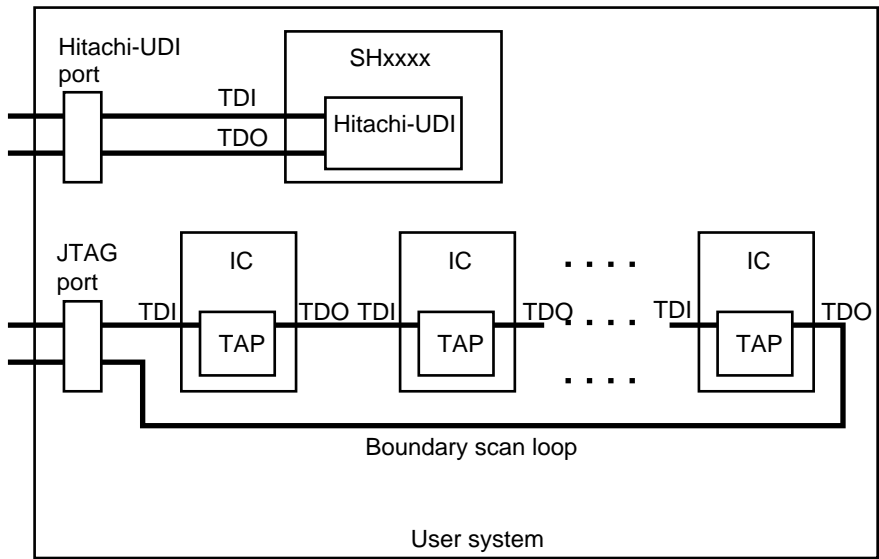
Figure 2.5   User System Example

RENESAS

2.5　System Check

When the HDI program is executed, use the procedure below to check that the emulator is operating correctly.

1. Check that the emulator card is inserted in the host computer's slot.
2. Connect the user system interface cable to the connector of the card emulator.
3. Connect the user system interface cable to the Hitachi-UDI port connector.
4. Supply power to the host computer and select [HDI for E10A SHxxxx] -> [Hitachi Debugging Interface] from the [Start] menu.
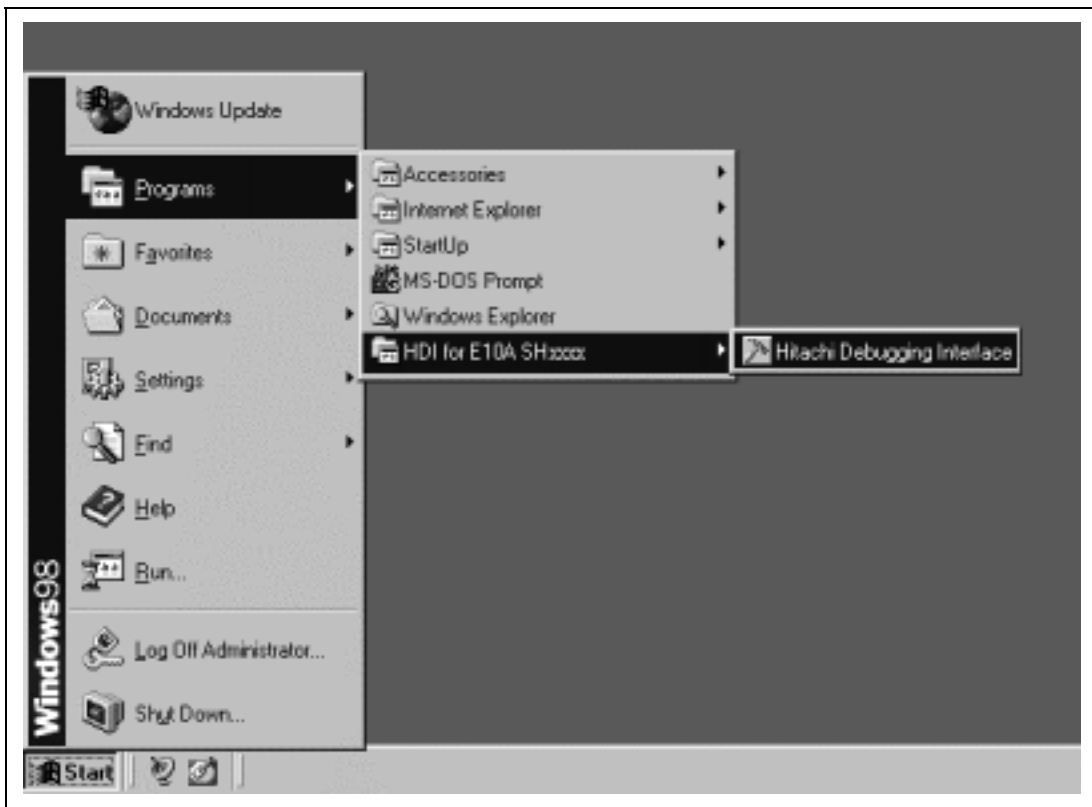
Figure 2.6　[Start] Menu

RENESAS

5. Select the setting to be used.
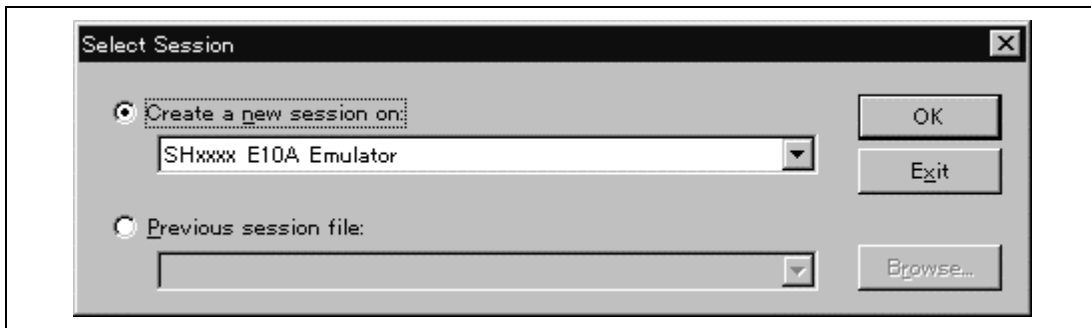
Figure 2.7 [Select Session] Dialog Box

RENESAS

6. The [E10A Driver Details] dialog box is displayed. With the [Driver] combo box, select the driver to connect the HDI with the emulator. [Interface] displays the interface name of the PC interface board to be connected, and [Channel] displays the interface to which the board is connected. Once the driver is selected in the [E10A Driver Details] dialog box, this dialog box is not displayed when the HDI is run next time. (This procedure will not be executed by target devices.)
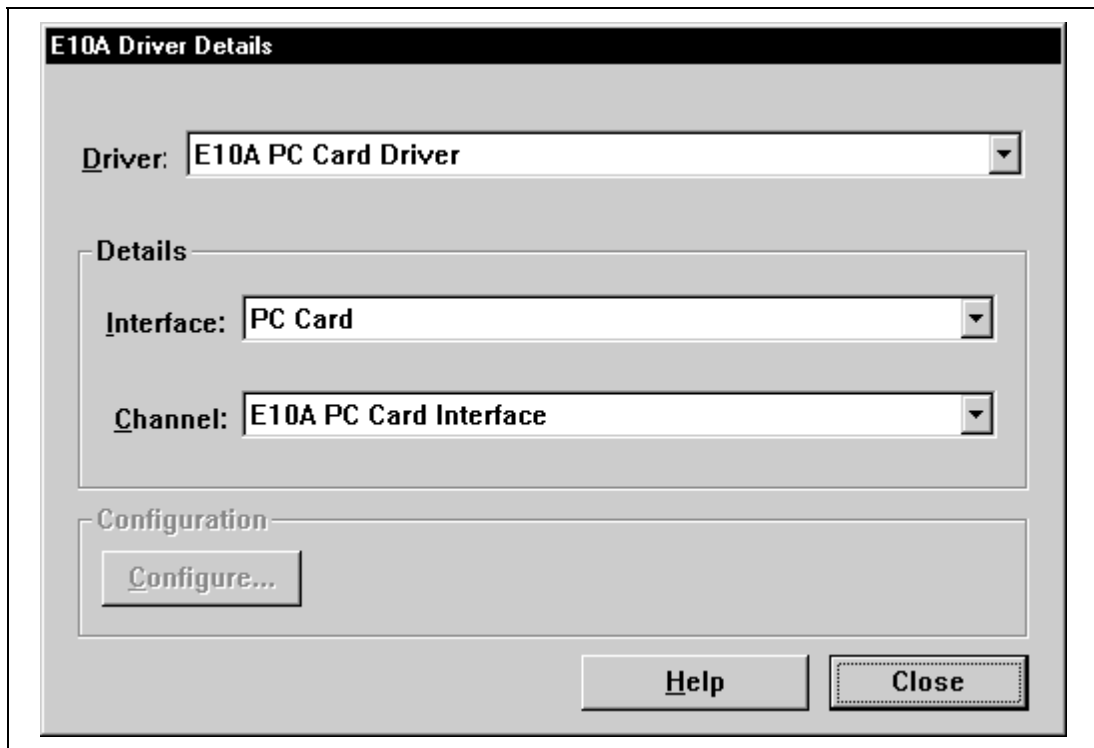
Figure 2.8   [E10A Driver Details] Dialog Box

— With the [Driver] combo box, select the driver to connect the HDI with the emulator.

— [Interface] displays the interface name of the card emulator to be connected, and [Channel] displays the interface to which the board is connected.

[Driver] combo box: Select [E10A PC Card Driver] to use the PCMCIA card emulator.

Select [E10A PCI Card Driver] to use the PCI card emulator. For details, refer to table 6.3 in section 6.5.1, Emulator Driver Selection.

[Interface] combo box: Select [PC Card] to use the PCMCIA card emulator.

[PCI] is displayed to use the PCI card emulator. (If the driver is not installed, the [PC Card] or [PCI] is not displayed.)

— Click the [Close] button.

RENESAS

7. The HDI window is displayed, and the dialog box is displayed as shown in figure 2.9.
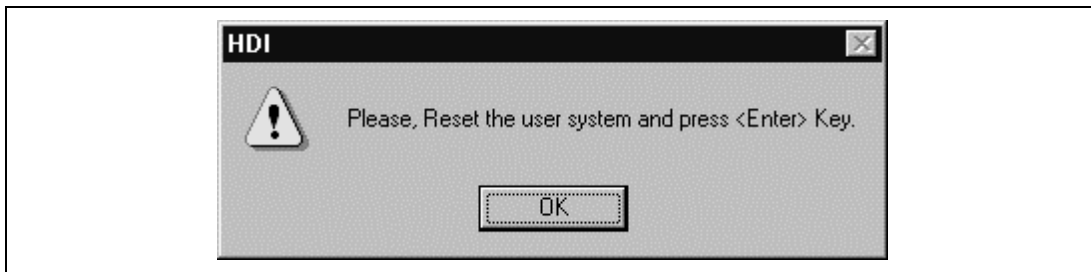
Figure 2.9   Dialog Box of the RESET Signal Input Request Message

8. Power on the user system.
9. Input the reset signal from the user system, and click the [OK] button.
10. When "Link Up" is displayed on the status bar, the HDI initiation is completed.
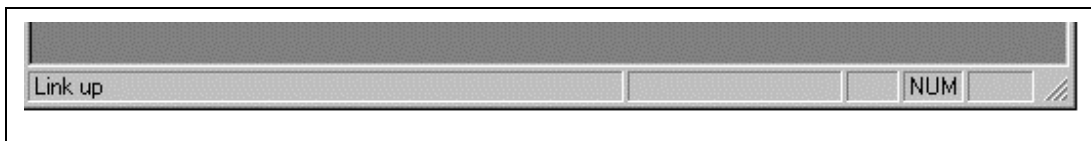
Figure 2.10   [HDI] Status Bar

Notes: 1. When the HDI is not linked up even if the above procedure has been executed, the driver will not be set correctly. Install drivers provided under the /DRIVERS directory in the CD-R according to the screen instructions. For instructions on how to check the driver has been correctly set, refer to the OS manual for your host computer or the following URL:
   http://www.hitachi.co.jp/Sicd/English/Products/micom/dev_env/tool/eml/e10a/ she10aqa.htm
   2. If the user system interface cable is disconnected to the Hitachi-UDI port connector on the user system during user program execution, the following dialog box will be displayed.

Figure 2.11   [JTAG Connector Disconnected] Dialog Box

RENESAS

3. **If the emulator is not initiated, the following dialog boxes shown in figures 2.12 through 2.16 will be displayed.**

(a) **If the following dialog box is displayed, the power of the user system may not be input or the RESET signal may not be input to the device. Check the input circuits for the power of the user system and the reset pin.**

Figure 2.12   [Can not find /RESET signal] Dialog Box

(b) **If the following dialog box is displayed, check that the Hitachi-UDI port connector on the user system is correctly connected.**
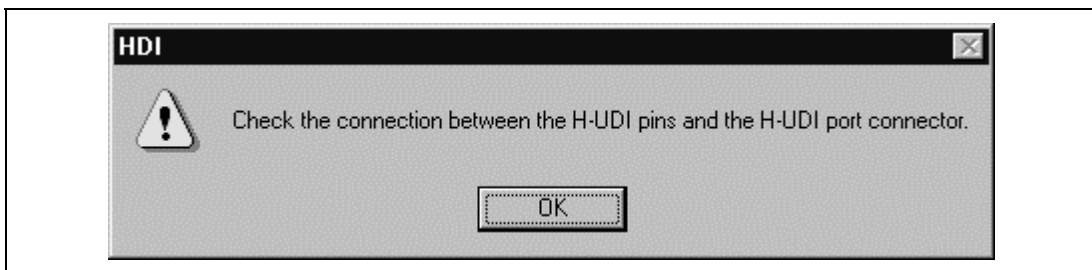
Figure 2.13   [Check the connection] Dialog Box

(c) **If the following dialog box is displayed, the device may not correctly operate. Check if there are reasons for illegal device operation.**

Figure 2.14   [COMMUNICATION TIMEOUT ERROR] Dialog Box

Figure 2.15 [INVALID ASERAM FIRMWARE!] Dialog Box

Figure 2.16 [Error JTAG boot] Dialog Box

4. If the driver is not correctly connected, the following dialog box will be displayed.
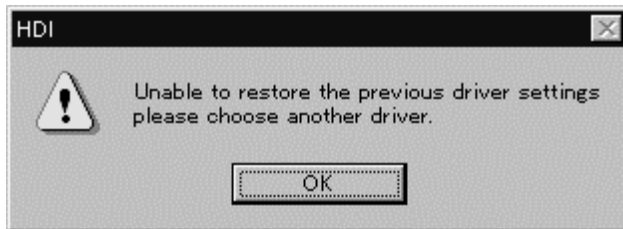
Figure 2.17 [Unable to restore the previous driver settings] Dialog Box

The [E10A Driver Details] dialog box is displayed when the [OK] button is clicked. Select the correct driver. For details, refer to section 6.5.1, Emulator Driver Selection.

RENESAS

2.6 Ending the HDI

Exit the HDI by using the following procedure:

1. Select [Exit] from the [File] menu to end the HDI. When the [Exit HDI] dialog box is displayed, click the [Yes] button.
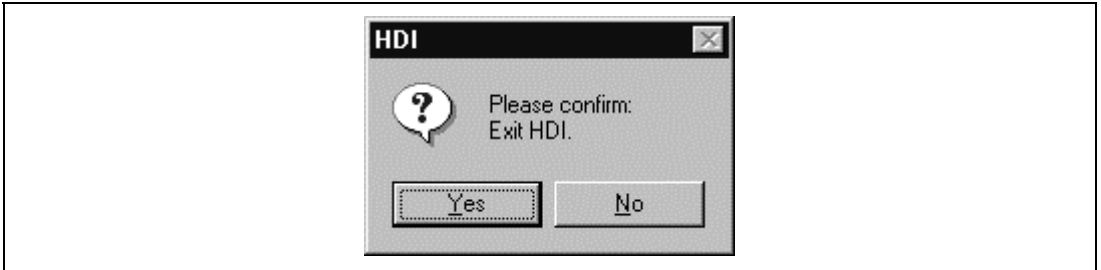
Figure 2.18   [Exit HDI] Dialog Box

2. Then, the [Save session] dialog box is displayed. If necessary, click the [Yes] button to save session. After saving session, the HDI ends. If not necessary, click the [No] button to end the HDI.
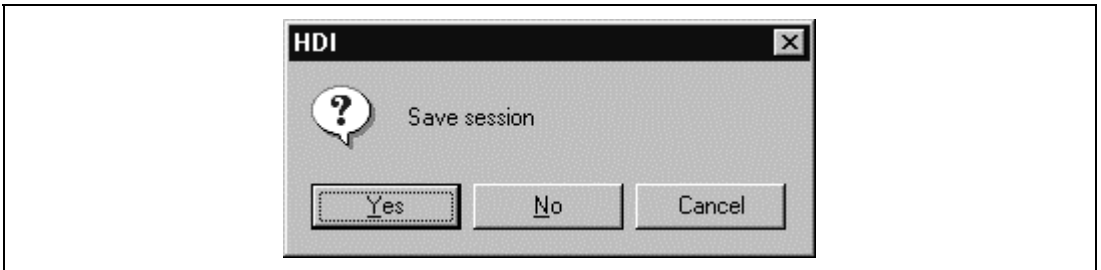
Figure 2.19   [Save session] Dialog Box

3. Turn the user system off.

2.7 Uninstalling the HDI

Follow this procedure to remove the installed HDI from the user's host computer.

1. Open [Add/Remove Programs Properties] from the control panel. Select the HDI program from the list and click the [Add/Remove…] button.
2. The setup program is executed again and the installed application can be changed, modified, or removed. When the application is to be uninstalled, select removal.

CAUTION

A shared file may be detected while the program is being removed. If another HDI may be using the shared file, do not remove the file. When Microsoft® Windows NT®4.0 operating system is used, the removal of the registry information on the driver may be asked. If other HDI may use the target driver, do not remove the registry information. If another HDI does not start up after the removal process, re-install that HDI.

RENESAS

2.8 CD-R

2.8.1 Configuration of the CD-R

The root directory of the CD-R contains a setup program for HDI installation. The folders contain the files and programs listed below.

Table 2.2 Contents of the CD-R Directories

Directory Name	Contents	Description
Dlls	Microsoft® runtime library	A runtime library for the HDI. The version is checked at installation and this library is copied to the hard disk as part of the installation process.
Drivers	E10A emulator driver	The E10A emulator drivers.
Help	Online help for the E10A emulator	An online help file. This is copied to the hard disk as part of the installation process.
Manual	E10A emulator manual	Precautions on Using the E10A Emulator and the user's manual. These are provided as PDF files.
Pdf_read	Adobe® Acrobat® Reader setup program	Adobe® Acrobat® Reader is an application for displaying, viewing, and printing PDF files.

2.9 Support

Information on the latest version of the HDI and other supporting information for the emulator can be found on the web site. Access the following URL:

http://www.hitachi.co.jp/Sicd/English/Products/micom/dev_env/tool/eml/e10a/e10atop.htm

RENESAS

RENESAS

Section 3   Tutorial

3.1　　　Introduction

The following describes the main functions of the HDI by using a tutorial program.

The tutorial program is based on the C program that sorts ten random data items in ascending or descending order. The tutorial program performs the following actions:

- The `main` function generates random data to be sorted.
- The `sort` function sorts the generated random data in ascending order.
- The `change` function then sorts the data in descending order.

The file `tutorial.c` contains source code for the tutorial program. The file `tutorial.abs` is a compiled load module in the Dwarf2 format.

Table 3.1 is a list of the parts of the tutorial program and an outline of their configuration on the hard disk.

Table 3.1　　Tutorial Program: Configuration and Parts

Item	Contents
Workspace for HEW V1.2	[Installation directory]\tutorial\tutorial.hws
Load module	[Installation directory]\tutorial\tutorial\Debug\tutorial.abs
Main program (source file)	[Installation directory]\tutorial\tutorial\tutorial.c
Stack information file	[Installation directory]\tutorial\tutorial\Debug\tutorial.sni

Use area 3 (CS3 space) as the operating environment. The MMU function is not used.

RENESAS

Notes: 1. Operation of `tutorial.abs` is big endian. For little-endian operation, `tutorial.abs` must be recompiled. After recompilation, the addresses may differ from those given in this section.

2. This program was created by using Hitachi Embedded Workshop (hereafter referred to as HEW) V1.2. Older versions of HEW will not open the workspace included with the package, so create a new workspace in such situations.

3. This program was compiled without optimization for the SH2 CPU. If recompiled with different settings, the addresses may differ from those given in this section.

4. `tutorial.abs` is a load module in the Dwarf2 format. If a load module is recreated in the Sysrof format, the amount of information displayed on the HDI screen during the program's execution will be reduced.

5. This section describes general usage examples for the emulator. For the specifications of particular products, refer to section 6 or the online help file.

RENESAS

3.2　Running the HDI

To run the HDI, select the [HDI for E10A SHxxxx] -> [Hitachi Debugging Interface] from the [Start] menu.
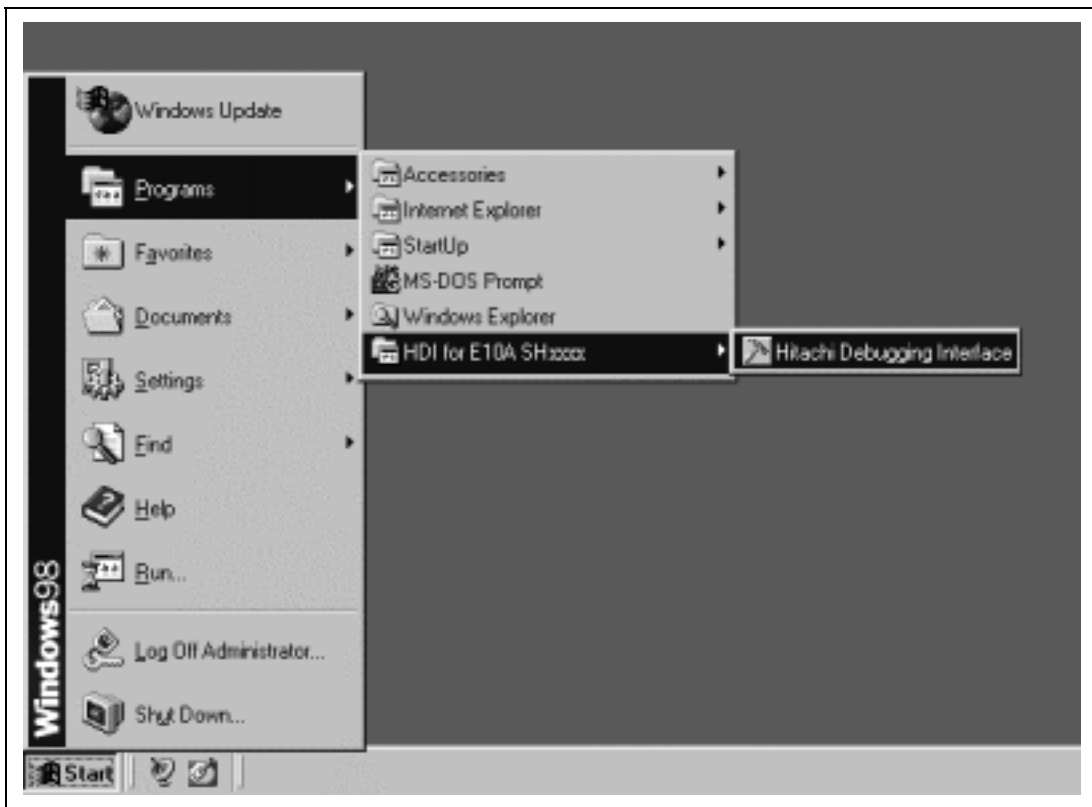
Figure 3.1　[Start] Menu

For the procedure of running the HDI, refer to section 2.5, System Check.

RENESAS
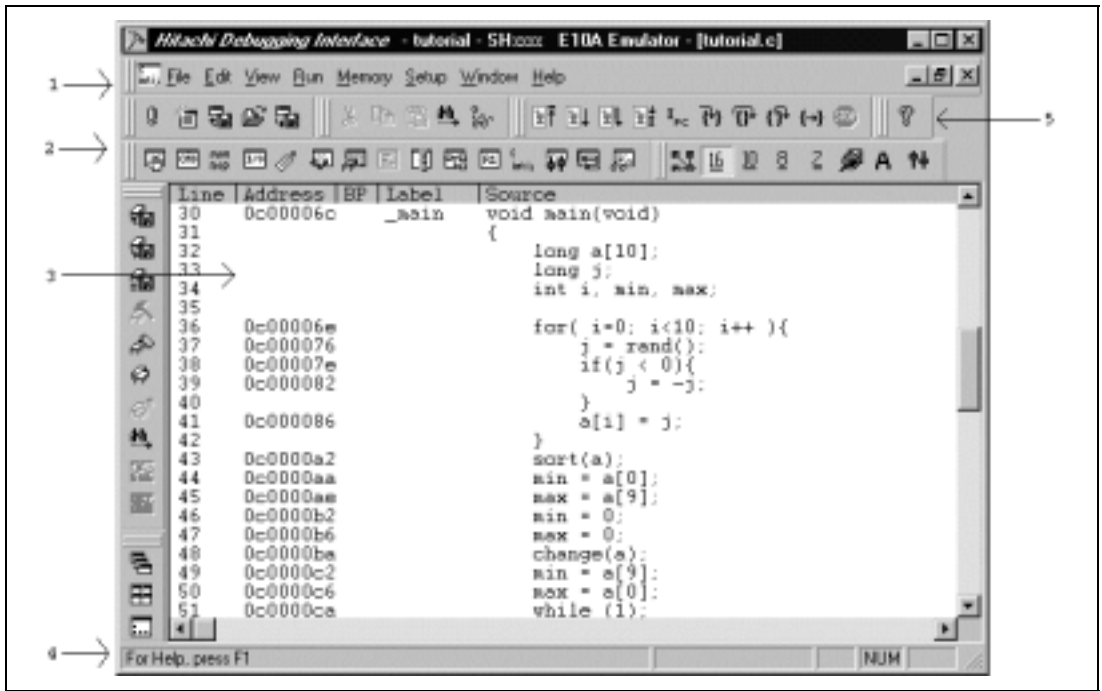
3.3   [HDI] Window

Figure 3.2   [HDI] Window

The key functions of the HDI are described in section 4, Descriptions of Windows. Numbers in figure 3.2 indicate the following:

1. Menu bar: Gives the user access to the HDI commands for using the HDI debugger.
2. Toolbar: Provides convenient buttons as shortcuts for the most frequently used menu commands.
3. Source window: Displays the source program being debugged.
4. Status bar: Displays the status of the emulator, and progress information about downloading.
5. [Help] button: Activates online help about any features of the HDI user interface.

RENESAS

3.4 Setting up the Emulator

The clocks which are used for data communications must be set up on the emulator before the program is downloaded.

- AUD clock

  A clock used in acquiring AUD traces.

  If its frequency is set too low, complete data may not be acquired during realtime tracing.

  If the frequency is set too high, the upper limit for the device's AUD clock may be exceeded.

  The AUD clock is only needed for emulators that have an AUD trace function.

- JTAG clock (TCK)

  A communication clock for downloading data to the emulator except for acquiring AUD trace.

  If its frequency is set too low, the speed of downloading will be lowered.

  If its frequency is set too high, the upper limit for the device's TCK clock may be exceeded.

For details of the limitations on both clocks, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK). The following is a description of the procedure used to set the clocks.

RENESAS

3.5 Setting the [Configuration] Dialog Box

- Select [Configure Platform...] from the [Setup] menu to set a communication clock. The [Configuration] dialog box is displayed.
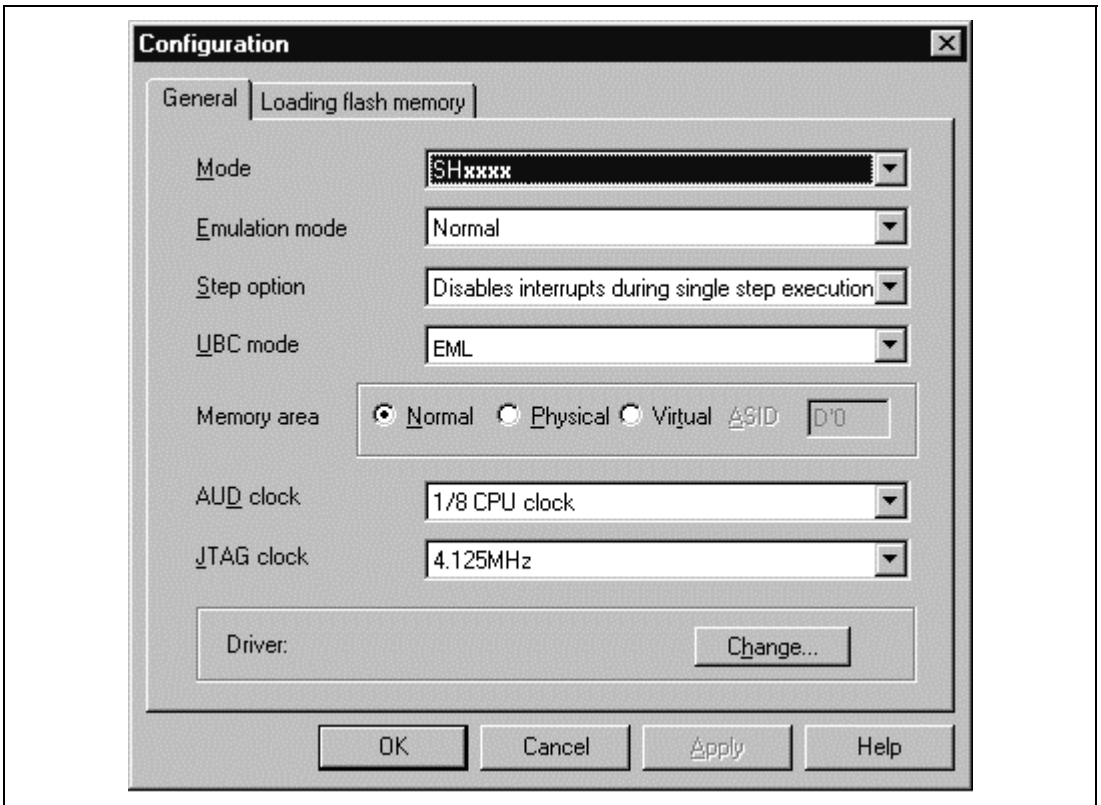
Figure 3.3 [Configuration] Dialog Box

- Set any value in the [AUD clock] and [JTAG clock] combo boxes. The clock also operates with the default value.

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

- Click the [OK] button to set a configuration.

RENESAS

3.6 Checking the Operation of the Target Memory for Downloading

Check that the destination memory area for downloading is operating correctly.

When the destination memory is SDRAM or DRAM, a register in the bus controller must be set before downloading. Set the bus controller correctly in the [I/O Registers] window according to the memory type. For details, refer to section 8.6, I/O Register Display, in the Hitachi Debugging Interface User's Manual.

When the required settings, such as the settings for the bus controller, have been completed, display and edit the contents of the destination memory in the [Memory] window to check that the memory is operating correctly.

Note: The above way of checking the operation of memory may be inadequate. It is recommended that a program for checking the memory be created.

• Select [Memory…] from the [View] menu, enter H'0c000000 in the [Address] edit box, and set the format in the [Format] combo box to Byte.
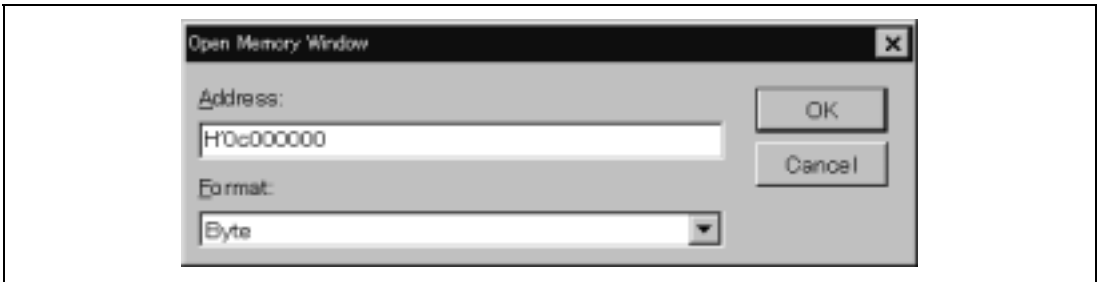
Figure 3.4   [Open Memory Window] Dialog Box

• Click the [OK] button. The [Memory] window is displayed and shows the specified memory area.
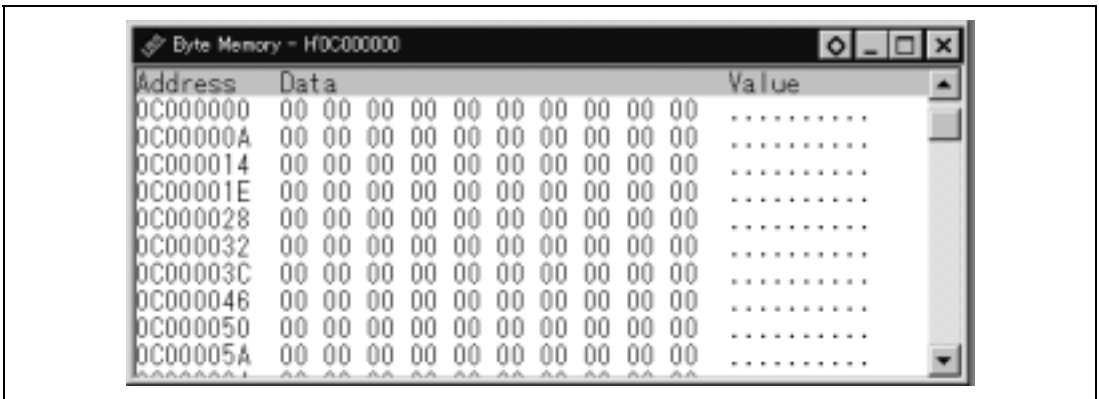
Figure 3.5   [Memory] Window

RENESAS

- Placing the mouse cursor on a point in the display of data in the [Memory] window and double clicking allows the values at that point to be changed. Data can also be directly edited around the current position of the text cursor.

RENESAS

3.7 Downloading the Tutorial Program

3.7.1 Downloading the Tutorial Program

Download the object program to be debugged.

- Select [Load Program...] from the [File] menu. The [Load Program] dialog box is displayed. Enter '[installation directory]\tutorial\tutorial\Debug\tutorial.abs' in the [File name] list box as shown in figure 3.6, then click the [Open] button.
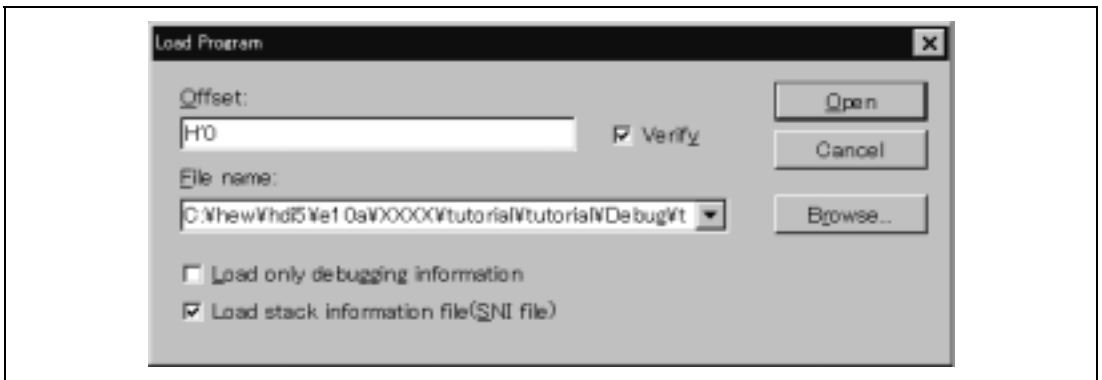
Figure 3.6 [Load Program] Dialog Box

Notes: 1. When installing the emulator, if no directory is specified and the HEW is in use, the program is installed under '\Hew\hdi5\e10a'. When the HEW is not in use, the program is installed under '\root directory\E10A'.
   2. The SNI file is required so that the profiler function can be used.
   3. The [Verify] check box is disabled in this product.

After the file has been loaded, the following dialog box displays information about the memory areas to which the program code has been transferred.

Figure 3.7 [HDI] Dialog Box

- Click the [OK] button to continue.

RENESAS

3.7.2 Displaying the Source Program

The HDI allows the user to debug a program at the source level.

- Select [Source...] from the [View] menu. The [Open] dialog box is displayed.
- Select the C source file that corresponds to the object file the user has loaded.
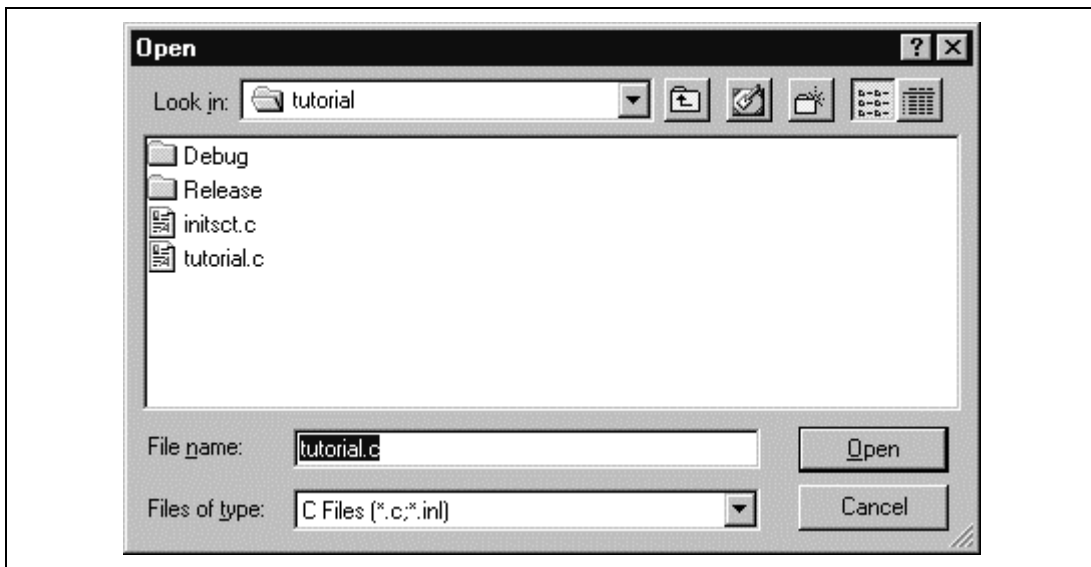
Figure 3.8   [Open] Dialog Box

- Select [tutorial.c] and click the [Open] button. The [Source] window is displayed.

RENESAS

```
┌─────────────────────────────────────────────────────────────────────┐
│ ▒ tutorial.c                                              _ □ ×      │
├─────────────────────────────────────────────────────────────────────┤
│ Line │ Address │ BP │ Label  │ Source                          ▲     │
│ 30    0c00006c       _main    void main(void)                        │
│ 31                             {                                      │
│ 32                                    long a[10];                     │
│ 33                                    long j;                         │
│ 34                                    int i, min, max;                │
│ 35                                                                    │
│ 36    0c00006e                        for( i=0; i<10; i++ ){    ▓     │
│ 37    0c000076                            j = rand();           ▓     │
│ 38    0c00007e                            if(j < 0){            ▓     │
│ 39    0c000082                                j = -j;                 │
│ 40                                         }                          │
│ 41    0c000086                            a[i] = j;                   │
│ 42                                     }                              │
│ 43    0c0000a2                        sort(a);                        │
│ 44    0c0000aa                        min = a[0];                     │
│ 45    0c0000ae                        max = a[9];                     │
│ 46    0c0000b2                        min = 0;                        │
│ 47    0c0000b6                        max = 0;                        │
│ 48    0c0000ba                        change(a);                      │
│ 49    0c0000c2                        min = a[9];                      │
│ 50    0c0000c6                        max = a[0];                      │
│ 51    0c0000ca                        while (1);                       │
│ 52                             }                                      │
│ 53                                                                    │
│ 54                             void abort(void)                       │
│ 55                             {                                ▼     │
│ ◄│                                                            │► /   │
└─────────────────────────────────────────────────────────────────────┘
```

Figure 3.9   [Source] Window (Displaying the Source Program)

• If necessary, select the [Font] option from the [Customise] submenu on the [Setup] menu to select a font and size that are legible.

Initially the [Source] window shows the start of the main program, but the user can use the scroll bar to scroll through the program and look at the other statements.

3.8   Setting a Software Breakpoint

A breakpoint is a simple debugging function.

The [Source] window provides a very simple way of setting a software breakpoint at any point in a program. For example, to set a breakpoint at the sort function call:

- Select by double-clicking the [BP] column on the line containing the sort function call.

```
tutorial.c                                                    _ □ ×
Line  Address   BP  Label    │Source
30    0c00006c      _main     void main(void)
31                            {
32                                    long a[10];
33                                    long j;
34                                    int i, min, max;
35
36    0c00006e                        for( i=0; i<10; i++ ){
37    0c000076                            j = rand();
38    0c00007e                            if(j < 0){
39    0c000082                                j = -j;
40                                        }
41    0c000086                            a[i] = j;
42                                    }
43    0c0000a2  ●                 sort(a);
44    0c0000aa                    min = a[0];
45    0c0000ae                    max = a[9];
46    0c0000b2                    min = 0;
47    0c0000b6                    max = 0;
48    0c0000ba                    change(a);
49    0c0000c2                    min = a[9];
50    0c0000c6                    max = a[0];
51    0c0000ca                    while (1);
52                                }
53
54                            void abort(void)
55                            {
```

Figure 3.10   [Source] Window (Setting a Software Breakpoint)

The symbol ● will appear on the line containing the sort function, and the word ● Break will appear when the [BP] column is extended. This shows that a software breakpoint has been set.

Note: The software breakpoint cannot be set in the ROM area.

RENESAS

3.9 Setting Registers

Set values of the program counter and the stack pointer before executing the program.

- Select [Registers] from the [View] menu. The [Registers] window is displayed.

```
R1 Registers                                    ◇ _ □ ×
Register     │Value                                    ▲
   R0         00000000
   R1         00000000
   R2         00000000
   R3         00000000
   R4         00000000
   R5         00000000
   R6         00000000
   R7         00000000
   R8         00000000
   R9         00000000
   R10        00000000
   R11        00000000
   R12        00000000
   R13        00000000
   R14        00000000
   R15        00000000
   PC         A0000000
 + SR         -P1B--------------------IIII---F
   GBR        00000000                                 ▼
◄ ┃                                              ►
```
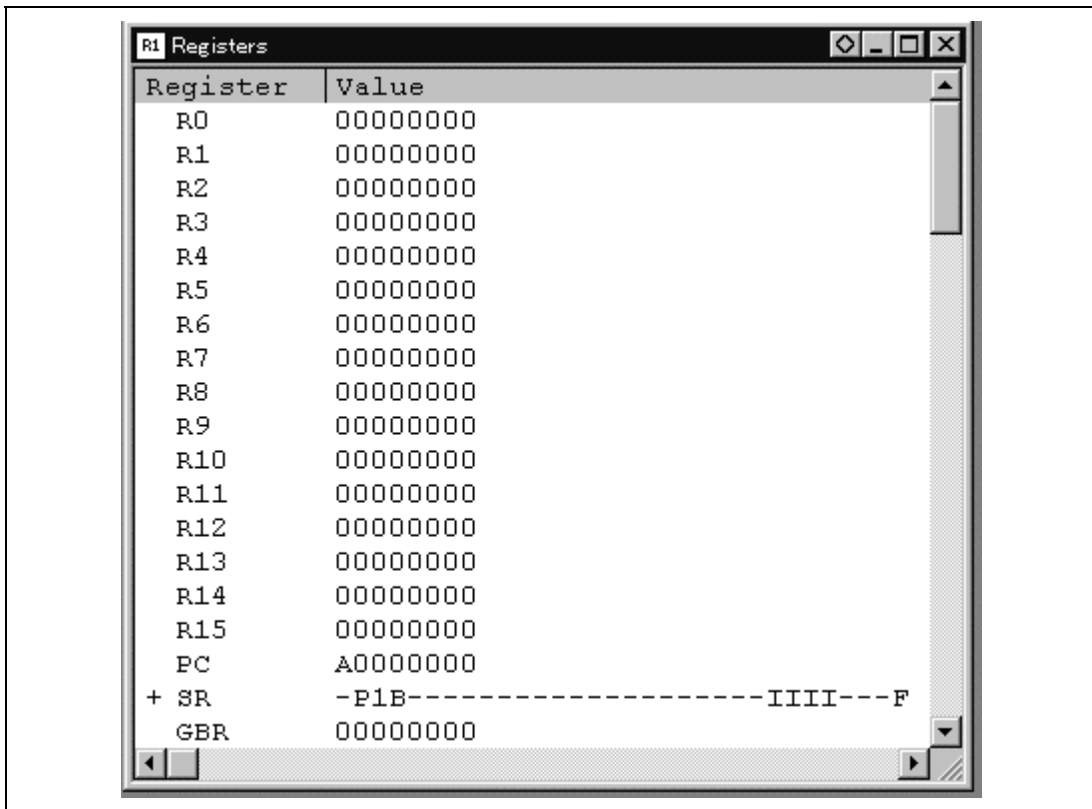
Figure 3.11   [Registers] Window

- To change the value of the program counter (PC), double-click the value area in the [Registers] window with the mouse. The following dialog box is then displayed, and the value can be changed. Set the program counter to H'0c00006c in this tutorial program, and click the [OK] button.
- Move the mouse pointer on the value to be changed in the [PC] value area and enter the new value by the keyboard.
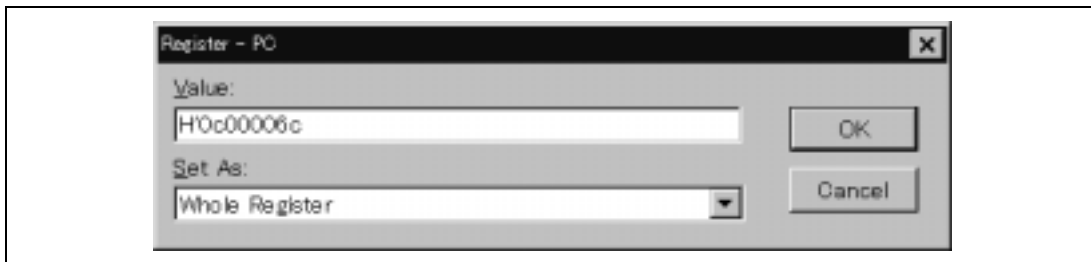
Figure 3.12 [Register] Dialog Box (PC)

- Change the value of the stack pointer (SP) in the same way. Set H'0c000c00 for the value of the stack pointer in this tutorial program.

RENESAS

3.10　Executing the Program

Execute the program as described in the following:

- To execute the program, select [Go] from the [Run] menu, or click the [Go] button on the toolbar.

Figure 3.13　[Go] Button

The program will be executed up to the breakpoint that has been inserted, and a statement will be highlighted in the [Source] window to show the position that the program has halted, with the message [Break=BREAKPOINT] in the status bar.


```
tutorial.c                                              _ □ ×
Line | Address  | BP | Label  | Source
30     0c00006c       _main     void main(void)
31                              {
32                                      long a[10];
33                                      long j;
34                                      int i, min, max;
35
36     0c00006e                         for( i=0; i<10; i++ ){
37     0c000076                             j = rand();
38     0c00007e                             if(j < 0){
39     0c000082                                 j = -j;
40                                          }
41     0c000086                             a[i] = j;
42                                      }
43     0c0000a2 ●                       sort(a);
44     0c0000aa                         min = a[0];
45     0c0000ae                         max = a[9];
46     0c0000b2                         min = 0;
47     0c0000b6                         max = 0;
48     0c0000ba                         change(a);
49     0c0000c2                         min = a[9];
50     0c0000c6                         max = a[0];
51     0c0000ca                         while (1);
52                                  }
53
54                              void abort(void)
55                              {
```

Figure 3.14　[Source] Window (Break Status)

RENESAS

The user can see the cause of the break that occurred last time in the [System Status] window.

- Select [Status] from the [View] menu. After the [System Status] window is displayed, open the [Platform] page, and check the status of Cause of last break.
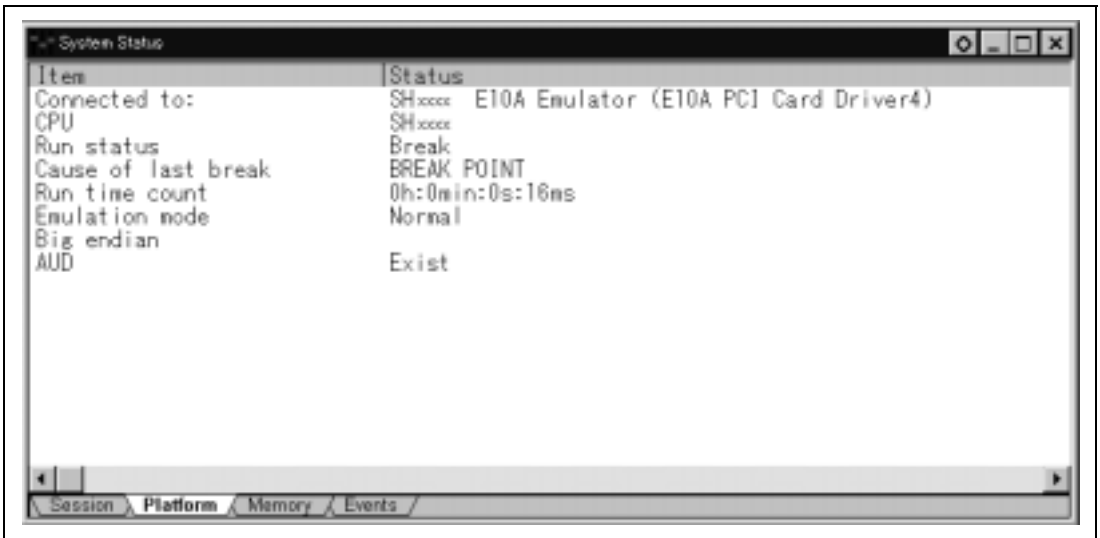
Figure 3.15 [System Status] Window

Note: The items that can be displayed in this window differ according to the product. For the items that can be displayed, refer to the online help.

RENESAS

3.11　Reviewing Breakpoints

The user can see all the breakpoints set in the program in the [Breakpoints] window.

- Select [Breakpoints] from the [View] menu.
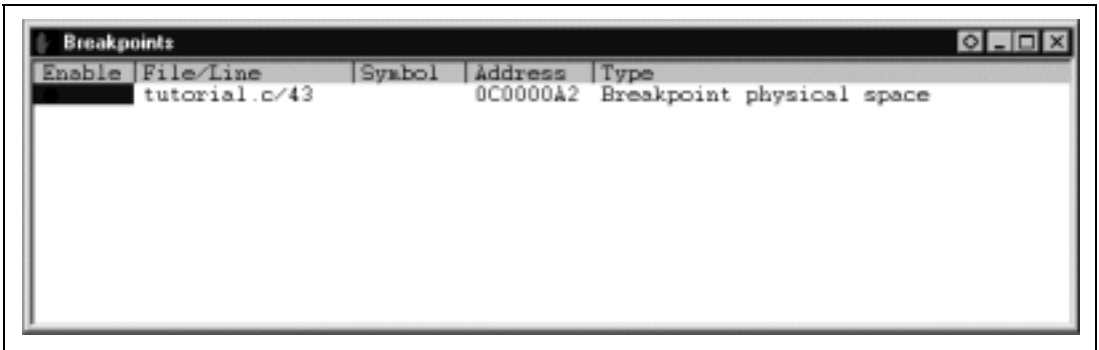
Figure 3.16　[Breakpoints] Window

The pop-up menu, opened by clicking the [Breakpoints] window with the right mouse button, also allows the user to set or change breakpoints, define new breakpoints, and delete, enable, or disable breakpoints.

RENESAS

3.12 Viewing Memory

When the label name is specified, the user can view the memory contents that the label has been registered in the [Memory] window. For example, to view the memory contents corresponding to the _main in word size:

- Select [Memory …] from the [View] menu, enter **_main** in the [Address] edit box, and set Word in the [Format] combo box.
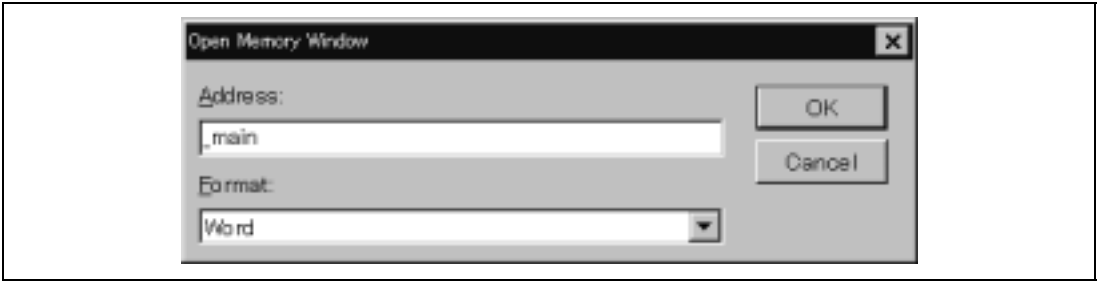
Figure 3.17   [Open Memory Window] Dialog Box

- Click the [OK] button. The [Memory] window showing the specified area of memory is displayed.
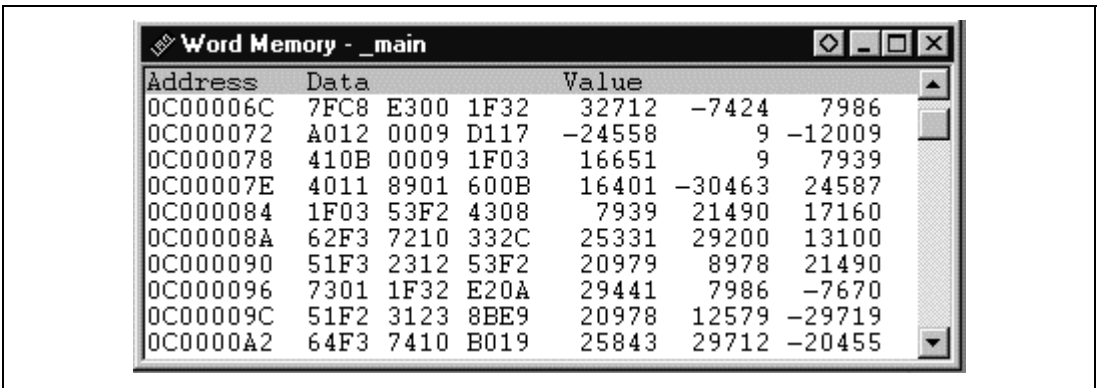
Figure 3.18   [Memory] Window

RENESAS

3.13   Watching Variables

As the user steps through a program, it is possible to watch that the values of variables used in the user program are changed. For example, set a watch on the long-type array a declared at the beginning of the program, by using the following procedure:

- Click the left of displayed array a in the [Source] window to position the cursor.
- Click the [Source] window with the right mouse button and select [Instant Watch...] from a pop-up menu.
  The following dialog box will be displayed.
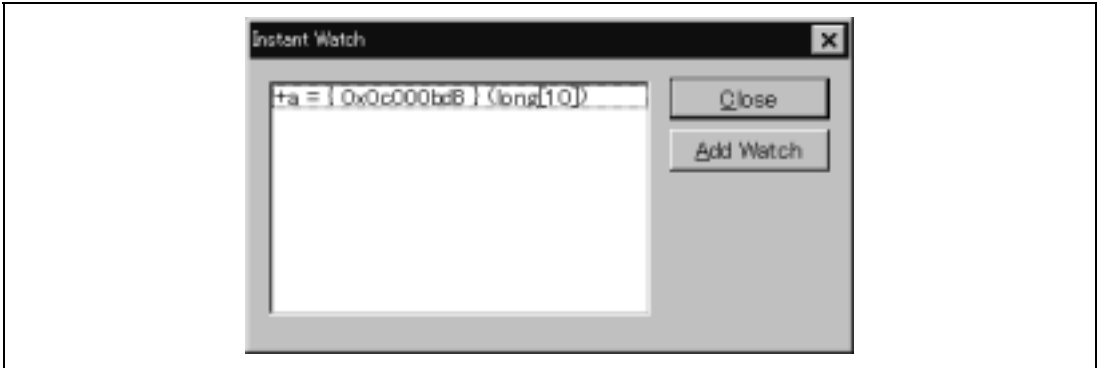
Figure 3.19   [Instant Watch] Dialog Box

RENESAS

- Click [Add Watch] button to add a variable to the [Watch] window.
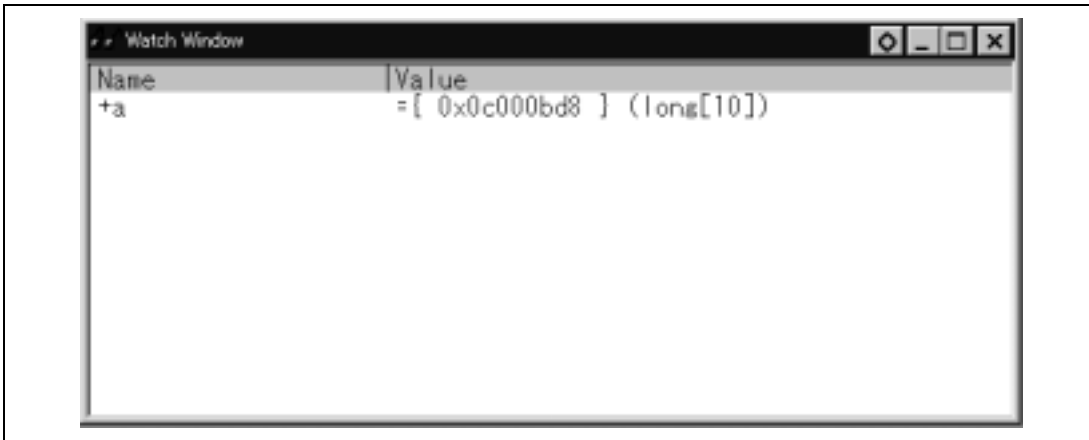
Figure 3.20   [Watch] Window (Displaying the Array)

The user can also add a variable to the [Watch] window by specifying its name.

- Click the [Watch] window with the right mouse button and select [Add Watch…] from the pop-up menu.

  The following dialog box will be displayed.

Figure 3.21   [Add Watch] Dialog Box

- Input variable *max* and click the [OK] button.

RENESAS

The [Watch] window will now also show the int-type variable max.

Figure 3.22　[Watch] Window (Displaying the Variable)

The user can double-click the + symbol to the left of any variable in the [Watch] window to watch the all elements in array a.
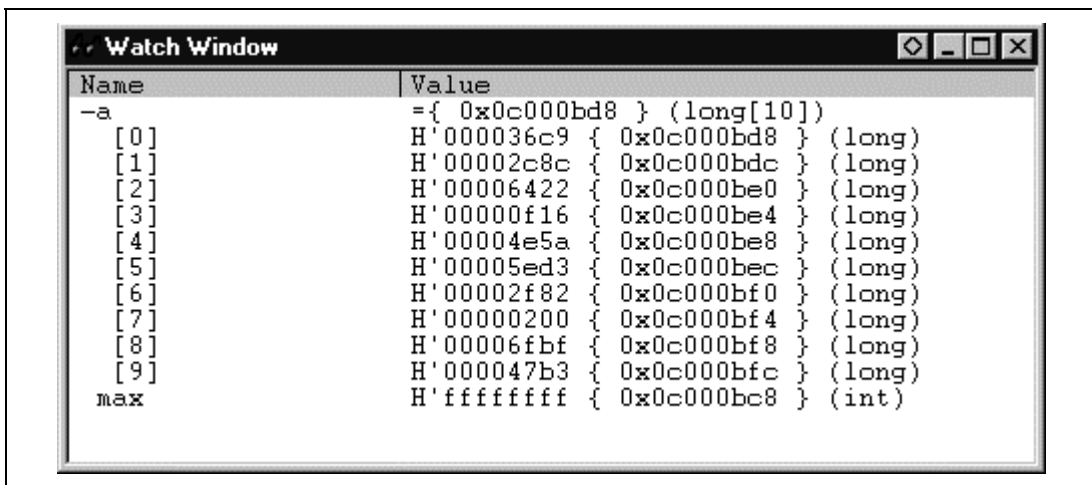
Figure 3.23　[Watch] Window (Displaying Array Elements)

3.14 Stepping Through a Program

The HDI provides a range of step menu commands that allow efficient program debugging.

Table 3.2   Step Option

Menu Command	Description
Step In	Executes each statement, including statements within functions.
Step Over	Executes a function call in a single step.
Step Out	Steps out of a function, and stops at the statement following the statement in the program that called the function.
Step…	Steps the specified times repeatedly at a specified rate.

3.14.1   Executing [Step In] Command

The [Step In] steps into the called function and stops at the first statement of the called function.

- To step through the sort function, select [Step In] from the [Run] menu, or click the [Step In] button in the toolbar.

Figure 3.24   [Step In] Button

RENESAS

```
┌────────────────────────────────────────────────────────────────────────┐
│ ▒▒▒ tutorial.c                                              _ □ ×        │
├────────────────────────────────────────────────────────────────────────┤
│ Line │ Address  │ BP │ Label  │ Source                               ▲  │
│  47    0c0000b6                         max = 0;                         │
│  48    0c0000ba                         change(a);                       │
│  49    0c0000c2                         min = a[9];                       │
│  50    0c0000c6                         max = a[0];                       │
│  51    0c0000ca                         while (1);                        │
│  52                             }                                        │
│  53                                                                      │
│  54                             void abort(void)                         │
│  55                             {                                        │
│  56                                                                      │
│  57    0c0000d8     _abort      }                                        │
│  58                                                                      │
│  59    0c0000dc     _sort       void sort(long *a)                       │
│  60                             {                                        │
│  61                                     long t;                          │
│  62                                     int i, j, k, gap;                │
│  63                                                                      │
│  64    0c0000e0                         gap = 5;                         │
│  65    0c0000e4                         while( gap > 0 ){                │
│  66    0c0000e8                             for( k=0; k<gap; k++){       │
│  67    0c0000f0                                 for( i=k+gap; i<10;      │
│  68    0c0000fc                                     for(j=i-gap; j>      │
│  69    0c000108                                         if(a[j]>a[j-     │
│  70    0c000120                                             t = a[j      │
│  71    0c00012a                                             a[j] = ¿     │
│  72    0c00013e                                             a[j+gap ▼    │
│ ◄ │                                                              ► │    │
└────────────────────────────────────────────────────────────────────────┘
```

Figure 3.25   [Source] Window (Step In)

• The highlighted line moves to the first statement of the sort function in the [Source] window.

RENESAS

3.14.2 Executing [Step Out] Command

The [Step Out] steps out of the called function and stops at the next statement of the calling statement in the main function.

- To step out of the `sort` function, select [Step Out] from the [Run] menu, or click the [Step Out] button in the toolbar.

Note: It takes time to execute this function.

Figure 3.26 [Step Out] Button
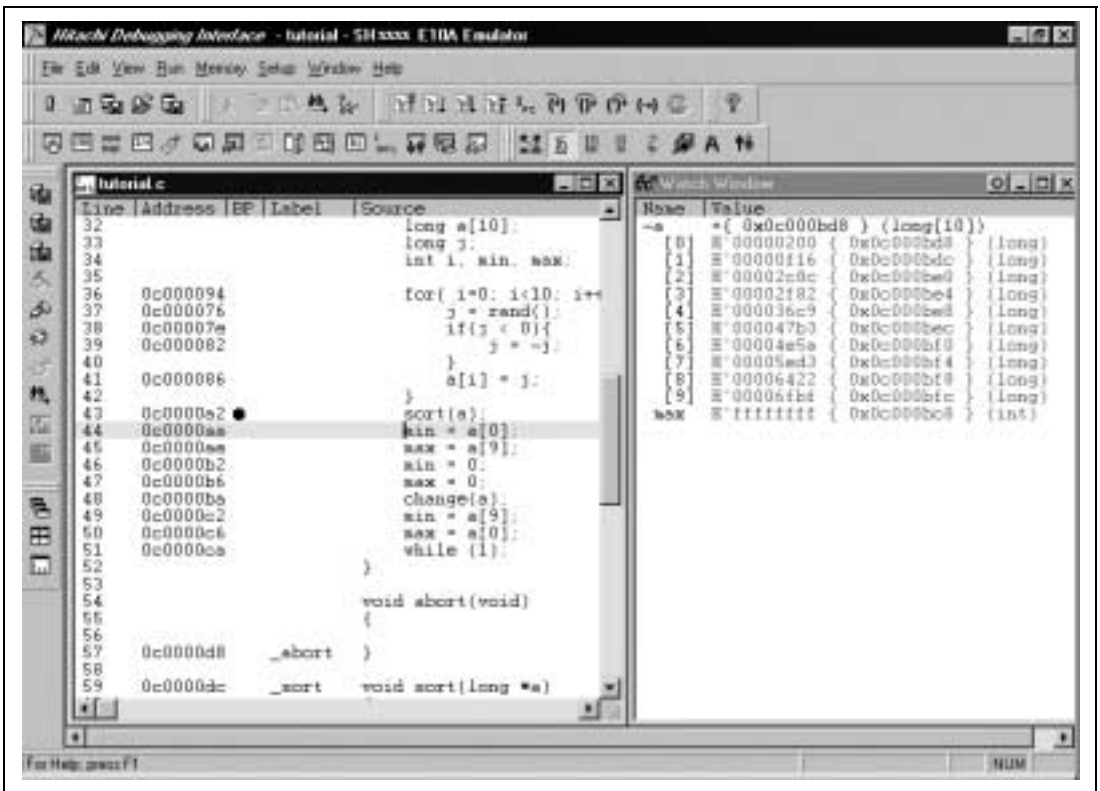
Figure 3.27 [HDI] Window (Step Out)

- The data of variable `a` displayed in the [Watch] window is sorted in ascending order.

RENESAS

- To execute two steps, use [Step In] twice.
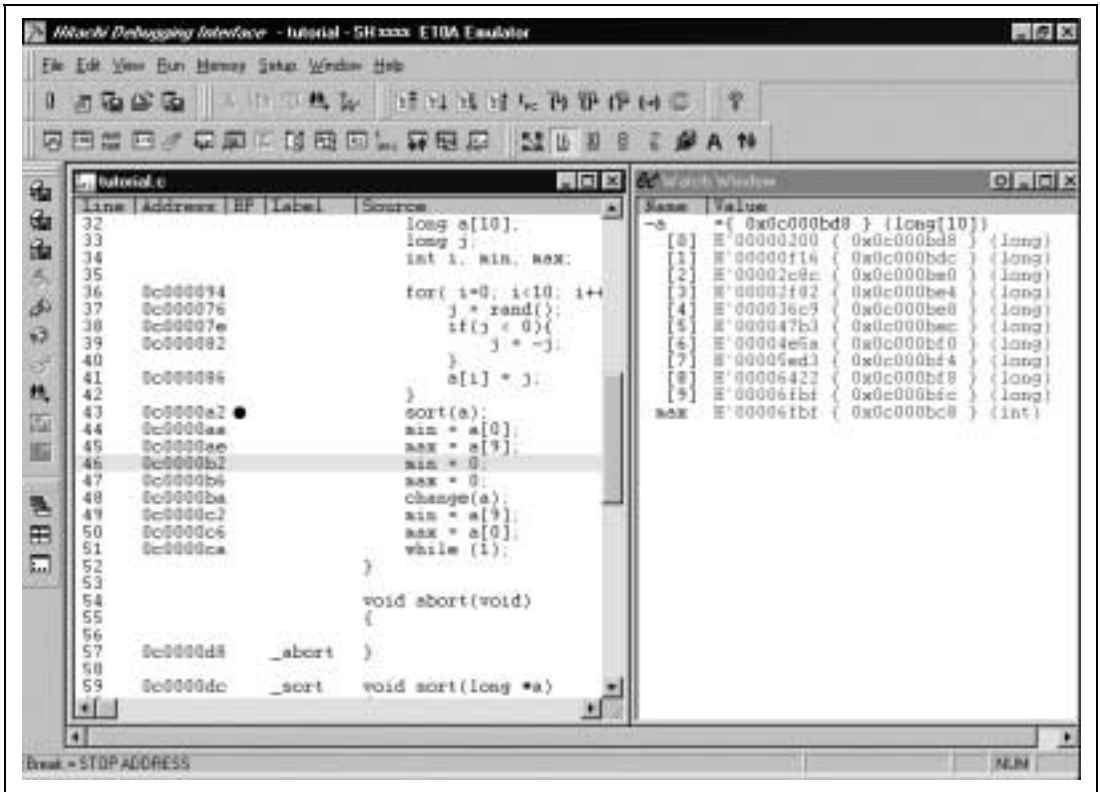
Figure 3.28 [HDI] Window (Step In –> Step In)

- The value of max displayed in the [Watch] window is changed to the maximum data value.

RENESAS

3.14.3 Executing [Step Over] Command

The [Step Over] executes a function call as a single step and stops at the next statement of the main program.

- Using [Step Over], execute two steps to reach the change function statement.
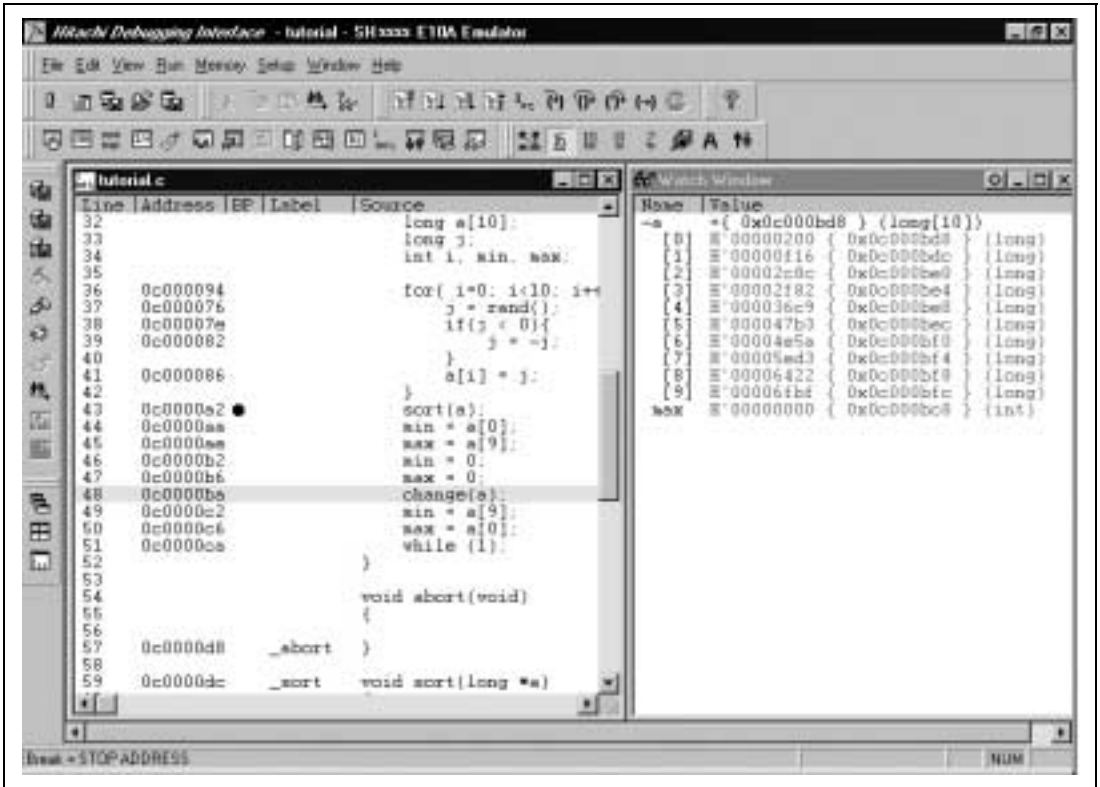
Figure 3.29 [Source] Window (Before Step Over Execution)

- To step through all statements in the change function at a single step, select [Step Over] from the [Run] menu, or click the [Step Over] button in the toolbar.

Figure 3.30 [Step Over] Button

RENESAS
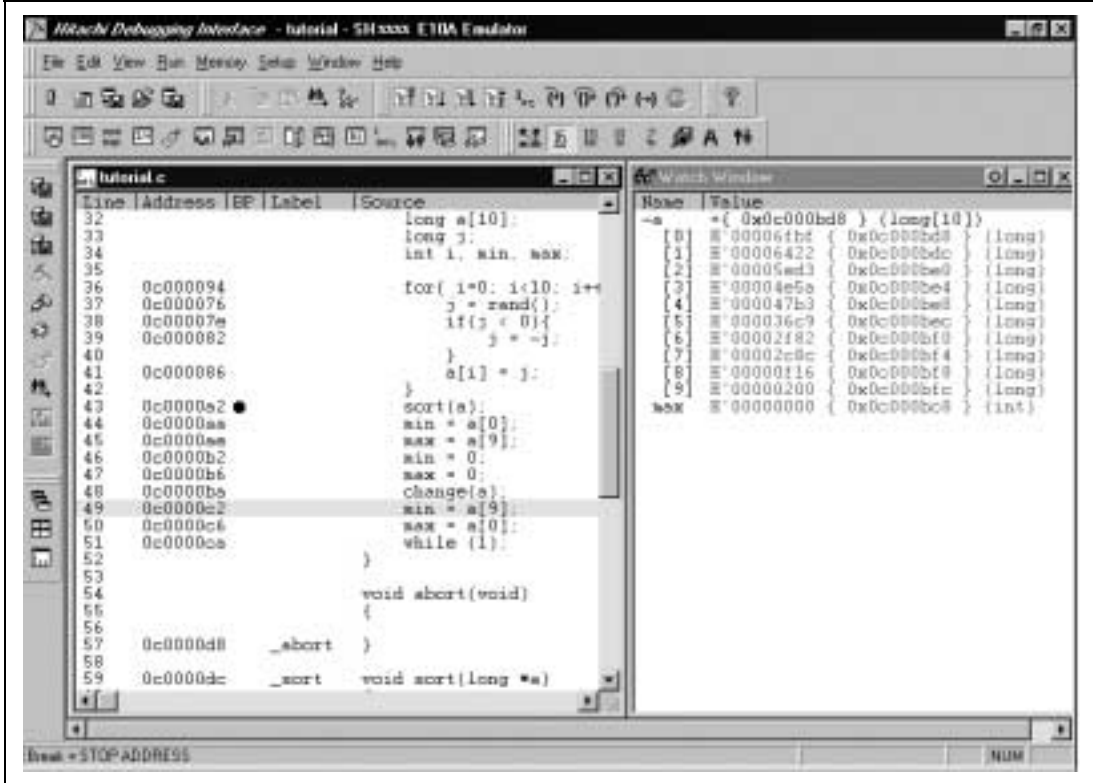
Figure 3.31 [HDI] Window (Step Over)

RENESAS

3.15　Forced Breaking of Program Executions

The HDI can force a break in the execution of a program.

- To execute the remaining sections of the main function, select [Go] from the [Run] menu or the [Go] button on the toolbar.

Figure 3.32　[Go] Button

- The program goes into an endless loop. To force a break in execution, select [Halt] from the [Run] menu or the [Stop] button on the toolbar.

Figure 3.33　[Stop] Button

- The highlighted line of the [Program] window moves to the `while` statement, and the value of `max` displayed in the [Watch] window is updated to the most recent value.

RENESAS

3.16   Displaying Local Variables

The user can display local variables in a function using the [Locals] window. For example, we will examine the local variables in the main function, which declares five local variables: a, j, i, min, and max.

- Select [Locals] from the [View] menu. The [Locals] window is displayed.

    Initially, the [Locals] window is empty because local variables have not yet been declared.

    The [Locals] window will now show the local variables and their values.

```
Locals                                    ◇ _ □ ✕
Name                    Value
+a                      ={ 0x0c000bd8 } (long[10])
 j                      D'18355 { 0x0c000bd4 } (long)
 i                      D'10 { 0x0c000bd0 } (int)
 min                    D'512 { 0x0c000bcc } (int)
 max                    D'28607 { 0x0c000bc8 } (int)
```
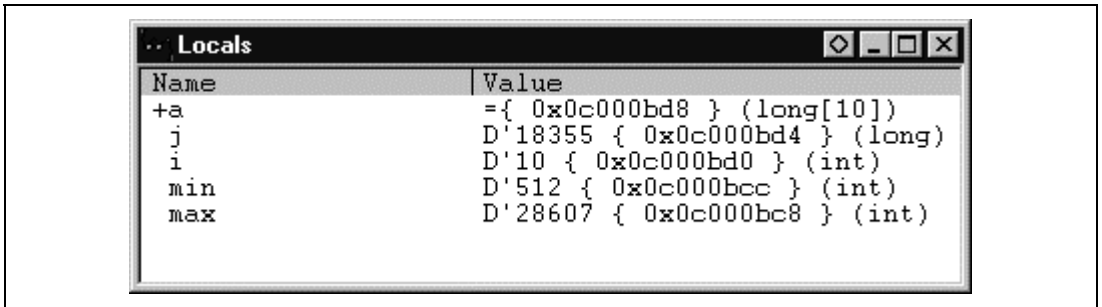
Figure 3.34   [Locals] Window

- Double-click the + symbol in front of array a in the [Locals] window to display the elements of array a.
- Refer to the elements of array a before and after the execution of the sort function, and confirm that random data is sorted in descending order.

3.17 Break Function

The emulator has software and hardware break functions. With the HDI, a software breakpoint can be set using the [Breakpoints] window, and a hardware break condition can be set using the [Break Condition] dialog box.

An overview and setting of the break function are described below.

3.17.1 Software Break Function

The emulator can set up to 255 software breakpoints. Other methods for setting a software breakpoint than in section 3.8 are described below.

- Select [Breakpoints] from the [View] menu. The [Breakpoints] window is displayed.
- Click the [Breakpoints] window with the right mouse button and select [Delete All] from the pop-up menu to cancel all the breakpoints that have been set.
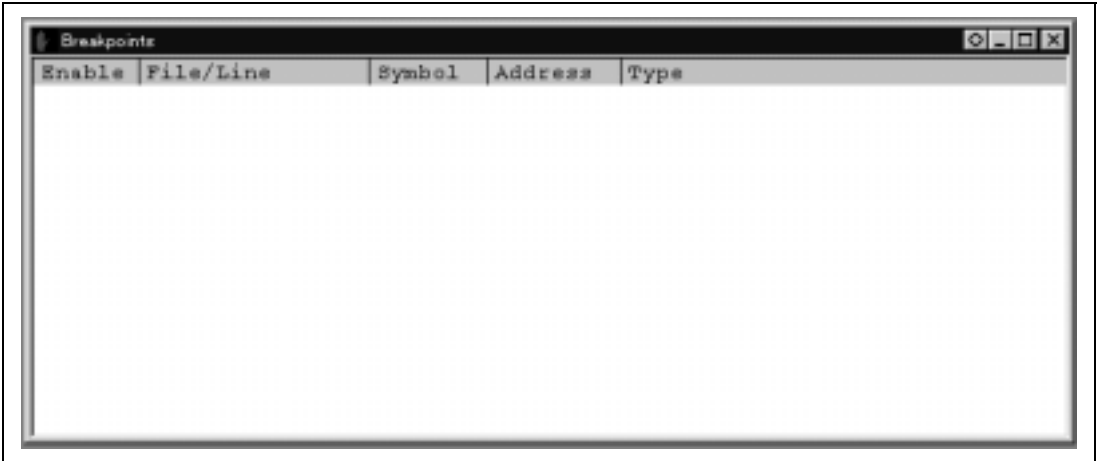
Figure 3.35   [Breakpoints] Window (Before Software Breakpoint Setting)

RENESAS

- Click the [Breakpoints] window with the right mouse button and select [Add] from the pop-up menu.

  The [Break] dialog box is displayed. The [Point] page is displayed as a default.
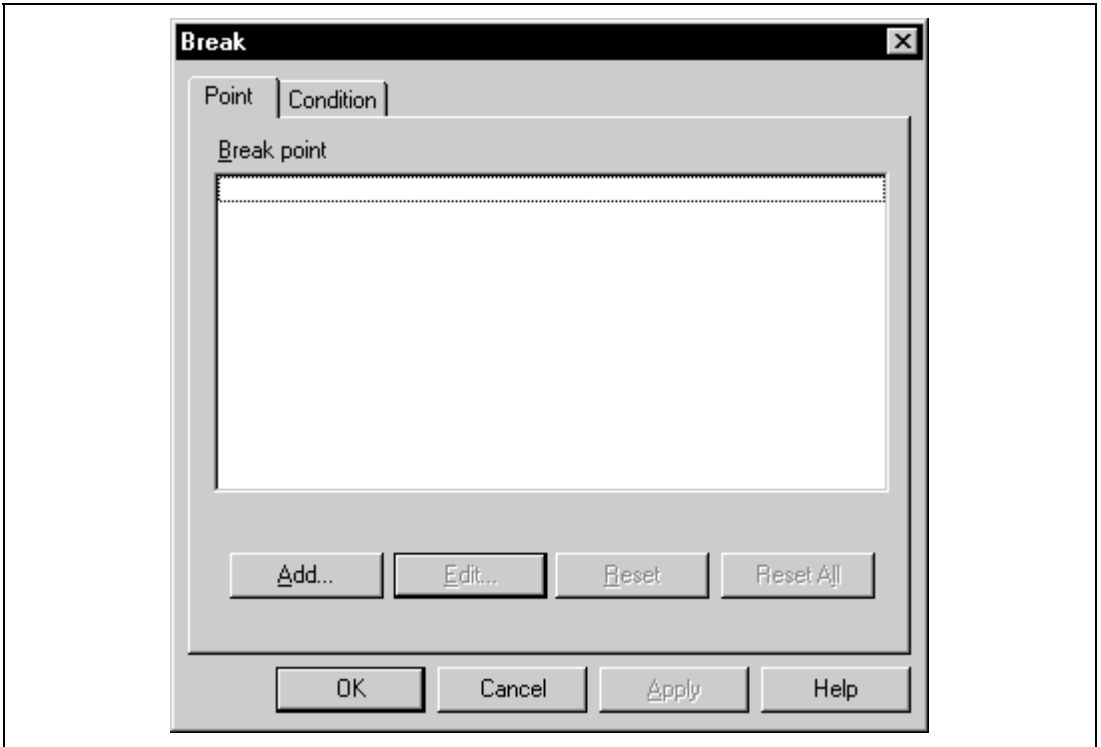
Figure 3.36   [Point] Page ([Break] Dialog Box)

- Click the [Add...] button to display the [Breakpoint] dialog box.
- Enter *H'0c0000c2* to the [Value] edit box.
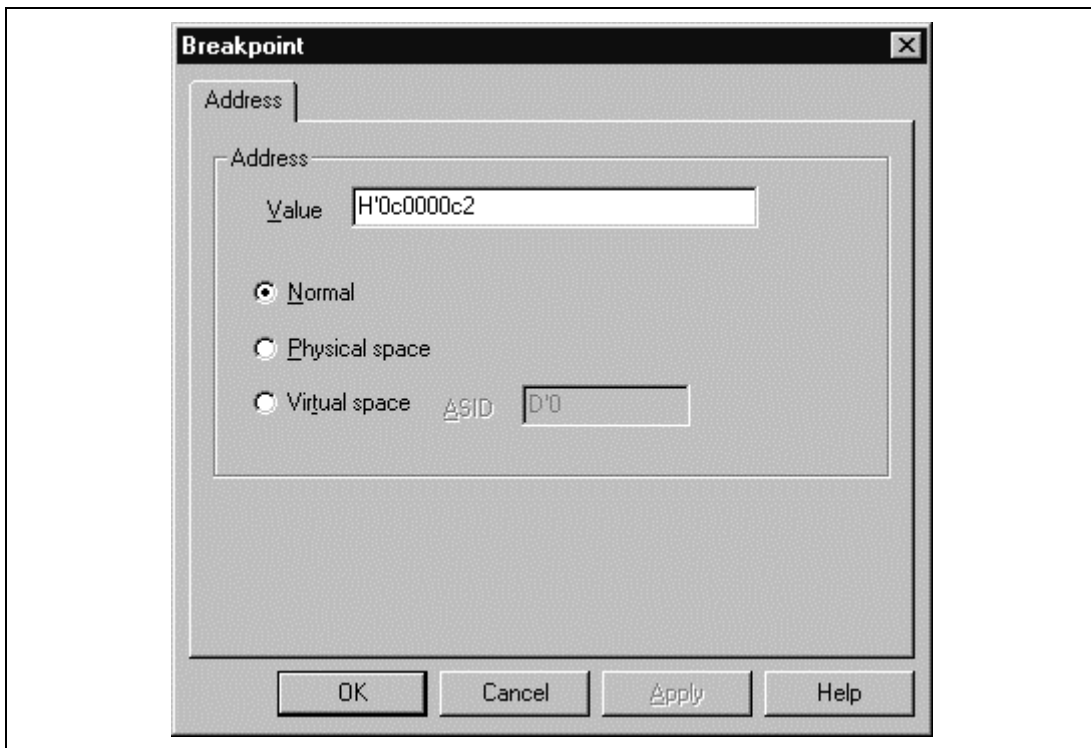
Figure 3.37 [Break Point] Dialog Box

- Click the [OK] button.

RENESAS

The [Break] dialog box is displayed. The address set in the value field of [Breakpoint] and the memory space are displayed.
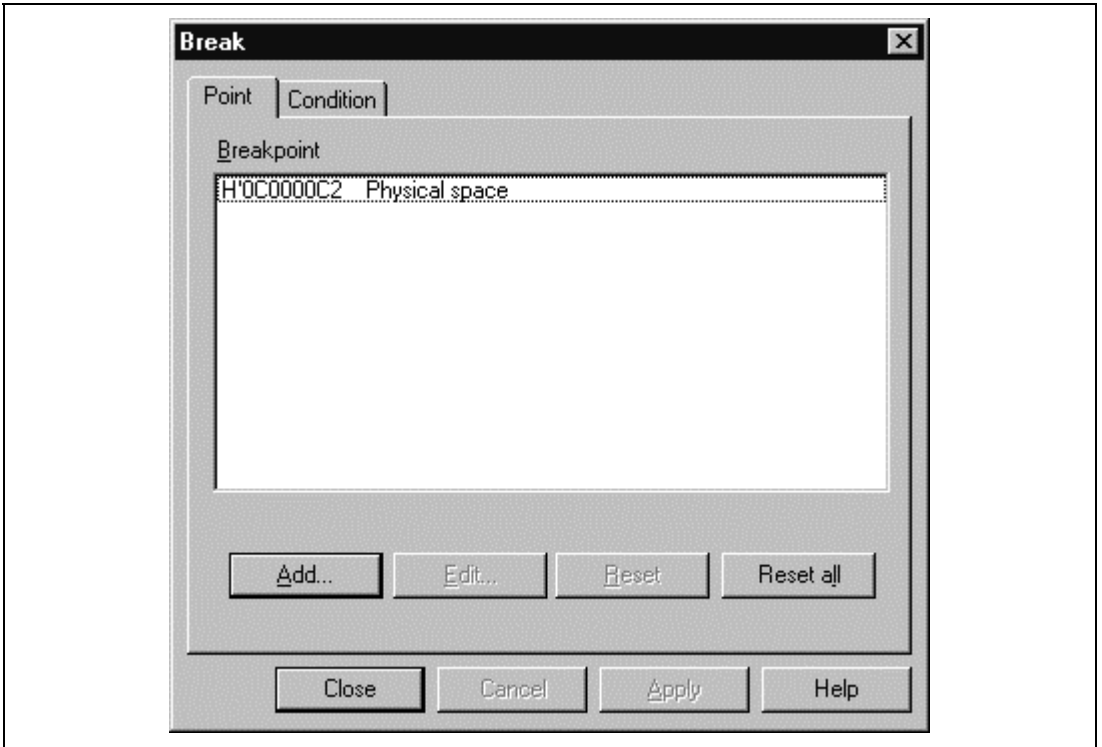
Figure 3.38   [Point] Page ([Break] Dialog Box) (After Software Breakpoint Setting)

- Click the [Close] button (or [OK] button in some emulator products).

The software breakpoint that has been set is displayed in the [Breakpoints] window.
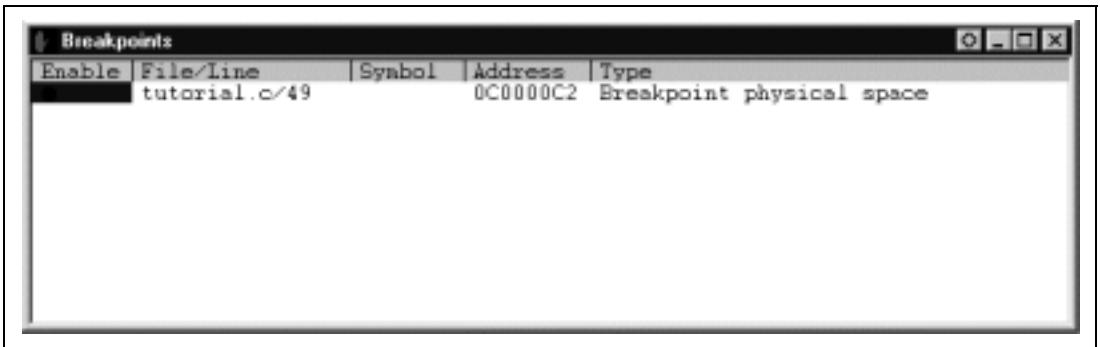
Figure 3.39   [Breakpoints] Window (Software Breakpoint Setting)

To stop the tutorial program at the breakpoint, the following procedure must be executed:

- Close the [Breakpoints] window.
- Set the program counter and stack pointer values (PC = H'0c00006c and R15 = H'0c000c00) that have been set in section 3.9, Setting Registers, in the [Registers] window. Click the [Go] button.

The program runs, and stops at the set breakpoint.

Figure 3.40   [Source] Window at Execution Stop (Software Break)

RENESAS

The [System Status] window displays the following contents.

Figure 3.41   Displayed Contents of the [System Status] Window (Software Break)

Note: The items that can be displayed in this window differ according to the product. For the items that can be displayed, refer to the online help.

3.18 Hardware Break Function

A method is given below in which the address bus condition and the read cycles for the bus status condition are set under Break Condition 1 as hardware break conditions.

- Select [Breakpoint Window] from the [View] menu. The [Breakpoints] window is displayed.
- Click the [Breakpoints] window with the right mouse button and select [Delete All] from the pop-up menu to cancel all breakpoints that have been set.
- Click the [Breakpoints] window with the right mouse button and select [Add] from the pop-up menu.
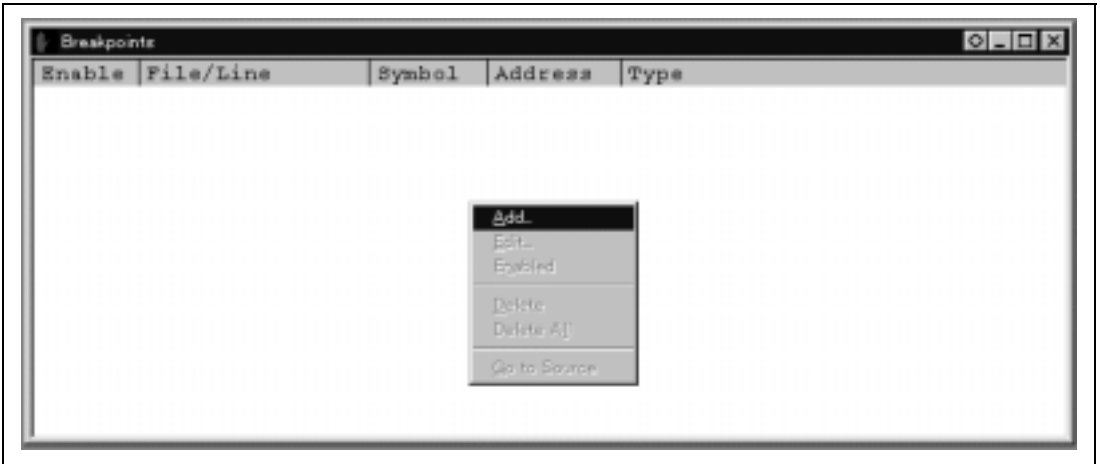
Figure 3.42   [Breakpoints] Window (Before Hardware Break Condition Setting)

RENESAS

The [Break] dialog box is displayed. To set hardware break conditions, select [Condition] in the [Break] dialog box to display the [Condition] page.
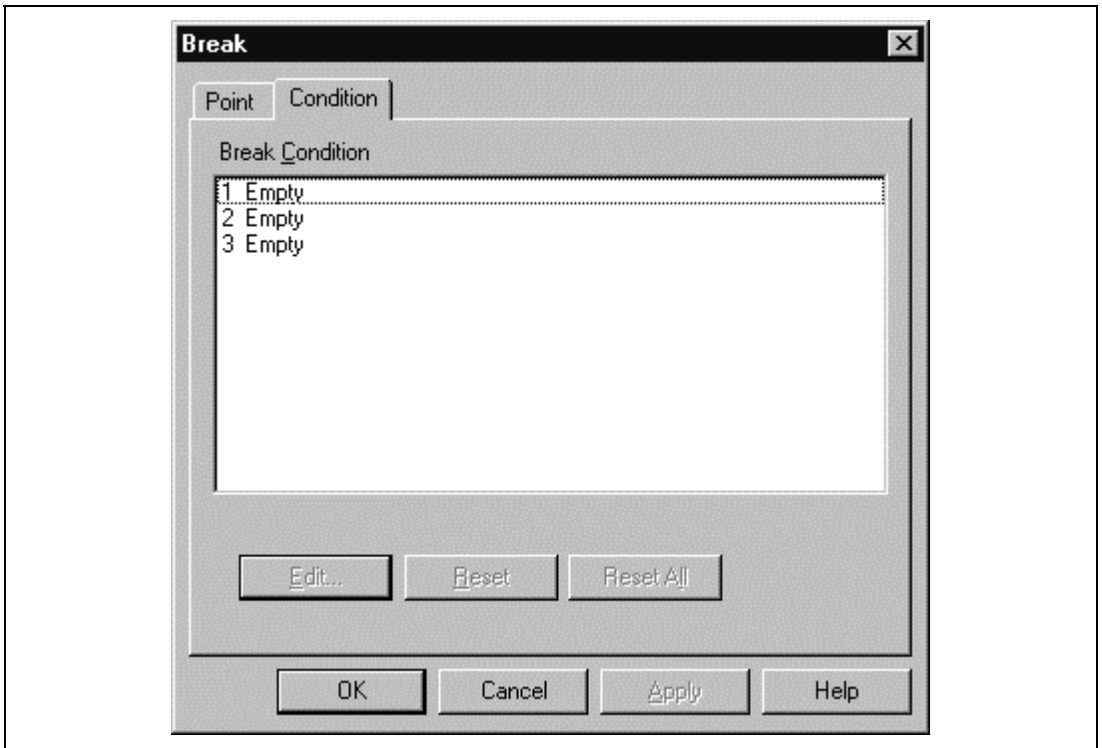
Figure 3.43   [Condition] Page ([Break] Dialog Box)

Up to three breakpoints can be set independently for the Break Condition hardware break condition. In this example, set the hardware break condition for Break Condition 1.

Note: The number of hardware break conditions differs according to the product. For the number that can be specified for each product, refer to the online help.

- Highlight the first point in the [Break Condition] list box.
- Click the [Edit...] button. The [Break Condition 1] dialog box is displayed.

RENESAS

- Clear the [Don't Care] check box in the [Address] page.
- Select the [Address] radio button and enter **H'0c0000b2** as the value in the [Address] edit box.
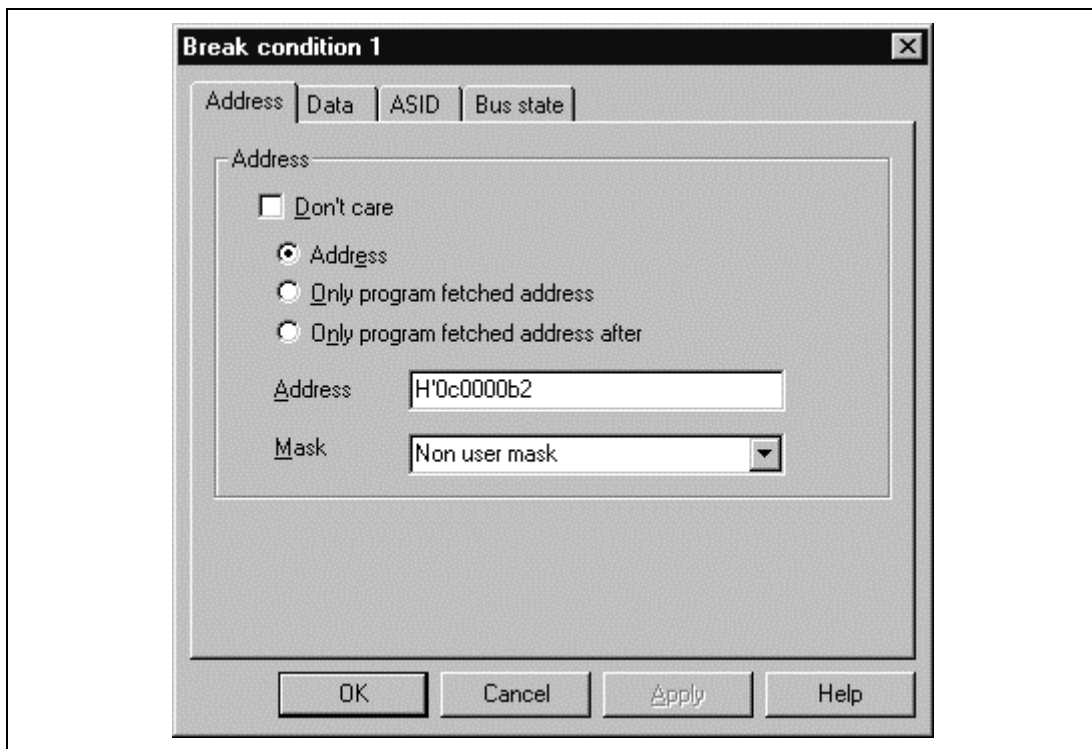
Figure 3.44   [Address] Page ([Break Condition 1] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

RENESAS

- Select [Bus State] to display the [Bus State] page.
- Select the [Read] radio button in the [Read/Write] group box.
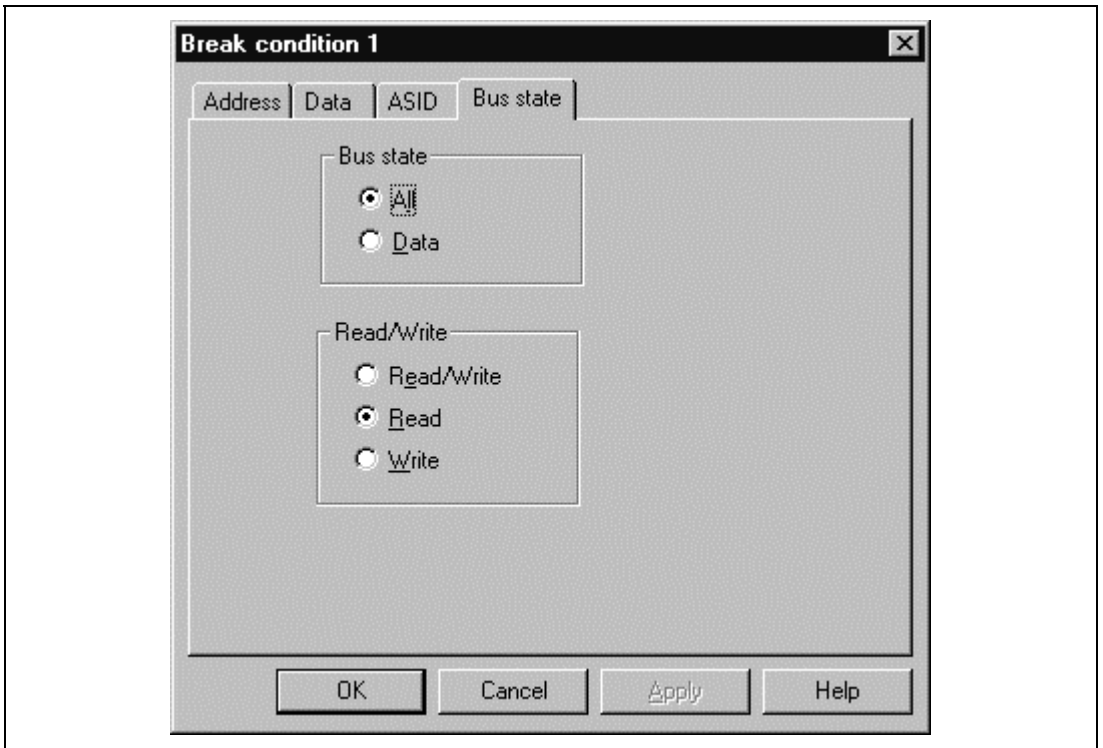
Figure 3.45   [Bus State] Page ([Break Condition 1] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

- Click the [OK] button.
- The [Break] dialog box is displayed, and the first point display in the [Break Condition] list box changes from Empty to Enable.
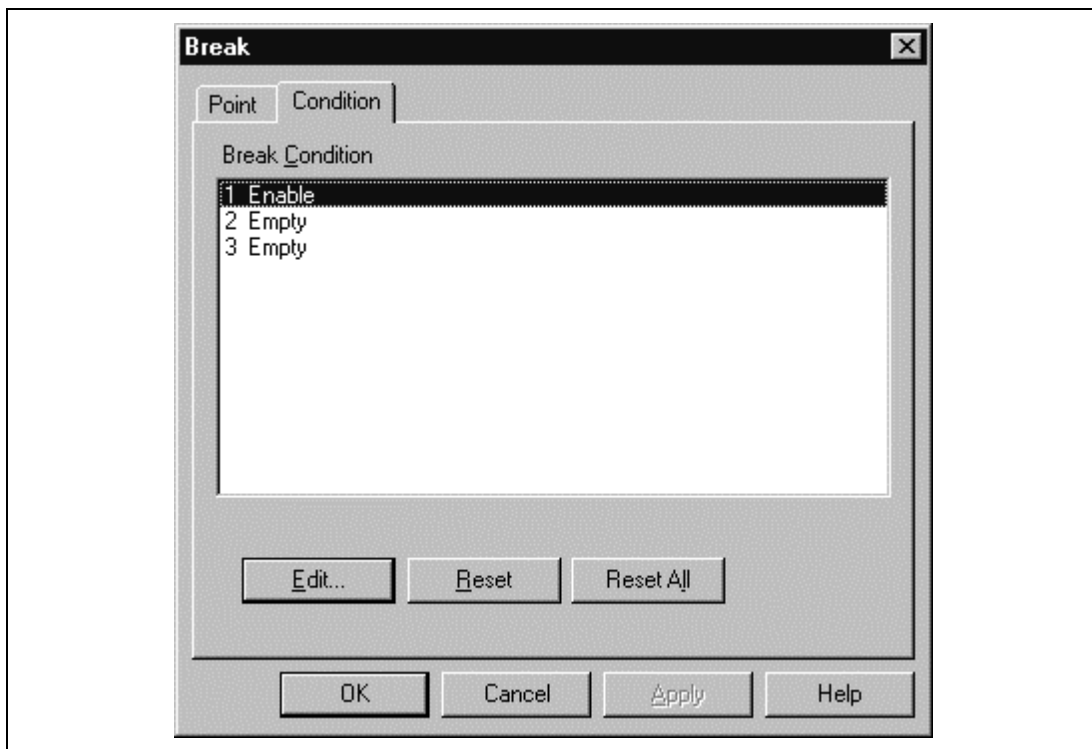
Figure 3.46   [Break] Dialog Box (After Hardware Break Condition Setting)

Note: The number of hardware break conditions differs according to the product. For the number that can be specified for each product, refer to the online help.

- Click the [OK] button.

RENESAS

The newly set hardware breakpoint is displayed in the [Breakpoints] window. With this setting, `Break Condition 1` is displayed in [Type] in the [Breakpoints] window.

This completes the setting of the Break Condition 1 hardware break condition. When the program is executed, a break will occur when address H'0c0000b2 is accessed in a read cycle.
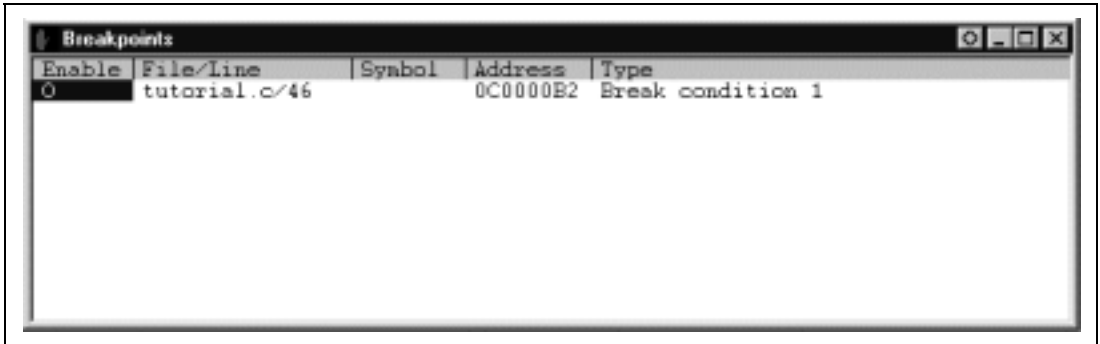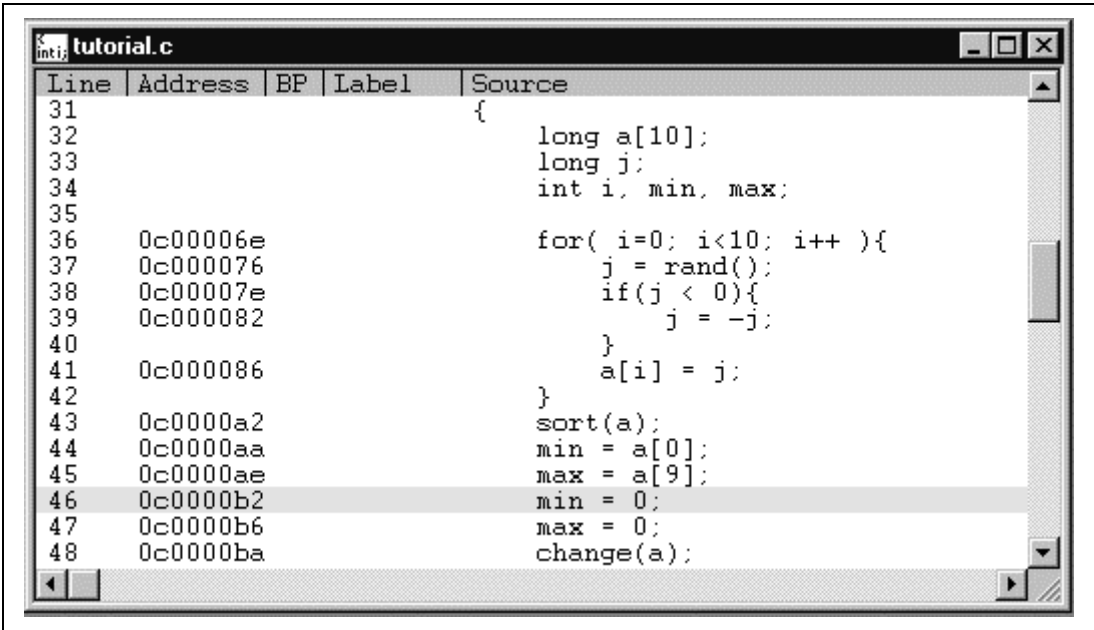
Figure 3.47   [Breakpoints] Window ([Break Condition 1] Setting)

- Close the [Breakpoints] window.
- Set the program counter and stack pointer values (PC = H'0c00006c and R15 = H'0c000c00) that have been set in section 3.9, Setting Registers, in the [Registers] window. Click the [Go] button.

RENESAS

The program runs then stops at the condition specified under Break Condition 1.

Figure 3.48 [Source] Window at Execution Stop (Break Condition 1)

RENESAS

The [System Status] window displays the following contents.

Figure 3.49   Displayed Contents of the [System Status] Window (Break Condition 1)

Note: The items that can be displayed in this window differ according to the product. For the items that can be displayed, refer to the online help.

RENESAS

3.18.1　Setting the Sequential Break Condition

The emulator has sequential break functions. When the hardware break conditions listed in table 3.3 are satisfied, program execution is halted. This mode is called sequential break.

Table 3.3　Sequential Break Conditions

Break Condition	Description
Sequential break condition 2-1	Program is halted when Break Condition 2 and Break Condition 1 are satisfied in that order.

Sequential break condition 2-1 is described below as an example.

Before executing the program, change setting in the [Configuration] dialog box. Otherwise, the sequential break does not function.

- Select [Configure Platform...] from the [Setup] menu. The [Configuration] dialog box is displayed.
- Select Sequential break condition 2-1 from the [Emulation mode] combo box.

RENESAS
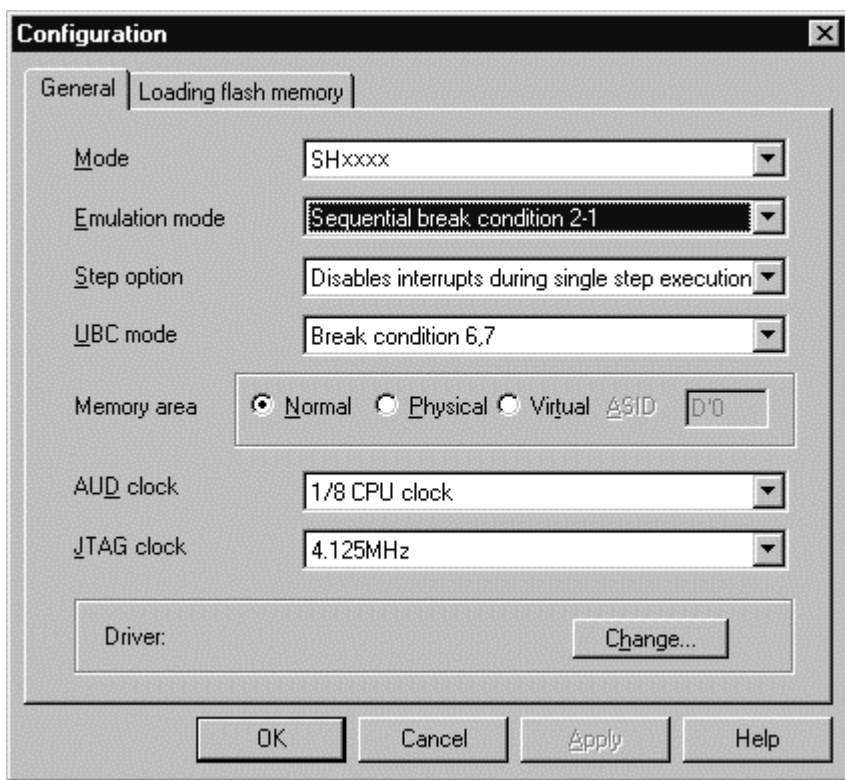
Figure 3.50 [Configuration] Dialog Box (Sequential Break Setting)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

Click the [OK] button and close the [Configuration] dialog box.

Set hardware break conditions as follows:

Break condition 1: When address H'0c0000c6 is accessed in a read cycle, a break condition is satisfied.

Break condition 2: When address H'0c0000b2 is accessed in a read cycle, a break condition is satisfied.

Follow the setting method described in the previous section.
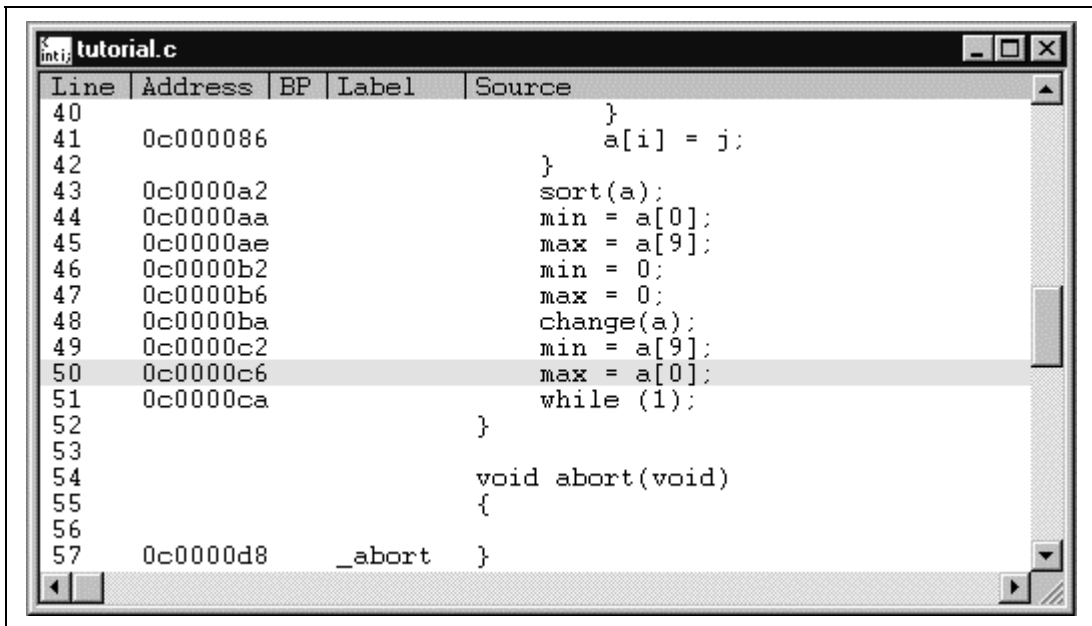
- When Break Condition 1,2 setting has been completed, the state of the [Breakpoints] window is as follows.

Figure 3.51   [Breakpoints] Window (After Sequential Break Condition Setting)

- Close the [Breakpoints] window.
- Set the program counter and stack pointer values (PC = H'0c00006c and R15 = H'0c000c00) that have been set in section 3.9, Setting Registers, in the [Registers] window. Click the [Go] button.

RENESAS

The program runs then stops at the condition specified under Break Condition 1.


```
 tutorial.c                                                    _ □ X
Line  Address  BP  Label    Source                                   ▲
40                                              }
41     0c000086                         a[i] = j;
42                                     }
43     0c0000a2                         sort(a);
44     0c0000aa                         min = a[0];
45     0c0000ae                         max = a[9];
46     0c0000b2                         min = 0;
47     0c0000b6                         max = 0;
48     0c0000ba                         change(a);
49     0c0000c2                         min = a[9];
50     0c0000c6                         max = a[0];
51     0c0000ca                         while (1);
52                                     }
53
54                              void abort(void)
55                              {
56
57     0c0000d8     _abort     }                                     ▼
◄ █                                                               ► //
```

Figure 3.52   [Source] Window at Execution Stop (Sequential Break)

The [System Status] window displays the following contents.
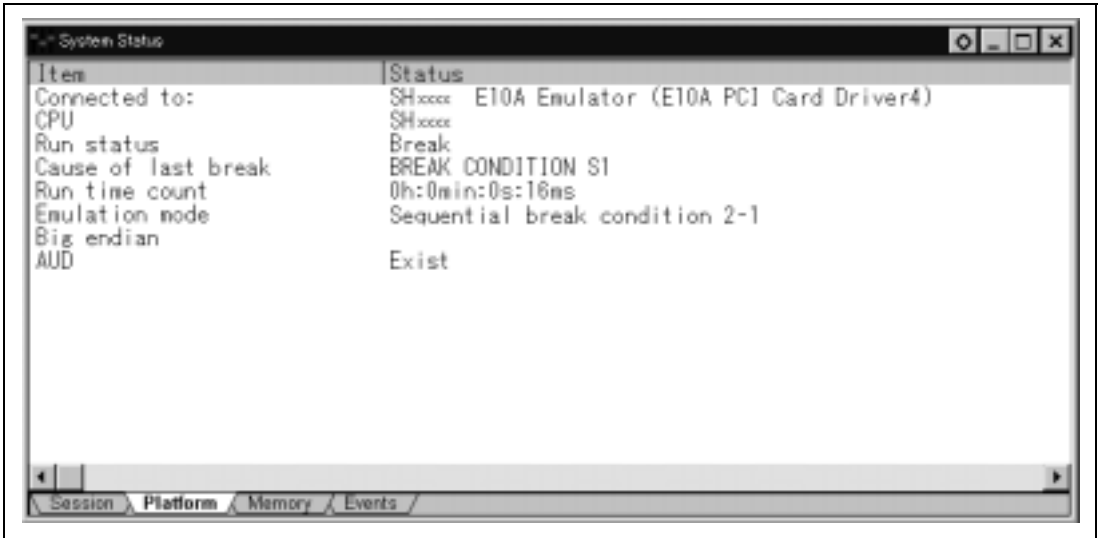
Figure 3.53   Displayed Contents of the [System Status] Window (Sequential Break)
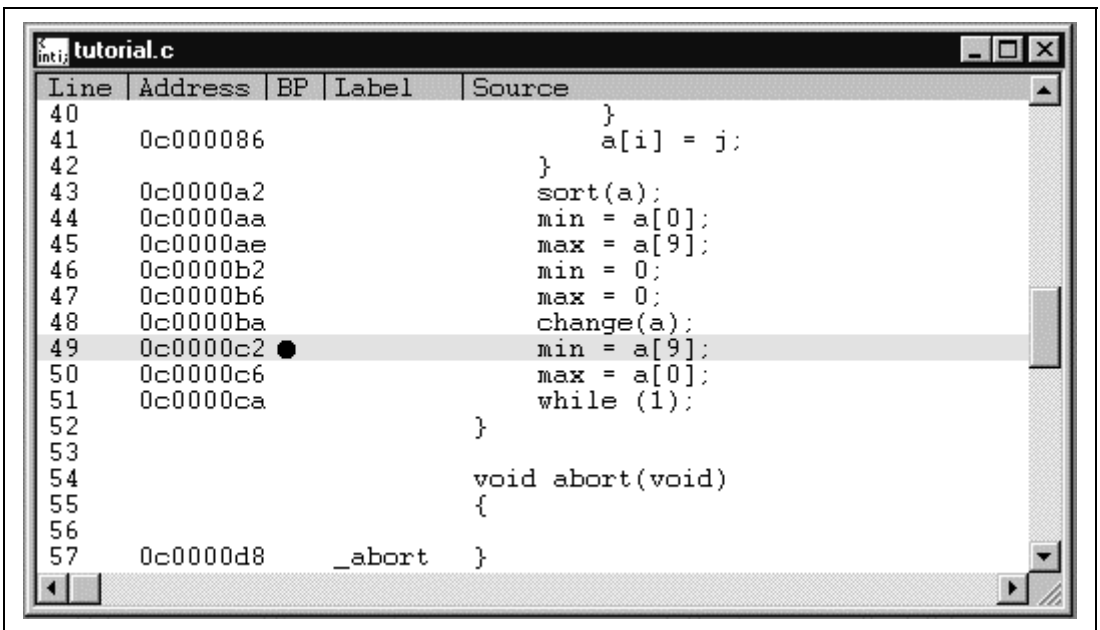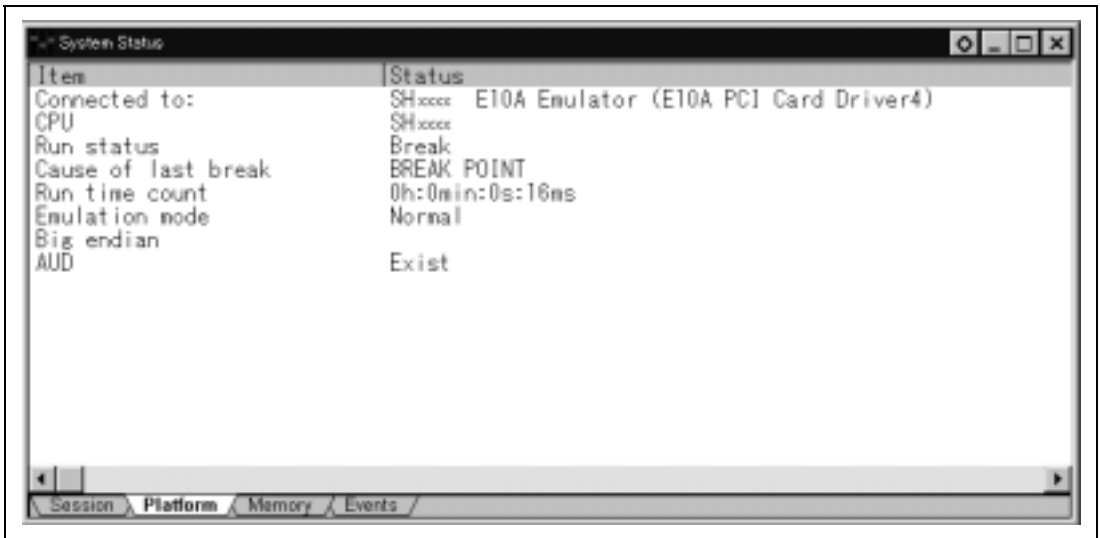
Note: The items that can be displayed in this window differ according to the product. For the items that can be displayed, refer to the online help.

RENESAS

3.19 Trace Functions

The E10A emulator has two branch-instruction trace functions.

(1) Internal Trace Function

The branch source and branch destination addresses, mnemonics, operands, and source lines are displayed. Since this function uses the trace buffer built into the MCU, a realtime trace can be acquired.

Notes: 1. The number of branch instructions that can be acquired by a trace differs according to the product. For the number that can be specified for each product, refer to the online help.

2. The internal trace function is not supported for all products. For the specifications of each product, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help.

3. The internal trace function is not extended for all products. For the specifications of each product, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help.

(2) AUD Trace Function

This is the large-capacity trace function that is enabled when the AUD pin is connected to the emulator. This function displays the branch source and branch destination addresses, mnemonics, operands, and source lines.

When the branch source and branch destination instructions are one branch, the number of branch instructions acquired by a trace is a maximum of 4,096 in the PCMCIA-type emulator and a maximum of 16,384 in the PCI-type emulator.

Table 3.4 shows the AUD trace function.

Notes: 1. The AUD trace function is not supported for all products. For the specifications of each product, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help.

2. The AUD trace function is not extended for all products. For the specifications of each product or the number of acquired branches, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help.

RENESAS

Table 3.4   AUD Trace Functions

Type	Mode	Description
Acquisition mode when branches continuously occur	Realtime trace	When the next branch occurs while the trace information is being output, the output is stopped and the next trace information is output. The user program can be executed in realtime, but some trace information will not be output.
	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.
Acquisition mode when the trace buffer of the emulator becomes full	Trace continue	This function always overwrites the oldest trace information to acquire the latest trace information.
	Trace stop	The trace information is not acquired. The user program is continuously executed.

RENESAS

3.19.1   Internal Trace Function

The branch source and branch destination information for the latest several branch instructions are displayed.

The following is a procedure to set the internal trace function (this function is not needed to be set in the emulator that does not support the AUD trace function):

1. Select [Trace] from the [View] menu.
2. Click the [Trace] window with the right mouse button and select [Acquisition] from the pop-up menu to display the [Trace Acquisition] window.
3. Select the [Internal trace] radio button in the [Trace type] group box.
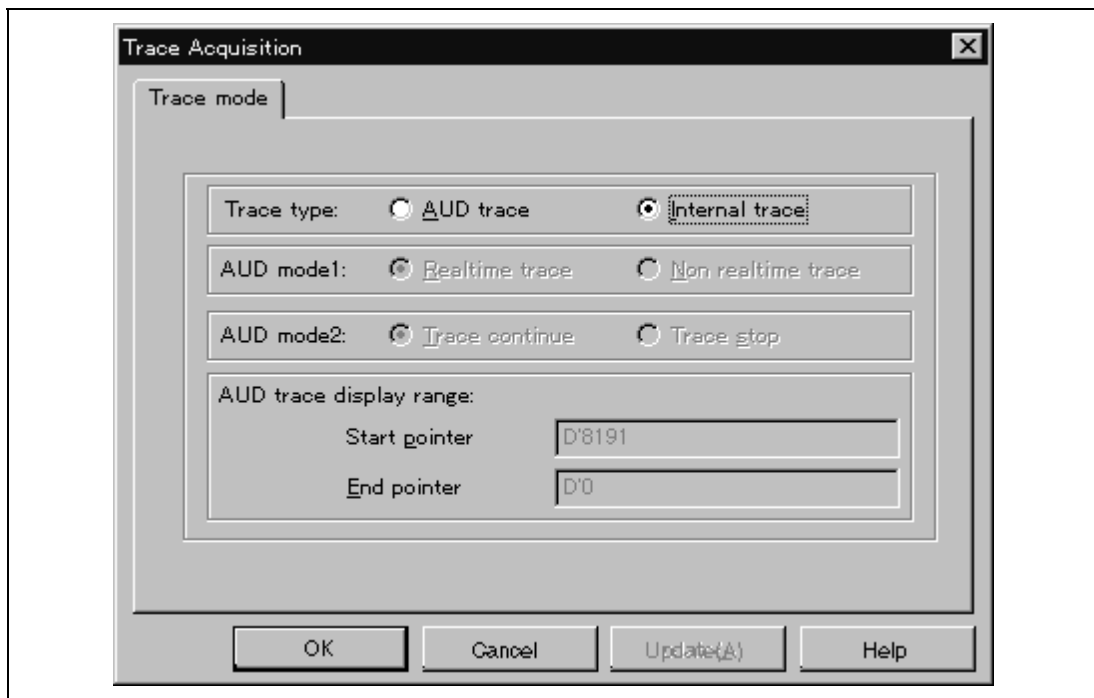
Figure 3.54   [Trace mode] Window

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

RENESAS

Run the program as shown in the example of section 3.17.1, Software Break Function. The trace results are displayed in the [Trace] window after the program execution is completed.

```
Trace - 16 records (no filter)
No.      IP          TYPE  ADDR     DATA MNEMO OPERAND            Source
-000015 -D'000007 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000014           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000013 -D'000006 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000012           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000011 -D'000005 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000010           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000009 -D'000004 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000008           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000007 -D'000003 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000006           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000005 -D'000002 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000004           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000003 -D'000001 BRANCI 0C0001F4 **** BF    @H'C0001D0:8
-000002           DESTI  0C0001D0 **** MOV.L @R15,R1              a[i] = tmp[9 -
-000001 -D'000000 BRANCI 0C0001F8 **** RTS
+000000           DESTI  0C0000C2 **** MOV.L @(H'34:4,R15),R2     min = a[9];
```

Figure 3.55 [Trace] Window

- If necessary, adjust the column width by dragging the header bar immediately below the title bar.

Note: The number of branch instructions that can be acquired by a trace differs according to the product. For the number that can be specified for each product, refer to the online help.

RENESAS

3.19.2　AUD Trace Function

This function is operational when the AUD pin of the MCU is connected to the emulator.

The following is the procedure for setting the AUD trace function (this function does not need to be set in an emulator that does not support the internal trace function):

1. Select [Trace] from the [View] menu.
2. Click the [Trace] window with the right mouse button and select [Acquisition] from the pop-up menu to display the [Trace Acquisition] window.
3. Select the [AUD trace] radio button in the [Trace type] group box.
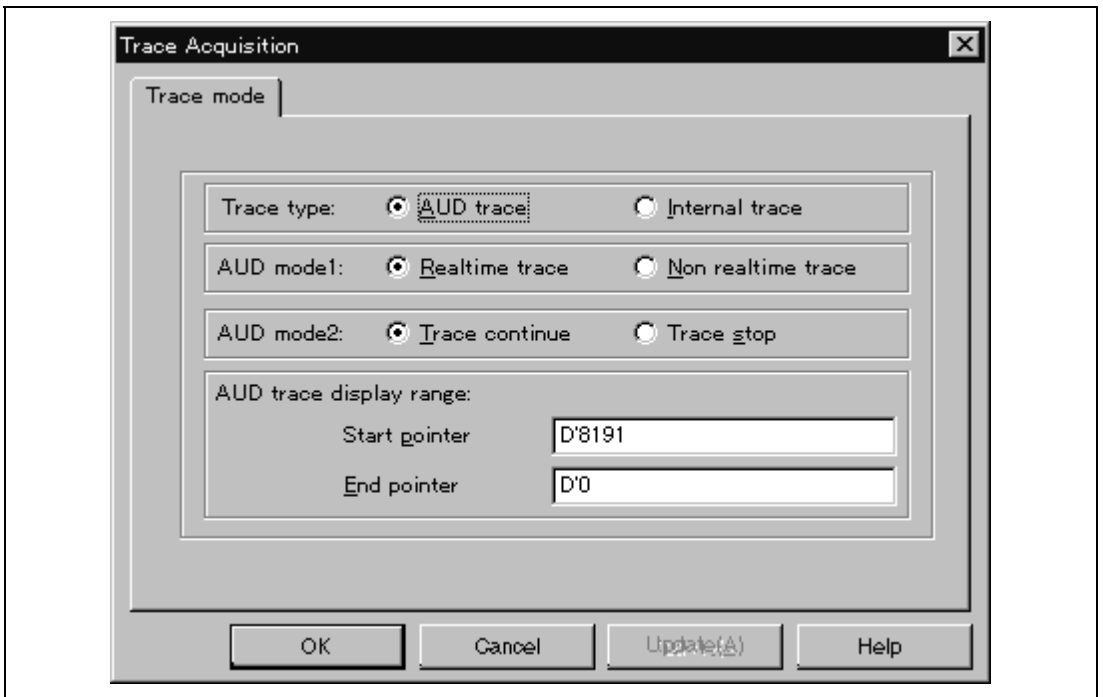
Figure 3.56　[Trace mode] Window

Note:　For a description of each option, refer to table 3.4.

Note:　The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

The trace results are displayed in the [Trace] window after the program execution is completed. The display specifications in the [Trace] window are the same as the internal trace function.

RENESAS

The following is an example of the display in the SH7751 E10A emulator.

Figure 3.57   [Trace] Window in the SH7751 E10A Emulator

RENESAS

3.19.3   VP_MAP Translation

The MCU, which has an MMU, translates internal addresses (virtual addresses) to actual memory addresses (physical addresses). Address translation is performed according to the address translation table (translation look-aside buffer: TLB) in the MCU. The MMU operates during command input wait state as well as during user program execution. When a command for memory access is executed while the MMU address translation function is enabled, the address translated by the MMU is accessed. If the specified address is not within the TLB, a TLB miss occurs, and the TLB must be updated by the user program.

The emulator has address translation functions according to the VP_MAP tables. The VP_MAP tables are the address translation tables for the emulator created with the VPMAP_SET command.

The following shows an example of how to use the VP_MAP tables.

Example:

1. Create VP_MAP tables for translating virtual addresses H'10000 to H'10FFF to physical addresses H'4000000 to H'4000FFF and virtual addresses H'11000 to H'11FFF to physical addresses H'0 to H'FFF.
    ```
    >vs 10000 10FFF 4000000 (RET)
    >vs 11000 11FFF 0 (RET)
    >vd (RET)
    <VADDR_TOP>   <VADDR_END>   <PADDR_TOP>
    00010000      00010FFF      04000000
    00011000      00011FFF      00000000
    DISABLE
    ```
2. Then, enable the VP_MAP tables. (When the tables are disabled, addresses are not translated.)
    ```
    >ve ;enable (RET)
    >vd (RET)
    <VADDR_TOP>   <VADDR_END>   <PADDR_TOP>
    00010000      00010FFF      04000000
    00011000      00011FFF      00000000
    ENABLE
    ```

    Here, virtual addresses correspond to physical addresses as shown in figure 3.58.

RENESAS
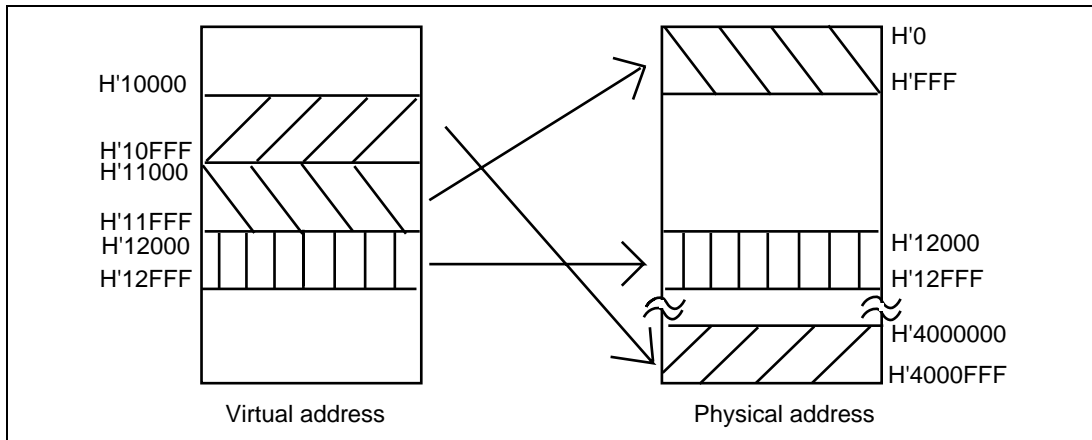
Figure 3.58   Address Translation according to VP_MAP Tables

How to translate addresses depends on the settings of the radio buttons of the memory area group in the [Configuration] dialog box. The following shows how to translate addresses in each setting state.

- When the Normal radio button is selected:

    The VP_MAP table has a priority over the TLB. When the VP_MAP table is enabled and the specified address is within the VP_MAP table settings, the emulator translates the address according to the VP_MAP table. If the specified address is outside the VP_MAP table settings even when the VP_MAP table is enabled, or when the VP_MAP table is disabled, the emulator translates the address according to the MMU state.

- When the Virtual radio button is selected:

    The address is translated according to the TLB. If the specified address is outside the TLB table settings, a TLB error will occur.

- When the Physical radio button is selected:

    The address is not translated.

RENESAS

Table 3.5 Address Translation Tables

Radio Button*	VP_MAP		MMU		Table Used for Translation
	Enabled/ Disabled	Within/ Outside the range	Enabled/ Disabled	Within/Outside the TLB Range	
Normal	Enabled	Within the Range	Enabled	Within the Range	Translated according to the VP_MAP table
				Outside the range	Translated according to the VP_MAP table
			Disabled	Within/outside the range	Translated according to the VP_MAP table
		Outside the Range	Enabled	Within the Range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within/outside the range	Not translated
	Disabled	Within/ outside the range	Enabled	Within the Range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within/outside the range	Not translated
Virtual	Enabled/ disabled	Within/ outside the range	Enabled	Within the Range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within the Range	Translated according to the TLB table
				Outside the range	TLB error
Physical	Enabled/ disabled	Within/ outside the range	Enabled/ disabled	Within/outside the range	Not translated

Note: Specified by the [Memory area] group box in the [Configuration] dialog box.

RENESAS
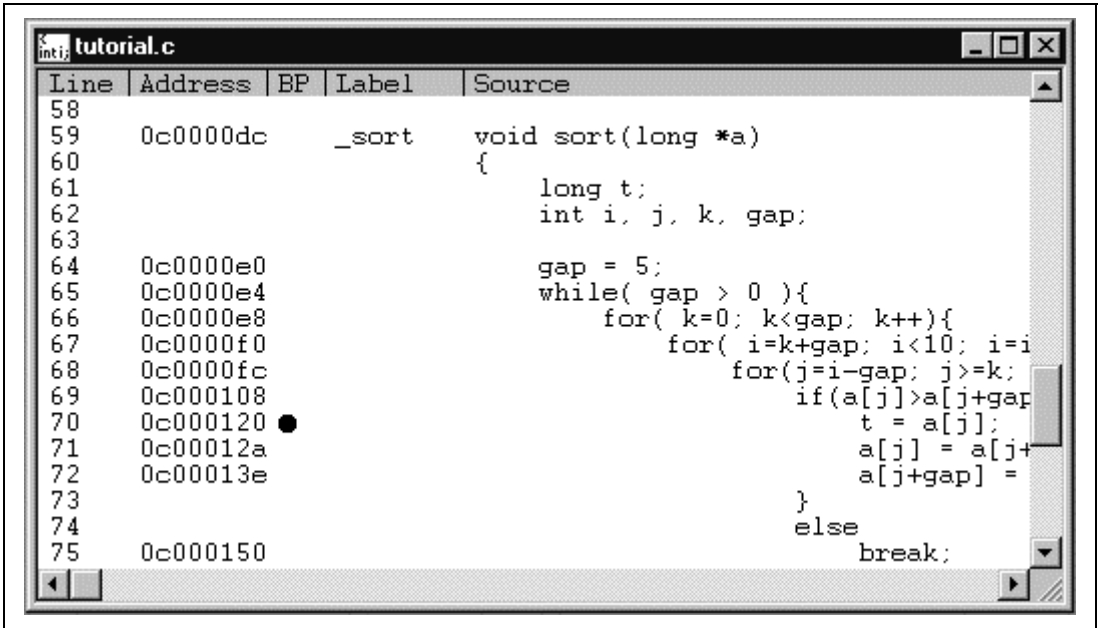
3.20 Stack Trace Function

The emulator uses the stack's information to display the name of the calling function for a function at which the program counter is currently pointing.

Notes: 1. This function can be used only when the load module that has the Dwarf2-type debugging information is loaded.
2. For details on the stack trace function, refer to the online help.

• Double-click the [BP] column in the sort function and set a software breakpoint.

Figure 3.59   [Source] Window (Software Breakpoint Setting)

• Set the same program counter and stack pointer values (PC = H'0c00006c and R15 = H'0c000c00) as were set in section 3.9, Setting Registers (again, use the [Registers] window). Click the [Go] button.
• After the break in execution, select [Stack Trace] from the [View] menu to open the [Stack Trace] window.

RENESAS

```
Stack Trace                                        _ □ ×
Kind Name                        Value
F     sort(long*)              { 0x0c000120 }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
F     main()                   { 0x0c0000aa }
```
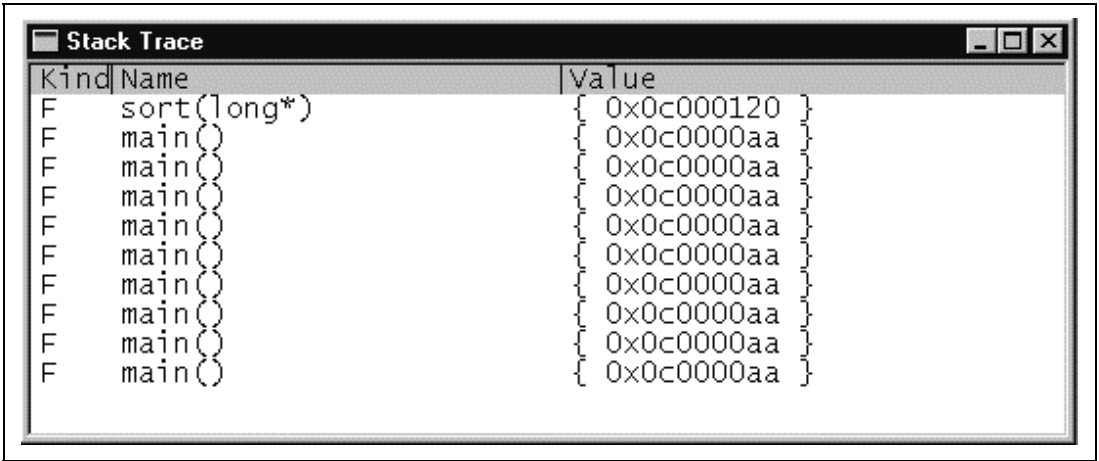
Figure 3.60   [Stack Trace] Window

Figure 3.60 shows that the position of the program counter is currently at the selected line of the
sort() function, and that the sort() function is called from the main() function.

Notes: 1. If the function is not deeply nested (lower than 10), the main() function will be
            displayed multiple times.

        2. For details on this function, refer to the online help. The online help is opened when
            the [F1] key is pressed in the [Stack Trace] window.

3.21 Profiling Function

The profile function can measure performance for each function.

Notes: 1. Realtime operation is not possible while this function is in operation, since internal breaks are generated during program execution. Measuring the profile itself affects the measurements. For details, refer to section 6, SHxxxx Emulator Specifications.

2. When this function is to be used, click the [Load stack information file (SNI file)] check box in the [Load Program] dialog box to load the stack information file.

3. Performance profile measurement is not supported for all products. On those products for which it is supported, its characteristics differ according to the product. For specifications for each product, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help file.

4. For details, refer to section 13 in the Hitachi Debugging Interface User's Manual.

- Select [Profile-List] from the [View] menu to open the [Profile-List] window. A different set of data to be measured can be set for each function.

Figure 3.61 [Profile-List] Window

RENESAS

- The profile function is now enabled. Place the mouse cursor on an entry in the [Profile-List] window, click the right-hand mouse button, then select [Enable Profiler] from the pop-up menu.
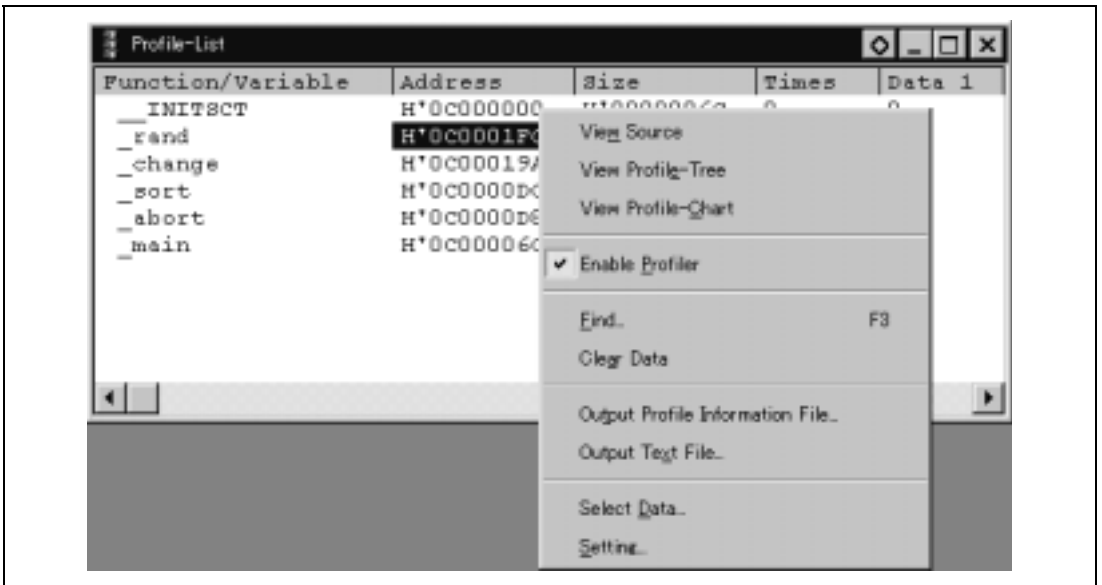
Figure 3.62   Selection of [Enable Profiler]

RENESAS

- Data to be measured for the selected function is now set. Select [Select Data] from the pop-up menu by clicking with the right mouse button. The [Select Data] dialog box is displayed.
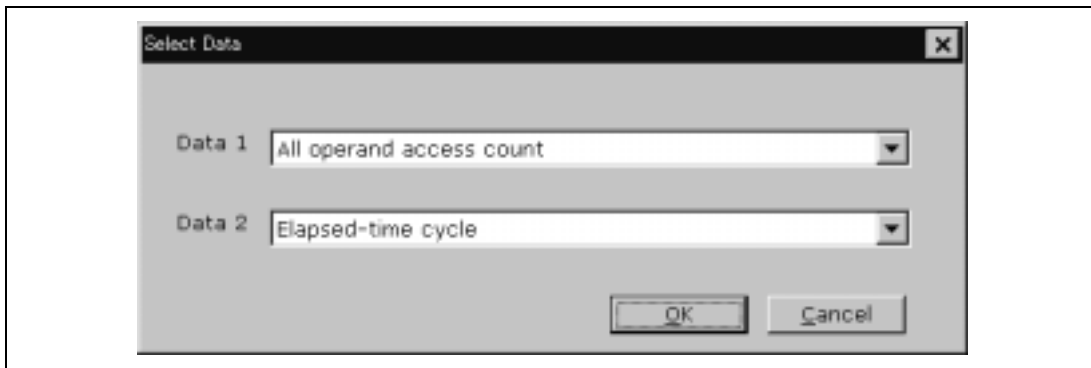
Figure 3.63   [Select Data] Dialog Box

- Use the [Select Data] dialog box to select the data to be measured. [All operand access count] is selected for Data1 as a first item to be measured. [Elapsed-time cycle] is selected for Data2 as a second item to be measured.
- After the data has been selected, press the [OK] button.
- Double-click the [BP] column for the while statement of the main function to set a software breakpoint.

RENESAS

Figure 3.64 [Source] Window (Software Break Setting)

- Set the same program counter and stack pointer values (PC = H'0c00006c and R15 = H'0c000c00) as were set in section 3.9, Setting Registers (again, use the [Registers] window). Click the [Go] button.
- After the break in execution, the results of the measurements are displayed in the [Profile-List] window.

Figure 3.65 [Profile-List] Window

- Figures 3.66 and 3.67 show the [Profile-Tree] and [Profile-Chart] windows, respectively.

Figure 3.66   [Profile-Tree] Window
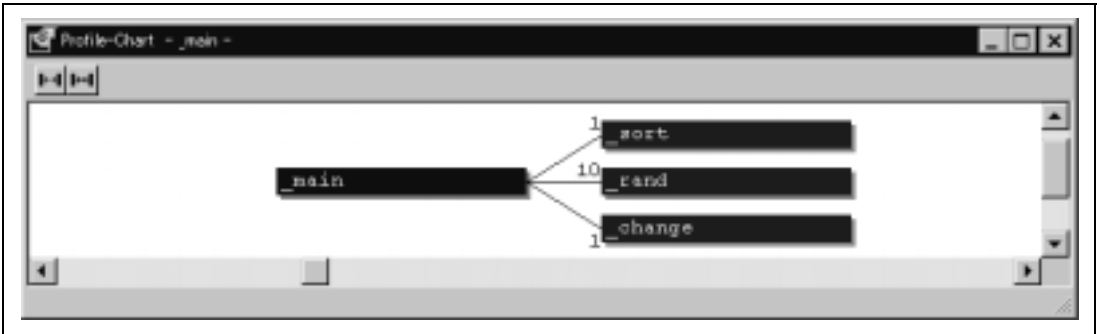
Figure 3.67   [Profile-Chart] Window

RENESAS

3.22 Download Function to the Flash Memory Area

The E10A emulator enables downloading to the flash memory area. This function requires a program for writing the flash memory (hereinafter referred to as a write module), a program for erasing the flash memory (hereinafter referred to as an erase module), and the RAM area for downloading and executing these modules.

Note: The write/erase module must be prepared by users.

- Interface with write/erase module and E10A emulator firmware

  The write/erase module is branched from the E10A emulator firmware. To branch from the E10A emulator firmware to the write/erase module or to return from the write/erase module to the E10A emulator firmware, the following conditions must be observed:

- Describe all the write/erase modules with the assembly language.
- Guarantee all the general/control register values before and after calling the write/erase module.
- Return the write/erase module to the calling source after processing.

  The module interface must be as follows to pass correctly the information that is required for flash memory accessing.

Table 3.6 Module Interface

Module Name	Argument	Return Value
Write module	R4(L): Write address	R0(L): End code
	R7(L): Verify option 0 = no verify, 1 = verify	Normal end = 0, Abnormal end = other than 0, Verify error = BT
	R5(L): Access size 0x4220 = byte, 0x5720 = word, 0x4C20 = longword	
	R6(L): Write data	
Erase module	R4(L): Access size 0x4220 = byte, 0x5720 = word, 0x4C20 = longword	None

Note: The (L) means the longword size.

Note: **Write module:**
The write data for the access size is set to the R6 register. When the access size is word or byte, 0 is set to the upper bit of the R6 register.

RENESAS

— Flash memory download method

It is required to perform necessary settings on the [Loading flash memory] page in the [Configuration] window for downloading to the flash memory.
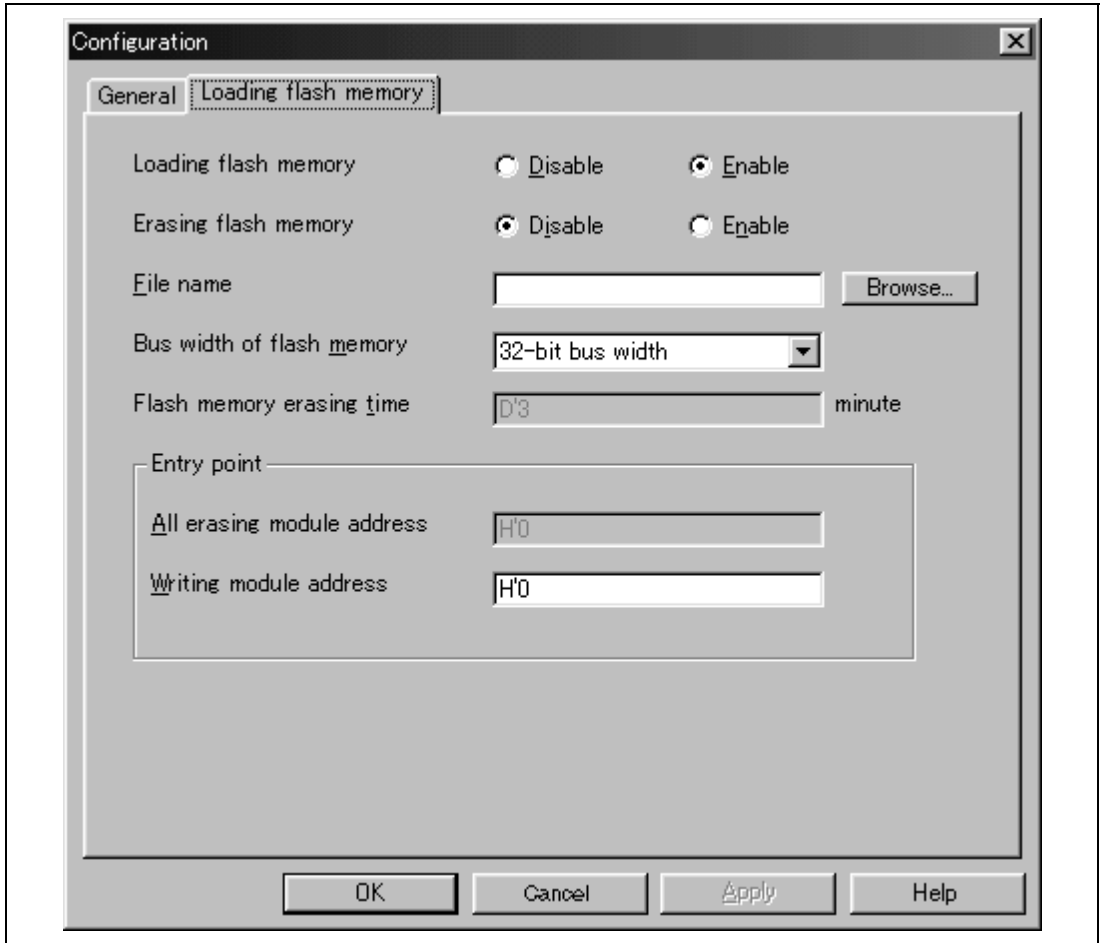
Figure 3.68 [Loading flash memory] Page

RENESAS

Table 3.7 shows the options for the [Loading flash memory] page.

Table 3.7   [Loading flash memory] Page Options

Option	Description
[Loading flash memory] radio button	Sets Enable for flash memory downloading.
	At Enable, when [File load] is selected from the [File] menu for downloading, the write module is always called. Enable: Download to the flash memory Disable: Not download to the flash memory
[Erasing flash memory] radio button	Sets Enable for erasing before the flash memory is written.
	At Enable, the erase module is called before calling the write module. Enable: Erase the flash memory Disable: Not erase the flash memory
[File name] edit box	Sets the write/erase module name. The file that has been set is loaded to the RAM area before loading to the flash memory. A maximum of 128 characters can be input for the file name.
[Bus width of flash memory] list box	Sets the bus width of the flash memory.
[Flash memory erasing time] edit box*	Sets the TIMEOUT value at flash memory erasing. Increase the value if erasing requires much time; the default time is three minutes. The radix for the input value is decimal. It becomes hexadecimal by adding H'.
[Entry point] group box	Sets the calling destination address of the write/erase module.
	[All erasing module address] edit box: Inputs the calling destination address of the erase module. [Writing module address] edit box: Inputs the calling destination address of the write module.

Note: Although the values that can be set are D'0 to D'65535, the TIMEOUT hours may be extended according to the set value. Therefore, it is recommended to input the minimum value. The value to be input must only be positive integer.

— Notes on using the flash memory download function

The following are notes on downloading to the flash memory.

- When the flash memory download is enabled, downloading other than to the flash memory area is disabled.
- Downloading is only enabled to the flash memory area. Perform memory write or software break only to the RAM area.
- When the flash memory erase is enabled, the [Stop] button cannot stop erasing.
- The area for the write/erase module must be MMU-disabled space.

RENESAS

— Examples of downloading to the flash memory

The following shows examples of downloading to the flash memory manufactured by Intel Corporation (type number: G28F640J5-150) that has been mounted on Hitachi's SH7751 CPU board (type number: HS7751STC01H). A sample is provided in the \Fmtool folder in the installation destination folder. Create the program for user specification according to this sample. The SH7751 E10A emulator must be used when the SH7751 CPU board is used.

Table 3.8   Example of Board Specifications

Item		Contents
SDRAM address		H'0C000000 to H'0FFFFFFF
Flash memory address		H'01000000 to H'01FFFFFF
Bus width of flash memory		32 bits
Operating environments	CPU internal frequency	167 MHz
	Bus frequency	55.7 MHz
	CPU internal module frequency	27.83 MHz
	Endian	Big endian
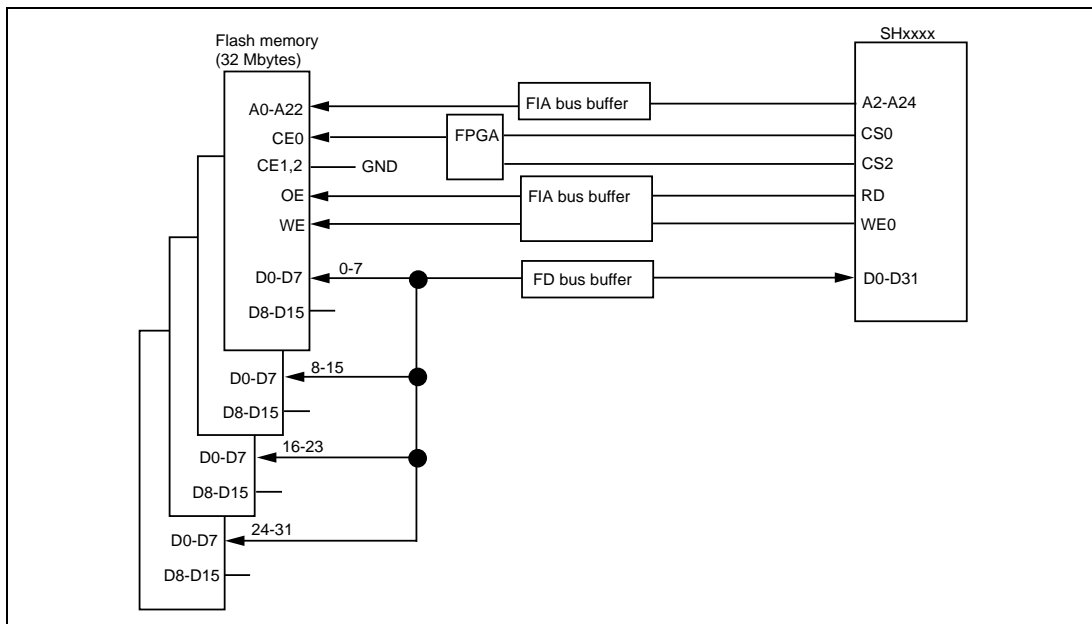
Figure 3.69   Flash Memory Wiring

Table 3.9   Sample Program Specifications

Item	Contents
RAM area to be used	H'0C001000 to H'0C0015BF
Write module start address	H'0C001100
Erase module start address	H'0C001000

(i) Since the SDRAM is used, the bus controller is set.

(ii) Options on the [Loading flash memory] page in the [Configuration] window are set as follows:
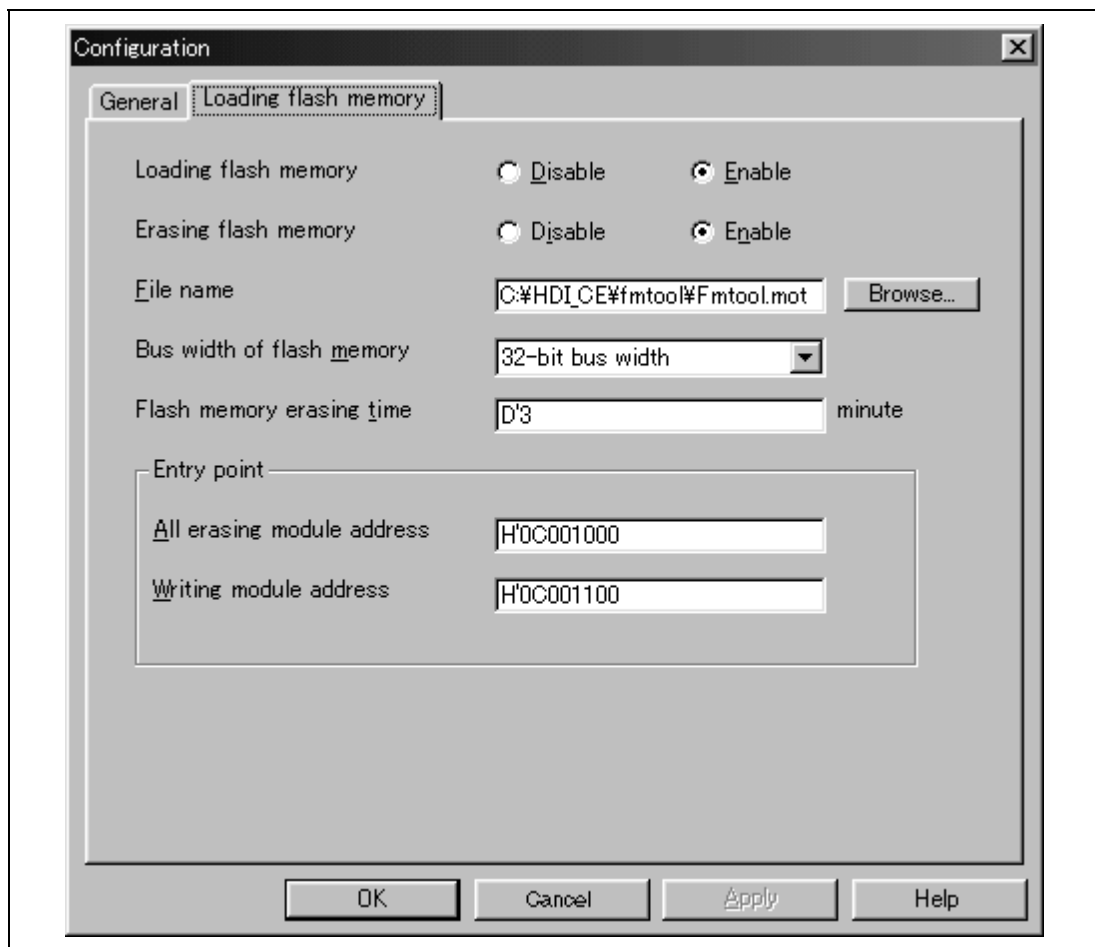
Figure 3.70   [Loading flash memory] Page

Notes: 1. When the data has already been written in the flash memory, be sure to select [Enable] for [Erasing flash memory]. If [Disable] is selected, a verify error occurs.

2. When [Erasing flash memory] is selected, it takes about one minute.

(iii) [Load Program…] is selected from the [File] menu for downloading to the flash memory area.

RENESAS

3.23   What Next?

This tutorial has described the major features of the emulator and the use of the HDI.

Sophisticated debugging can be carried out by using the emulation functions that the emulator offers. This provides for effective investigation of hardware and software problems by accurately isolating and identifying the conditions under which such problems arise.

Further details on the use of the HDI can be found in the separately issued Hitachi Debugging Interface User's Manual.

Section 4   Descriptions of Windows

4.1   HDI Windows

HDI window menu bars and the corresponding pull-down menus are listed in table 4.1. Where a description of a menu is included in the Hitachi Debugging Interface User's Manual or in this manual, a O mark or the relevant section number is shown. Related commands in the E10A Emulator User's Manual are also shown.

Table 4.1   HDI Window Menus and Related Manual Entries

Menu Bar	Pull-Down Menu	Hitachi Debugging Interface User's Manual	This Manual
File menu	New Session…	O	—
	Load Session…	O	—
	Save Session	O	2.6
	Save Session As…	O	—
	Load Program…	O	3.7.1
	Initialize	O	—
	Exit	O	—
Edit Menu	Cut	O	—
	Copy	O	—
	Paste	O	—
	Find…	O	—
	Evaluate…	O	—

RENESAS

Table 4.1   HDI Window Menus and Related Manual Entries (cont)

Menu Bar	Pull-Down Menu	Hitachi Debugging Interface User's Manual	This Manual
View Menu	Breakpoints	O	3.11, 3.17.1, 4.2.4, 6.5.5
	Command Line	O	—
	Disassembly...	O	—
	I/O Area	O	—
	Labels	O	—
	Locals	O	3.16
	Memory...	O	3.12
	Performance Analysis	O	—
	Profile-List	O	3.21
	Profile-tree	O	3.21
	Registers	O	3.9
	Source…	O	3.7.2
	Stack Trace	X	3.20
	Status	O	3.10, 3.17.1, 4.2.9
	Trace	O	4.2.7, 6.5.3, 6.5.7
	Watch	O	3.13
Run Menu	Reset CPU	O	—
	Go	O	3.10
	Reset Go	O	—
	Go to Cursor	O	—
	Set PC To Cursor	O	—
	Run…	O	—
	Step In	O	3.14.1
	Step Over	O	3.14.3
	Step Out	O	3.14.2
	Step…	O	—
	Halt	O	—

RENESAS

Table 4.1   HDI Window Menus and Related Manual Entries (cont)

Menu Bar	Pull-Down Menu	Hitachi Debugging Interface User's Manual	This Manual
Memory Menu	Refresh	O	—
	Load	O	—
	Save	O	—
	Verify	O	—
	Test	O	—
	Fill	O	—
	Copy	O	—
	Compare	O	—
Setup Menu	Status bar	O	—
	Options	O	—
	Radix	O	—
	Customise	O	—
	Configure Platform…	O	3.5, 4.2
Window Menu	Cascade	O	—
	Tile	O	—
	Arrange Icons	O	—
	Close All	O	—
Help Menu	Index	O	—
	Using Help	O	—
	Search for Help on	O	—
	About HDI	O	—

RENESAS

4.2 Descriptions of Each Window

This section describes each window. Figures in this section are used as examples. Each E10A emulator type has explanatory notes. Read section 6, SHxxxx E10A Emulator Specifications.

4.2.1 [Configuration] Dialog Box

Function:

This dialog box sets the emulation conditions of the emulator.

Window:
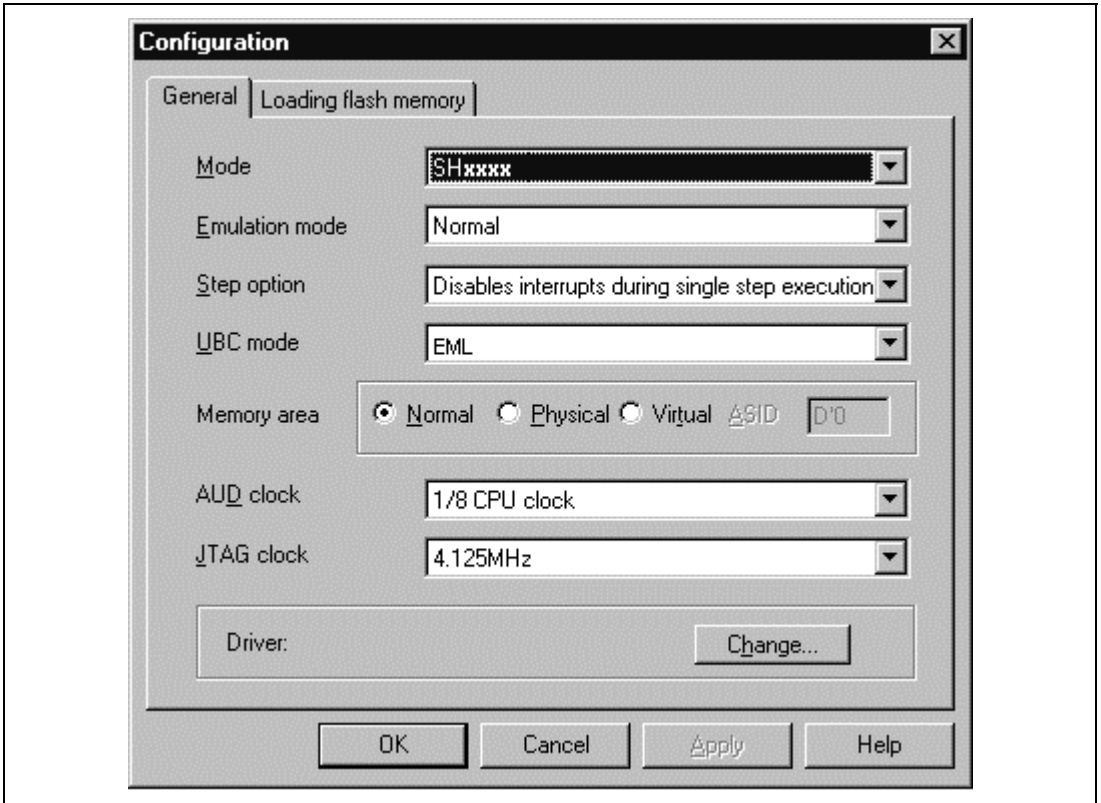
Figure 4.1 [Configuration] Dialog Box

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

RENESAS

Description:

The [Configuration] dialog box consists of the [General] page listed in table 4.2.

Table 4.2   [Configuration] Dialog Box Page

Page Name	Description
[General]	Sets and displays the emulation mode conditions.
[Loading flash memory]	Sets the download function for the flash memory.

Clicking the [OK] button sets the emulation conditions. If the [Cancel] button is clicked, this dialog box is closed without setting the emulation conditions.

RENESAS

(1) [General] Page ([Configuration] Dialog Box)

Function:

This page sets the emulator operation conditions, displays the device name, sets the emulation mode, UBC mode, and memory area (only for a product that supports a device with the MMU function), sets and displays the AUD clock (AUDCK) and JTAG clock (TCK), and selects the driver.

Window:
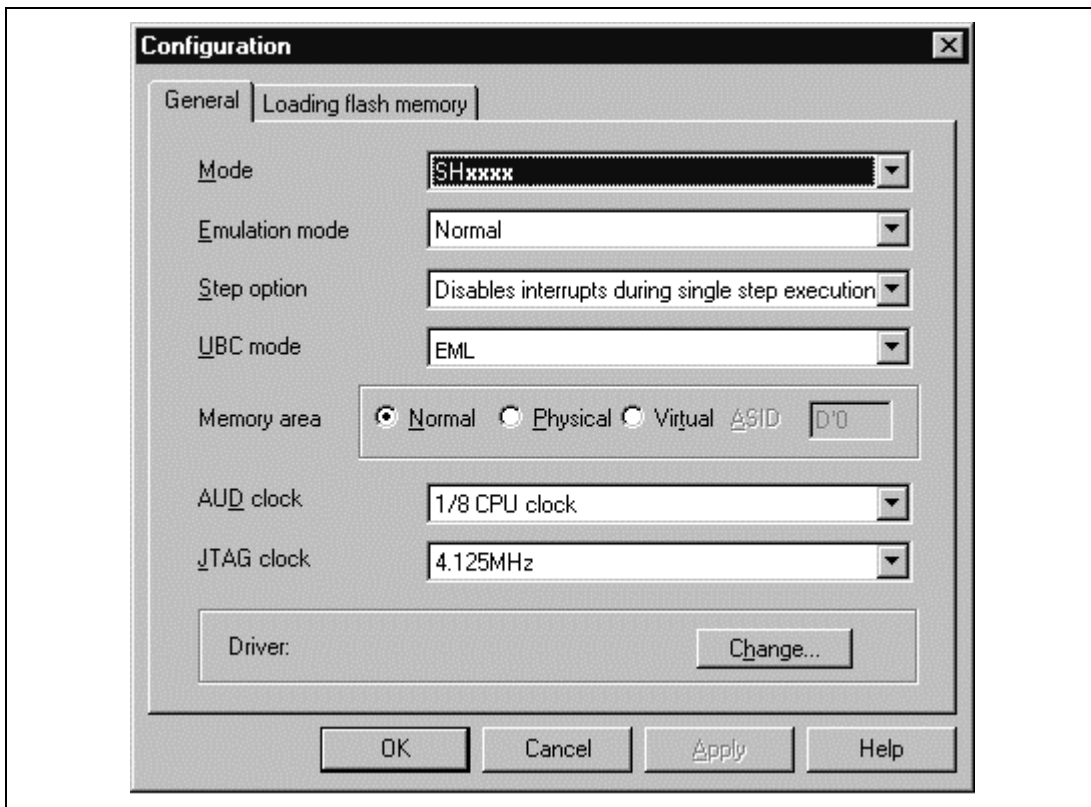
Figure 4.2   [General] Page ([Configuration] Dialog Box)

Note: The items and displayed contents that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

RENESAS

Description:

Table 4.3　[General] Page Options

Option	Description
[Mode] combo box	Displays the device name.
[Emulation mode] combo box	Selects the execution mode. Select Normal to perform normal emulation. Select No Break to disable breakpoint settings. Select Sequential break Condition 2-1, etc. to use the sequential break function*[1]. (For Sequential break Condition 2-1, execution stops when conditions are satisfied in the order of Break Condition 2 and Break Condition 1.)
[Step option] combo box	Enables or disables interrupts during step execution.
	Disables interrupts during single step execution: Interrupts during step execution are masked. Enables interrupts during single step execution: Interrupts during step execution are released.
[UBC mode] combo box	EML: The UBC is used as a Break Condition by the emulator.
	USER: The UBC is released for users. In this case, the [Break Condition] page becomes non-active.
[Memory area] group box	Sets the address setting mode to the memory area.
	The default is Normal. When the VP_MAP is enabled and the address is within the table range, address translation is done according to the VP_MAP table. For other cases, address translation is done according to the MMU state.
	Select Physical when setting with a physical address. Select Virtual when address translation is done by the TLB table.
[AUD clock] combo box	Selects the AUD clock*[2].
[JTAG clock] combo box	Sets the JTAG frequency*[3].
[Driver] group box	Displays the driver currently selected.
[Change...] button	Displays the [E10A Driver Details] dialog box. Use when a driver currently connected is changed.

Notes: 1. When using the sequential break function, set the corresponding hardware break conditions.
　　　2. The range of frequencies that the AUD operates under is different according to the devices used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).
　　　3. The range of frequencies that the JTAG operates at is different according to the devices used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).

RENESAS

When a driver is to be changed with the [Change..] button, the following message is displayed.
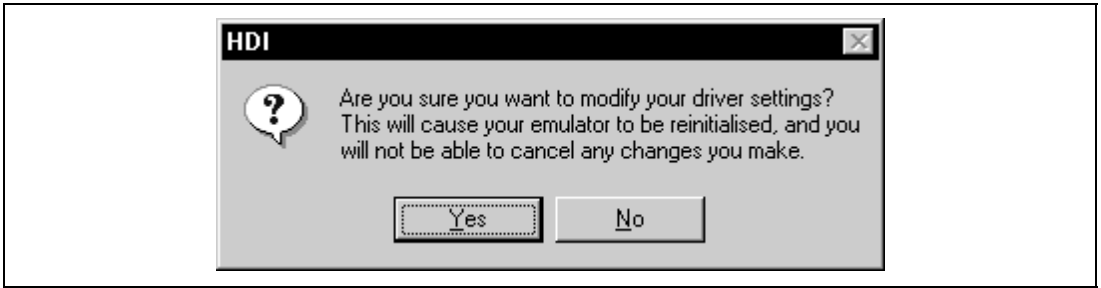
Figure 4.3   Warning Message Box

When the [Yes] button is clicked, the [E10A Driver Details] dialog box is displayed. When the [No] button is clicked, the display returns to the [Configuration] dialog box.

Related Command:

GO_OPTION command

RENESAS

(2) [E10A Driver Details] Dialog Box Function

Function:

When the [Change] button in the [Driver] group box is clicked on the [General] page in the [Configuration] dialog box, the [E10A Driver Details] dialog box is displayed.
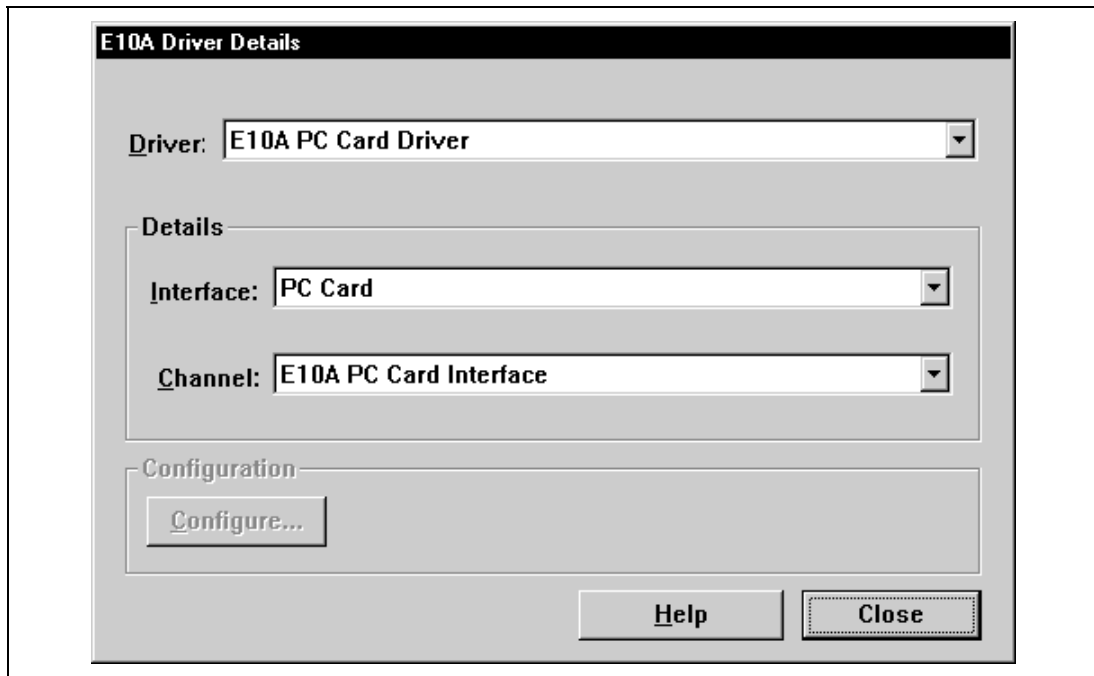
Figure 4.4   [E10A Driver Details] Dialog Box

Description:

Table 4.4   Options for the [E10A Driver Details] Dialog Box

Option	Description
[Driver] combo box	Selects the driver to connect the HDI with the emulator. Selects [E10A PC Card Driver] to use the PCMCIA card emulator. Selects [E10A PCI Card Driver] to use the PCI card emulator. For details, refer to section 6.5.1, Emulator Driver Selection.
[Interface] combo box	Displays the interface name of the card emulator to be connected. Selects [PC Card] to use the PCMCIA card emulator. Selects [PCI] to use the PCI card emulator. (If the driver is not installed, the [PC Card] or [PCI] is not displayed.)
[Channel] combo box	Displays the interface to which the board is connected.

(3) [Loading flash memory] Page ([Configuration] Dialog Box)

Function:

Downloading to the flash memory is set on the [Loading flash memory] page.

Window:
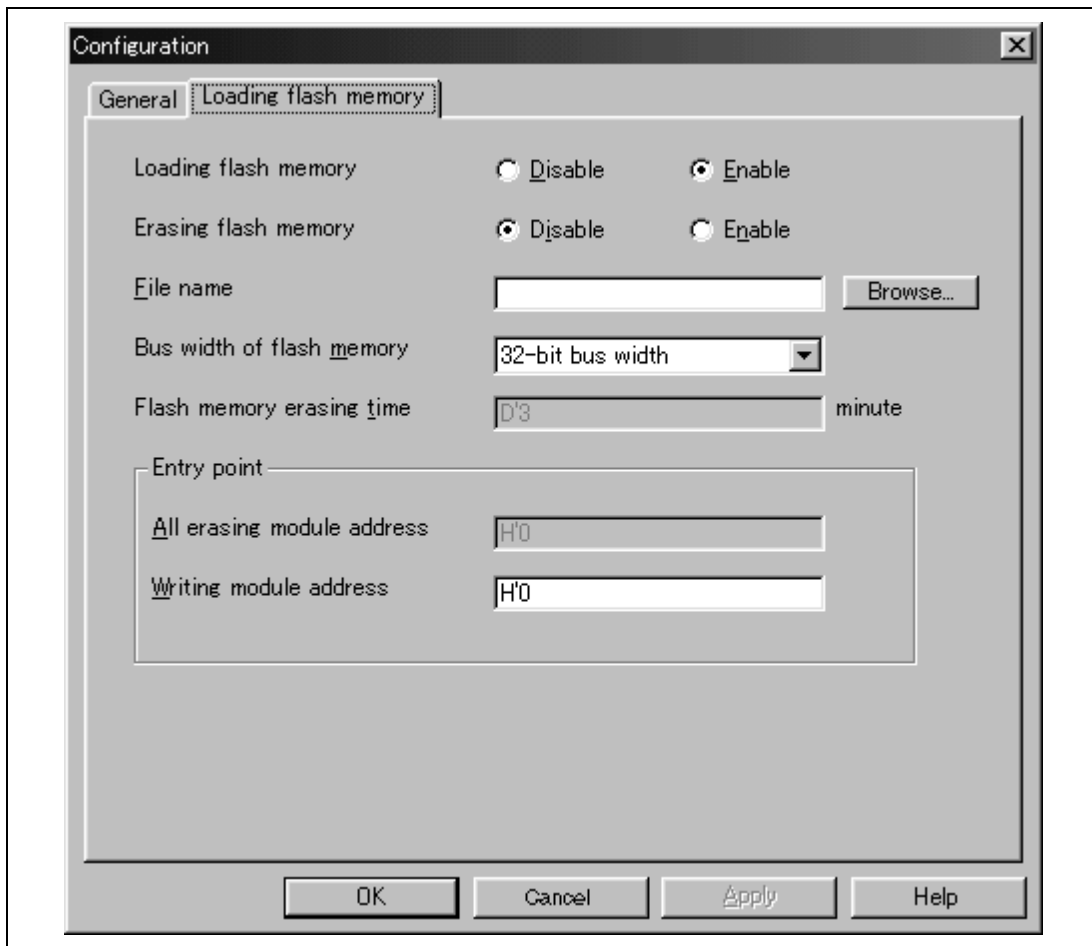
Figure 4.5   [Loading flash memory] Page ([Configuration] Dialog Box)

RENESAS

Description:

Table 4.5 [Loading flash memory] Page Options

Option	Description
[Loading flash memory] radio button	Sets Enable for flash memory downloading.
	At Enable, when [File load] is selected from the [File] menu for downloading, the write module is always called. Enable: Download to the flash memory Disable: Not download to the flash memory
[Erasing flash memory] radio button	Sets Enable for erasing before the flash memory is written.
	At Enable, the erase module is called before calling the write module. Enable: Erase the flash memory Disable: Not erase the flash memory
[File name] edit box	Sets the write/erase module name. The file that has been set is loaded to the RAM area before loading to the flash memory. A maximum of 128 characters can be input for the file name.
[Bus width of flash memory] list box	Sets the bus width of the flash memory.
[Flash memory erasing time] edit box*	Sets the TIMEOUT value at flash memory erasing. Increase the value if erasing requires much time; the default time is three minutes. The radix for the input value is decimal. It becomes hexadecimal by adding H'.
[Entry point] group box	Sets the calling destination address of the write/erase module.
	[All erasing module address] edit box: Inputs the calling destination address of the erase module. [Writing module address] edit box: Inputs the calling destination address of the write module.

Note: Although the values that can be set are D'0 to D'65535, the TIMEOUT hours may be extended according to the set value. Therefore, it is recommended to input the minimum value. The value to be input must only be positive integer.

Notes on using the flash memory download function

The following are notes on downloading to the flash memory.

- When the flash memory download is enabled, downloading other than to the flash memory area is disabled.
- Downloading is only enabled to the flash memory area. Perform memory write or software break only to the RAM area.
- When the flash memory erase is enabled, the [Stop] button cannot stop erasing.
- The entry area for the user module must be MMU-disabled space.

RENESAS

4.2.2 [Breakpoints] Window

Function:

This window lists all break conditions that have been set.

Window:

Figure 4.6 [Breakpoints] Window

RENESAS

Description:

The [Breakpoints] window displays breakpoint setting information. The items listed in the following tables are displayed.

Table 4.6   [Breakpoints] Window Display Items

Item	Description
[Enable]	Displays whether the break condition is enabled or disabled. BREAKPOINT: • Break Condition: o (If the address is the same as the one that has been set to the BREAKPOINT, the mark is •.)
[File/Line]	Displays the file name and line number where the breakpoint is set.
[Symbol]	Displays the symbol corresponding to the breakpoint address. If no symbol has been defined for the address, nothing is displayed.
[Address]	Displays the address where the breakpoint is set.
[Type]	Displays the break condition type as follows: Break Point: Software breakpoint (Virtual or physical address is determined according to the MMU state at setting.) Break Point Virtual Space ASID = D'xxx: Software breakpoint (Virtual address. ASID value is displayed in decimal.) Break Point Physical Space: Software breakpoint (Physical address.) Break Condition 1 to Break Condition 3: Hardware break condition

Note: Only "Break Point" is displayed in the [Type] item when the device does not support the MMU.

RENESAS

The pop-up menu, which is opened by clicking the right mouse button, can be used to set, change, and clear breakpoints, and to enable or disable break conditions. The pop-up menu functions are described in the following table.

Table 4.7   [Breakpoints] Window Pop-up Menu Operation

Menu Name	Description
[Add]	Sets break conditions. Clicking this button will display the [Break] dialog box, enabling break conditions to be set.
[Edit]	Changes break conditions. Select break conditions to be changed and click this button. The break condition setting dialog box will be displayed, enabling the break condition to be changed.
[Disable] ([Enable])	Enables or disables break conditions. Select break conditions to be enabled or disabled and click this button.
[Delete]	Clears break conditions. Select break conditions to be cleared and click this button.
[Del All]	Clears all break conditions.
[Go to Source]	Jumps to the address which sets the break in the [Source] window.

RENESAS

4.2.3　[Break] Dialog Box

Function:

This dialog box displays the break condition settings.

Window:
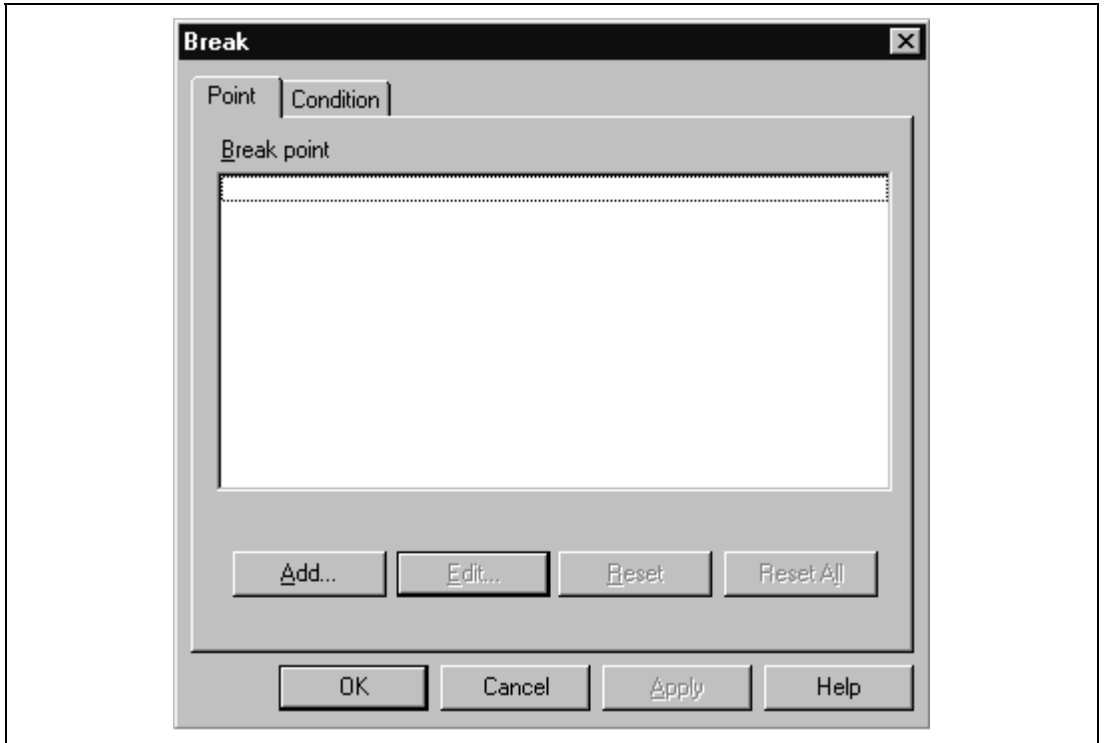
Figure 4.7　[Break] Dialog Box

Description:

The [Break] dialog box consists of the pages listed in table 4.8.

Table 4.8   [Break] Dialog Box Pages

Page Name	Description
[Point]	Displays software breakpoint settings.
[Condition]	Displays Break Condition settings.

The dialog boxes which set or modify break conditions can be displayed from the pages above.

Clicking the [OK] button (or [Close] button in some emulator products) will close this dialog box.

RENESAS

(1) [Point] Page ([Break] Dialog Box)

Function:

This page displays software breakpoint settings. In this page, software breakpoints can be set, changed, and cleared.

Window:
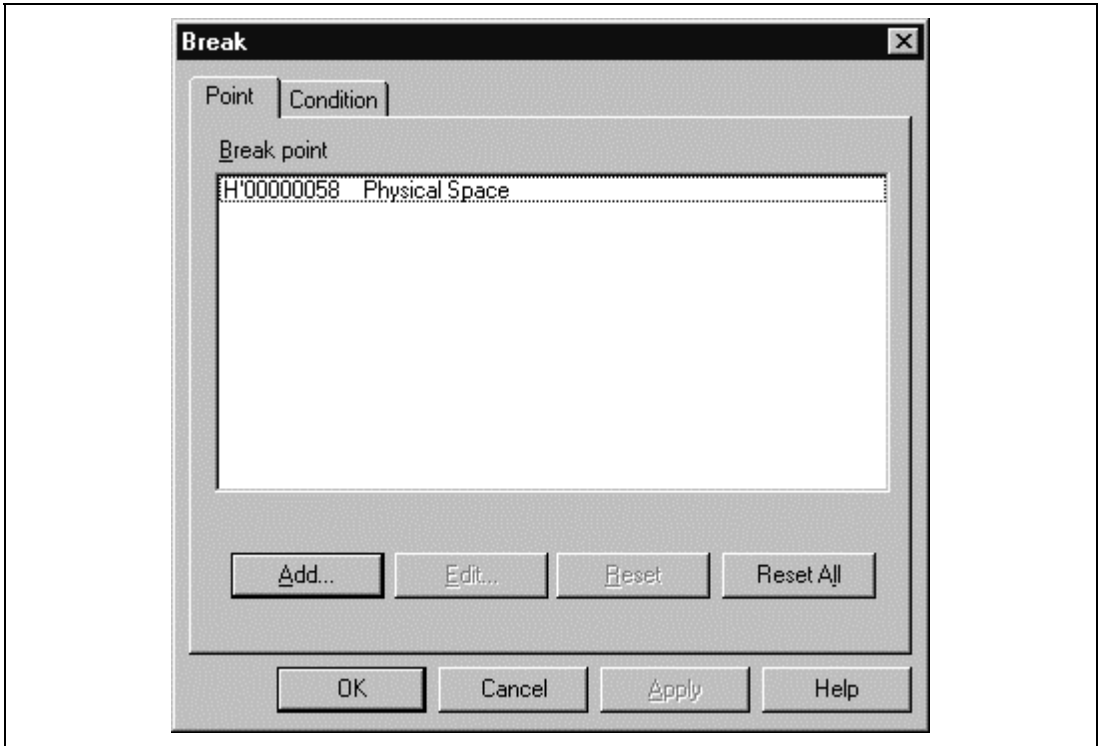
Figure 4.8   [Point] Page ([Break] Dialog Box)

Description:

Table 4.9　[Point] Page Options

Option	Description
[Break point] list box	Lists the software breakpoints currently being set. The display contents are <breakpoint address> and <address space>. <address space> is displayed as follows: • Physical Space • Virtual Space ASID = D'xxx (xxx is the ASID value displayed in decimal form.)
[Add...] button	Sets software breakpoints. Clicking this button displays the [Break Point] dialog box.
[Edit...] button	Changes the software breakpoint selected in the [Break point] list box. Clicking this button displays the [Break Point] dialog box.
[Reset] button	Clears the software breakpoint selected in the [Break Point] list box.
[Reset All] button	Clears all software breakpoints displayed in the [Break Point] list box.

Related Commands:

BREAKPOINT command
BREAKPOINT_CLEAR command
BREAKPOINT_ENABLE command
BREAKPOINT_DISPLAY command

RENESAS

(2) [Condition] Page ([Break] Dialog Box)

Function:

This page displays the Break Condition settings. These conditions can also be set or cleared in this page.

Window:
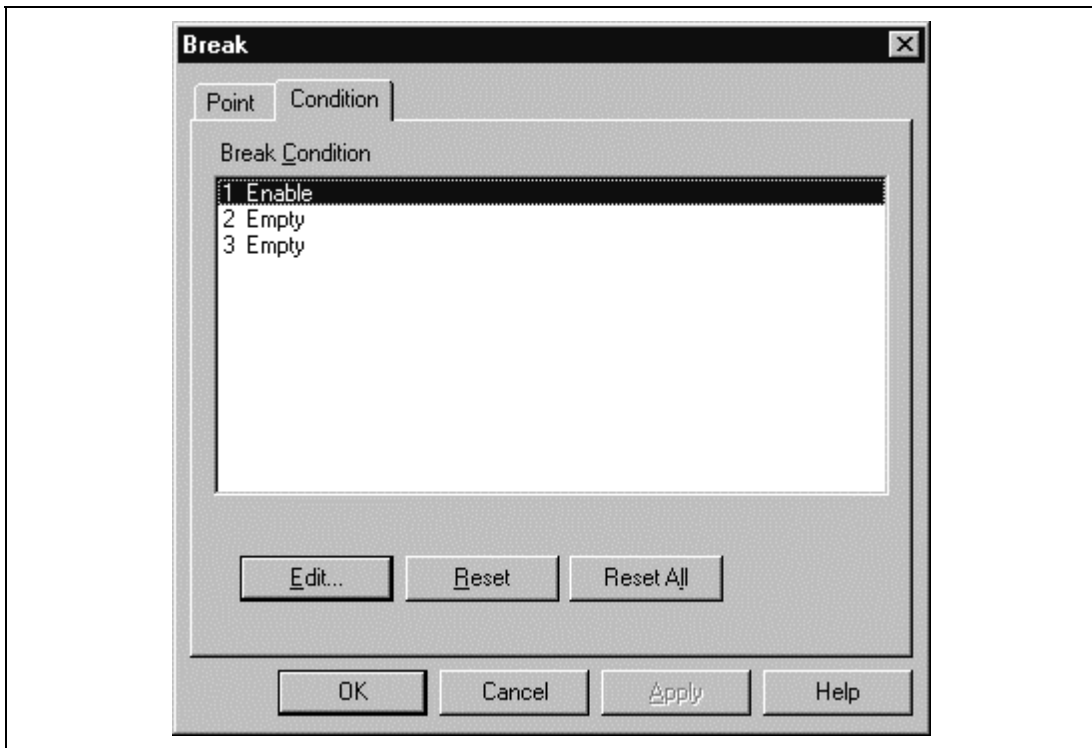
Figure 4.9   [Condition] Page ([Break] Dialog Box)

Note: The number of hardware break conditions differs according to the product. For the number that can be specified for each product, refer to the online help.

Description:

Table 4.10   [Condition] Page Options

Option	Description
[Break Condition] list box	Displays the Break Condition settings. The display at system initiation is as follows: When conditions are set, Enable is displayed. When no conditions are set, Empty is displayed. 1 Empty (setting of Break Condition 1) 2 Empty (setting of Break Condition 2) :
[Edit...] button	Changes the Break Condition settings selected in the [Break Condition] list box. Clicking this button displays the [Break Condition] dialog boxes.
[Reset] button	Clears the Break Condition settings selected in the [Break Condition] list box.
[Reset All] button	Clears all Break Condition settings in the [Break Condition] list box.

Related Commands:

BREAKCONDITION_CLEAR command
BREAKCONDITION_DISPLAY command
BREAKCONDITION_ENABLE command
BREAKCONDITION_SET command

RENESAS

4.2.4 [Break Point] Dialog Box

Function:

This dialog box sets software breakpoints.

Window:
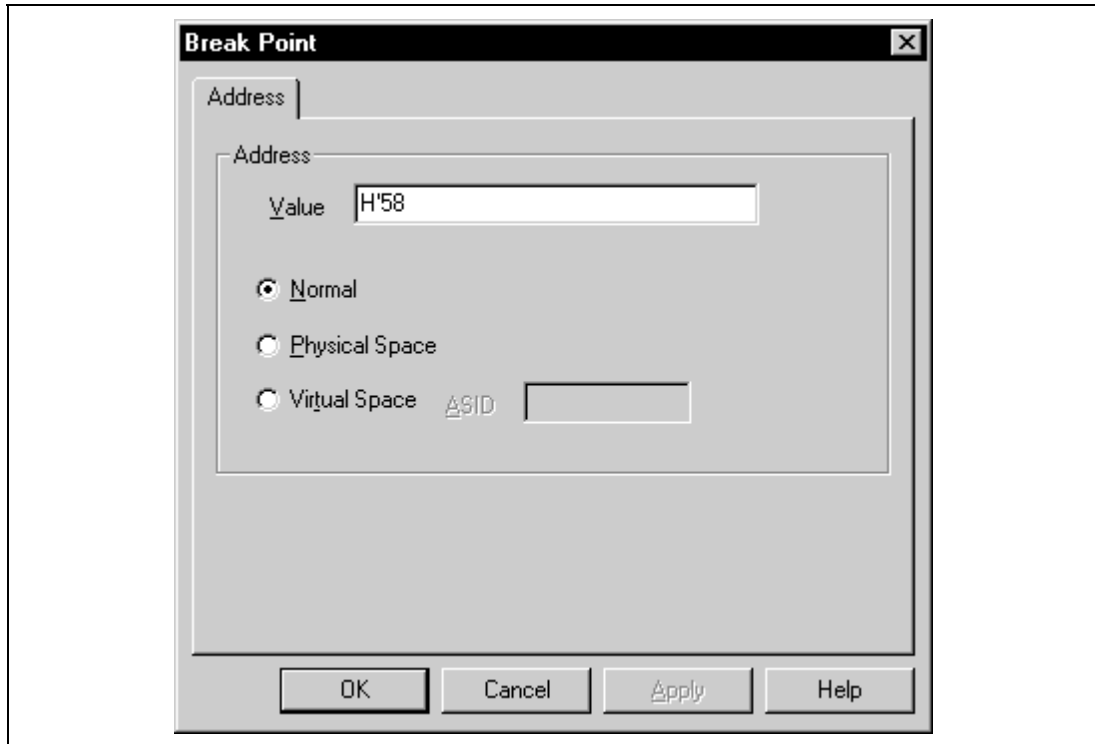
Figure 4.10 [Break Point] Dialog Box

Note: **The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.**

RENESAS

Description:

The [Break Point] dialog box consists only of the [Address] page. This dialog box sets address conditions and address areas. The [Address] page options are as follows:

Table 4.11   [Address] Page Options

Option	Description
[Value] edit box	Sets a breakpoint address with a number or a symbol.
[Normal] radio button	Does not set an address area.*
[Physical Space] radio button	Shows that the break condition is the physical area.*
[Virtual Space] radio button	Shows that the break condition is the virtual area.*
[ASID] edit box	Sets an ASID value (0 to 255) when the breakpoint address is in the virtual area. Nothing is set as default.*

Note: These options are not supported in a device in which the MMU is not built-in.

Clicking the [OK] button enables breakpoints to be set. If the [Cancel] button is clicked, this dialog box is closed without setting breakpoints.

Related Commands:

BREAKPOINT command
BREAKPOINT_CLEAR command
BREAKPOINT_DISPLAY command
BREAKPOINT_SET command

RENESAS

4.2.5 [Break Condition] Dialog Box

Function:

This dialog box sets hardware break conditions.

Window:
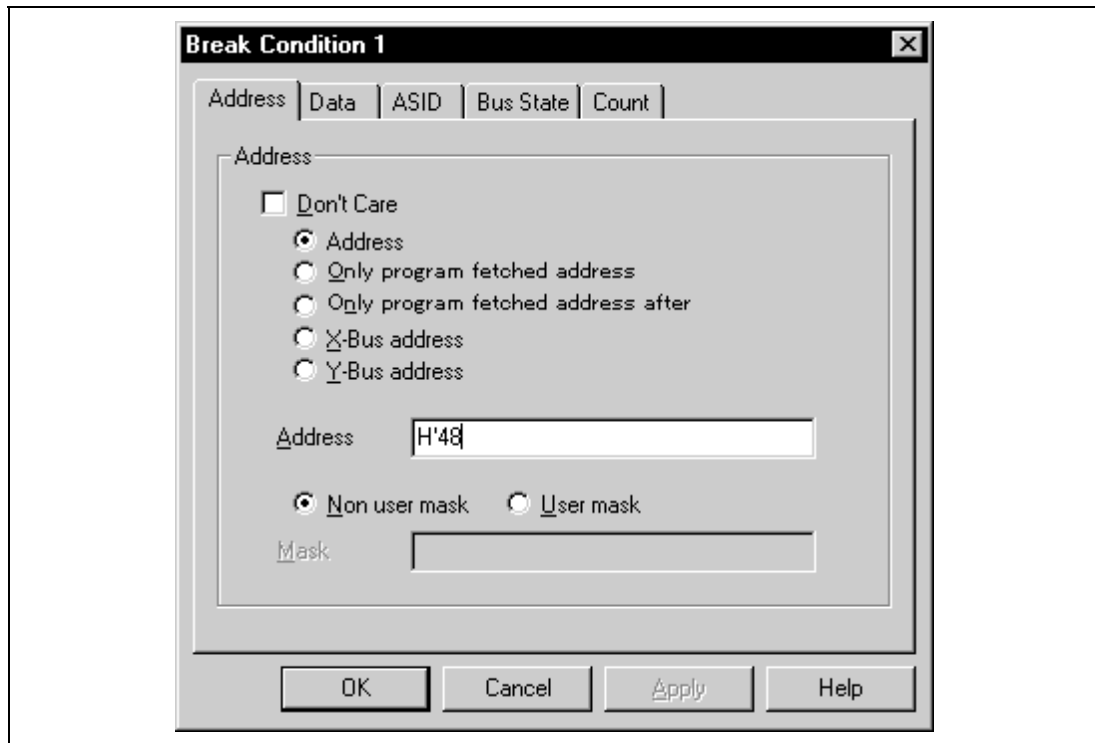
Figure 4.11 [Break Condition] Dialog Box

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

Description:

The [Break Condition] dialog box consists of multiple pages. A condition to halt the program can be set in each page.

Contents to be set by each page are described in section 4.2.6, [Break Condition] Dialog Box Pages.

Clicking the [OK] button sets the hardware break conditions. If the [Cancel] button is clicked, the dialog box is closed without setting the hardware break conditions.

Related Commands:

BREAKCONDITION_CLEAR command
BREAKCONDITION_DISPLAY command
BREAKCONDITION_ENABLE command
BREAKCONDITION_SET command

RENESAS

4.2.6 [Break Condition] Dialog Box Pages

Function:

The [Break Condition] dialog box pages allow a number of hardware break conditions to be set. Some functions may not be supported by some types of emulators. The setting conditions may differ from the dialog box name in table 4.12. For details, refer to section 6.5.2, Break Condition Functions.

Table 4.12 Setting Conditions in [Break Condition] Dialog Boxes

Dialog Box	Type				
	Address Bus Condition	Data Bus Condition	Bus State and Read/Write Conditions	Count Condition	LDTLB Instruction Break and Internal I/O Access Break Conditions
[Break Condition 1] dialog box	O	O	O	O	X
[Break Condition 2] dialog box	O	X	O	X	X
[Break Condition 3] dialog box	X	X	X	X	O

Note: O: Can be set by checking the radio button in the dialog box.
X: Cannot be set in the dialog box.

RENESAS

Table 4.13 shows all the [Break Condition] dialog box pages.

Table 4.13 [Break Condition] Dialog Box Pages

Page Name	Function
[Address]	Sets the address conditions of Break Condition 1 and Break Condition 2. (Address condition is not displayed in the [Break Condition 3] dialog box page.)
[Data]	Sets the data conditions of Break Condition 1. (Data condition is not displayed in the [Break Condition 2] and [Break Condition 3] dialog box pages.)
[ASID]	Sets the ASID conditions of Break Condition 1 and Break Condition 2. (ASID condition is not displayed in the [Break Condition 3] dialog box page.)
[Bus State]	Sets the bus state conditions and read/write cycle conditions of Break Condition 1 and Break Condition 2. (Bus state condition is not displayed in the [Break Condition 3] dialog box page.)
[Count]	Sets the satisfaction count conditions of Break Condition 1. (Count condition is not displayed in the [Break Condition 2] and [Break Condition 3] dialog box pages.)
[General]	Sets the conditions of Break Condition 3. (Data condition is not displayed in the [Break Condition 1] and [Break Condition 2] dialog box pages.)

Note: This function differs according to the product. For the specifications of each product, refer to section 6.5.2, Break Condition Functions, or to the online help.

RENESAS

(1) [Address] Page ([Break Condition] Dialog Box)

Function:

This page sets the address bus conditions.

Window:
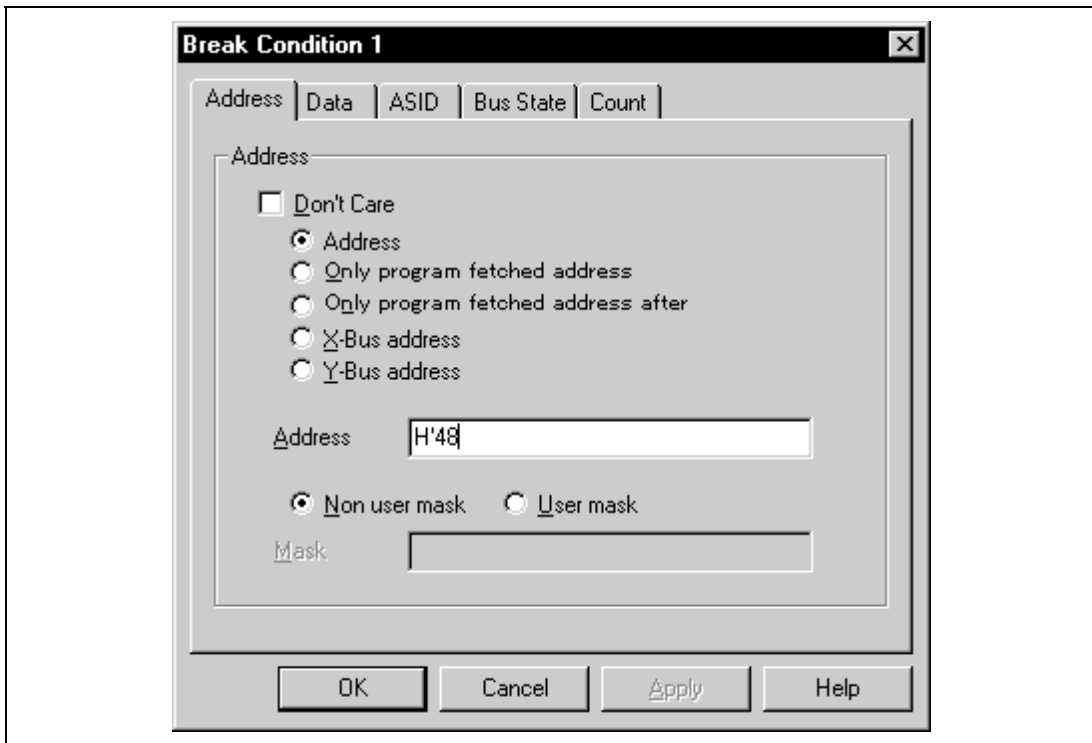
Figure 4.12   [Address] Page ([Break Condition 1] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

Description:

Table 4.14   [Address] Page Options

Option	Description
[Don't Care] check box	Does not set address conditions.
[Address] radio button	Sets use of the normal address bus as break conditions.
[Only program fetched address] radio button	Sets a break before prefetched address execution as break conditions.
[Only program fetched address after] radio button	Sets a break after prefetched address execution as break conditions.
[X-bus address] radio button	Sets the X-bus address as a break condition. Can be set only with Break Condition 1.
[Y-bus address] radio button	Sets the Y-bus address as a break condition. Can be set only with Break Condition 1.
[Address] edit box	Sets the address bus value with a number or a symbol.
[Non user mask] radio button	Sets no mask conditions.
[User mask] radio button	Sets mask conditions.
[Mask] edit box	Sets the mask bits if [User mask] is selected. For masked bits, the break condition is satisfied regardless of the address values.

Note: This page is displayed when the conditions of Break Condition 1 and Break Condition 2 are set.

A page name to be displayed and the contents of an option that can be set will change depending on the radio button selected.

Table 4.15   Address Options

Option	Description
[Address] radio button, [X-Bus address] radio button, and [Y-Bus address] radio button	All pages can be selected and masks can be set.
[Only program fetched address] radio button	The [Address] and [ASID] pages can be set; however, no mask can be set.
[Only program fetched address after] radio button	The [Address] and [ASID] pages can be set.

Note: This function differs according to the product. For the specifications of each product, refer to section 6.5.2, Break Condition Functions, or to the online help.

RENESAS

(2) [Data] Page ([Break Condition] Dialog Box)

Function:

This page sets the data bus conditions.

Window:
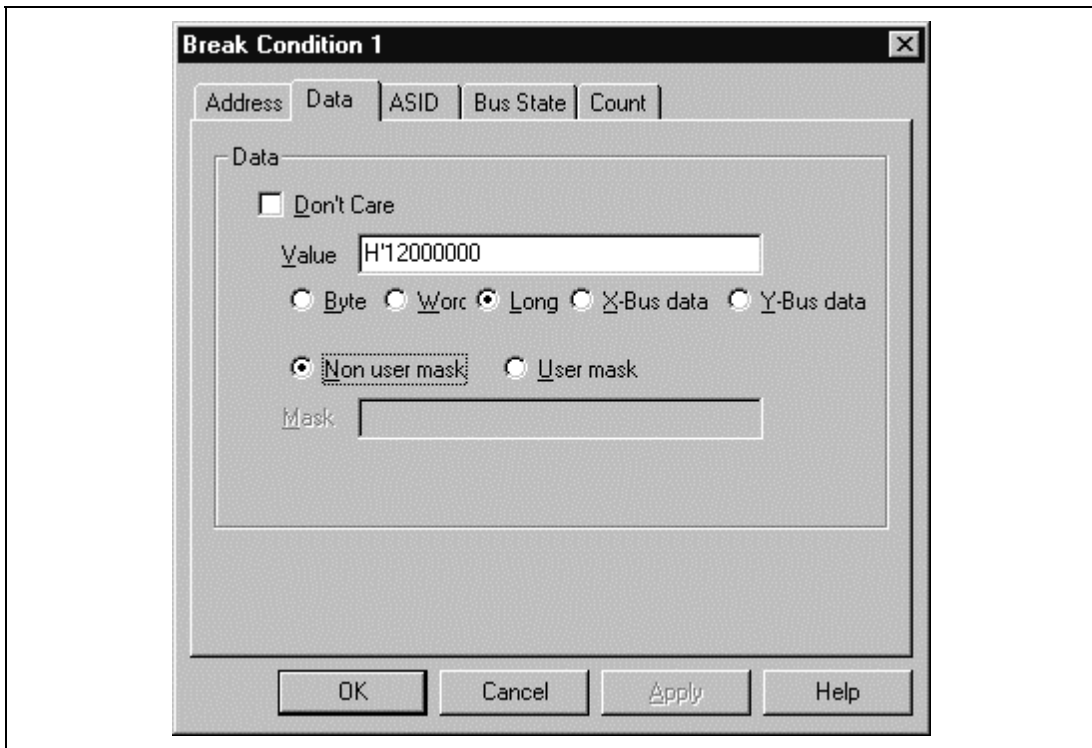
Figure 4.13   [Data] Page ([Break Condition 1] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

Description:

Table 4.16   [Data] Page Options

Option	Description
[Don't Care] check box	Does not set data conditions.
[Value] edit box	Sets the data bus value with a number.
[Byte] radio button	Sets byte data access cycles.
[Word] radio button	Sets word data access cycles.
[Long] radio button	Sets longword data access cycles.
[X-bus data] radio button	Sets X-bus data access cycles.
[Y-bus data] radio button	Sets Y-bus data access cycles.
[Non user mask] radio button	Does not set mask conditions.
[User mask] radio button	Sets mask conditions.
[Mask] edit box	Sets the mask bits when [User mask] is selected. Mark a bit to be masked with *. For masked bits, the break conditions will be satisfied regardless of the data values.

Note: This page is displayed when the conditions of Break Condition 1 are set.

RENESAS

(3) [ASID] Page ([Break Condition] Dialog Box)

Function:

This page sets the ASID conditions.

Window:
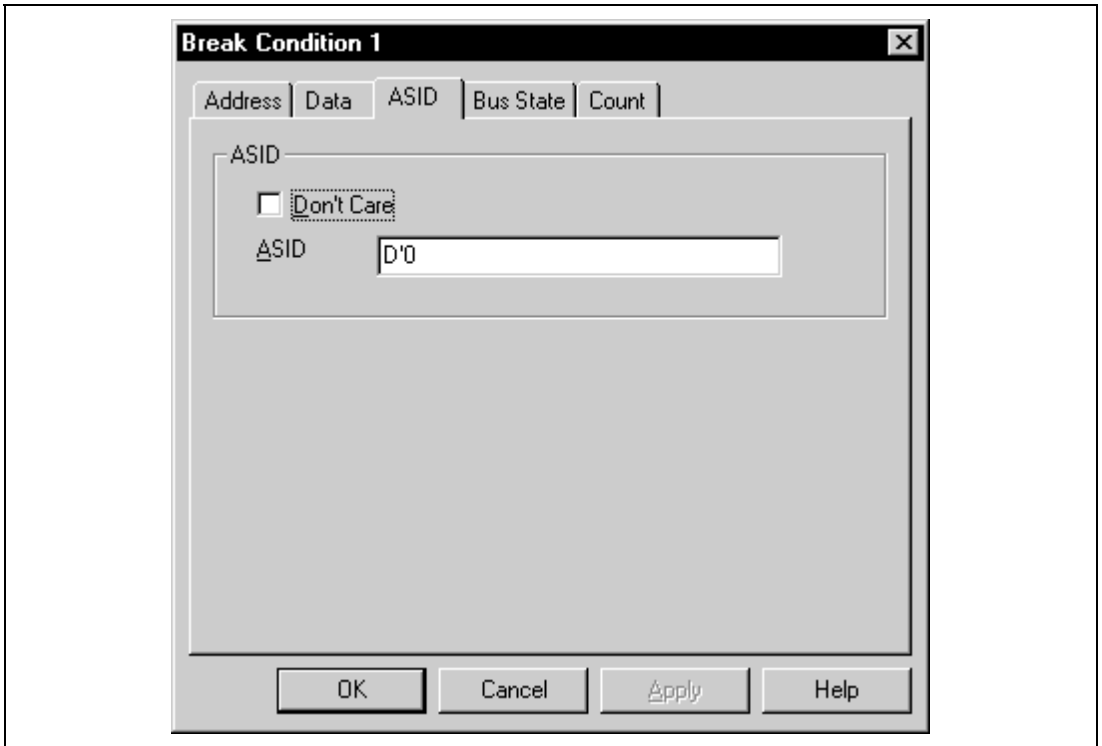
Figure 4.14   [ASID] Page ([Break Condition] Dialog Box)

Description:

Table 4.17   [ASID] Page Options

Option	Description
[Don't Care] check box	Does not set ASID conditions.
[ASID] edit box	Sets the ASID condition value. The default is 0.

Note: This page is displayed when the conditions of Break Condition 1 and Break Condition 2 are set.

Note: These options are not supported in a device in which the MMU is not built-in.

(4) [Bus State] Page ([Break Condition] Dialog Box)

Function:

This page sets bus state conditions and read/write cycle conditions.

Window:
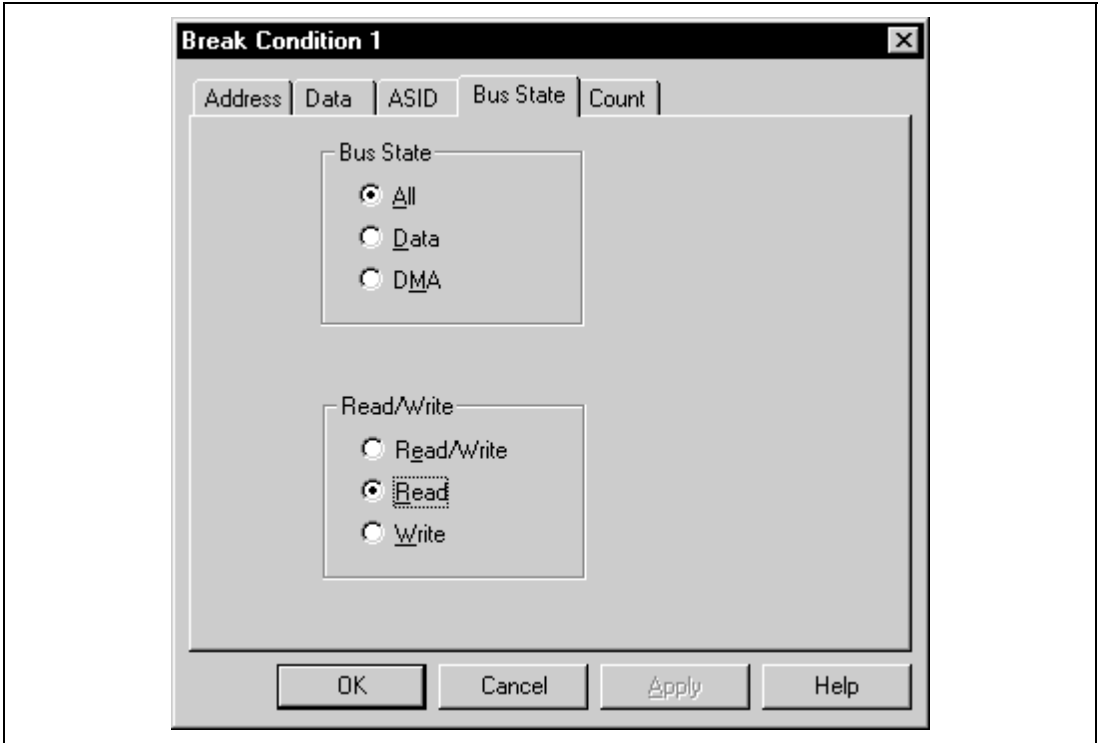
Figure 4.15   [Bus State] Page ([Break Condition] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the online help.

RENESAS

Description:

Table 4.18   [Bus State] Page Options

Group Box	Option	Description
[Bus State] group box	[All] radio button	Sets the bus state conditions as break conditions.
	[Data] radio button	Sets the execution cycle as break conditions.
	[DMA] radio button	Sets DMA cycles as a break condition.
[Read/Write] group box	[Read/Write] radio button	Sets the read/write cycle conditions as break conditions.
	[Read] radio button	Sets read cycles as break conditions.
	[Write] radio button	Sets write cycles as break conditions.

Note: This page is displayed when the conditions of Break Condition 1 and Break Condition 2 are set.

RENESAS

(5) [Count] Page ([Break Condition] Dialog Box)

Function:

This page sets the conditions for Break Condition 1.

Window:
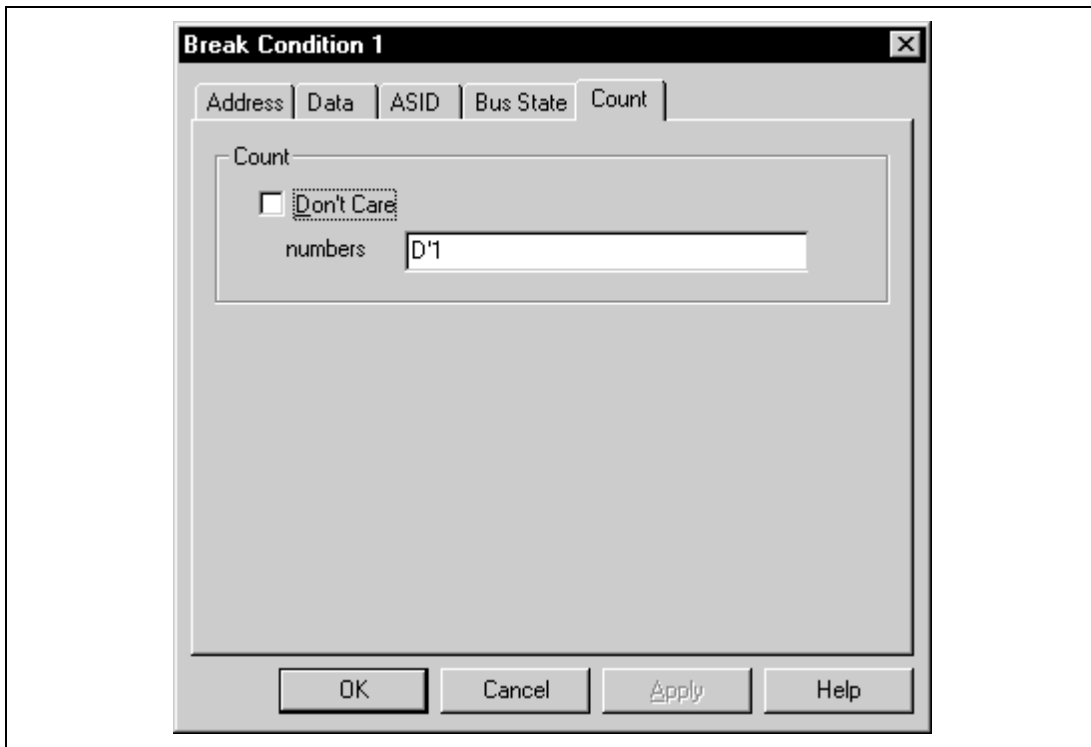
Figure 4.16   [Count] Page ([Break Condition] Dialog Box)

Table 4.19   [Count] Page Options

Option	Description
[Don't Care] check box	Sets no satisfaction count conditions.
Input area	Sets the satisfaction count as a break condition. The maximum count is 4,095. Breaks when the conditions set by the [Break Condition] dialog box for the specified times are satisfied. The default is D'1.

Note: Some products are not supported by this function. For the specifications of each product, refer to the online help.

RENESAS

(6) [General] Page ([Break Condition] Dialog Box)

Function:

This page sets the conditions for Break Condition 3.

Window:
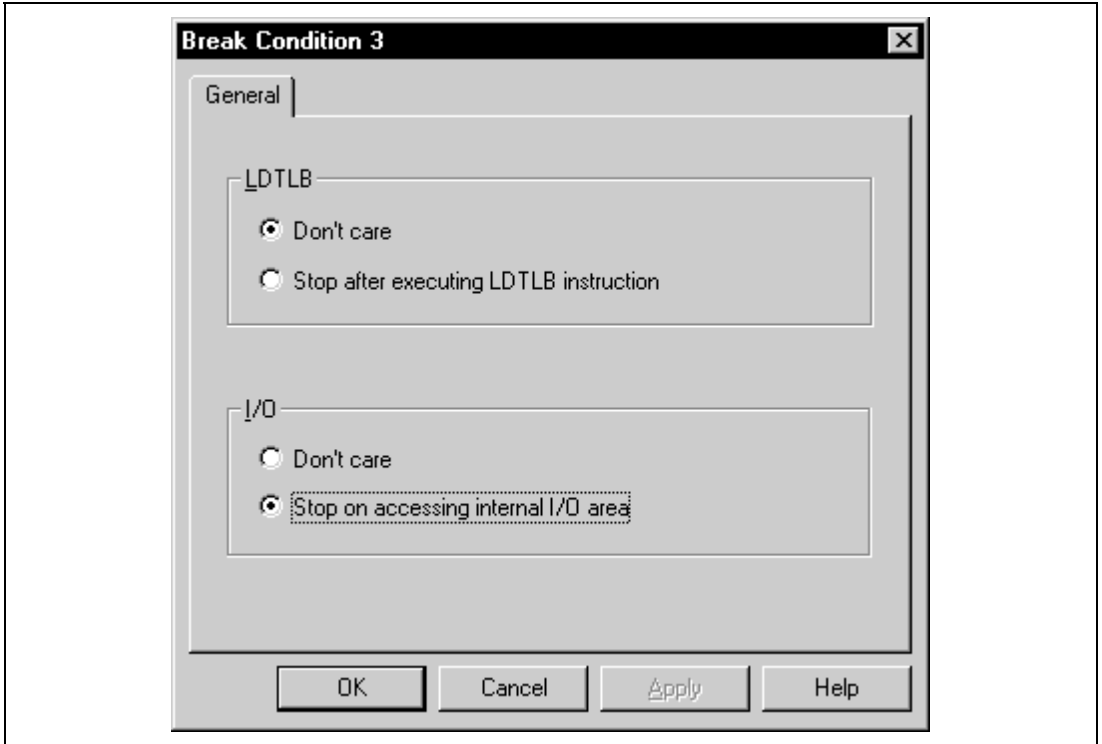
Figure 4.17   [General] Page ([Break Condition] Dialog Box)

RENESAS

Description:

Table 4.20　[General] Page Options

Group Box	Option	Description
[LDTLB] group box	[Don't Care] radio button	Does not set break conditions when the LDTLB instruction is executed.
	[Stop after executing LDTLB instruction] radio button	Sets the LDTLB instruction execution as break conditions.
	[DMA] radio button	Sets the DMA cycle as break conditions.
[I/O] group box	[Don't Care] radio button	Does not set break conditions when the internal I/O area is accessed.
	[Stop on accessing internal I/O area] radio button	Sets the internal I/O area access as break conditions.
	[Write] radio button	Sets only write cycle as break conditions.
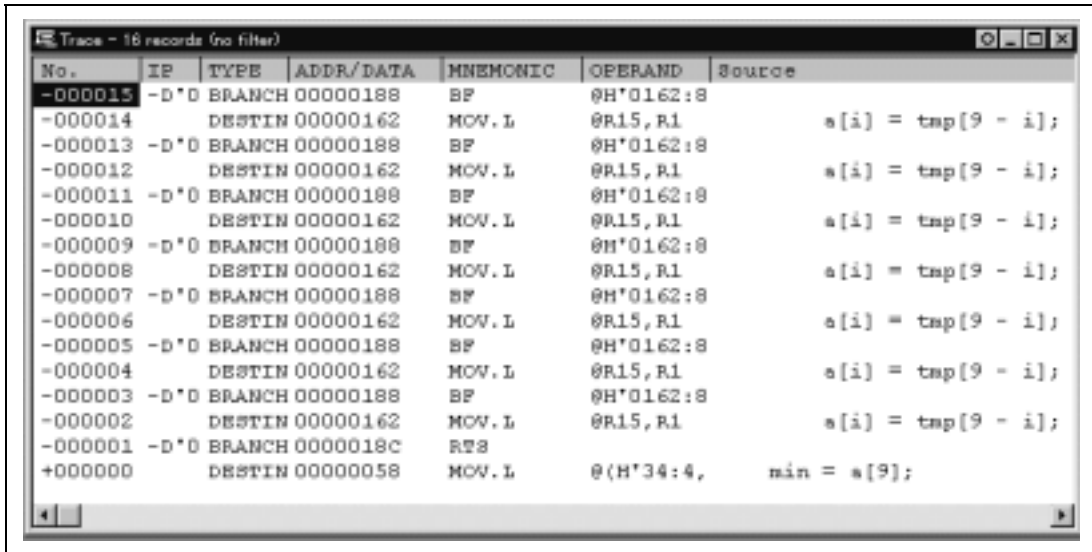
Note:　Some products are not supported by this function.　For the specifications of each product, refer to the online help.

RENESAS

4.2.7 [Trace] Window

Function:

This window displays the trace buffer contents.

Window:

Figure 4.18 [Trace] Window

Note: The types of information and the number of branch instructions differ according to the product. For the settings for each product, refer to the online help.

Description:

This window displays the trace buffer contents. The items listed in table 4.21 are displayed.

Table 4.21   [Trace] Window Display Items

Item	Description
[No.]	Displays the number in ascending order as the trace stop point is 0 (signed decimal).
[IP]	Displays the instruction pointer (signed decimal).
[TYPE]	For the branch instruction trace, displays the information type, that is, branch source or branch destination. BRANCH: Branch source DESTINATION: Branch destination
[ADDR/DATA]	For the branch instruction trace, displays the branch source or branch destination address.
[MNEMONIC]	Displays the execution instruction mnemonic.
[OPERAND]	Displays the execution instruction operand.
[Source]	Displays the C-source line of the address that the trace has been acquired.

The pop-up menu, opened by clicking the right mouse button, can be used to set, change, and clear trace conditions. For details, refer to the Hitachi Debugging Interface User's Manual.

Notes:

1. In some cases, the emulator address may be acquired by trace. In such a case, the following message will be displayed. Ignore this address because it is not a user program address.

        *** EML ***

2. The [Halt] menu in the pop-up menu is active only when the [Trace] window is open during user program execution. When the internal trace is used, realtime emulation cannot be performed by using the [Halt] menu.

Related Command:

TRACE_DISPLAY command

RENESAS

4.2.8   [Trace Acquisition] Dialog Box

Function:

This dialog box sets trace acquisition conditions. When the [Acquisition] menu is selected from
the pop-up menu, which is displayed by clicking the right mouse button in the [Trace] window,
the [Trace Acquisition] dialog box is displayed.

Table 4.22   [Trace Acquisition] Dialog Box Page Options

Page Name	Description
[Trace Mode]	Sets the conditions of trace mode.

(1) [Trace Mode] Page ([Trace Acquisition] Dialog Box)

Function:

This page sets the conditions for trace mode.

Window:
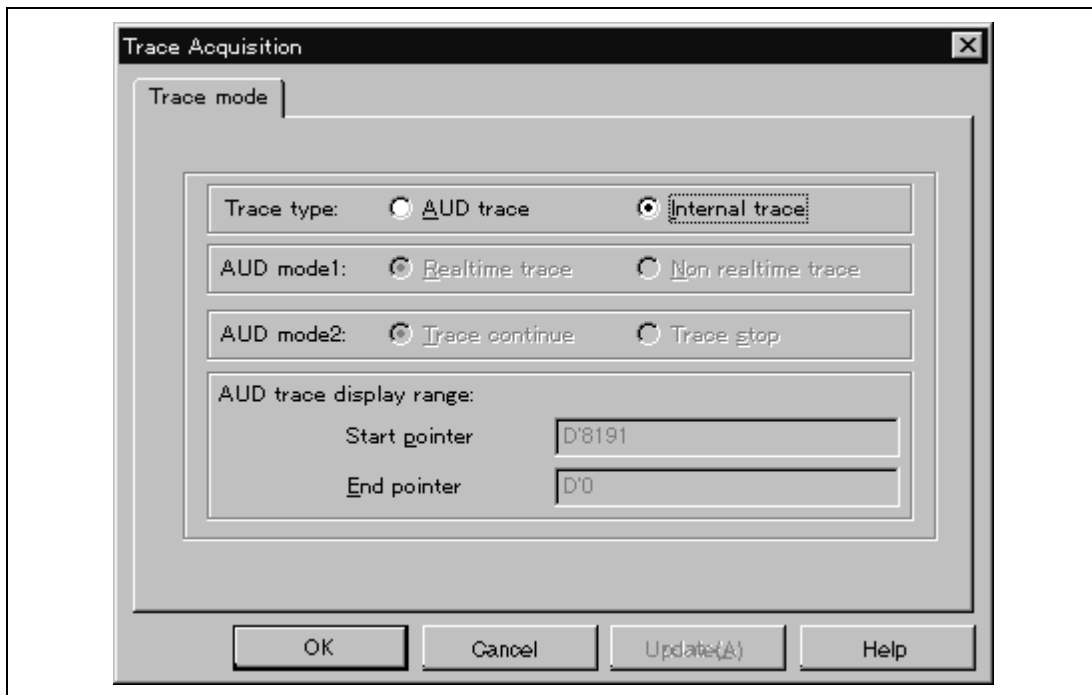
Figure 4.19   [Trace mode] Page ([Trace Acquisition] Dialog Box)

Note: This function differs according to the product. For the specifications of each product, refer to the section related to the trace functions in section 6, SHxxxx E10A Emulator Specifications, or to the online help.

RENESAS

Description:

Table 4.23   [Trace mode] Page Options

Option	Description
[AUD trace] radio button	Uses AUD trace functions. By default, this box is not checked.
[Internal trace] radio button	Uses the internal trace functions. By default, this box is checked.
[Realtime trace] radio button	When the next branch occurs while the trace information is being output, the information is stopped and the next trace information is output. The user program can be executed in realtime, but some trace information will not be output. By default, this box is checked.
[Non realtime trace] radio button	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. By default, this box is not checked.
[Trace continue] radio button	When the trace buffer becomes full, this function always overwrites the oldest trace information to acquire the latest trace information.
[Trace stop] radio button	When the trace buffer becomes full, the trace information is not acquired.
[AUD trace display range] group box	Inputs the start or end pointer value in the trace display range as numerical values. By default, the start pointer is –D'8191 and the end pointer is –D'0000. In the PCMCIA card emulator, –D'8191 to D'0 can be set to the trace pointer. In the PCI card emulator, –D'32767 to D'0 can be set.

Related Command:

AUD_MODE command

4.2.9 [System Status] Window

Function:

This window lists information, such as conditions that have been set to the emulator and execution results.

Window:
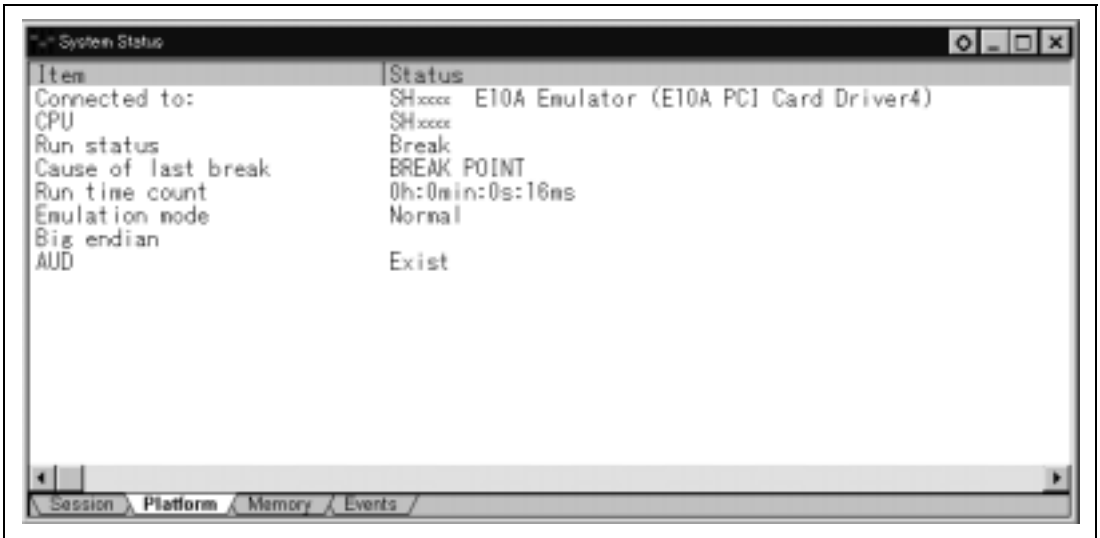
Figure 4.20 [System Status] Window

Note: The items that can be displayed in this window differ according to the product. For the items that can be displayed, refer to the online help.

The items listed in the following table are displayed in the [System Status] window.

Table 4.24　[System Status] Window Display Items

Page	Item	Description
[Session]	Target System	Always displays Connected.
	Session Name	Displays the session file name.
	Program Name	Displays the load module file name.
[Platform]	Connected To	Displays the name of the connected emulator and the selected driver name.
	CPU	Displays the target device name.
	Run status	Displays the execution status: RUNNING: Being executed Break: Stopped
	Cause of last break	Displays the cause of the emulator stopping at break. In this example, the cause of the stop is BREAK POINT.
	Run time count	Displays the program execution time. The display format is h: hours, min: minutes, s: seconds, and ms: milliseconds. In this example, 0h:0min:0s:16ms is displayed.
	Emulator mode	Displays the emulator operating mode (setting information for [Emulation Mode] of the [Configuration] dialog box).
	Big Endian/Little Endian	Displays the endian state (Big Endian or Little Endian). In this example, Big Endian is displayed.
	AUD	Displays whether the AUD function can be used. This item is displayed by the emulator with the AUD function.
[Memory]	Loaded Memory Areas	Displays the loaded area of the load module.
[Events]	Resources	Displays the usage states of BREAKPOINT and Break Condition.

RENESAS

RENESAS

Section 5 Command-line Functions

5.1 Table and Symbol Description

This section describes the format used in section 5.2, Command Descriptions. The descriptions of some commands are given over two or more pages.

5.1.1 Format

The input format for each command is as follows. Characters shown in bold-italics are to be input.

[ ] : Parameters enclosed by [ ] can be omitted.

< > : Contents shown in < > are set.

< >=: The parameter to the left of the "=" sign is input in the format shown to the right.

| : This represents a non-exclusive selection.

|| : This represents an exclusive selection.

The command parameter details are described in the parameter table.

5.1.2 Parameter Input

Numerical Parameters:

A binary, octal, decimal, or hexadecimal value, a symbol, or a formula can be input. A symbol can contain up to 32 characters. Terms in a formula are separated with operators (such as + or –).

Keyword Parameters:

One of the bold characters given in the description column of the table can be input. If a character string not shown in the description is input, an error will occur.

Character-String Parameters:

Character-string parameters are used to input mask data or a file name. In the mask data, set a radix (H': hexadecimal or B': binary) at the top of a character string and set * at the digit to be masked.

RENESAS

5.1.3 Examples

These are actual input examples. For commands whose execution results in a specific display output, an example of the display is given.

5.1.4 Related Items

Related E10A HDI commands (abbreviations) and dialog boxes are shown. (Refer to section 4, Descriptions of Windows.)

5.2   Command Descriptions

The command list of the E10A emulator is shown below.

Table 5.1   E10A HDI Commands

No.	Command	Abb.	Function
1	AUD_CLOCK	AUCL	Sets the AUD clock (AUDCK).
2	AUD_MODE	AUM	Sets AUD trace conditions.
3	AUD_TRACE	AUT	Displays trace information.
4	BREAKCONDITION_ CLEAR	BCC	Clears hardware breakpoints that have been set.
5	BREAKCONDITION_ DISPLAY	BCD	Displays hardware breakpoints that have been set.
6	BREAKCONDITION_ ENABLE	BCE	Enables or disables hardware breakpoints that have been set.
7	BREAKCONDITION_ SET	BCS	Sets hardware breakpoints.
8	BREAKPOINT	BP	Sets software breakpoints.
9	BREAKPOINT_CLEAR	BC	Clears software breakpoints that have been set.
10	BREAKPOINT_DISPLAY	BD	Displays software breakpoints that have been set.
11	BREAKPOINT_ENABLE	BE	Enables or disables software breakpoints that have been set.
12	DEVICE_TYPE	DE	Displays device type currently selected.
13	GO_OPTION	GP	Displays or sets the emulation mode during user program execution.
14	JTAG_CLOCK	JCK	Displays or sets a JTAG clock (TCK) frequency.
15	MEMORYAREA_SET	MAS	Displays or sets memory area at command input, such as load, verify, save, memory display, or memory change.

RENESAS

Table 5.1   E10A HDI Commands (cont)

No.	Command	Abb.	Function
16	REFRESH	RF	Updates the HDI memory information to the latest contents.
17	RESTART	RST	Restarts the emulator.
18	STATUS	STS	Displays emulator state information.
19	STEP_INTERRUPT	SI	Displays or sets the enable or disable status of interrupts during step execution.
20	TRACE_DISPLAY	TD	Displays acquired trace buffer information.
21	UBC_MODE	UM	Displays or sets UBC use states.
22	VPMAP_CLEAR	VC	Clears the emulator address translation (VP_MAP) table which has been set.
23	VPMAP_DISPLAY	VD	Displays the emulator address translation (VP_MAP) table.
24	VPMAP_ENABLE	VE	Enables or disables the emulator address translation (VP_MAP) table.
25	VPMAP_SET	VS	Sets emulator address translation (VP_MAP) table.

Note: **Support for these commands varies with the product. For the specifications of each product, refer to the online help.**

RENESAS

5.2.1 AUD_CLOCK:AUCL

Description:

Sets or displays the AUD clock (AUDCK) values that have been set.

Format:

```
aucl [<option>]
```

 <option> = <aud_clock>

Table 5.2   AUD_CLOCK Command Parameter

Parameter	Type	Description
<aud_clock>	Numerical value	Sets values from 1 to 7.
		1: 5 MHz (PCI), 7.5 MHz (PCMCIA)
		2: 10 MHz (PCI), 15 MHz (PCMCIA)
		3: 20 MHz (PCI), 30 MHz (PCMCIA)
		4: 30 MHz (PCI), 60 MHz (PCMCIA)
		5: 40 MHz (PCI)
		6: 50 MHz (PCI)
		7: 60 MHz (PCI)

Notes: 1. When <option> is omitted, the AUD clock (AUDCK) values that have been set are
          displayed.
       2. The range of frequencies that the AUD operates under differs according to the devices
          used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and
          AUD Clock (AUDCK).
       3. The AUD clock (AUDCK) value, which can be set with this command, may differ
          according to emulator products. For details, refer to section 6.5.4, Notes on Using the
          JTAG Clock (TCK) and AUD Clock (AUDCK).

**Note: The parameters for this command differ according to the product. For the
          specifications of each product, refer to the online help.**

RENESAS

Examples:

To set AUD clock (AUDCK) to 15 MHz:

>*AUD_CLOCK 2 (RET)*
AUD CLOCK = 15MHz

The AUD clock (AUDCK) is displayed:

>*AUD_CLOCK (RET)*
AUD CLOCK = 15MHz

Related Item:

[Configuration] dialog box

RENESAS

5.2.2 AUD_MODE:AUM

Description:

Sets or displays AUD trace acquisition conditions.

Format:

```
aum [<option1>] [<option2>]
```

<option1> = mode<mode>
<option2> = full<full>

Table 5.3 AUD_MODE Command Parameter

Parameter	Type	Description
<mode>	Keyword	Selects the trace mode.
		N: Internal trace
		F: Non realtime trace
		R: Realtime trace
<full>	Keyword	Continues or stops emulation when the trace memory is full.
		C: Always overwrites the oldest information to acquire the latest information.
		S: When the trace buffer memory is full, information acquisition stops.

Note: When <option1> and <option2> are omitted, the current setting conditions are displayed.

Note: The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

RENESAS

Examples:

To select realtime trace mode and set continue option:

>*aum mode R full c (RET)*

To display settings:

>*aum (RET)*
mode = Realtime trace, continue

To use internal trace mode:

>*aum (RET)*
mode = Internal trace

Related Item:

[Trace Acquisition] dialog box

RENESAS

5.2.3　AUD_TRACE:AUT

Description:

Displays the trace information.

Format:

```
aut [<option1>] [<option2>]
```

 <option1> = start<start_pointer>
 <option2> = end<end_pointer>

Table 5.4　AUD_TRACE Command Parameter

Parameter	Type	Description
<start_pointer>	Numerical value (-n)	Start pointer value for trace display.
<end_pointer>	Numerical value (-m)	End pointer value for trace display.

Notes: 1.　In the PCMCIA card emulator, –D'8191 to D'0 can be set to the trace pointer. In the PCI card emulator, –D'32767 to D'0 can be set.
　　　　2.　When the internal trace is selected, the AUT command displays the information that has been acquired by using the AUD function.

RENESAS

Example:

To display trace information according to the information acquired during user program execution:

```
>AUD_TRACE (RET)
IP             TYPE           ADDR          MNEMONIC     OPERAND
-D'xxxxxx      BRANCH         ******10
               DESTINATION    01000020      MOV.L        R1, @R1
(a)            (b)            (c)           (d)          (e)
```

(a) Instruction pointer (signed decimal)
(b) Types of branch source or branch destination
        BRANCH: Branch source
        DESTINATION: Branch destination
(c) Address of instruction word
(d) Instruction mnemonic
(e) Instruction operand

Related Item:

[Trace] dialog box

RENESAS

5.2.4 BREAKCONDITION_CLEAR: BCC

Description:

Clears hardware breakpoints that have been set.

Format:

```
bcc [<channel>]
```

 <channel> = channel <channel_number>

Table 5.5   BREAKCONDITION_CLEAR Command Parameter

Parameter	Type	Description
<channel number>	Numerical value	Hardware break channel number

Note: When <channel> is omitted, all hardware breakpoints that have been set are cleared.

Examples:

To clear all hardware breakpoints:

>*bcc (RET)*

To clear a hardware breakpoint set at channel 2:

>*bcc channel 2 (RET)*

Related Items:

BCD, BCE, and BCS commands
[Breakpoints] window
[Break] and [Break Condition] dialog boxes

RENESAS

5.2.5　BREAKCONDITION_DISPLAY: BCD

Description:

Displays hardware breakpoints that have been set. The display contents include a hardware breakpoint channel number, enable or disable of the setting, and setting conditions.

Format:

```
bcd [<channel>]
```

 <channel> = channel <channel_number>

Table 5.6　BREAKCONDITION_DISPLAY Command Parameter

Parameter	Type	Description
<channel_number>	Numerical value	Hardware breakpoint channel number

Note:　When <channel> is omitted, all hardware breakpoints that have been set are displayed.

Examples:

To display all hardware breakpoint settings:

>*bcd (RET)*

Break Condition 1:Enable data 20 long
Break Condition 2:Disable address 126
Break Condition 3:Disable LDTLB break

To display the hardware breakpoint set at channel 1:

>*bcd channel 1 (RET)*

Break Condition 1:Enable data 20 long

Note:　The items displayed with this command differ according to the product. For the display specifications of each product, refer to the online help.

Related Items:

BCC, BCE, and BCS commands
[Breakpoints] window
[Break] and [Break Condition] dialog boxes

RENESAS

5.2.6 BREAKCONDITION_ENABLE: BCE

Description:

Enables or disables hardware breakpoints that have been set.

Format:

```
bce [<channel>] <mode>
```

  `<channel> = channel <channel_number>`

Table 5.7　BREAKCONDITION_ENABLE Command Parameters

Parameter	Type	Description
<channel_number>	Numerical value	Hardware break channel number
<mode>	Keyword	Enables or disables hardware break settings.
		Set either of the following keywords: **enable**: Enables hardware break settings. **disable**: Disables hardware break settings.

Note:　When <channel> is omitted, all hardware breakpoints that have been set are enabled or disabled.

Examples:

To enable all hardware breakpoints:

     >***bce enable (RET)***

To disable the hardware breakpoints set at channel 1:

     >***bce channel 1 disable (RET)***

Related Items:

BCC, BCD, and BCS commands
[Breakpoints] window
[Break] and [Break Condition] dialog boxes

5.2.7 BREAKCONDITION_SET: BCS

Description:

Sets hardware breakpoints.

Note: The function will differ according to the devices used. For functions of each emulator product, refer to section 6.5.2, Break Condition Functions.

Format:

```
bcs <channel>  <option> [<option> ... ]
```

<channel>	=	channel <channel_number>
<option>	=	[<addropt> | <dataopt> | <asidopt> | <r/wopt> |
		<accessopt>] | [<countopt>] || [<ldtlbopt> | <ioopt>]
<addropt>	=	address <address> [<addrcycle>]|| address mask <maskdata>
		<addrcycle>
<dataopt>	=	data <data> <datawidth> ||
		data mask <maskdata> <datawidth>
<asidopt>	=	asid <asid>
<r/wopt>	=	direction <r/w>
<accessopt>	=	access <access>
<countopt>	=	count <count>
<ldtlbopt>	=	ldtlb <lbtlb>
<ioopt>	=	io <io>

RENESAS

Table 5.8   BREAKCONDITION_SET Command Parameters

Parameter	Type	Description
<channel_number>	Numerical value	Hardware break condition channel number
		Specifiable options change depending on the channel number. For details, refer to section 6.5.2, Break Condition Functions.
		1: <addropt>, <dataopt>, <asidopt>, <r/wopt>, and <accessopt> can be set.
		2: <addropt>, <asidopt>, <r/wopt>, and <accessopt> can be set.
		3: <ldtlbopt> and <ioopt> can be set.
<address>	Numerical value	Virtual address as an address bus value
<addrcycle>	Keyword	Address bus access conditions for program fetch cycles
		Set either of the following keywords: **pc**: Breaks before the address set by the <address> parameter is executed. When this keyword is set, only the <addropt> and <asidopt> cannot be set as conditions. In addition, when pc is set, the <maskdata> parameter cannot be set.
		pcafter: Breaks after the address set by the <address> parameter is executed. When this keyword is set, only the <addropt> and <asidopt> cannot be set as conditions. When pcafter is not set, the address bus during data access cycles and program fetch cycles is targeted as the access condition. **x:** X-Bus address bus access **y:** Y-Bus address bus access
<maskdata>	Character string	Mask specification for desired bits in the data
		Set a radix (**H'** for hexadecimal or **B'** for binary) at the top of a character string and set * in the bit to the masked. Conditions are satisfied regardless of the values of masked bits.
<data>	Numerical value	Data bus value

RENESAS

Table 5.8   BREAKCONDITION_SET Command Parameters (cont)

Parameter	Type	Description
<datawidth>	Keyword	Data bus access conditions
		Set one of the following keywords:
		byte: byte access
		word: word access
		long: longword access
		x: X-Bus data bus access
		y: Y-Bus data bus access
<asid>	Numerical value	ASID value from 0 to H'FF.
<r/w>	Keyword	Bus cycle read/write conditions
		Set either of the following keywords:
		read: read cycles
		write: write cycles
<access>	Keyword	Bus cycle access type
		dat: execution cycles
<count>	Numerical value	Set satisfaction count from 1 to H'FFFF
<ldtlb>	Keyword	Set LDTLB instruction execution as a break condition
		break: Breaks when the LDTLB instruction is executed.
<io>	Keyword	Set internal I/O access condition as a break condition.
		break: Breaks when the internal I/O area is accessed.

Note: The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

RENESAS

Examples:

To set the following conditions for channel 1 hardware breakpoint:

> <addropt> item: An address bus value of H'1000000,
> <dataopt> item: D0 bit of the byte access data is 0,
> <r/wopt> item: write cycle.

> **>bcs channel 1 address H'1000000 data mask B'*******0 byte**
> **direction write (RET)**

To set the following conditions for channel 2 hardware breakpoint:

> <addropt> item: Sets an address bus value of H'1000000 during the program fetch cycles,
> and breaks before execution,
> <asidopt> item: The ASID value is H'0.

> **>bcs channel 2 address H'1000000 pc asid H'0 (RET)**

To set the following conditions for channel 1 hardware breakpoint:

> <addropt> item: Sets an address bus value of H'1000000 during the program fetch cycles
> with a mask set to the lower 10 bits, and breaks after execution,
> <asidopt> item: H'10 to the ASID value.

> **>bcs channel 1 address H'1000000 pcafter m1 asid H'10 (RET)**

To set the following conditions for channel 2 hardware breakpoint:

> <accessopt> item: Execution cycles,
> <r/wopt> item: Read cycles.

> **>bcs channel 2 access dat direction read (RET)**

To set the following conditions for channel 3 hardware breakpoint:

> <ldtlbopt> item: Breaks during LDTLB instruction execution,
> <ioopt> item: Breaks when the internal I/O area is accessed.

> **>bcs channel 3 ldtlb break io (RET)**

Related Items:

BCC, BCD, BCE, and TM commands
[Breakpoints] window
[Break] and [Break Condition] dialog boxes

RENESAS

5.2.8   BREAKPOINT: BP

Description:

Sets software breakpoints.

Note: The function will differ according to the devices used.

Format:

```
bp <address> [<address_space> [<asidopt>]]
```

  <address_space> = space <space>
  <asidopt> = asid <asid>

Table 5.9   BREAKPOINT Command Parameters

Parameter	Type	Description
<address>	Numerical value	Breakpoint address
		When an odd address is set, the address is rounded down to an even address.
<space>	Keyword	Breakpoint address area
		Set either of the following keywords: **physical**: physical address **virtual**: virtual address
<asid>	Numerical value	ASID value of a breakpoint when virtual is set to the <space> parameter.

Note: When virtual is set and the <asidopt> item is omitted in the <address_space> item, a
       breakpoint is set to a virtual address corresponding to the ASID value at command input.

**Note: The parameters for this command differ according to the product. For the
       specifications of each product, refer to the online help.**

RENESAS

Examples:

To set a software breakpoint at physical address H'10002C8:

> >*bp H'10002C8 space physical (RET)*

To set a software breakpoint at logical address H'1000000, whose ASID value is H'10:

> >*bp H'1000000 space virtual asid H'10 (RET)*

Related Items:

BC, BD, BE, VC, VD, VE, and VS commands
[Breakpoints] window
[Break] dialog box

RENESAS

5.2.9　BREAKPOINT_CLEAR: BC

Description:

Clears software breakpoints that have been set.

Format:

```
bc [<address> [<address_space> [<asidopt>]]]
```

　<address_space> = space <space>
　<asidopt> = asid <asid>

Table 5.10　BREAKPOINT_CLEAR Command Parameters

Parameter	Type	Description
<address>	Numerical value	Breakpoint address
<space>	Keyword	Address area of a breakpoint
		Set either of the following keywords: **physical**: physical address **virtual**: virtual address
<asid>	Numerical value	ASID value of a breakpoint when virtual is set to the <space> parameter.

Notes: 1. When no parameters are set, all software breakpoints are cleared.
2. When <address_space> and <asidopt> are not set, all software breakpoints that match the specified address are cleared.

Note:　The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

RENESAS

Examples:

To clear all breakpoints:

>*bc (RET)*

To clear all software breakpoints whose address value is H'1000000:

>*bc H'1000000 (RET)*

To clear a software breakpoint whose virtual address is H'1000000, according to the ASID value at command input:

>*bc H'1000000 space virtual (RET)*

To clear the software breakpoint at virtual address H'1000000, whose ASID value is H'10:

>*bc H'1000000 space virtual asid H'10 (RET)*

Related Items:

BP, BD, BE, VC, VD, VE, and VS commands
[Breakpoints] window
[Break] dialog box

5.2.10   BREAKPOINT_DISPLAY: BD

Description:

Displays software breakpoints that have been set.

Format:

bd

Table 5.11   BREAKPOINT_DISPLAY Command Parameter

Parameter	Type	Description
None		

Example:

To display the software breakpoints that have been set:

>*bd (RET)*

```
  H'00000110 physical enable
  H'0000011c virtual asid H'0 disable
  H'00000250 physical enable
```

**Note: The items displayed with this command differ according to the product. For the
display specifications of each product, refer to the online help.**

Related Items:

BP, BC, and BE commands
[Breakpoints] window
[Break] dialog box

RENESAS

5.2.11 BREAKPOINT_ENABLE: BE

Description:

Enables or disables software breakpoints that have been set.

Format:

```
be <address> <address_space> <asidopt> <mode>
```

 <address_space> = space <space>
 <asidopt> = asid <asid>

Table 5.12 BREAKPOINT_ENABLE Command Parameters

Parameter	Type	Description
<address>	Numerical value	Breakpoint address
<space>	Keyword	Address area
		Set either of the following keywords: **physical**: physical address **virtual**: virtual address
<asid>	Numerical value	ASID value of a breakpoint when virtual is set to the <space> parameter.
<mode>	Keyword	Enables or disables breakpoints.
		Set either of the following keywords: **enable**: Enables breakpoints. **disable**: Disables breakpoints.

Examples:

To enable a software breakpoint at physical address H'1002:

>*be H'1002 space physical enable (RET)*

To enable a software breakpoint at logical address H'1000000, whose ASID value is H'10:

>*be H'1000000 space virtual asid H'10 enable (RET)*

Notes: 1. The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

2. The items displayed with this command differ according to the product. For the display specifications of each product, refer to the online help.

RENESAS

Related Items:

BC, BD, BP, VC, VD, VE, and VS commands
[Breakpoints] window
[Break] dialog box

RENESAS

5.2.12 DEVICE_TYPE: DE

Description:

Displays the currently selected device.

Format:

```
de
```

Table 5.13   DEVICE_TYPE Command Parameter

Parameter	Type	Description
None		

Example:

To display the currently selected device:

>*de (RET)*

```
 Current device = SHxxxx
```

5.2.13　GO_OPTION: GP

Description:

Displays or sets the emulation mode.

Format:

Displays emulation mode.

```
gp
```

Sets emulation mode.

```
gp <eml_opt>
```
<eml_opt> = eml_mode <eml_mode>

Table 5.14　GO_OPTION Command Parameter

Parameter	Type	Description
<eml_mode>	Keyword	Specifies the emulation mode.
		normal: Normal execution
		sequence1: Stops the user program only when the conditions are satisfied in the order of hardware breakpoints 2 to 1. Hardware breakpoints 1 and 2 must be set.
		no_break: Makes software breakpoints and hardware breakpoints temporarily invalid and executes the user program.

Notes:　1.　The sequential break function differs according to emulator products. For details, refer to online help.
　　　　2.　The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

RENESAS

Examples:

To display the currently set emulation mode for user program execution:

>*gp (RET)*

    Emulator execution mode = Sequential break Condition 2-1

To set the normal emulation mode for user program execution:

>*gp eml_mode normal (RET)*

Note: The items displayed with this command differ according to the product. For the display specifications of each product, refer to the online help.

Related Items:

BCS and BS commands,
[Breakpoints] window,
[Break], [Break Condition], and [Configuration] dialog boxes

RENESAS

5.2.14　JTAG_CLOCK: JCK

Description:

Displays or sets the JTAG clock (TCK) frequency.

Format:

Displays the JTAG clock (TCK) frequency.

```
jck
```

Sets the JTAG clock (TCK) frequency.

```
jck <jck_opt>
```

Table 5.15　JTAG_CLOCK Command Parameter

Parameter	Type	Description
<jck_opt>	Numerical value	Sets one of the JTAG clock (TCK) frequency.
		(PCMCIA used: 3.75 MHz, 7.5 MHz, or 15 MHz) **3**: 3.75 MHz **7**: 7.5 MHz **15**: 15 MHz
		(PCI used: 4.125 MHz, 8.25 MHz, or 16.5 MHz) **4**: 4.125 MHz **8**: 8.25 MHz **16**: 16.5 MHz

Note: The range of frequencies that the Hitachi-UDI operates at differs according to the devices used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).

RENESAS

Examples:

(when PCMCIA used):

To set the JTAG clock (TCK) frequency:

>*jck 15 (RET)*

```
JTAG Clock  15MHz
```

To display the JTAG clock (TCK) frequency:

>*jck (RET)*

```
JTAG Clock  15MHz
```

(when PCI used):

To set the JTAG clock (TCK) frequency:

>*jck 16 (RET)*

```
JTAG Clock  16.5MHz
```

To display the JTAG clock (TCK) frequency:

>*jck (RET)*

```
JTAG Clock  16.5MHz
```

RENESAS

5.2.15   MEMORYAREA_SET: MAS

Description:

Displays and sets memory area at command input, such as load, verify, save, memory display, or memory change.

Format:

Displays memory area.

mas

Sets memory area.

mas \<memory_area\> [\<asidopt\>]
\<asidopt\> = **asid** \<asid\>

Table 5.16   MEMORYAREA_SET Command Parameters

Parameter	Type	Description
\<memory_area\>	Keyword	Sets memory area.
		normal: Does not set memory area. **physical**: Sets physical address area. **virtual**: Sets virtual address area.
\<asid\>	Numerical value	Sets an ASID value from 0 to H'FF when virtual is set to the \<memory_area\> parameter.

Notes: 1. When virtual is set and \<asid\> is omitted in \<memory_area\>, a virtual address corresponding to the ASID value at command input is accessed.
2. When a memory is accessed, the contents in the instruction cache are disabled after this command is executed.

RENESAS

Examples:

To display a memory area for command input, such as load, verify, save, memory display, and memory change:

>*mas (RET)*
memoryarea_set virtual asid H'10

To set a memory area for command input, such as load, verify, save, memory display, and memory change, to a physical address area:

>*mas physical (RET)*

To set a memory area for command input, such as load, verify, save, memory display, and memory change, to a virtual address area whose ASID value is H'10:

>*mas virtual asid H'10 (RET)*

5.2.16　REFRESH: RF

Description:

Updates the HDI memory information.

Format:

```
rf
```

Table 5.17　REFRESH Command Parameter

Parameter	Type	Description
None		

Example:

To update the HDI memory information:

>*rf (RET)*

RENESAS

5.2.17 RESTART: RST

Description:

Restarts the emulator. The settings of breakpoints or trace acquisition conditions are not reset here.

Format:

```
rst
```

Table 5.18   RESTART Command Parameter

Parameter	Type	Description
None		

Example:

To restart the emulator:

>*rst (RET)*

5.2.18 STATUS: STS

Description:

Displays status information of the emulator.

Format:

```
sts
```

Table 5.19 STATUS Command Parameter

Parameter	Type	Description
None		

Example:

To display status information of the emulator:

```
>sts (RET)
Emulator Status
Connected to:      SHxxxx E10A Emulator (E10A PC Card Driver)
CPU                SHxxxx
Run status         Break
Cause of last break  BREAK POINT
Run time count     0h:0min:0s:10ms
Emulator mode      Normal
Big endian
AUD                Exist
```

Note: The items displayed with this command differ according to the product. For the display specifications of each product, refer to the online help.

RENESAS

5.2.19   STEP_INTERRUPT: SI

Description:

Sets or displays the enable or disable status of interrupts during step execution. If enabled, interrupts occur and stop at the top address in the interrupt routine.

Format:

Displays the enable or disable status of interrupts during step execution.

si

Sets the enable or disable status of interrupts during step execution.

si <mode>

Table 5.20   STEP_INTERRUPT Command Parameter

Parameter	Type	Description
<mode>	Keyword	Enables or disables interrupts during step execution.
		Set either of the following: **enable**: Enables interrupts. **disable**: Disables interrupts.

Example:

To enable interrupts during step execution:

> *si enable (RET)*

To display interrupt status during step execution:

> *>si (RET)*
> Emulator step interrupt mode = ENABLE

RENESAS

5.2.20   TRACE_DISPLAY: TD

Description:

Displays the acquired trace information. The information to be acquired is the branch source and branch destination addresses when a branch is made during the user program execution.

Format:

```
td
```

Table 5.21   TRACE_DISPLAY Command Parameter

Parameter	Type	Description
None		

Notes:

1. In some cases, the emulator address may be acquired. In such a case, the following message will be displayed at the place where the mnemonic or operand is displayed. Ignore this address because it is not a user program address.

    *** EML ***

2. If a TLB error occurs while acquired trace information is displayed, the following error message will be displayed:
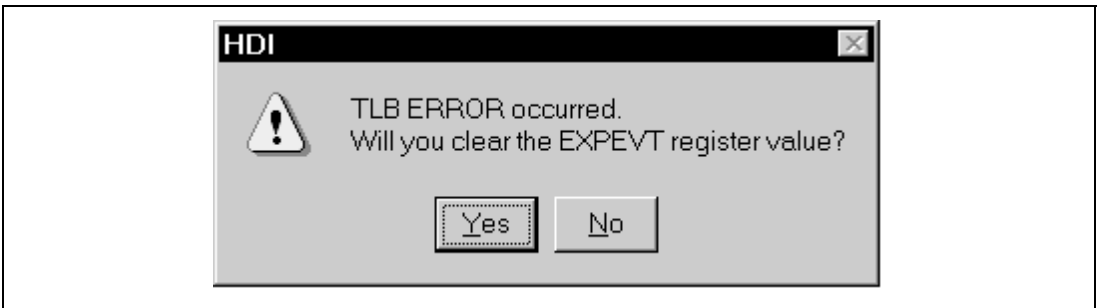
Figure 5.1   TLB Error Message Dialog

RENESAS

Example:

To display trace information according to information acquired during user program execution:

```
>td (RET)
IP              TYPE            ADDR            MNEMONIC        OPERAND
-D'xxxxxx       BRANCH          01000010        JSR             @R0
                DESTINATION     01000020        MOV.L           R1, @R1
(a)             (b)             (c)             (d)             (e)
```

(a) Instruction pointer (signed decimal)
(b) Types of branch source or branch destination
       BRANCH: Branch source
       DESTINATION: Branch destination
(c) Address of instruction word
(d) Instruction mnemonic
(e) Instruction operand

Related Items:

TM command
[Trace] window
[Trace Acquisition] dialog box

5.2.21 UBC_MODE:UM

Description:

Sets or displays the current UBC state.

Format:

```
um [<ubc_mode>]
```

Table 5.22 UBC_MODE Command Parameter

Parameter	Type	Description
<ubc_mode>	Keyword	Selects the UBC mode.
		EML: Uses the UBC as Break Condition by the emulator.
		USER: Releases the UBC to the user. (Break Condition cannot be used.)

Note: When <option> is omitted, the current setting conditions are displayed.

Note: The parameters for this command differ according to the product. For the specifications of each product, refer to the online help.

Examples:

To release the UBC to the user:

>*UBC_mode user (RET)*
UBC_mode = USER

To display the current UBC state:

>*UBC_mode (RET)*
UBC_mode = EML

Note: The items displayed with this command differ according to the product. For the display specifications of each product, refer to the online help.

Related Item:

[Configuration] dialog box

RENESAS

5.2.22 VPMAP_CLEAR: VC

Description:

Clears the address translation (VP_MAP) table that is set in the emulator.

Format:

```
vc [<address>]
```

Table 5.23 VPMAP_CLEAR Command Parameter

Parameter	Type	Description
<address>	Numerical value	Sets the virtual start address of the VP_MAP table range to be cleared.

Note: All contents in the VP_MAP table are cleared if <address> is omitted.

Examples:

To clear all the contents in the VP_MAP table:

> *vc (RET)*

To clear the contents in the VP_MAP table range starting from virtual address H'4000:

> *vc H'4000 (RET)*

Related Items:

VD, VE, and VS commands

Note: This command is not supported in a device in which the MMU is not built-in.

RENESAS

5.2.23 VPMAP_DISPLAY: VD

Description:

Displays the address translation (VP_MAP) table set in the emulator.

Format:

vd

Table 5.24 VPMAP_DISPLAY Command Parameter

Parameter	Type	Description
None		

Example:

To display the VP_MAP table:

```
>vd (RET)
<VADDR_TOP>          <VADDR_END>          <PADDR_TOP>
01000000             0100ffff             02000000
01010000             0101ffff             03000000
ENABLE
```

<VADDR_TOP>, <VADDR_END>, and <PADDR_TOP> represent the virtual start address, the virtual end address, and the physical start address, respectively. ENABLE or DISABLE in the last line indicates that the VP_MAP table is valid or invalid.

Related Items:

VC, VE, and VS commands

Note: This command is not supported in a device in which the MMU is not built-in.

RENESAS

5.2.24 VPMAP_ENABLE: VE

Description:

Enables or disables the setting of the address translation (VP_MAP) table in the emulator.

Format:

```
ve <enable>
```

Table 5.25   VPMAP_ENABLE Command Parameter

Parameter	Type	Description
<enable>	Keyword	Enables or disables the setting of the VP_MAP table.
		enable: Enables the setting of the VP_MAP table. **disable**: Disables the setting of the VP_MAP table.

Note: The setting of the VP_MAP table is disabled at the emulator initiation.

Example:

To enable the setting of the VP_MAP table:

>***ve enable (RET)***

Related Items:

VC, VD, and VS commands

Note: This command is not supported in a device in which the MMU is not built-in.

5.2.25   VPMAP_SET: VS

Description:

Sets the address translation (VP_MAP) table in the emulator.

Format:

```
vs <lsaddress> <leaddress> <paddress>
```

Table 5.26   VPMAP_SET Command Parameters

Parameter	Type	Description
<lsaddress>	Numerical value	Specifies the virtual start address to be set in the VP_MAP table in the page size units supported by the MMU. Setting a physical fixed area or an internal I/O area as a virtual address will result in an error.
<leaddress>	Numerical value	Specifies the virtual end address to be set in the VP_MAP table in the page size units supported by the MMU. Setting a physical fixed area or an internal I/O area as a virtual address will result in an error.
<paddress>	Numerical value	Specifies the physical start address to be set in the VP_MAP table.

Note: The virtual address range to be newly set cannot overlap a virtual address that has already been set. Clear the previous set range when making a new setting.

Example:

To set the virtual address range H'4000 to H'4FFF to be translated into the physical address range H'400000 to H'400FFF:

>*vs H'4000 H'4fff H'400000 (RET)*

Related Items:

VC, VD, and VE commands
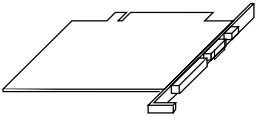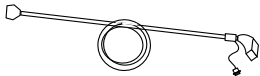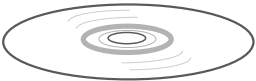
Note: This command is not supported in a device in which the MMU is not built-in.

RENESAS

Section 6   SH7750 E10A Emulator Specifications

6.1   Components of the Emulator

The SH7750 E10A emulator supports the SH7750 and SH7750S. Table 6.1 lists the components of the emulator.

Table 6.1   Components of the Emulator (HS7750KCM01H or HS7750KCI01H)

Classi-fication	Component	Appearance	Quan-tity	Remarks
Hard-ware	Card emulator	(PCMCIA) or (PCI)	1	HS7750KCM01H (PCMCIA: 14-pin type): Depth: 85.6 mm, Width: 54.0 mm, Height: 5.0 mm, Weight: 27.0 g
				HS7750KCI01H (PCI: 14-pin type): Depth: 144.0 mm, Width: 105.0 mm, Mass: 93.0 g
	User system interface cable		1	HS7750KCM01H (PCMCIA: 14-pin type): Length: 80 cm*, Weight: 45.0 g
				HS7750KCI01H (PCI: 14-pin type): Length: 150 cm, Mass: 86.0 g
Soft-ware	SH7750 E10A emulator setup program, SH7750 E10A Emulator User's Manual, and Hitachi Debugging Interface User's Manual		1	HS7750KCM01SR, HS7750KCM01HJ, HS7750KCM01HE, HS6400DIIW5SJ, and HS6400DIIW5SE (provided on a CD-R)

Note: The length may be 50 cm according to the time for purchasing the product.

6.2 Pin Arrangement of the Hitachi-UDI Port Connector

Figure 6.1 shows the pin arrangement of the Hitachi-UDI port connector (14 pins).

CAUTION

Note that the pin number assignment of the Hitachi-UDI differs from that of the connector manufacturer.
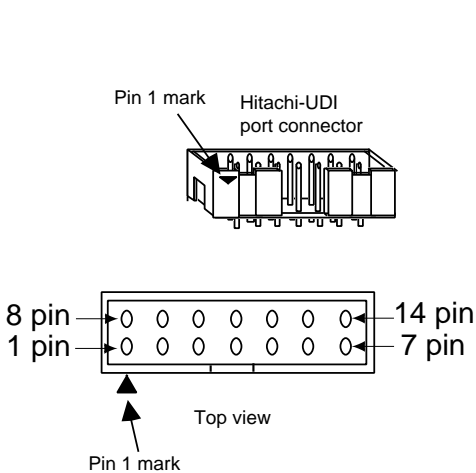
Pin No.	Signal	Input/ Output*[1]	SH7750 Pin No.	
			BGA256	HQFP208
1	TCK	Input	A-5	198
2*[2]	/TRST	Input	C-4	200
3	TDO	Output	A-6	194
4	/ASEBRK*[2] BRKACK	Input /Output	B-7	193
5	TMS	Input	B-6	197
6	TDI	Input	B-5	199
7*[2]	/RESET	Output	B-1	2
11	Not connected	—	—	—
8 to 10 and 12 to 13	GND	—	—	—
14*[3]	GND	Output	—	—

Notes: 1. Input to or output from the user system.
      2. The slash (/) means that the signal is low-active.
      3. The emulator monitors the GND signal of the user system and detects whether or not the user system is connected.

Figure 6.1 Pin Arrangement of the Hitachi-UDI Port Connector (14 Pins)

RENESAS

Notes: 1. Handling of the TCK, TMS, TDI, TDO, /TRST, and /ASEBRK pins depend on the use conditions of the Hitachi-UDI as follows:

   (a) When the user system is used by connecting the emulator, the TCK, TMS, TDI, TDO, /ASEBRK pins must be pulled up by a resistance of several kilo-ohms. The /TRST pin must be pulled down by a resistance of several kilo-ohms.

   (b) When an interrupt and reset are used through the Hitachi-UDI and the user system is independently used, the TCK, TMS, TDI, TDO, /ASEBRK pins must be pulled up by a resistance of several kilo-ohms. The /TRST pin must be pulled down by a resistance of several kilo-ohms.

   (c) When the user system is independently used without using the emulator and Hitachi-UDI, the TCK, TMS, TDI, TDO, /ASEBRK pins must be pulled up by a resistance of several kilo-ohms. The /TRST pin must be grounded.

2. The /RESET signal in the user side is input to the B-1 pin (BGA256) or 2 pin (HQFP208) of the SH7750. Connect this pin to the Hitachi-UDI port connector as the output from the user system.

RENESAS

6.3 User System Interface Circuit

The emulator is connected to the user system via the user interface cable. Figure 6.2 shows the user system interface circuit of the emulator (HS7750KCM01H).
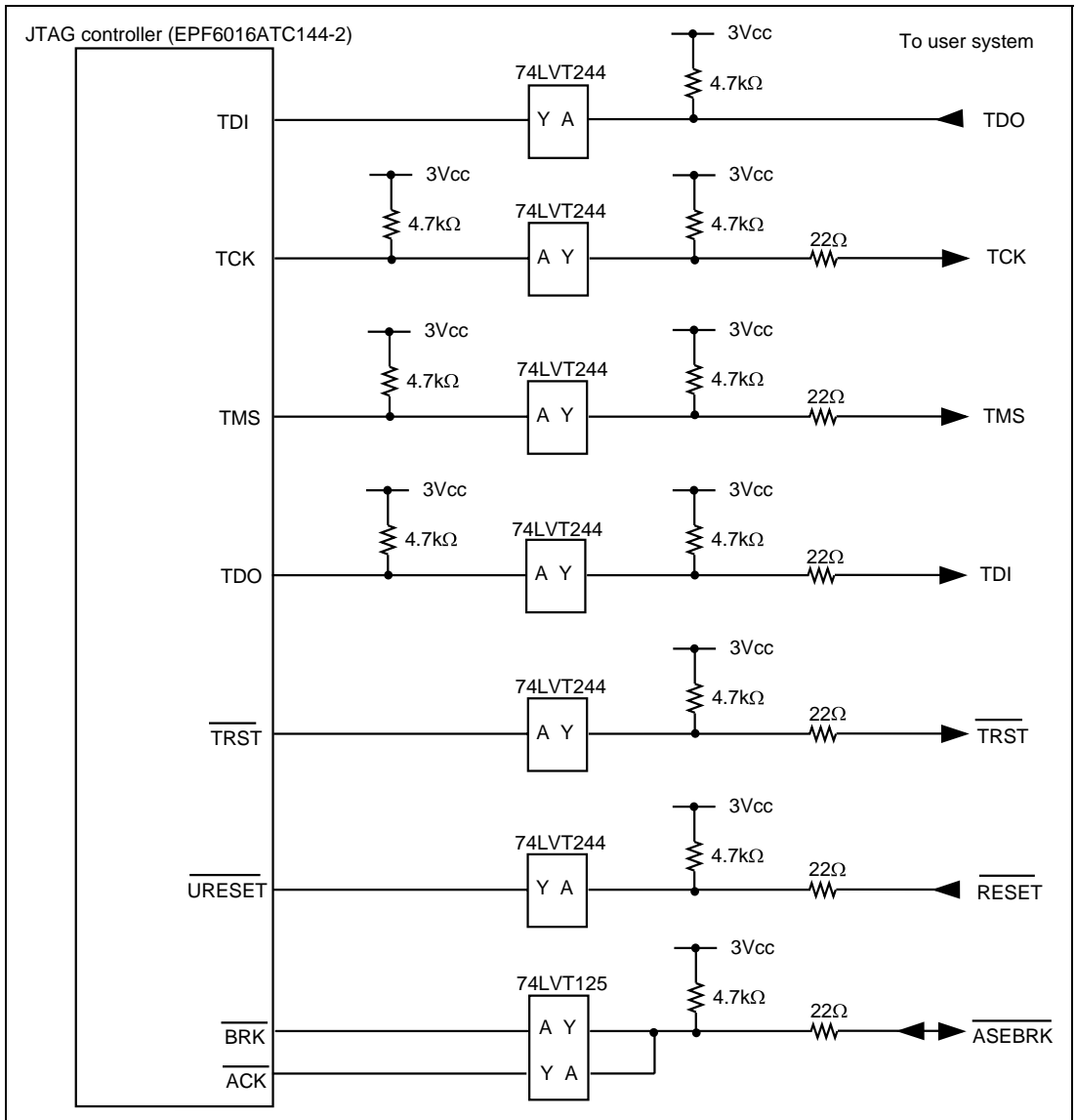
Figure 6.2   User System Interface Circuit (HS7750KCM01H)
(Model Name: HS0005KCM03H)

RENESAS

Figure 6.3 shows the user system interface circuit of the emulator (HS7750KCI01H).
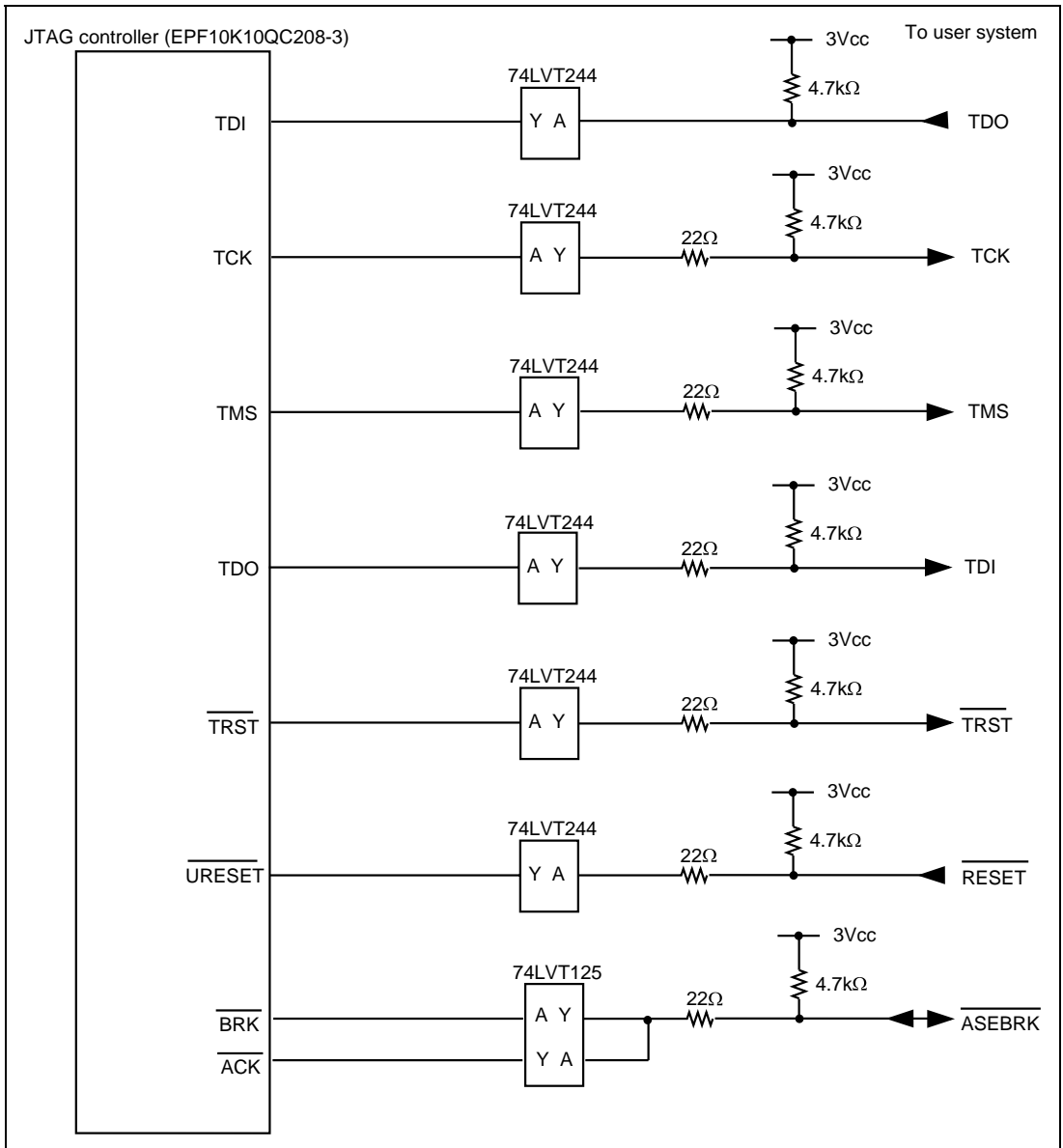
**Figure 6.3   User System Interface Circuit (HS7750KCI01H)
(Model Name: HS0005KCI03H)**

RENESAS

6.4 Differences between the SH7750 and SH7750S 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 6.2.

Table 6.2 Register Initial Values at Emulator Power-On

Register	Emulator at Power-on
R0_BANK0 to R7_BANK0	H'00000000
R0_BANK1 to R7_BANK1	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 Hitachi-UDI; do not access the Hitachi-UDI by the user program.

RENESAS

3. Low-Power Modes (Sleep, Standby, and Module Standby)

For low-power consumption, the SH7750 and SH7750S have sleep, standby, and module standby modes.

The sleep and standby modes are switched using the SLEEP instruction. When the emulator is used, the sleep mode can be cleared by either normal clearing or by the satisfaction of a break condition (including (BREAK) or (CTRL) + C key input). In the latter case, the user program breaks. The standby mode can be cleared with the normal clearing function or (BREAK) or (CTRL) + C key input, and after the standby mode is cleared, the user program operates correctly. Note, however, that if a command has been entered in standby mode or module standby mode, no commands can be used from the emulator after the standby mode is cleared.

Notes: 1. After the sleep mode is cleared by a break, execution restarts at the instruction following the SLEEP instruction.

2. If the memory is accessed or modified in sleep mode, the sleep mode is cleared and execution starts at the instruction following the SLEEP instruction.

3. Although the SH7750S supports the hardware standby function, if the SH7750 E10A emulator enters the hardware standby mode, a TIMEOUT error will occur.

4. RESET Signals (/RESET and /MRESET)

The SH7750 and SH7750S RESET signals (/RESET and /MRESET) are only valid during user program execution started with clicking the GO or STEP-type button. If these signals are input from the user system in command input wait state, they are not sent to the SH7750 or SH7750S.

Note: Do not start user program execution while control input signals (/RESET, /RDY, and /BREQ) are being low. A TIMEOUT error will occur.

5. Direct Memory Access Controller (DMAC)

The DMAC operates even in the command wait state. When a data transfer request is generated, the DMAC executes DMA transfer.

6. Internal I/O Registers

In the emulator, the internal I/O registers can be accessed from the [I/O registers] window. However, pay attention when accessing the SDMR register of the bus-state controller. Before accessing the SDMR register, specify addresses to be accessed in the I/O-register definition file (SH7750.IO or SH7750S.IO) and then activate the HDI. For details on I/O-register definition files, refer to the Hitachi Debugging Interface User's Manual.

Note: As default, SDMR2 and SDMR3 are specified in the I/O-register definition file as the area-2 SDMR register and area-3 SDMR register, respectively.

When the SH7750S is used, the IPRD register is not displayed in the [I/O Registers] window. To get it to display, edit the I/O-register definition file (SH7750.IO) as follows and start the HDI:

RENESAS

In SH7750.IO, locate 'IPRC = 0xFFD0000C W A'. Under this, add 'IPRD = 0xFFD00010 W A' and save the file.

7. Memory Access during Emulation

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 E10A emulator to access the memory. Therefore, realtime emulation cannot be performed.

The stopping time of the user program is as follows:

Environment:

     Host computer: 500 MHz (Pentium® III)

     SH7750: 50 MHz (CPU clock)

     JTAG clock: 15 MHz

When a one-byte memory is read from the command-line window, the stopping time will be about 47 ms.

8. Interrupt

When the NMIB bit in the ICR register is 1, the NMI interrupt is accepted during break and the program is executed from the NMI interrupt vector. If the program cannot return normally from the NMI interrupt routine or the value in the general-purpose register is not guaranteed, a communication timeout error will occur.

9. Memory Access during User Program Break

The emulator can download the program for the flash memory area. Other memory write operations are enabled for the RAM area. Therefore, an operation such as memory write, BREAKPOINT, or user program download should be set only for the RAM area. When the memory area can be written by the MMU, do not perform memory write, BREAKPOINT, or downloading.

10. 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 writes to the memory.

— At memory read: Does not change the cache write mode that has been set.

Therefore, when memory read or write is performed during user program break, the cache state will be changed.

6.5 Specific Functions for the SH7750 E10A Emulator

The SH7750 E10A emulator includes the following useful functions.

- Performance analysis function
  Measures several performances, such as the cache-miss count.
- Interrupt enable function during user program break
  Accepts interrupts during user program breaks.

For details, refer to section 6.5.7, SH7750 E10A Emulator Useful Functions.

The SH7750 E10A emulator does not support the following function:

- AUD trace function

6.5.1 Emulator Driver Selection

Table 6.3 shows drivers which are selected in the [E10A Driver Details] dialog box.

Table 6.3 Type Number and Driver

Type Number	Driver
HS7750KCM01H	E10A PC Card Driver 3
HS7750KCI01H	E10A PCI Card Driver 3

RENESAS

6.5.2 Break Condition Functions

Break Conditions: In the SH7750 E10A emulator, five break conditions can be set (Break Condition 1, 2, 3, 4, 5). Table 6.4 lists the items that can be specified for each.

Table 6.4 Types of Break Conditions

Break Condition Type	Description
Address bus condition (Address)	Breaks when the SH7750 or SH7750S address bus value or the program counter value matches the specified value.
Data bus condition (Data)	Breaks when the SH7750 or SH7750S data bus value matches the specified value. Byte, word, or longword can be specified as the access data size.
ASID condition (ASID)	Breaks when the SH7750 or SH7750S ASID value matches the specified condition.
Bus state condition (Bus State)	There are two bus state condition settings:
	Read/write condition: Breaks when the SH7750 or SH7750S RD or RDWR signal level matches the specified condition.
	Bus state condition: Breaks when the operating state in an SH7750 or SH7750S bus cycle matches the specified condition.
LDTLB instruction break condition	Breaks when the SH7750 or SH7750S executes the LDTLB instruction.
Internal I/O break condition	Breaks when the SH7750 or SH7750S accesses the internal I/O.

Note: **For details on window function and command-line syntax, refer to the on-line help function.**

Table 6.5 lists the combinations of conditions that can be set under Break Condition 1, 2, 3, 4, 5.

RENESAS

Table 6.5 Dialog Boxes for Setting the Break Condition

	Dialog Box		
	[Break Condition 1] Dialog Box	[Break Condition 2, 3, 4] Dialog Box	[Break Condition 5] Dialog Box
Address bus condition (Address)	O	O	X
Data bus condition (Data)	O	X	X
ASID condition (ASID)	O	O	X
Read/write specification	O	O	X
Data access	O	O	X
Before/after execution	O	O	X
Sequential break	O	O	X
LDTLB instruction break	X	X	O
Internal I/O break	X	X	O

Note: O: Can be set in the dialog box.
       X: Cannot be set in the dialog box.

Notes: 1. If the BL bit of the SR register is 1, do not use BREAKPOINTs.

2. If a break is specified for an address that is close to an address whose instruction generates a manual reset, a manual reset may be generated instead of a break. Therefore, to ensure the performance of a break, specify a break for an address that is four addresses before the address whose instruction generates an exception.

The SH7750 E10A emulator has sequential break functions. When the break conditions are satisfied in the specified order (table 6.6), program execution is halted. This mode is called sequential break.

RENESAS

Table 6.6   Sequential Break Conditions

Break Condition	Description
Sequential break condition 2-1	Program is halted when Break Condition 2 and Break Condition 1 are satisfied in that order.
Sequential break condition 3-2-1	Program is halted when Break Condition 3, Break Condition 2, and Break Condition 1 are satisfied in that order.
Sequential break condition 4-3-2-1	Program is halted when Break Condition 4, Break Condition 3, Break Condition 2, and Break Condition 1 are satisfied in that order.

Note: Sequential breaks can be specified by the [Configuration] dialog box.

Notes on Setting the [Break Condition] Dialog Box and BREAKCONDITION_SET Command:

1. When [Go to cursor], [Step In], [Step Over], or [Step Out] is selected, the settings of Break Condition 4 are disabled.
2. Break Condition 4 is disabled when an instruction to which a BREAKPOINT has been set is executed. Accordingly, do not set a BREAKPOINT to an instruction which satisfies Break Condition 4.
3. When a Break Condition is satisfied, emulation may stop after two or more instructions have been executed.
4. If a PC break before execution 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.

6.5.3   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. Accordingly, it can be set only to the internal RAM area. However, a BREAKPOINT cannot be set to the following addresses:
   — An address whose memory content is H'003B
   — An area other than the CS0 to CS6 areas and the internal RAM area
   — An instruction in which Break Condition 4 is satisfied
   — A slot instruction of a delayed branch instruction

   In addition, do not perform memory write, BREAKPOINT, or download even if the memory space can only be written by the MMU.
3. During step execution, a BREAKPOINT is disabled.

RENESAS

4. Conditions set at Break Condition 4 are disabled when an instruction to which a BREAKPOINT has been set is executed. Do not set a BREAKPOINT to an instruction in which Break Condition 4 is satisfied.

5. When execution resumes from the address where a BREAKPOINT is specified, single-step execution is performed at the address before execution resumes. Therefore, realtime operation cannot be performed.

6. 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.

7. 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 SH7750 or SH7750S MMU status during command input when the VPMAP_SET command setting is disabled. The ASID value of the SH7750 or SH7750S 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 SH7750 or SH7750S MMU status during command input. Even when the VP_MAP table is modified after BREAKPOINT setting, the address translated at BREAKPOINT setting is valid.

8. 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 SH7750 or SH7750S MMU during 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 [System Status] window is ILLEGAL INSTRUCTION, not BREAKPOINT.

9. 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 SH7750 or SH7750S MMU during 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.

10. If a TLB error occurs during virtual address setting, the following message box will be displayed.

RENESAS

Figure 6.4   Message Box for Clearing a TLB-Error

If a program is executed again without clearing the BREAKPOINT set at the address in which the TLB error occurs, a TLB error will occur again. Accordingly, clear the BREAKPOINT before execution.

11. 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 [System Status] window is ILLEGAL INSTRUCTION, not BREAKPOINT.

12. When a BREAKPOINT is set to the cacheable area, the cache block containing the BREAKPOINT address is filled immediately before and after user program execution.

13. While a BREAKPOINT is set, the contents of the instruction cache are disabled at execution completion.

6.5.4   Notes on Using the JTAG Clock (TCK)

Set the JTAG clock (TCK) frequency to lower than the frequency of half of the SH7750 or SH7750S peripheral module clock (CKP). Do not set the frequency as 16.5 MHz or more.

Note: The SH7750 E10A emulator does not support the AUD function.

6.5.5   Trace Functions

The SH7750 E10A emulator does not support the AUD function. Table 6.7 shows the emulator type number and AUD function.

Table 6.7   Type Number and AUD Function

Type Number	AUD Function
HS7750KCM01H	Not supported
HS7750KCI01H	Not supported

RENESAS

Table 6.8 shows the trace functions.

Table 6.8   Trace Functions

Function	Description
Branch instruction trace	Traces and displays the branch instructions. The branch source address and branch destination address for the eight latest branch instructions are displayed. There are three kinds of branch instruction trace: • Normal branch instruction trace Traces and displays the normal branch instructions. The normal branch instructions are the BF, BF/S, BT/S, BRA, BRAF, and JMP instructions. To use this function, select the [Acquire normal branch instruction trace] radio button in the [Trace Acquisition] dialog box. • Subroutine branch instruction trace Traces and displays the subroutine branch instructions. The subroutine branch instructions are the BSR, BSRF, JSR, and RTS instructions. To use this function, select the [Acquire subroutine branch instruction trace] radio button in the [Trace Acquisition] dialog box. • Exception branch instruction trace Traces and displays the exception branch instruction. The exception branch instruction is the RTE instruction. In addition, all the exception and interrupt operations are traced. To use this function, select the [Acquire exceptional branch instruction trace] radio button in the [Trace Acquisition] dialog box.
Internal I/O trace	Traces and displays the address and data that access the internal I/O area. To use this function, select the [Get trace information of internal I/O Area] radio button in the [Break Condition 5] dialog box and the [Acquire continuous trace] check box in the [Trace Acquisition] dialog box.

RENESAS

Table 6.8   Trace Functions (cont)

Function	Description
LDTLB instruction execution trace	Traces and displays the address that executes the LDTLB instruction. To use this function, select the [Get trace information of LDTLB instruction] radio button in the [Break Condition 5] dialog box and the [Acquire continuous trace] check box in the [Trace Acquisition] dialog box.
Continous trace	Acquires the trace information continuously. This is called continuous trace. For the branch instruction trace, eight-branch information can be repeatedly acquired a maximum of four times. Select the [Acquire continuous trace] radio button in the [Trace Acquisition] dialog box. If continuous trace is selected, realtime trace cannot be performed.

Notes: 1. For the SH7750 E10A emulator, trace acquisition of the eight latest branch instructions is enabled.

2. If an interrupt is generated at the program execution start or end, including a step execution, 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 ***

3. If a TLB error occurs while acquired trace information is displayed, the following error message will be displayed.

Figure 6.5   Message Box for Clearing a TLB Error

After a TLB error, trace acquisiton cannot be performed.

4. When MMU settings are modified or when a user program is changed between GO command completion and trace display, the displayed mnemonics or operand may not be correct.

5. If a completion-type exception occurs during exception branch acquisition, the next address to the address in which an exception occurs is acquired.

RENESAS

6. When a user interrupt is enabled by the INTERRUPT command during the emulator command wait state or user program execution, an interrupt that is generated at the program execution start or end, including a step execution, can be traced in realtime.

1. Notes on the [Trace] Window
   (1) If an interrupt is generated at the program execution start or end, including a step execution, 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 TLB error occurs while acquired trace information is displayed, the following error message will be displayed.

Figure 6.6   Message Box for Clearing a TLB Error

   After a TLB error, trace acquisition cannot be performed.
   (3) When MMU settings are modified or when a user program is changed between GO command completion and trace display, the displayed mnemonics or operand may not be correct.
   (4) If a completion-type exception occurs during exception branch acquisition, the next address to the address in which an exception occurs is acquired.
   (5) When a user interrupt is enabled by the INTERRUPT command during the emulator command wait state or user program execution, an interrupt that is generated at the program execution start or end, including a step execution, can be traced in realtime.

RENESAS

2. Notes on Setting the [Trace Acquisition] Window

   (1) When the [Acquire continuous trace] check box is selected, do not perform memory access during emulation.

   (2) When internal I/O trace or LDTLB instruction trace is performed, select the [Acquire continuous trace] check box.

   (3) When the [Acquire continuous trace] check box is selected, 32 trace information data can be acquired. In this case, however, since the user program stops at constant intervals, the processing speed is decreased compared with the case where the [Acquire continuous trace] check box is not selected.

   (4) 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

   (5) When the [Acquire continuous trace] check box is selected, and when either the [Get trace information of internal I/O area] radio button (internal I/O trace enabled) or the [Get trace information of LDTLB instruction] radio button is selected (LDTLB instruction trace enabled) with the [Break Condition 5] dialog box,
   - An internal I/O trace cannot be made with the Step In command.
   - The LDTLB instruction and internal I/O trace cannot be performed with the Step Over command.

   (6) In the SH7750 E10A emulator, the contents of the SGR register are lost when the user program breaks. Since the user program execution stops at constant intervals while the continuous trace is selected, the contents of the SGR register will be lost. Accordingly, do not use the continuous trace for a program in which an SGR value is referred to with the interrupt handler.

   (7) When continuous trace is used, do not enable user interrupt by the INTERRUPT command during the emulator command wait state or user program execution.

6.5.6   Notes on Using the Profile Function

1. Errors

   The profile function internally breaks user program execution, collects the measured data, and re-executes the user program.

   Since the function also counts when the measured item is generated at break or re-execution, an error will be included in the measured profile value.

   The measured value of this function should be the target.

RENESAS

2. Functions that cannot be used while the profile function is being used

   (a) Performance measurement function

   The profile function is implemented by using the performance measurement function described in section 6.5.7 (1), Performance measurement function. This function cannot be used when the profile function is enabled.

   (b) Step function

   When the profile function is enabled, do not use the step function. The profile data cannot be measured correctly.

   (c) Memory access during user program execution

   When the profile function is enabled, memory access is disabled during user program execution.

   (d) Continuous trace function

   When the profile function is enabled, do not use the continuous trace function that can be used in the internal trace function. The profile data cannot be measured correctly.

   (e) Internal trace function

   When the profile function is enabled, mode selection of the internal trace is disabled since all items of the internal trace modes are selected in the emulator.

   (f) Halt function

   When the profile function is enabled, do not use the halt function for the internal or AUD trace.

3. Others

   (a) When the profile function is used, the contents that have been set in the performance measurement function or data that has been measured will be deleted.

   (b) Since the profile function is implemented with the internal break, it takes a long time to start and end the user program execution. The user program execution times under the following environment are shown below:

   Environment:

   Host computer: 800 MHz (Pentium® III)

   Memory: 130 Mbytes

   OS: Windows® 2000

   Execution program: 10,000 nested calls

   (i) When the profile function is not used: 1 second or lower

   (ii) When the profile function is used in the setting without including a child function:
   20 seconds

   (iii) When the profile function is used in the setting including a child function:
   211 seconds

RENESAS

6.5.7 SH7750 E10A Emulator Useful Functions

1. Performance analysis function

   The SH7750 E10A emulator can measure the performances of the SH7750 or SH7750S. Display and initialization can be performed by the PERFORMANCE_ANALYSIS command, cancellation can be performed by the PERFORMANCE_CLEAR command, and setting can be performed by the PERFORMANCE_SET command. The performance analysis method is described below.

Notes: 1. When <channel> is omitted for the initialization and cancellation of the performance measurement results, all performance measurement results are initialized and cancelled.

   2. When performance measurement conditions are canceled, the settings in the UBC are not guaranteed.

   3. When setting start and end PC values for both channel 1 and channel 2, set the same start and end PC values for PERFORMANCE_ANALYSIS channel 1 and PERFORMANCE_ANALYSI channel 2. If different values are set, the last setting is valid.

   4. When start and end PC values are set with this command, the UBC contents set before using this command are ignored.

   The emulator measures how many times the events specified with the performance analysis function are satisfied. The emulator uses measurement channels 1 and 2 to specify the measurement start and end PC values. Therefore, the UBC function cannot be used by the user when the PC values are specified for the start and end conditions. For this function, two events can be measured simultaneously and the following conditions can be specified:

   — Measurement range

   One of the following ranges can be specified by either of measurement channels 1 and 2.

   1. From the start to the end of the user program execution

   2. From the occurrence of the event set in channel A of the UBC to the occurrence of the event set in channel B

   When the first range is specified, the measurement result includes a several-cycle error for one user program execution. Therefore, do not specify this range when the step is to be executed. In addition, the user program execution stops when continuous trace is used; again, do not specify the first range in this case.

   — Events

   Cache miss count, TLB miss count, branch count, instruction execution count, operand access count, internal I/O access count, interrupt count, cache fill count, and elapsed cycle count can be measured. Table 6.9 lists the specifiable events.

RENESAS

Table 6.9   Measurement Events

Event	Keyword	Description
Operand access count	OARW*	The number of times the operand access is performed on the cacheable area when the cache is enabled (both read and write accesses).
	OARAM	The number of times the internal RAM area is accessed.
	OA	The number of all operand accesses.
Internal I/O access count	IOA	The number of times the internal I/O is accessed.
Cache miss count	DCRW	The number of times operand cache misses occur at data reading or writing.
	EC	The number of times instruction cache misses.
TLB miss count	DT	The number of times UTLB misses occur at data access.
	ET	The number of times UTLB and ITLB misses occur at instruction access.
Instruction fetch count	EF*	The number of times instructions are fetched from the cacheable area when the cache is enabled.
	EA	The number of times all instructions are fetched.
Branch count	B	The number of times branch instructions are issued (instructions to be counted: BF (other than displacement 0), BF/S, BT (other than displacement 0), BT/S, BRA, BRAF, JMP).
	BT	The number of times branches are taken (branches to be counted are the same as mode B).
Instruction execution count	E	The number of times instructions are issued.
	E2	The number of times two instructions are issued at the same time.
	EFP	The number of times FPU instructions are issued.
	ETR	The number of times the TRAPA instruction is executed.
Interrupt count	INT	The number of interrupts except NMI.
	NMI	The number of NMI interrupts.
Cache fill cycle count	ECF	The number of instruction cache fill cycles.
	OCF	The number of operand cache fill cycles.
Elapsed time count	TM	The number of cycles for elapsed time.

Note: For the non-cache operand accesses due to the PREF instruction or TLB.c=0, the correct value cannot be counted.

The events can be counted even in the conditions shown in table 6.10, in addition to the normal count conditions.

RENESAS

Table 6.10 Performance Count Conditions

Event	Count Condition	Target Mode
All count conditions	When the event to be counted up is canceled by an exception.	All
Instruction cache miss count	• Includes instruction fetch for the cache-off area to count the number of times the instruction has not been fetched in one cycle. • When a cache miss occurs during an overrun fetch generated at exception.	EC
TLB miss count	When the TLB miss is canceled by an exception having a higher priority than that of the TLB miss	DT and ET
Instruction fetch count	• When the instruction fetch request by the CPU is accepted. • Does not count when the cache is bypassed from the external bus to supply the instruction to the CPU at instruction cache miss.	EF and EA
Instruction issue count	Counts one when two instructions are issued at the same time.	E
	Counts one to three when instruction fetch exception (instruction address error, instruction TLB miss exception, or instruction TLB protection violation exception) occurs.	E and E2
FPU instruction issue count	• Counts one when two instructions are issued at the same time. • The following shows the FPU instructions: LDS Rm, FPUL, LDS.L @Rm+, FPUL, LDS Rm, FPSCR, LDS.L @Rm+, FPSCR, STS FPUL, Rn, STS.L FPUL, @-Rn, STS FPSCR, Rn, STS.L FPSCR, @-Rn Others: instructions that the instruction code is H'Fxxx	EFP

— Counting method

One of the following methods can be specified by each of measurement channels 1 and 2.

1. Counted by the CPU operating clock
2. Counted by the ratio of the CPU operating clock to the bus clock

When the above method 1 is specified, one CPU operating clock cycle is counted as one. When method 2 is specified, the count is incremented by 3, 4, 6, 8, 12, or 24, according to the clock frequency ratio (ratio of the CPU clock to the bus clock). In this case, the execution time can be calculated by the following expression:

RENESAS

$$T = C \times B / 24 \quad \text{(T: Execution time; B: Time of one bus clock cycle; C: Count)}$$

When the ratio of the CPU clock to the bus clock is changed in the user program, it is recommended to select method 2, above, to count the number of cycles.

The following shows examples to measure the performance of the user program by the performance measurement function.

1. **Measuring cache hit ratio**

   Specify measurement channel 1 to count the cache misses (for data read and write) and specify measurement channel 2 to count operand accesses (read and write) to the cacheable area while the cache is enabled. Specify, with both the channels, the measurement from the start to the end of user program execution.

   With the above command settings, the cache miss count and the access count to the cacheable area can be measured, and the cache hit ratio in the executed user program can be obtained.

2. **Measuring ratio of execution time in specified program area to total execution time**

   Specify measurement channel 1 to measure the elapsed cycle count from the start to the end of user program execution. Specify measurement channel 2 to measure the elapsed cycle count during execution from the specified start PC to the specified end PC.

   With both the channels, the total elapsed cycle and the elapsed cycle for the specified program area can be measured, and the ratio of the execution time in the specified program area to the total execution time can be obtained.

Notes: 1. **The counter for performance measurement has 48 bits. A maximum of $2^{48} = 2.8 \times 10^{14}$ counts and 16.3-day cycles (when the CPU operating frequency is 200 MHz) can be measured. If a counter overflow occurs, the count becomes invalid.**

2. **When performance measurement conditions are canceled, the settings in the UBC are not guaranteed.**

Note: **For details on window function and command-line syntax, refer to the on-line help function.**

2. Interrupts

   During user program execution or in command input wait state, any interrupt to the SH7750 can be used. Whether or not to process interrupts during user program execution or in command input wait state can be specified.

— When no interrupt is processed during user program execution or in command input wait state

While the emulator is executing the user program or is in command input wait state, interrupts are not processed generally. However, if an internal interrupt or an edge sensitive external interrupt occurs in command input wait state, the emulator holds the interrupt and executes the interrupt processing routine when the GO command is entered.

— When interrupts are processed during user program execution or in command input wait state

To process non-maskable interrupts and peripheral module interrupts during emulator command execution and in command input wait state, use the INTERRUPT command. Input the INTERRUPT command into the command line window.

- To process only non-maskable interrupts
  Specify "nonmask" in the <interrupt_enable> option of the INTERRUPT command.

- To process non-maskable interrupts and peripheral module interrupts
  Specify "all" in the <interrupt_enable> option of the INTERRUPT command.

- To switch to the mode in which no interrupt is processed:
  Specify "disable" in the <interrupt_enable> option of the INTERRUPT command.

Notes: 1. When interrupts are accepted during user program execution and command input wait state, user interrupt processing is not traced. In this case, continuous trace cannot be enabled.

2. Use the NOP instruction at the delay slot after the RTE instruction in the interrupt handler.

3. If a user interrupt is inserted while the user program breaks until the processing ends, do not set a BREAKPOINT in the interrupt handler. The emulator may generate a Communication Timeout error. Use the Break Condition function.

Note: For details on window function and command-line syntax, refer to the on-line help function.

6.5.8   Notes on HDI

1. Moving Source File Position after Creating Load Module

When the source file is moved after creating the load module, the [Open] dialog box may be displayed to specify the source file during the debugging of the created load module. Select the corresponding source file and click the [Open] button.

RENESAS

2. Source-level Execution
   — Source file

   Do not display source files that do not correspond to the load module in the program window. For a file having the same name as the source file that corresponds to the load module, addresses are displayed in the program window but operation in the window may not work properly.

   — Step

   Even standard C libraries are executed. To return to a higher-level function, enter Step Out. In a for statement or a while statement, executing a single step does not move execution to the next line. To move to the next line, execute two steps.

3. Operation During Accessing Files

   Do not perform other operations during saving in the [Load Program], [Verify Memory], [Save Memory], or [Trace] window because this will not allow correct saving to be performed.

4. Source Window at Program Change

   When a program being displayed in the source window is changed and the source file and load module are reloaded, close and reopen the source window once. If the window is not closed and reopened, the display may be incorrect.

5. Watch
   — Local variables at optimization

   Depending on the generated object code, local variables in a C source file that is compiled with the optimization option enabled will not be displayed correctly. Check the generated object code by displaying the [Disassembly] window.

   If the allocation area of the specified local variable does not exist, displays as follows.

   Example: The variable name is asc.

   asc = ? - target error 2010 (xxxx)

   — Variable name specification

   When a name other than a variable name, such as a symbol name or function name, is specified, no data is displayed.

   Example: The function name is main.

   main =

   — Array display

   When array elements exceed 1000, elements from after 1000 will not be displayed.

RENESAS

6. Memory Load Function

When [Load...] is selected from the [Memory] menu, the Memory Load function can be used although it takes time to download. It is recommended that the File Load function ([Load Program...] selected from the [File] menu) is used to load the S-type file.

Note: The File Load function deletes the debugging information of the previously loaded program. When other load modules are loaded after the program to be debugged has been loaded, use the following sequence: When the program to be debugged is linked, save the debugging information in another file. Load the debugging information file after all the load modules have been loaded.

7. Line Assembly

— Input radix

Regardless of the Radix setting, the default for line assembly input is decimal. Specify H' or 0x as the radix for a hexadecimal input.

8. Command Line Interface

— Batch file

To display the message "Not currently available" while executing a batch file, enter the sleep command. Adjust the sleep time length which differs according to the operating environment.

> Example: To display "Not currently available" during memory_fill execution:
>
> sleep d'3000
>
> memory_fill 0 ffff 0

— Overwrite file

In Command Line Interface, a file having the same name as the output file is overwritten without asking the user.

— File specification by commands

The current directory may be altered by file specifications in commands. Absolute paths are recommended to be used to specify the files in a command file so that the current directory alteration is not affected.

> Example: FILE_LOAD C:\\HEW\\HDI5\\E10A\\7750\\TUTORIAL\\TUTORIAL \\DEBUG\\TUTORIAL.ABS

RENESAS

9. About Hitachi Debugging Interface User's Manual

This version of HDI does not support section 10, Selecting Functions, written in Hitachi Debugging Interface User's Manual.

10. Initiating HDI

When the emulator is initiated by using another card emulator after it has been initiated by using the PCI card emulator, delete the [TARGET] line from the C:\windows\HDI.INI file.

11. Usage with Another Version of HDI

— Automatic load of session files

Since the emulator cannot use another version of HDI, re-install this HDI whenever another version has been previously installed.

If another version has been used, initiate this HDI with "Run" as follows without using the session files.

```
<Directory path name in which HDI is installed>\hdi /n (RET)
```

/n initiates the HDI without loading the recently used session files.

If there is another session file in the different debug platform, the following error message is displayed:

```
invalid target system: <recently used debug platform name>
```

12. [Select Function] Dialog Box

This HDI does not support BREAKPOINT setting in the [Select Function] dialog box (described in section 10, Selecting Functions, in the Hitachi Debugging Interface User's Manual).

13. Memory Save During User Program Execution

Do not execute memory save or verifying during user program execution.

14. [Performance Analysis] Window

This HDI does not support the [Performance Analysis] window (described in section 13.9, Performance Analysis, in the Hitachi Debugging Interface User's Manual).

15. Load of Motorola S-type Files

This HDI does not support Motorola S-type files with only the CR code (H'0D) at the end of each record. Load Motorola S-type files with the CR and LF codes (H'0D0A) at the end of each record.

RENESAS

16. [Memory] Window

If the following memory contents are displayed, they will be incorrect.

Word access from address 2n + 1

Longword access from address 4n + 1, 4n + 2, or 4n + 3

17. Scrolling Window During User Program Execution

Do not scroll the [Memory] and [Disassembly] windows by dragging the scroll box during user program execution. This generates many memory reads causing the user program to stop execution until the memory reads have been completed.

18. [I/O Registers] 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 6.11   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 [I/O Registers] window or [Memory] window.

- The internal I/O registers can be accessed from the [I/O registers] window. However, note the following when accessing the SDMR register of the bus-state controller. Before accessing the SDMR register, specify addresses to be accessed in the I/O-register definition files (SH7750.IO and SH7750S.IO) and then activate the HDI. For details on I/O-register definition files, refer to the Hitachi Debugging Interface User's Manual. Note that, however, the E10A emulator does not support the invalid module and bit-information display functions described in section 8, Displaying Variables, on the Hitachi Debugging Interface User's Manual.

RENESAS

- When the SH7750S is used, the IPRD register is not displayed in the [I/O Registers] window. To get it to display, edit the I/O-register definition file (SH7750.IO) as follows and start the HDI:

  In SH7750.IO, locate 'IPRC = 0xFFD0000C W A'. Under this, add 'IPRD = 0xFFD00010 W A' and save the file.

- Verify

  In the [I/O Registers] window, the verify function of the input value is disabled.

19. Note on [Registers] Window Operation During Program Execution

Although a dialog box is displayed in which the register contents can be changed by double-clicking the [Registers] window, do not change the register contents during program execution.

20. Note on Session Save of [Registers] Window

When the RB bit in the SR register is 0 and the session save of the [Registers] window is performed, the contents of general registers R0 to R7 cannot be saved.

21. Note on Radix in the [Register] Dialog Box

The default input radix in the [Register] dialog box is hexadecimal irrespective of the Radix display. When a radix other than a hexadecimal is input, specify the prefix code such as B'. After the value has been input in the [Register] dialog box, the Radix setting is changed to hexadecimal. When the radix other than a hexadecimal is used as a default, reset the Radix display.

22. BREAKPOINT
— Session file

When the BREAKPOINT address set in the session file is H'0, the BREAKPOINT will not be set. If the address set as the BREAKPOINT is wrong, the error message is not output. The BREAKPOINT is registered as DISABLE in the [Breakpoints] window.

— Breakpoint cancellation

When the contents of the BREAKPOINT address is modified during user program execution, the following message is displayed when the user program stops.

   BREAKPOINT IS DELETED A=xxxxxxxx

If the above message is displayed, cancel all BREAKPOINT settings with the [Delete All] or [Disable] button in the [Breakpoints] window.

— [Run program] dialog box

If a disabled BREAKPOINT address is specified as a stop address in the [Run Program] dialog box, the disabled BREAKPOINT will become enabled after the user program has stopped.

RENESAS

— [Breakpoints] window

During user program execution, it is impossible to jump from the BREAKPOINT to the source or address line on the [Source] or [Disassembly] window by using [Go to Source] in the popup menu displayed on the [Breakpoints] window.

23. Number of BREAKPOINT and [Stop At] Settings in the [Run...] Menu

The maximum number of BREAKPOINTs and [Stop At] settings allowed in the [Run...] menu is 255. Therefore, when 255 BREAKPOINTs are set, specification by [Stop At] in the [Run...] menu becomes invalid. Use the BREAKPOINTs and [Stop At] in the [Run...] menu with 255 or less total settings.

24. Note on RUN-TIME Display

The execution time of the user program displayed in the [Status] window may not be accurate since the timer in the host computer is used.

25. Note on Displaying COMMUNICATION TIMEOUT ERROR

If COMMUNICATION TIMEOUT ERROR is displayed, the emulator cannot communicate with the chip. Select [Initialize] from the [File] menu to initialize the emulator.

26. Note on Downloading Program

In the [Load Program] dialog box, which is opened when [Load Program…] is selected, the verify function is invalid. After downloading the program, perform verify in the [Verify S-Record File with Memory] dialog box, which is opened when [Verify] is selected from the [Memory] menu.

27. Note on [Fill Memory] Dialog Box

The start and end addresses can be specified in the [Fill Memory] dialog box. When the start address value is larger than the end address value, note that the addresses are filled from the end to start.

28. Note on Using Old Version of Windows® 95

In using the old released version of Windows® 95 (such as 4.00.950a), if [Options...] is selected from the [Setup] menu, an application error occurs and the HDI abnormally exits. This is due to the old version of COMCTL32.DLL in the System directory of the Windows directory. Download the update program of COMCTL32.DLL from the Microsoft® homepage for installation, or update the version of Windows® 95.

29. Support of Double Float Format

In the following memory operations, the double float format is not supported:

— [Fill Memory] dialog box
— [Search Memory] dialog box

RENESAS

— MEMORY_FILL command

The [Format] specification in the [Copy Memory] dialog box is ignored. Memory is copied in a byte unit.

— Double float display at little endian operation

30. Note on Continuous Step Execution

When the step is continuously executed by selecting [Step...] from the [Run] menu, do not use the BREAKPOINT because this will cause the HDI to abnormally operate.

31. Note on Using the [Run program] Dialog Box

When [Run...] is selected from the [Run] menu to specify the stop address, there is the following note:

— When the BREAKPOINT that has been set as Disable is specified as the stop address, note that the BREAKPOINT becomes Enable when the user program stops.

32. Memory Test Function

This product does not support the memory test function, which is used by selecting [test] from the [memory] menu.

RENESAS

SH7750 E10A Emulator User's Manual

Publication Date: 1st Edition, June 1999
2nd Edition, January 2001
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.