To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

To all our customers

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

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

Cautions

Keep safety first in your circuit designs!

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

Notes regarding these materials

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

User's Manual

SH7055MCM E10A Emulator HS7055KCM01HE

User's Manual

Renesas Microcomputer Development Environment System

Renesas Electronics
www.renesas.com

Rev.1.0 2000.09

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.

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

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 SH7055. 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.

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:

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.

MCU names:

This user’s manual uses SHxxxx as an example of the MCU 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.

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.**

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.

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, SH7055 E10A Emulator Specifications describes the features of the E10A emulator for each MCU. Read this section 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.

Related Manuals:

- Super HRISC engine Assembler User's Manual
- H Series Linkage Editor, Librarian, and Object Converter 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.

Contents

Section 1 Overview 1
1.1 Warnings 3
1.2 Environmental Conditions 4
1.3 Components 6

Section 2 Preparation before Use 7
2.1 Emulator Preparation 7
2.2 HDI Installation 8
2.2.1 Installing the HDI 8
2.3 Connecting the Host Computer with the Card Emulator 17
2.4 Connecting the Card Emulator with the User System 18
2.5 System Check 21
2.6 Ending the HDI 27
2.7 Uninstalling the HDI 28

Section 3 Tutorial 31
3.1 Introduction 31
3.2 Running the HDI 32
3.3 [HDI] Window 33
3.4 Setting up the Emulator 34
3.5 Setting the [Configuration] Dialog Box 34
3.6 Downloading the Tutorial Program 36
3.6.1 Downloading the Tutorial Program 36
3.6.2 Displaying the Source Program 37
3.7 Setting the Software Breakpoint 39
3.8 Setting Registers 40
3.9 Executing the Program 42
3.10 Reviewing Breakpoints 45
3.11 Viewing Memory 46
3.12 Watching Variables 47
3.13 Stepping Through a Program 50
3.13.1 Executing [Step In] Command 51
3.13.2 Executing [Step Out] Command 52
3.13.3 Executing [Step Over] Command 54
3.14 Displaying Local Variables 56
3.15 Break Function 57
3.15.1 Software Break Function 57
3.16 Hardware Break Function 64
3.16.1 Setting the Sequential Break Condition 72

3.17 Trace Functions 82
3.17.1 Internal Trace Function 84
3.17.2 AUD Trace Function 86
3.17.3 VP_MAP Translation 87
3.18 What Next? 90

Section 4 Descriptions of Windows 91
4.1 HDI Windows 91
4.2 Descriptions of Each Window 94
4.2.1 [Configuration] Dialog Box 94
4.2.2 [E10A Driver Details] Dialog Box 99
4.2.3 [Breakpoints] Window 100
4.2.4 [Break] Dialog Box 103
4.2.5 [Break Point] Dialog Box 109
4.2.6 [Break Condition] Dialog Box 111
4.2.7 [Break Condition] Dialog Box Pages 113
4.2.8 [Trace] Window 125
4.2.9 [Trace Acquisition] Dialog Box 127
4.2.10 [System Status] Window 130

Section 5 Command-line Functions 133
5.1 Table and Symbol Description 133
5.1.1 Format 133
5.1.2 Parameter Input 133
5.1.3 Examples 134
5.1.4 Related Items 134
5.2 Command Descriptions 135
5.2.1 AUD_CLOCK:AUCL 137
5.2.2 AUD_MODE:AUM 139
5.2.3 AUD_TRACE:AUT 141
5.2.4 BREAKCONDITION_CLEAR: BCC 143
5.2.5 BREAKCONDITION_DISPLAY: BCD 144
5.2.6 BREAKCONDITION_ENABLE: BCE 145
5.2.7 BREAKCONDITION_SET: BCS 146
5.2.8 BREAKPOINT: BP 150
5.2.9 BREAKPOINT_CLEAR: BC 152
5.2.10 BREAKPOINT_DISPLAY: BD 154
5.2.11 BREAKPOINT_ENABLE: BE 155
5.2.12 DEVICE_TYPE: DE 157
5.2.13 GO_OPTION: GP 158
5.2.14 JTAG_CLOCK: JCK 160
5.2.15 MEMORYAREA_SET: MAS 162
5.2.16 REFRESH: RF 164

5.2.17 RESTART: RST 165
5.2.18 STATUS: STS 166
5.2.19 STEP_INTERRUPT: SI 167
5.2.20 TRACE_DISPLAY: TD 168
5.2.21 UBC_MODE:UM 170
5.2.22 VPMAP_CLEAR: VC 171
5.2.23 VPMAP_DISPLAY: VD 172
5.2.24 VPMAP_ENABLE: VE 173
5.2.25 VPMAP_SET: VS 174

Section 6 SH7055 E10A Emulator Specifications 175
6.1 Overview of the SH7055 E10A Emulator 175
6.2 Pin Arrangement of the Hitachi-UDI Port Connector 177
6.3 User System Interface Circuit 180
6.4 Differences between the SH7055 F-ZTAT™ and the Emulator 186
6.5 The SH7055 E10A Emulator Functions 188
6.5.1 Emulator Driver Selection 188
6.5.2 Hardware Break Functions 188
6.5.3 Notes on Setting the [Breakpoint] Dialog Box 191
6.5.4 Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK) 191
6.5.5 AUD Functions 192
6.5.6 Notes on Displaying the [Trace] Window 193
6.5.7 Notes on Executing the User Program 194
6.5.8 Notes on Setting the UBC_MODE Command 194
6.5.9 Notes on HDI 194

Section 7 Activation Sequence and Specific Commands of the SH7055 E10A Emulator 201
7.1 Activation Sequence 201
7.2 RAM Monitor Commands 203
7.2.1 RAM_R: RR 203
7.2.2 RAM_W: RW 205

Figures

Figure 1.1 System Configuration with the Emulator (PCMCIA Card Emulator Used) 1
Figure 1.2 System Configuration with the Emulator (PCI Card Emulator Used) 2
Figure 2.1 Emulator Preparation Flow Chart 7
Figure 2.2 [setup.EXE] Icon 8
Figure 2.3 [Welcome!] Dialog Box 8
Figure 2.4 [Read Me] Dialog Box (Example) 9
Figure 2.5 [Select Destination Directory] Dialog Box 10
Figure 2.6 [Install] Dialog Box 11
Figure 2.7 [Make Backups?] Dialog Box 11
Figure 2.8 [Select Backup Directory] Dialog Box 12
Figure 2.9 [Install] Dialog Box 12
Figure 2.10 [Installing] Dialog Box 13
Figure 2.11 [Select Driver Type] Dialog Box 13
Figure 2.12 [Installing] Dialog Box 14
Figure 2.13 [Select Program Manager Group] Dialog Box 14
Figure 2.14 [HDI for E10A Setup] Dialog Box 15
Figure 2.15 [HDI] Program Group (When Windows® 95 is Used) 15
Figure 2.16 [HDI] Program Group (When Windows NT® is Used) 16
Figure 2.17 Inserting the PCMCIA Card Emulator into the Host Computer 17
Figure 2.18 Inserting the PCI Card Emulator into the Host Computer 17
Figure 2.19 Connecting the User System Interface Cable to the User System when the 14-pin Straight Type Connector is Used 19
Figure 2.20 User System Example 20
Figure 2.21 [Start] Menu 21
Figure 2.22 [Select Session] Dialog Box 22
Figure 2.23 [E10A Driver Details] Dialog Box 23
Figure 2.24 Dialog Box of the RESET Signal Input Request Message 24
Figure 2.25 [HDI] Status Bar 24
Figure 2.26 [JTAG Connector Disconnected] Dialog Box 24
Figure 2.27 [Can not find /RESET signal] Dialog Box 25
Figure 2.28 [Check the connection] Dialog Box 25
Figure 2.29 [COMMUNICATION TIMEOUT ERROR] Dialog Box 25
Figure 2.30 [INVALID ASERAM FIRMWARE!] Dialog Box 26
Figure 2.31 [Unable to restore the previous driver settings] Dialog Box 26
Figure 2.32 [Save session] Dialog Box 27
Figure 2.33 [Start] Menu 28
Figure 2.34 [Uninstall Hitachi Debugging Interface for SHxxxx E10A Emulator] Dialog Box 29
Figure 2.35 [Uninstall Hitachi Debugging Interface item: Install File item: Install File for SHxxxx E10A Emulator] Dialog Box 30
Figure 3.1 [Start] Menu 32

Figure 3.2 [HDI] Window....33
Figure 3.3 [Configuration] Dialog Box....34
Figure 3.4 [Load Object File] Dialog Box....36
Figure 3.5 [HDI] Dialog Box....36
Figure 3.6 [Open] Dialog Box....37
Figure 3.7 [Program] Window (Displaying the Source Program)....38
Figure 3.8 [Program] Window (Setting a Software Breakpoint)....39
Figure 3.9 [Registers] Window....40
Figure 3.10 [Register] Dialog Box (PC)....41
Figure 3.11 [Register] Dialog Box (R15)....41
Figure 3.12 [Go] Button....42
Figure 3.13 [Program] Window (Break Status)....42
Figure 3.14 [System Status] Window....43
Figure 3.15 [Breakpoints] Window....45
Figure 3.16 [Open Memory Window] Dialog Box....46
Figure 3.17 [Memory] Window....46
Figure 3.18 [Instant Watch] Dialog Box....47
Figure 3.19 [Watch] Window (Displaying the Array)....47
Figure 3.20 [Add Watch] Dialog Box....48
Figure 3.21 [Watch] Window (Displaying the Variable)....48
Figure 3.22 [Watch] Window (Displaying Array Elements)....49
Figure 3.23 [Program] Window (Step Execution)....50
Figure 3.24 [Step In] Button....51
Figure 3.25 [Program] Window (Step In)....51
Figure 3.26 [Step Out] Button....52
Figure 3.27 [Program] Window (Step Out)....52
Figure 3.28 [Program] Window (Step In → Step In)....53
Figure 3.29 [Program] Window (Before Step Over Execution)....54
Figure 3.30 [Step Over] Button....54
Figure 3.31 [Program] Window (Step Over)....55
Figure 3.32 [Locals] Window....56
Figure 3.33 [Breakpoints] Window (Before Software Breakpoint Setting)....57
Figure 3.34 [Point] Page ([Break] Dialog Box)....58
Figure 3.35 [Break Point] Dialog Box....59
Figure 3.36 [Point] Page ([Break] Dialog Box) (After Software Breakpoint Setting)....60
Figure 3.37 [Breakpoints] Window (Software Breakpoint Setting)....61
Figure 3.38 [Program] Window at Execution Stop (Software Break)....62
Figure 3.39 Displayed Contents of the [System Status] Window (Software Break)....63
Figure 3.40 [Breakpoints] Window (Before Hardware Break Condition Setting)....64
Figure 3.41 [Condition] Page ([Break] Dialog Box)....65
Figure 3.42 [Address] Page ([Break Condition 1] Dialog Box)....66
Figure 3.43 [Bus State] Page ([Break Condition 1] Dialog Box)....67
Figure 3.44 [Break] Dialog Box (After Hardware Break Condition Setting)....68

Figure 3.45 [Breakpoints] Window ([Break Condition 1] Setting)....69
Figure 3.46 [Program] Window at Execution Stop (Break Condition 1)....70
Figure 3.47 Displayed Contents of the [System Status] Window (Break Condition 1)....71
Figure 3.48 [Configuration] Dialog Box (Sequential Break Setting)....73
Figure 3.49 [Breakpoints] Window (Before Sequential Break Condition Setting)....74
Figure 3.50 [Condition] Page ([Break] Dialog Box)....75
Figure 3.51 [Break Condition 2] Dialog Box (Condition 2 [Address] Page)....76
Figure 3.52 [Break Condition 2] Dialog Box (Condition 2 [Bus State] Page)....77
Figure 3.53 [Break] Dialog Box (After [Break Condition 2] Condition Setting)....78
Figure 3.54 [Breakpoints] Window (After Sequential Break Condition Setting)....79
Figure 3.55 [Program] Window at Execution Stop (Sequential Break)....80
Figure 3.56 Displayed Contents of the [System Status] Window (Sequential Break)....81
Figure 3.57 [Trace mode] Window....84
Figure 3.58 [Trace] Window....85
Figure 3.59 [Trace mode] Window....86
Figure 3.60 Address Translation according to VP_MAP Tables....88
Figure 4.1 [Configuration] Dialog Box....94
Figure 4.2 [General] Page ([Configuration] Dialog Box)....96
Figure 4.3 Warning Message Box....98
Figure 4.4 [E10A Driver Details] Dialog Box....99
Figure 4.5 [Breakpoints] Window....100
Figure 4.6 [Break] Dialog Box....103
Figure 4.7 [Point] Page ([Break] Dialog Box)....105
Figure 4.8 [Condition] Page ([Break] Dialog Box)....107
Figure 4.9 [Break Point] Dialog Box....109
Figure 4.10 [Break Condition] Dialog Box....111
Figure 4.11 [Address] Page ([Break Condition 1] Dialog Box)....115
Figure 4.12 [Data] Page ([Break Condition 1] Dialog Box)....117
Figure 4.13 [ASID] Page ([Break Condition] Dialog Box)....119
Figure 4.14 [Bus State] Page ([Break Condition] Dialog Box)....120
Figure 4.15 [Count] Page ([Break Condition] Dialog Box)....122
Figure 4.16 [General] Page ([Break Condition] Dialog Box)....123
Figure 4.17 [Trace] Window....125
Figure 4.18 [Trace mode] Page ([Trace Acquisition] Dialog Box)....128
Figure 4.19 [System Status] Window....130
Figure 5.1 TLB Error Message Dialog....168
Figure 6.1 Pin Arrangement of the Hitachi-UDI Port Connector (14 Pins)....177
Figure 6.2 Pin Arrangement of the Hitachi-UDI Port Connector (36 Pins)....178
Figure 6.3 Circuit between User Interface Connector (36 Pins) and SH7055 MCM....179
Figure 6.4 User System Interface Circuit (HS0005KCM03H)....180
Figure 6.5 User System Interface Circuit of the Hitachi-UDI Pin (HS0005KCM04H)....181
Figure 6.6 User System Interface Circuit of the AUD Pin (HS0005KCM04H)....182
Figure 6.7 User System Interface Circuit (HS0005KCI03H)....183

Figure 6.8 User System Interface Circuit of the Hitachi-UDI Pin (HS0005KCI04H).....184
Figure 6.9 User System Interface Circuit of the AUD Pin (HS0005KCI04H) 185

Tables

Table 1.1 Environmental Conditions 4
Table 1.2 Operating Environments 5
Table 2.1 Recommended Hitachi-UDI Port Connector 18
Table 3.1 Tutorial Program Configuration 31
Table 3.2 Setting the [Configuration] Dialog Box 35
Table 3.3 Contents of the [System Status] Window 44
Table 3.4 Step Option 50
Table 3.5 Sequential Break Conditions 72
Table 3.6 AUD Trace Functions 83
Table 3.7 Address Translation Tables 89
Table 4.1 HDI Window Menus and Related Manual Entries 91
Table 4.2 [Configuration] Dialog Box Page 95
Table 4.3 [General] Page Options 97
Table 4.4 [Breakpoints] Window Display Items 101
Table 4.5 [Breakpoints] Window Pop-up Menu Operation 102
Table 4.6 [Break] Dialog Box Pages 104
Table 4.7 [Point] Page Options 106
Table 4.8 [Condition] Page Options 108
Table 4.9 [Address] Page Options 110
Table 4.10 Setting Conditions in [Break Condition] Dialog Boxes 113
Table 4.11 [Break Condition] Dialog Box Pages 114
Table 4.12 [Address] Page Options 116
Table 4.13 Address Options 116
Table 4.14 [Data] Page Options 118
Table 4.15 [ASID] Page Options 119
Table 4.16 [Bus State] Page Options 121
Table 4.17 [Count] Page Options 122
Table 4.18 [General] Page Options 124
Table 4.19 [Trace] Window Display Items 126
Table 4.20 [Trace Acquisition] Page Options 127
Table 4.21 [Trace mode] Page Options 129
Table 4.22 [System Status] Window Display Items 131
Table 5.1 E10A HDI Commands 135
Table 5.2 AUD_CLOCK Command Parameter 137
Table 5.3 AUD_MODE Command Parameter 139
Table 5.4 AUD_TRACE Command Parameter 141
Table 5.5 BREAKCONDITION_CLEAR Command Parameter 143
Table 5.6 BREAKCONDITION_DISPLAY Command Parameter 144
Table 5.7 BREAKCONDITION_ENABLE Command Parameters 145
Table 5.8 BREAKCONDITION_SET Command Parameters 147
Table 5.9 BREAKPOINT Command Parameters 150

Table 5.10 BREAKPOINT_CLEAR Command Parameters 152
Table 5.11 BREAKPOINT_DISPLAY Command Parameter 154
Table 5.12 BREAKPOINT_ENABLE Command Parameters 155
Table 5.13 DEVICE_TYPE Command Parameter 157
Table 5.14 GO_OPTION Command Parameter 158
Table 5.15 JTAG_CLOCK Command Parameter 160
Table 5.16 MEMORYAREA_SET Command Parameters 162
Table 5.17 REFRESH Command Parameter 164
Table 5.18 RESTART Command Parameter 165
Table 5.19 STATUS Command Parameter 166
Table 5.20 STEP_INTERRUPT Command Parameter 167
Table 5.21 TRACE_DISPLAY Command Parameter 168
Table 5.22 UBC_MODE Command Parameter 170
Table 5.23 VPMAP_CLEAR Command Parameter 171
Table 5.24 VPMAP_DISPLAY Command Parameter 172
Table 5.25 VPMAP_ENABLE Command Parameter 173
Table 5.26 VPMAP_SET Command Parameters 174
Table 6.1 Components of the Emulator (HS7055KCM01H, HS7055KCM02H, HS7055KCI01H, or HS7055KCI02H) 176
Table 6.2 Register Initial Values at Emulator Power-On 186
Table 6.3 Type Name and Driver 188
Table 6.4 Hardware Break Condition Specification Items 189
Table 6.5 Conditions Set in the [Break Condition] Dialog Box 189
Table 6.6 Conditions Set with the BREAKCONDITION_SET Command 189
Table 6.7 AUD Functions 192
Table 6.8 Type Name and AUD Function 193
Table 6.9 Notes on Executing the User Program 194
Table 6.10 Watchdog Timer and MSTCR Registers 198
Table 7.1 Selection of the Operation Mode 202
Table 7.2 RAM_R Command Parameter 203
Table 7.3 RAM_W Command Parameter 205

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.

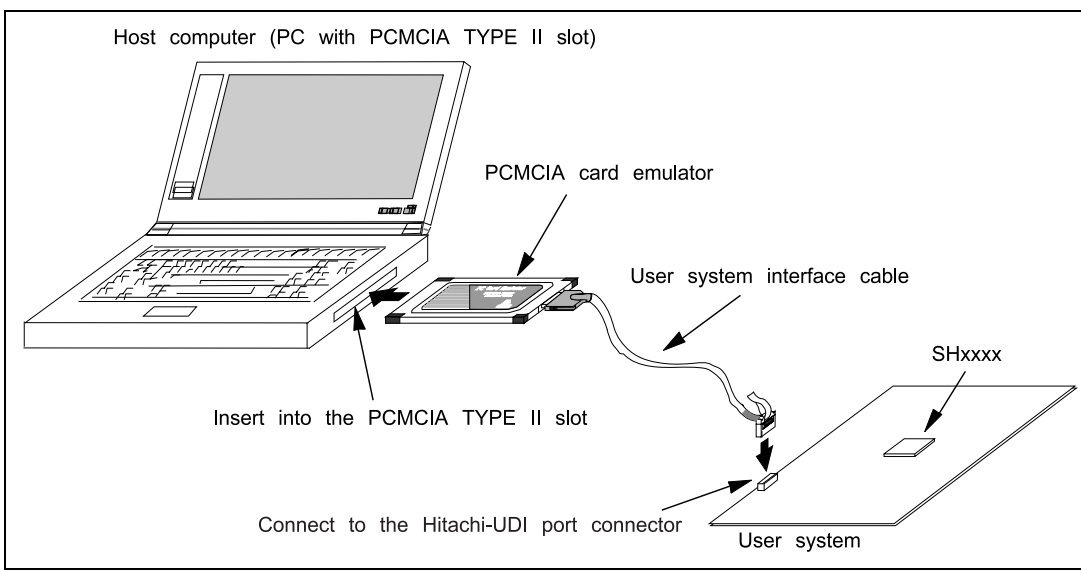


Figure 1.1 System Configuration with the Emulator (PCMCIA Card Emulator Used)

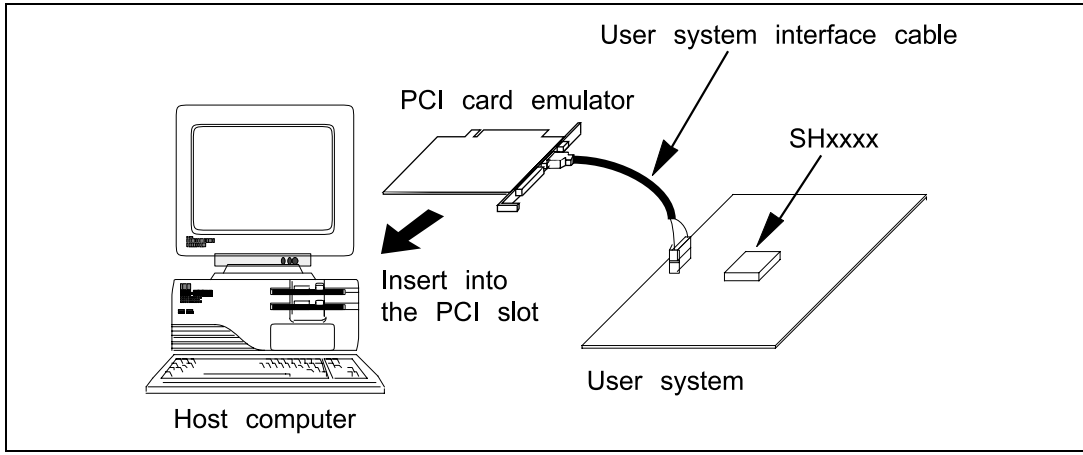


Figure 1.2 System Configuration with the Emulator (PCI Card Emulator Used)

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, and Microsoft® Windows NT® operating systems enable user program debugging using a pointing device such as a mouse. The HDI enables high-speed down-loading 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.

Note: The AUD is an abbreviation of the advanced user debugger.

1.1 Warnings

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.

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.

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 damage the user system and the emulator product. The USER PROGRAM will be LOST.

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	There must be no corrosive gases present

Table 1.2 lists the acceptable operating environments.

Table 1.2 Operating Environments

Item	Description
Host computer	Built-in Pentium or higher-performance CPU (166 MHz or higher recommended); IBM PC or compatible machine with the PCMCIA TYPE II slot or the PCI slot.
OS	Windows® 95, Windows® 98, 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, 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: 480 mA (max)
CD-ROM drive	Required to install the emulator or refer to the emulator user's manual.

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.

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:

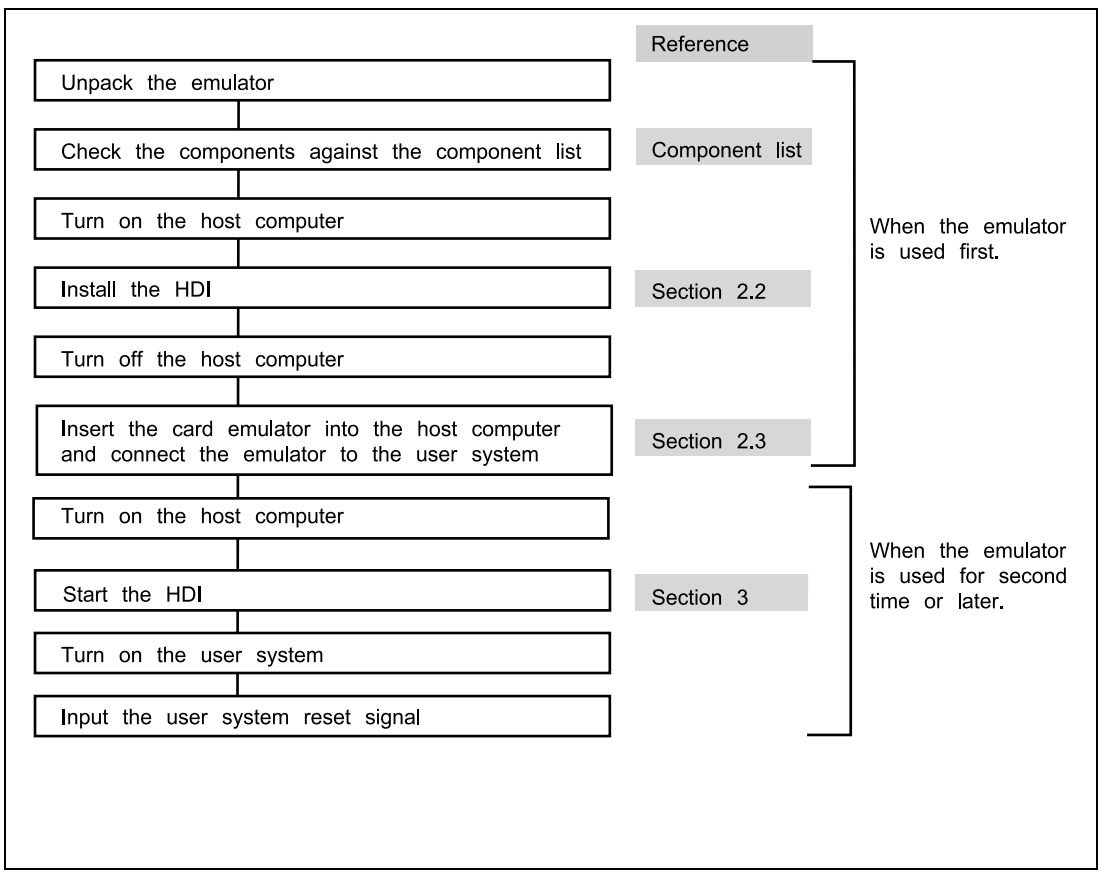


Figure 2.1 Emulator Preparation Flow Chart

2.2 HDI Installation

2.2.1 Installing the HDI

This section describes an example of installing the HDI on the IBM PC.

Start [setup.EXE] in the \SETUP directory of the CD-R.

Figure 2.2 [setup.EXE] Icon

- This runs the HDI installer, and the following [Welcome!] dialog box will be displayed.

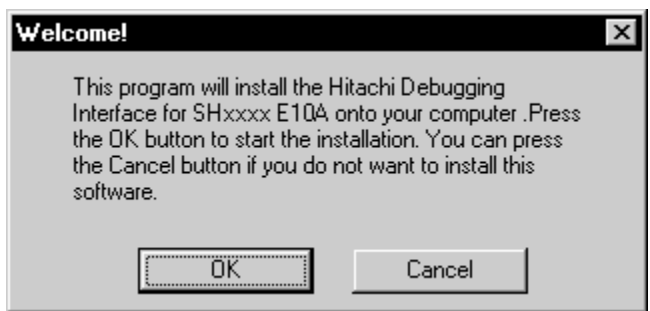


Figure 2.3 [Welcome!] Dialog Box

- Click the [OK] button to proceed with the installation.
- The [Read Me] dialog box is then displayed. Click the [OK] button to proceed.

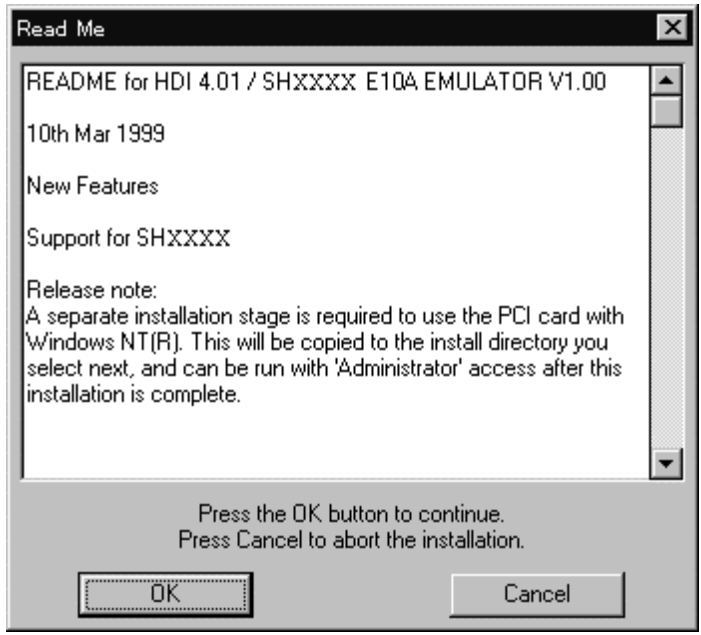


Figure 2.4 [Read Me] Dialog Box (Example)

- Then the [Select Destination Directory] dialog box is displayed. Select a directory for installing the HDI, and click the [OK] button. When installing the HDI into the default directory, just click the [OK] button.

  When the HEW (Hitachi Embedded Workshop) is installed in the host computer, the default directory is one that the HEW has been installed in. If the directory is changed and the emulator is installed, the HEW cannot automatically register the emulator. Figures 2.5 through 2.16 show examples of when the default directory is set as HDI_CE by using Windows® 95.

Figure 2.5 [Select Destination Directory] Dialog Box

- When the specified directory name already exists, the [Install] dialog box is displayed. When installing the HDI into the existing directory, click the [Yes] button. If the user wants to change the directory, click the [No] button. The [Select Destination Directory] dialog box then allows the user to select another directory.

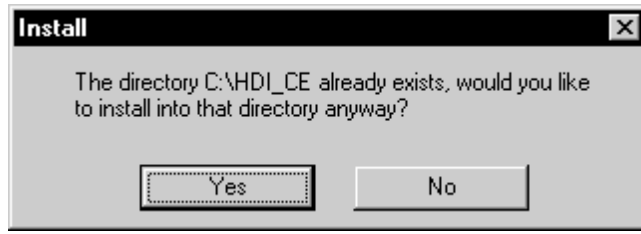


Figure 2.6 [Install] Dialog Box

- Clicking the [Yes] button in the [Install] dialog box displays the [Make Backups?] dialog box to ask the user whether backups should be made of the files replaced by the installation. Click the [Yes] button to save any files or the [No] button if the user does not want to make a backup.

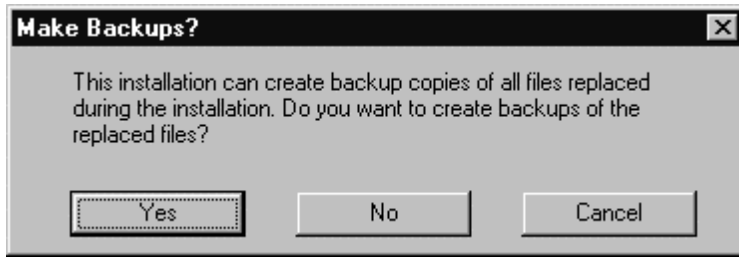


Figure 2.7 [Make Backups?] Dialog Box

- When the user selects the [Yes] button in the [Make Backups?] dialog box, the [Select Backup Directory] dialog box is displayed. Specify the backup file name then click the [OK] button to proceed. To save into the default directory, just click the [OK] button.

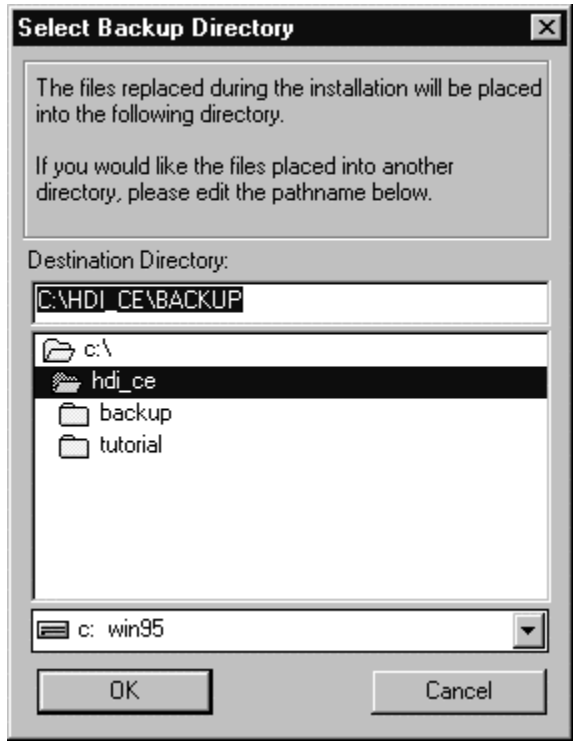


Figure 2.8 [Select Backup Directory] Dialog Box

- When the specified directory name already exists, the [Install] dialog box is displayed. When installing the HDI into the existing directory, click the [Yes] button. If the user wants to change the directory, click the [No] button. The [Select Backup Directory] dialog box then allows the user to select another directory.

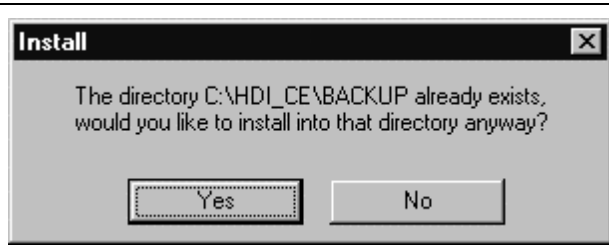


Figure 2.9 [Install] Dialog Box

- The installer then installs the HDI files into the specified directory.

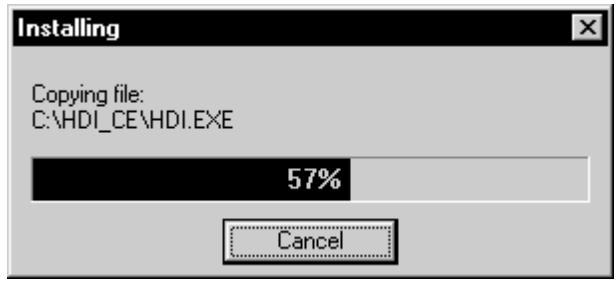


Figure 2.10 [Installing] Dialog Box

- During the installation, the [Select Driver Type] dialog box is displayed. To select whether the PCI card emulator or the PCMCIA card emulator is used, the following message is displayed. Then click the [OK] button. (This message is not displayed when Windows NT® is used.)

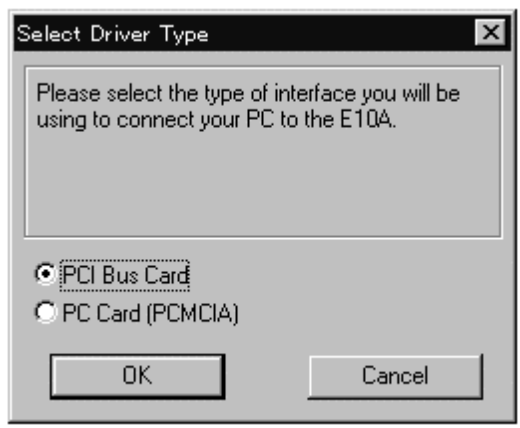


Figure 2.11 [Select Driver Type] Dialog Box

- The installer then installs the HDI files into the specified directory.

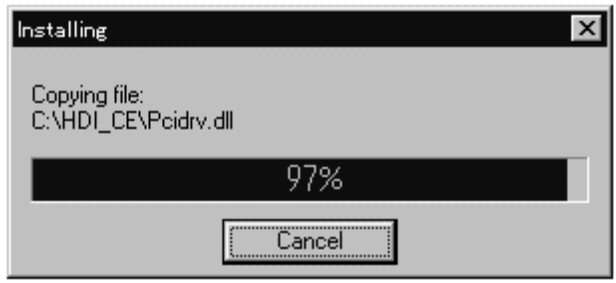


Figure 2.12 [Installing] Dialog Box

- The [Select Program Manager Group] dialog box allows the user to specify the program group name for the HDI icons. To specify `HDI` (the default group name) for a program group name, just click the [OK] button.

Figure 2.13 [Select Program Manager Group] Dialog Box

- When Windows NT® is used, the [HDI for E10A Setup] dialog box is displayed. Click the [OK] button.

Figure 2.14 [HDI for E10A Setup] Dialog Box

- Specifying the program group name enables the installer to create the following icons in the program group the user specified.

Figure 2.15 [HDI] Program Group (When Windows• 95 is Used)

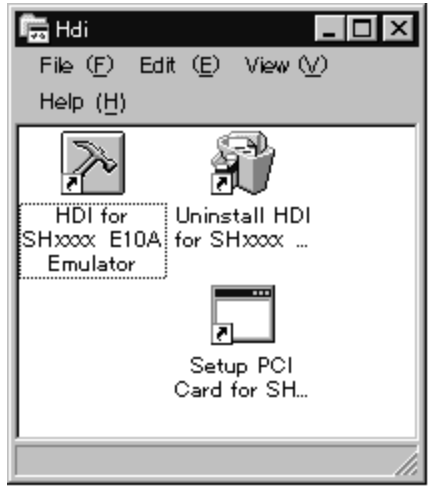


Figure 2.16 [HDI] Program Group (When Windows NT® is Used)

- These icons represent the following functions:

  [HDI for SHxxxx E10A Emulator] executes the HDI program.

  [Uninstall HDI for SHxxxx E10A Emulator] deletes the HDI software and the associated files when the HDI is uninstalled.

  [Setup PCI Card for SHxxxx E10A Emulator] sets up the PCI driver when Windows NT® is used.

Notes: 1. When Windows NT® is used, be sure to insert the card emulator and execute "Start Menu - Program - HDI - Setup PCI Card for SHxxxx E10A Emulator" after installation. If not executed, the PCI driver is incorrectly set.

2. When the PCMCIA card emulator is used, a message for installing the driver is displayed if the card emulator is being inserted for the first time. The driver is provided in the /SETUP directory on the CD-R. Install it according to the screen instructions.

2.3 Connecting the Host Computer with the Card Emulator

Insert the card emulator into the PCMCIA TYPE II slot or the PCI slot of the host computer (figures 2.17 and 2.18).

Note: Be sure to install the HDI before the card emulator is inserted.

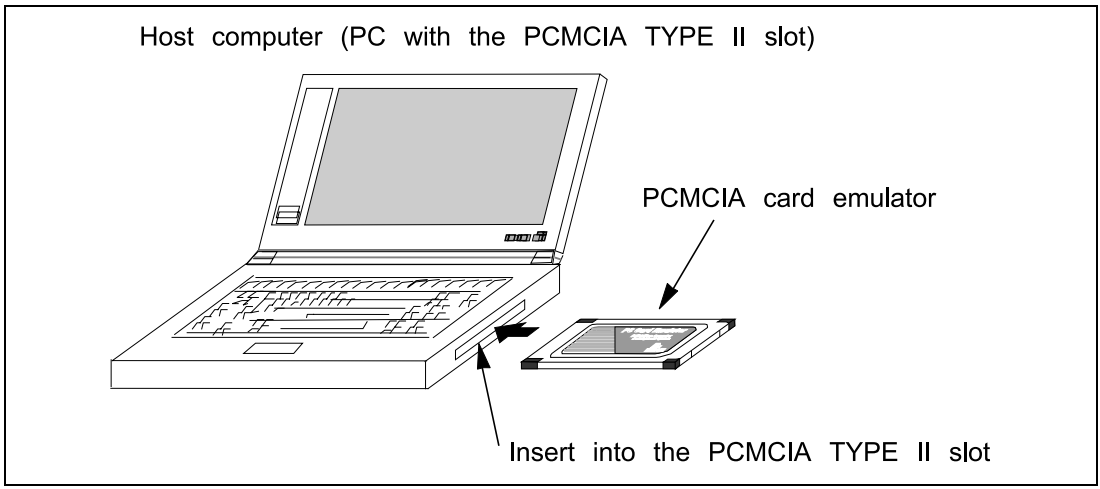


Figure 2.17 Inserting the PCMCIA Card Emulator into the Host Computer

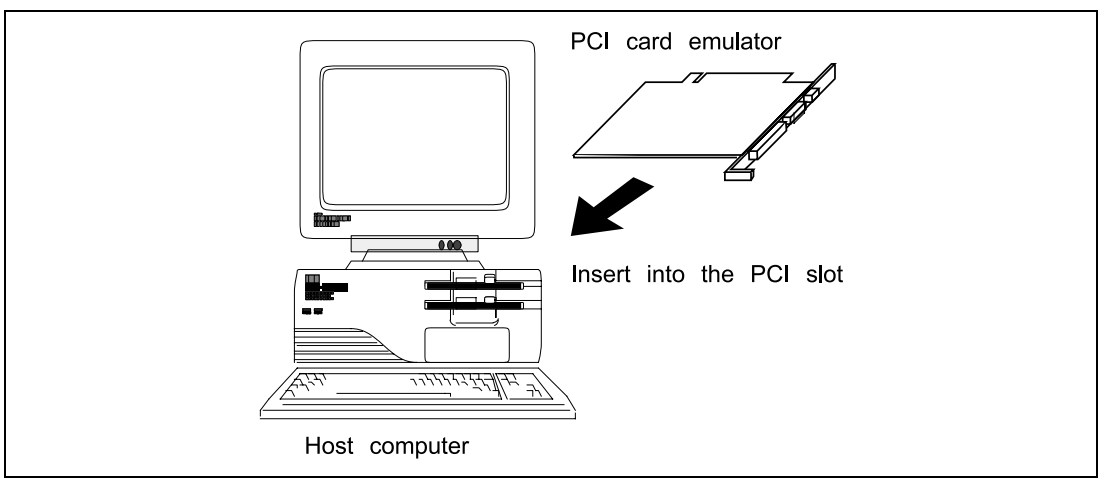


Figure 2.18 Inserting the PCI Card Emulator into the Host Computer

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.

Note: When installing the PCI card emulator, note the following:

1. Turn off the host computer.
2. Insert the emulator into the PCI slot in parallel.
3. Screw in the emulator after confirming the connector and cable positions.

2.4 Connecting the Card Emulator with 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	7614-6002, 2514-6002	Sumitomo 3M Limited	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 any components under the Hitachi-UDI 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 2.19 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.

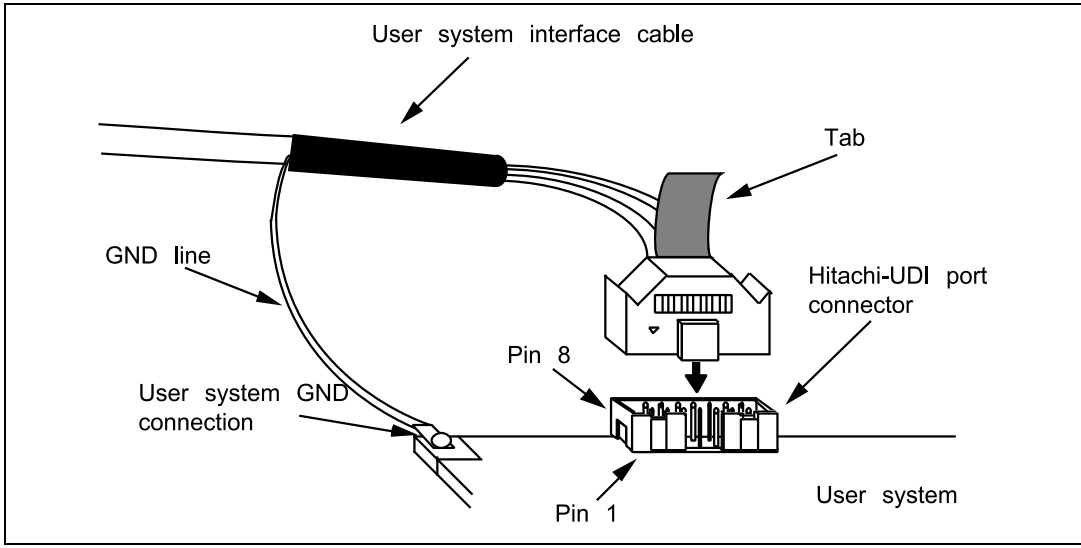


Figure 2.19 Connecting the User System Interface Cable to the User System when the 14-pin Straight Type Connector is Used

Notes: 1. To connect the signals output from the Hitachi-UDI port connector, refer to the MCU 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 MCUs 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 MCU.

5. When developing user systems, do not connect the TDI and TDO signals of the MCU to the boundary scan loop, or separate them by using a switch (figure 2.20).

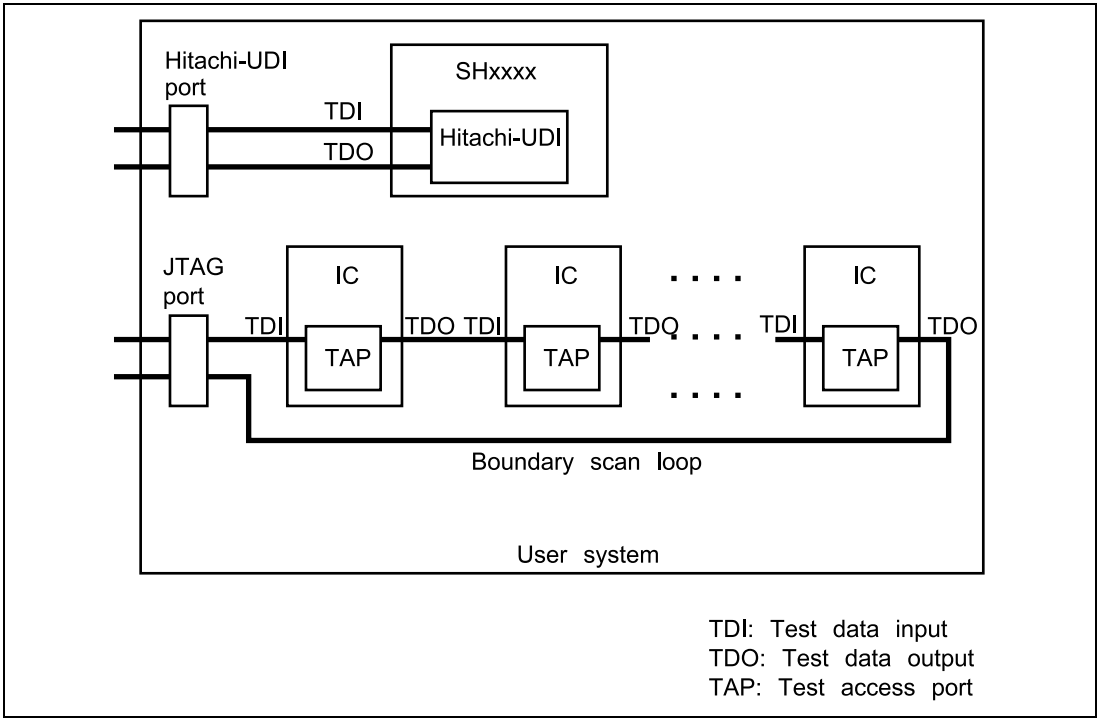


Figure 2.20 User System Example

2.5 System Check

When the HDI program is executed, check that the emulator operates correctly according to the following procedure:

1. Check that the card emulator is inserted into the host computer.
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. Power on the host computer and select [HDI for SHxxxx E10A Emulator] from the [Start] menu.

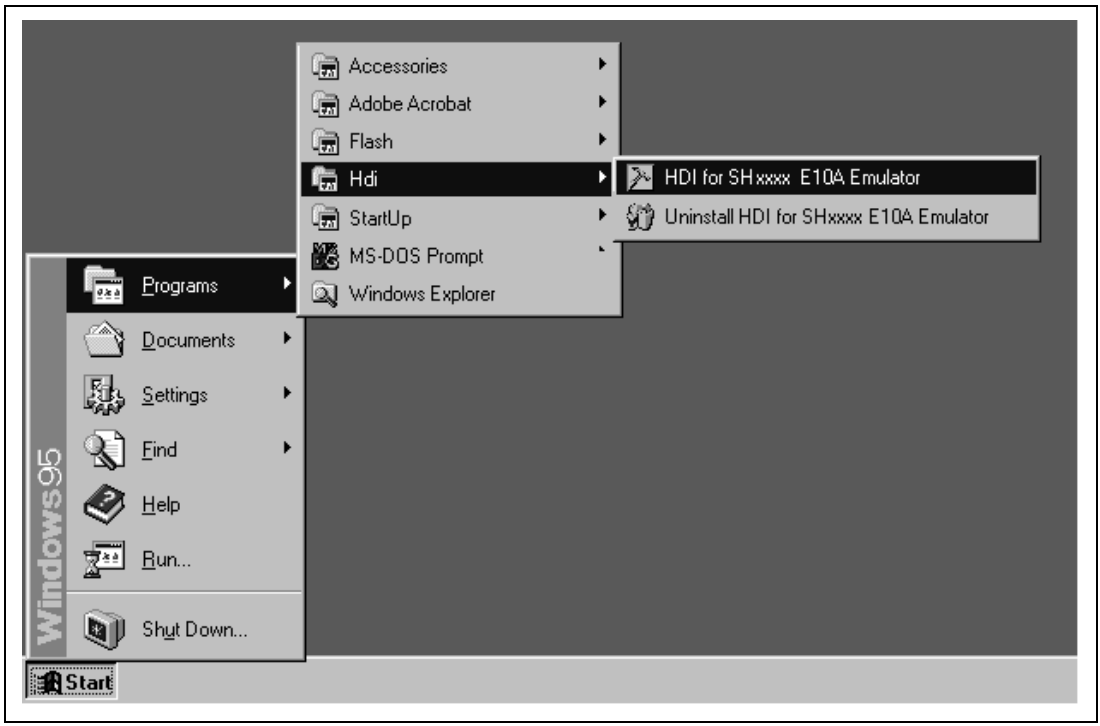


Figure 2.21 [Start] Menu

5. Select the setting to be used.

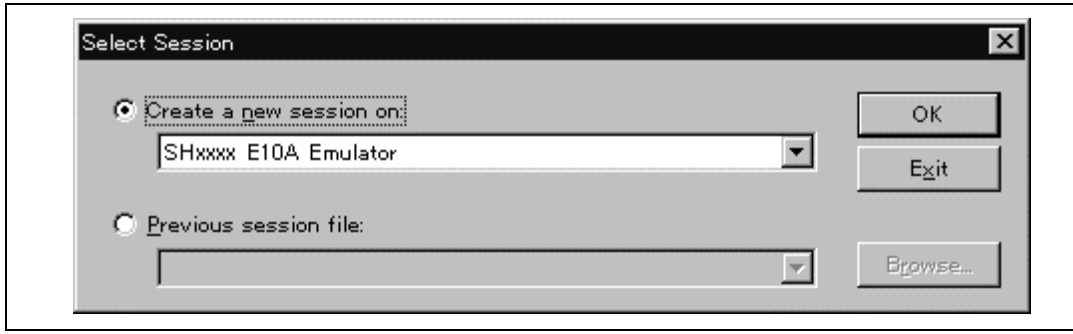


Figure 2.22 [Select Session] Dialog Box

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 MCUs.)

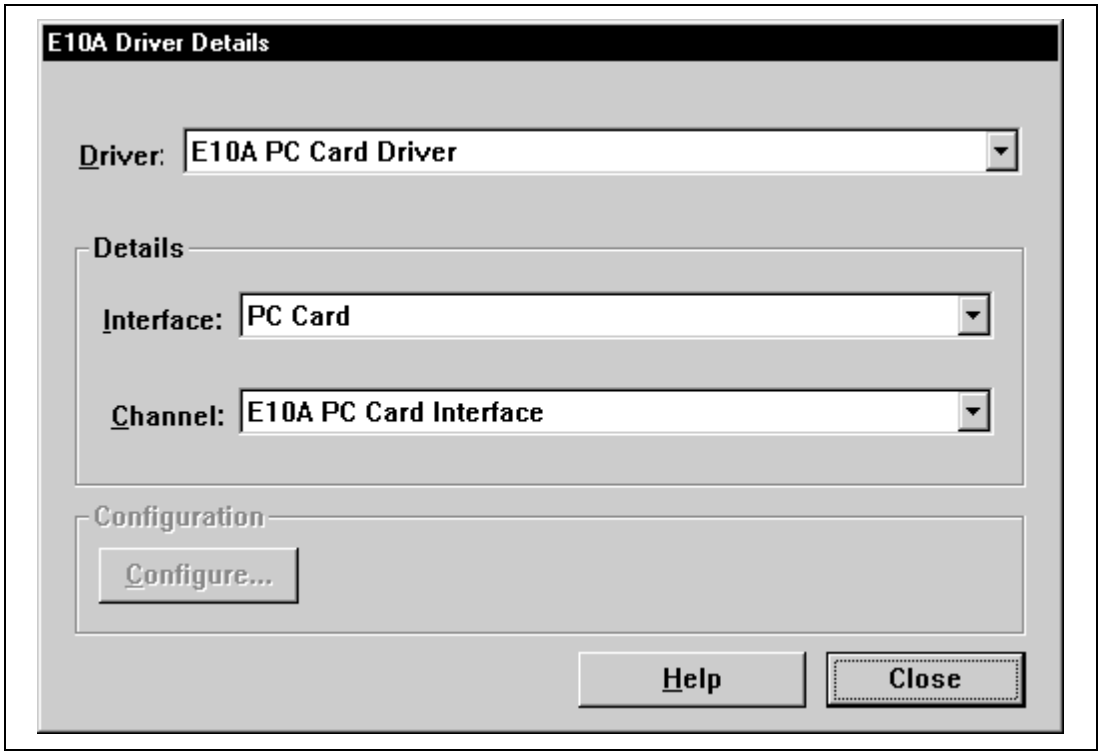


Figure 2.23 [E10A Driver Details] Dialog Box

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

— Click the [Close] button.

7. The HDI window is displayed, and the dialog box is displayed as shown in figure 2.24.

Figure 2.24 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.

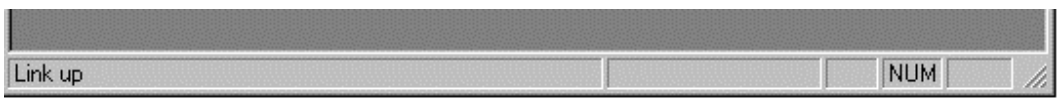


Figure 2.25 [HDI] Status Bar

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 with the /SETUP directory in the CD-R according to the screen instructions.

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 2.26 [JTAG Connector Disconnected] Dialog Box

3. **If the emulator is not initiated, the following dialog boxes shown in figures 2.27 through 2.30 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 MCU. Check the input circuits for the power of the user system and the reset pin.

Figure 2.27 [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.

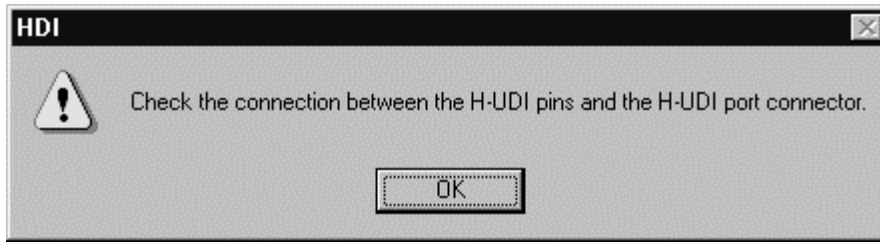


Figure 2.28 [Check the connection] Dialog Box

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

Figure 2.29 [COMMUNICATION TIMEOUT ERROR] Dialog Box

Figure 2.30 [INVALID ASERAM FIRMWARE!] Dialog Box

4. **If the driver is not correctly connected, the following dialog box will be displayed.**

Figure 2.31 [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.

2.6 Ending the HDI

Power off the emulator by using the following procedure:

1. Select [Exit] from the [File] menu to end the HDI. 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.

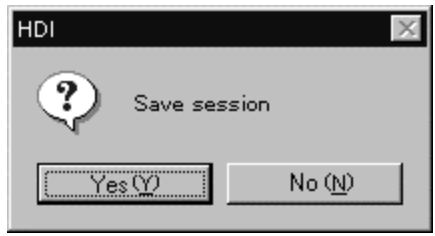


Figure 2.32 [Save session] Dialog Box

2. Power off the user system.

2.7 Uninstalling the HDI

Uninstallation of the HDI is described below.

1. First, select [Uninstall HDI for SHxxxx E10A Emulator] from the [Start] menu.

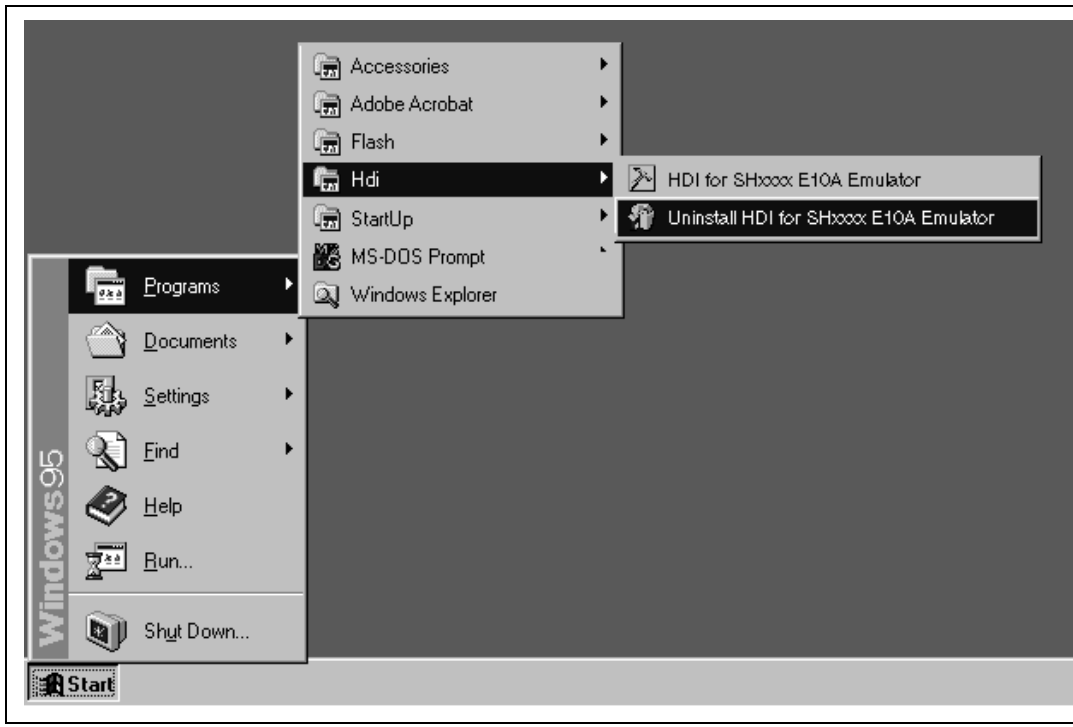


Figure 2.33 [Start] Menu

The [Uninstall Hitachi Debugging Interface for SHxxxx E10A Emulator] dialog box is displayed.

Figure 2.34 [Uninstall Hitachi Debugging Interface for SHxxxx E10A Emulator] Dialog Box

The function of each button is as follows:

Automatic: The HDI is automatically removed.

Custom: The files to be removed can be selected.

2. Always select the [Automatic] button and click [Next] to display the [Uninstall Hitachi Debugging Interface item: Install File item: Install File for SHxxxx E10A Emulator] dialog box.

Figure 2.35 [Uninstall Hitachi Debugging Interface item: Install File item: Install File for SHxxxx E10A Emulator] Dialog Box

3. Click the [Finish] button to uninstall the HDI.

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 tutorial program is included in the `sort.c` file. The compiled load module is provided in the SYSROF format and is included in the `sort.abs` file.

Table 3.1 lists the tutorial program configuration.

Table 3.1 Tutorial Program Configuration

Item	Contents
Tutorial file (load module)	c:\HDI_CE\tutorial\sort.abs
Tutorial file (source file)	c:\HDI_CE\tutorial\sort.c
Make file (DOS batch file)	c:\HDI_CE\tutorial\tutorial.bat
Subcommand file for linkage editor	c:\HDI_CE\tutorial\tutorial.sub

For the operating environment, use area 0 (CS0 space) and a memory bus width of 32 bits. The MMU function is not used.

Notes: 1. `sort.abs` operates in big endian. `sort.abs` must be recompiled to operate in little endian.

2. This section describes general usage examples of the emulator (SH7729 E10A emulator). For each product specifications, refer to section 6 or on-line help. The sample program is created by using SH C compiler V4.1B and H-series linkage editor V5.3.

3.2 Running the HDI

To run the HDI, select the [HDI for SHxxxx E10A Emulator] from the [Start] menu.

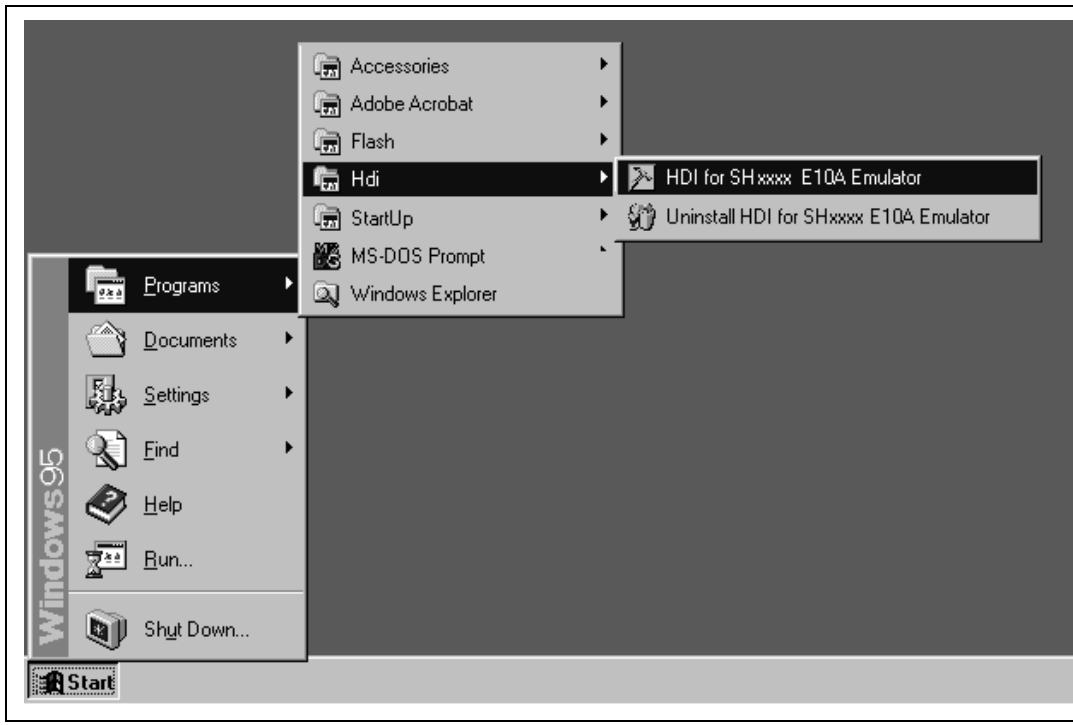


Figure 3.1 [Start] Menu

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

3.3 [HDI] Window

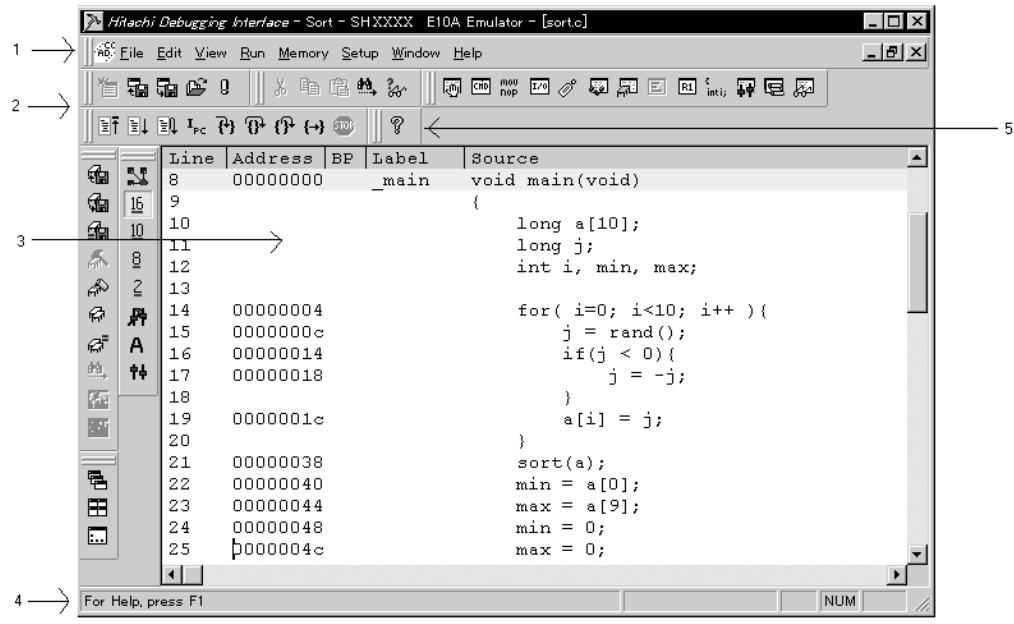


Figure 3.2 [HDI] Window

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. Program 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 on-line help about any features of the HDI user interface.

3.4 Setting up the Emulator

The following MCU conditions must be set up on the emulator before downloading the program:

- Device type
- Execution mode

The following describes how to set up the emulator for the tutorial programs.

3.5 Setting the [Configuration] Dialog Box

- Select [Configure Platform...] from the [Setup] menu to set configuration. The [Configuration] dialog box is displayed.

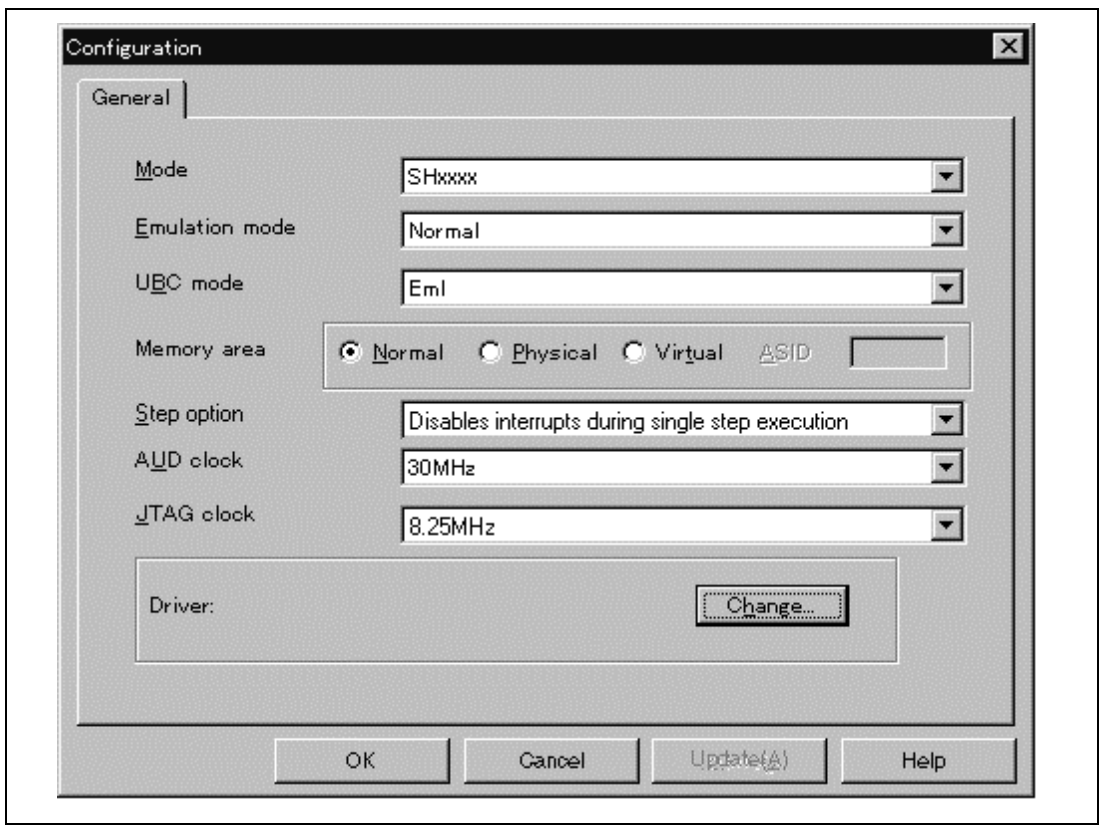


Figure 3.3 [Configuration] Dialog Box

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

Set options as follows:

Table 3.2 Setting the [Configuration] Dialog Box

Option	Value
Mode	SHxxxx (default)
Emulation mode	Normal (normal execution, default)
UBC mode	Eml (default)*
Memory area (address setting of memory space)	Normal (default)
Step option (interrupt setting during step execution)	Disables interrupts during single step execution (default)
AUD clock	30 MHz (default)
JTAG clock	8.25 MHz (default)

Note: When User is selected in the UBC mode, the break condition function cannot be used. Refer to section 6 for differences in emulator products.

- Click the [OK] button to set any changes in the configuration.

3.6 Downloading the Tutorial Program

3.6.1 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 the [Offset] edit box and [File name] list box as shown in figure 3.4 and click the [Open] button.

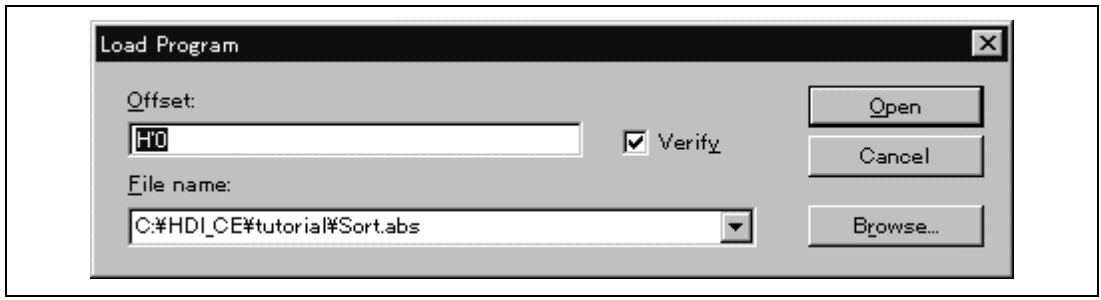


Figure 3.4 [Load Object File] Dialog Box

When the file has been loaded, the following dialog box displays information about the memory areas that have been filled with the program code.

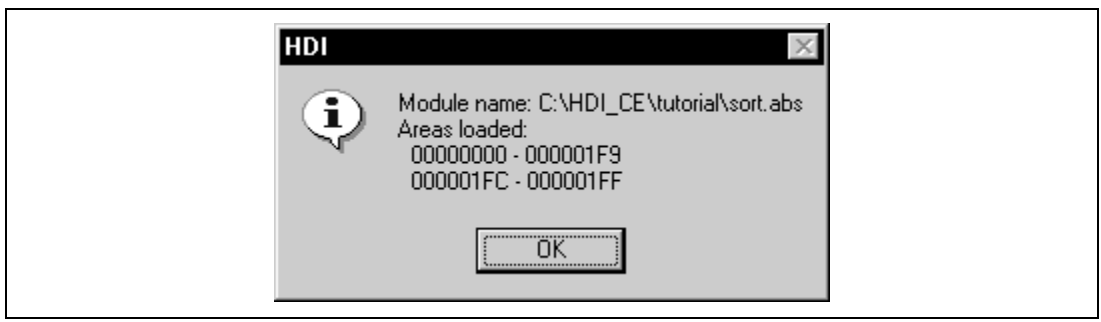


Figure 3.5 [HDI] Dialog Box

- Click the [OK] button to continue.

3.6.2 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.

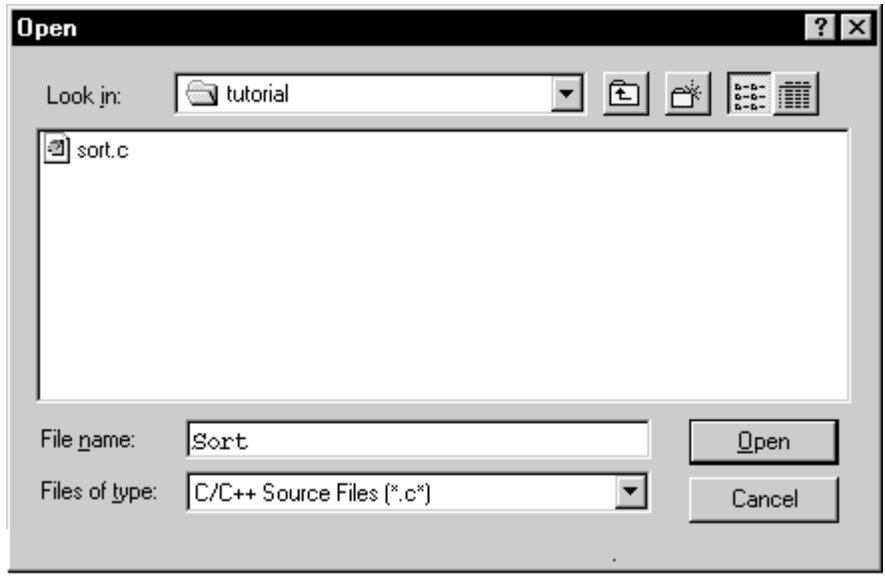


Figure 3.6 [Open] Dialog Box

- Select [sort.c] and click the [Open] button. The [Program] window is displayed.

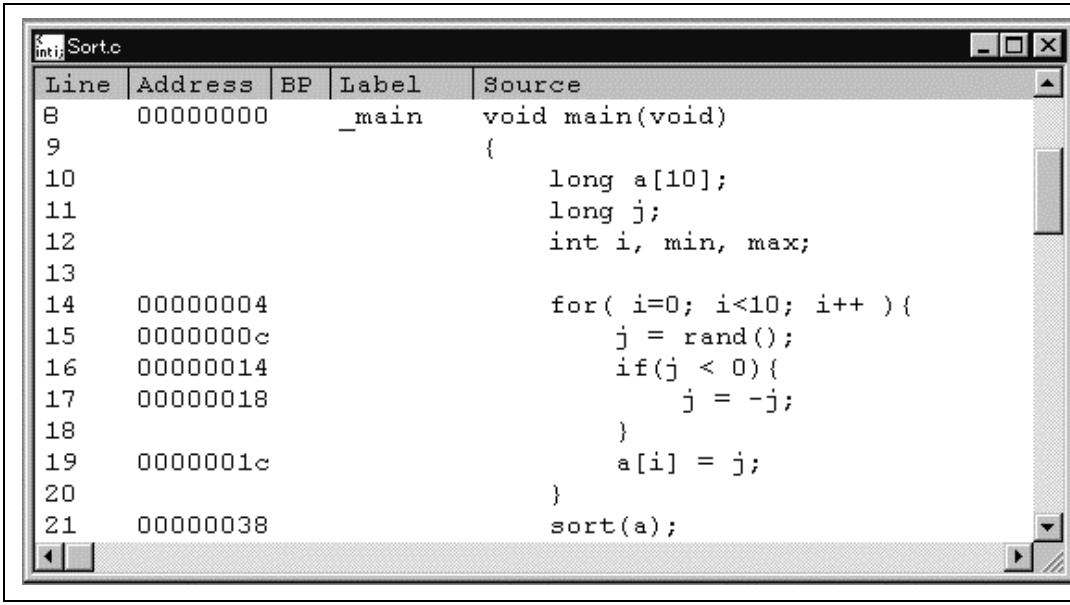


Figure 3.7 [Program] Window (Displaying the Source Program)

- If necessary, select the [Font] option from the [Customise] submenu on the [Setup] menu to select a clear font and size.

Initially the [Program] window shows the start of the main program, but the user can use the scroll bar to scroll through the program to see the other statements.

3.7 Setting the Software Breakpoint

A breakpoint is one of the easy debugging functions.

The [Program] 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.

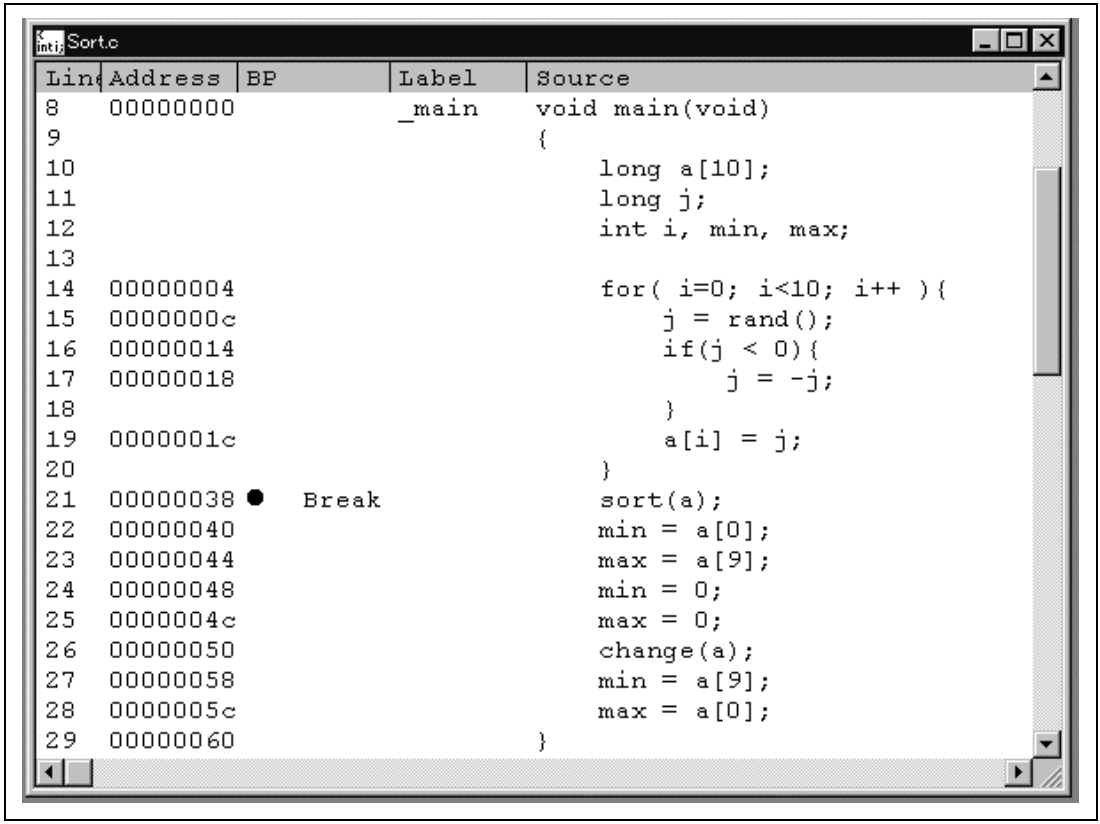


Figure 3.8 [Program] Window (Setting a Software Breakpoint)

The word • `Break` will be displayed on the line containing the `sort` function to show that a software breakpoint is set.

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

3.8 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.

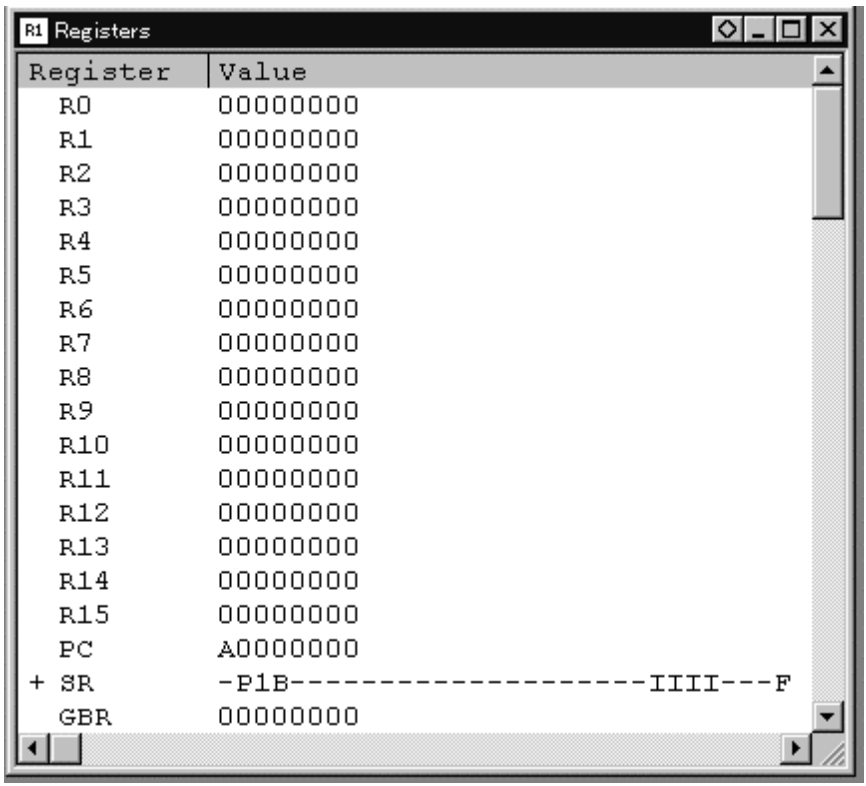


Figure 3.9 [Registers] Window

- 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.

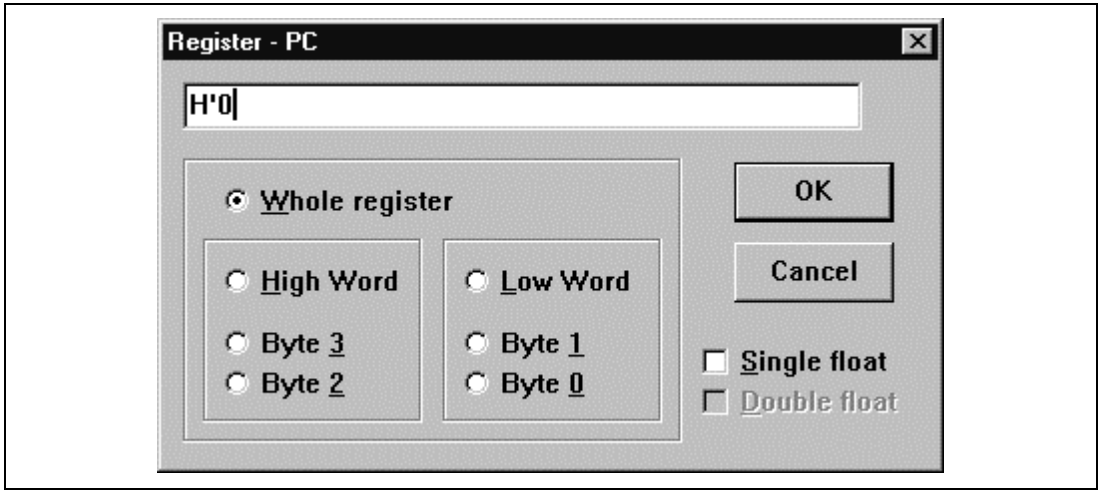


Figure 3.10 [Register] Dialog Box (PC)

- Set the program counter to H'0 in this tutorial program, and click the [OK] button.
- To change the value of the stack pointer (SP), move the mouse pointer on the value to be changed in the [R15] value area in the [Registers] window and enter the new value by the keyboard, or double-click the value area with the mouse. The following dialog box is then displayed.

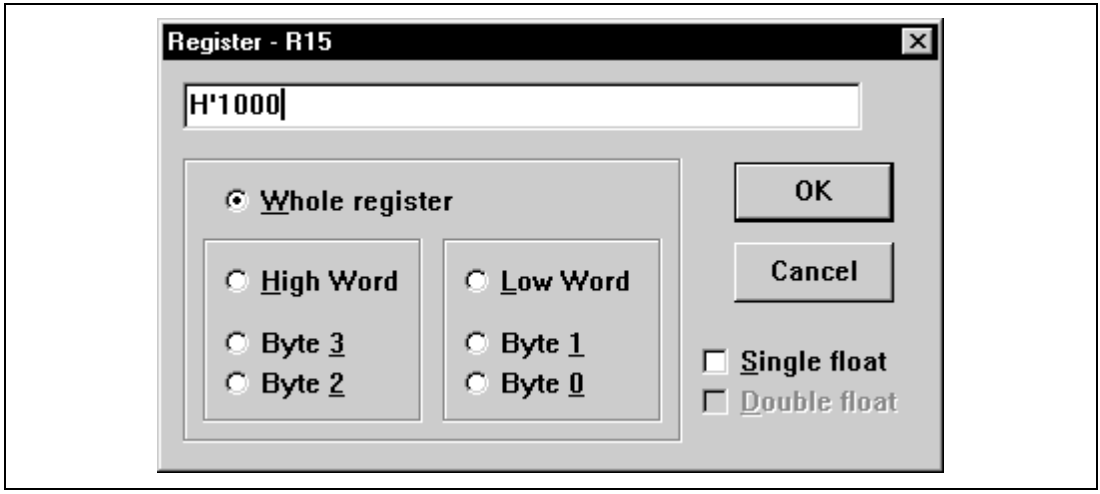


Figure 3.11 [Register] Dialog Box (R15)

- Set H'1000 for the value of the stack pointer in this tutorial program, and click the [OK] button.

3.9 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 3.12 [Go] Button

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

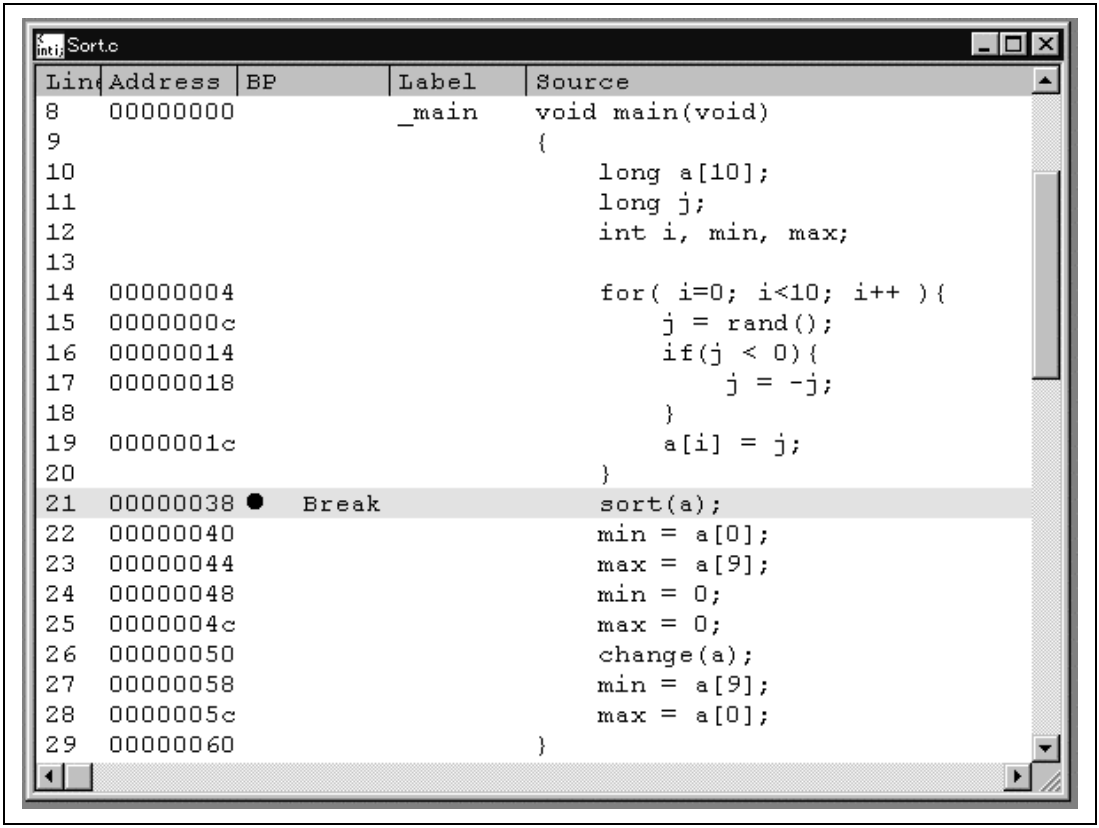


Figure 3.13 [Program] Window (Break Status)

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.

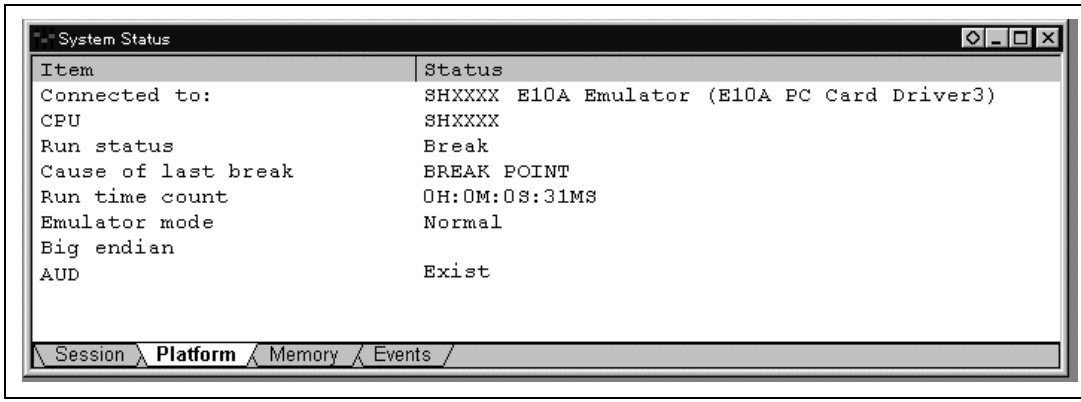


Figure 3.14 [System Status] Window

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 on-line help.

The [System Status] window displays the following items in each page.

Table 3.3 Contents of the [System Status] Window

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 MCU 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, M: minutes, S: seconds, and MS: milliseconds. In this example, 0H:0M:0S:31MS 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.

3.10 Reviewing Breakpoints

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

- Select [Breakpoints] from the [View] menu.

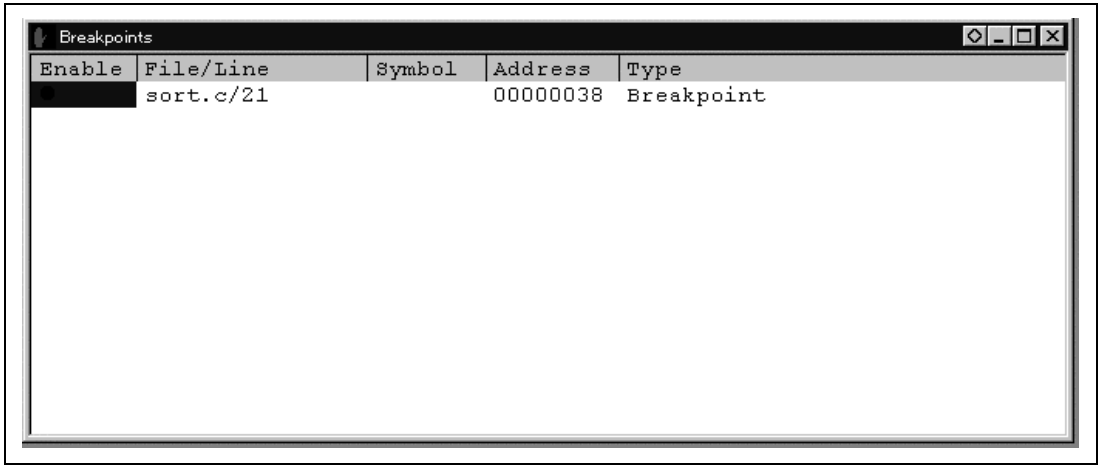


Figure 3.15 [Breakpoints] Window

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.

3.11 Viewing Memory

The user can view the contents of a memory block 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 3.16 [Open Memory Window] Dialog Box

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

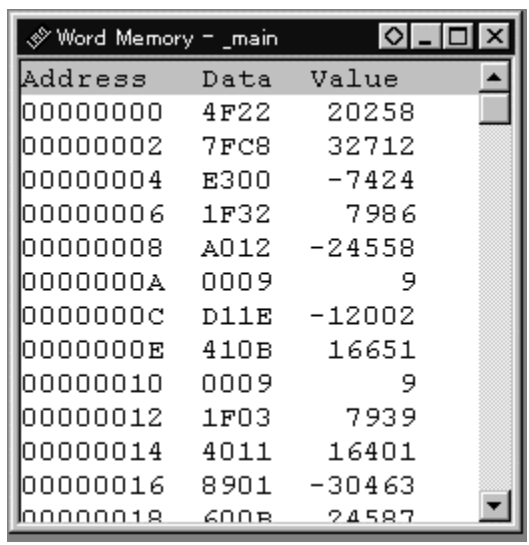


Figure 3.17 [Memory] Window

3.12 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 [Program] window to position the cursor.
- Click the [Program] window with the right mouse button and select [Instant Watch...] from a pop-up menu.

  The following dialog box will be displayed.

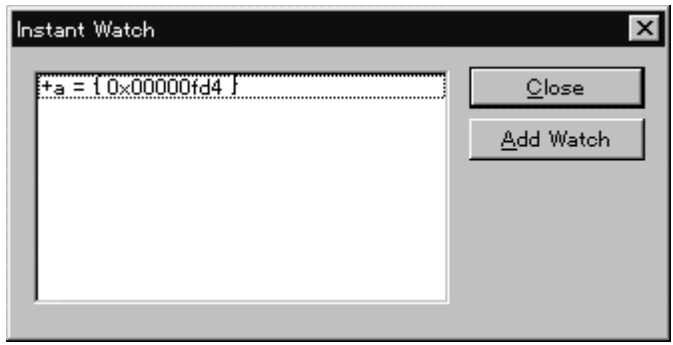


Figure 3.18 [Instant Watch] Dialog Box

- Click [Add Watch] button to add a variable to the [Watch] window.

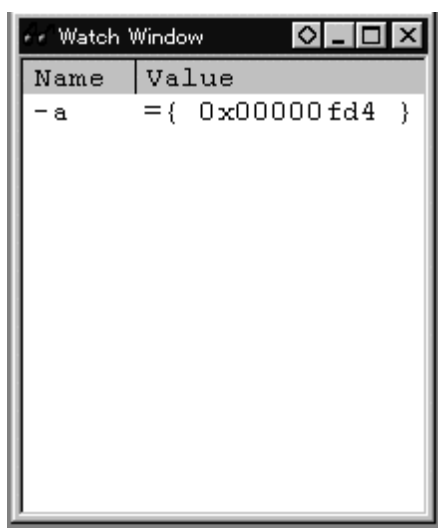


Figure 3.19 [Watch] Window (Displaying the Array)

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 3.20 [Add Watch] Dialog Box

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

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

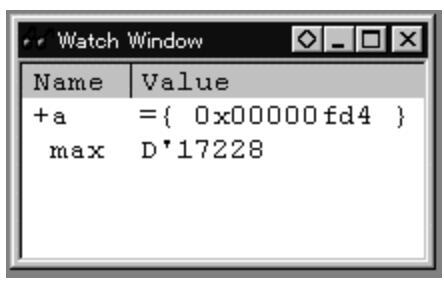


Figure 3.21 [Watch] Window (Displaying the Variable)

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.

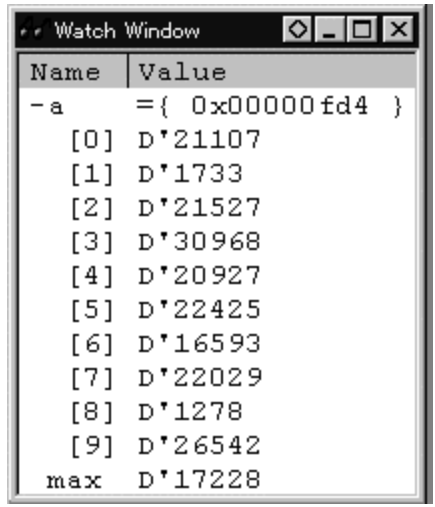


Figure 3.22 [Watch] Window (Displaying Array Elements)

3.13 Stepping Through a Program

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

Table 3.4 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.

Execute the step commands to confirm that the `sort` function statement at address H'38 has been executed.

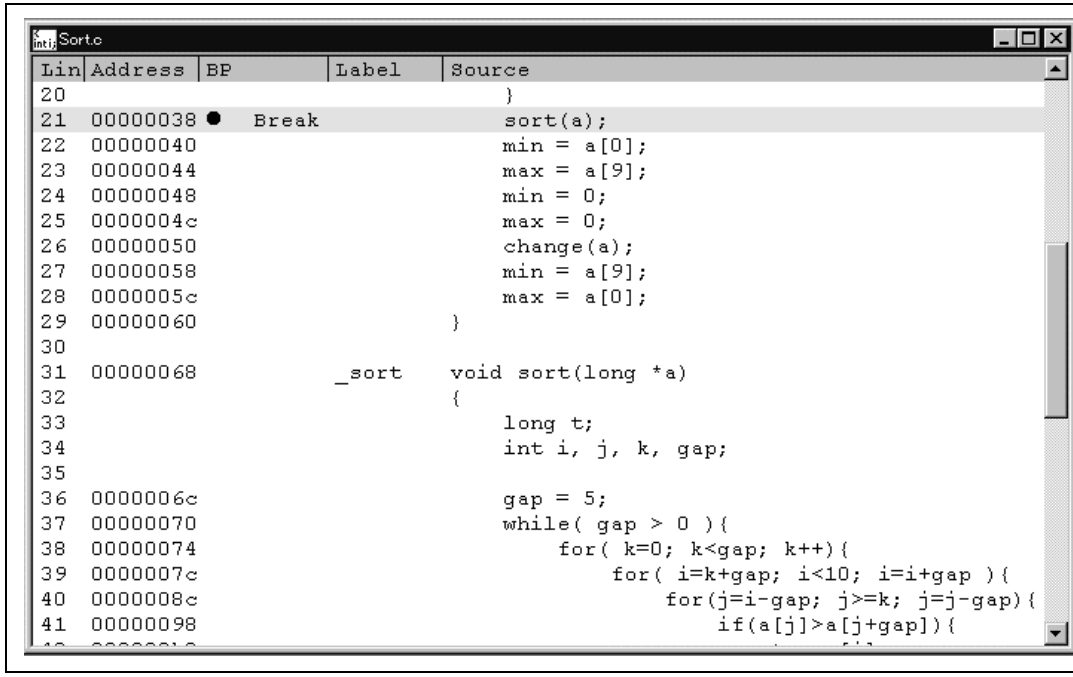


Figure 3.23 [Program] Window (Step Execution)

3.13.1 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

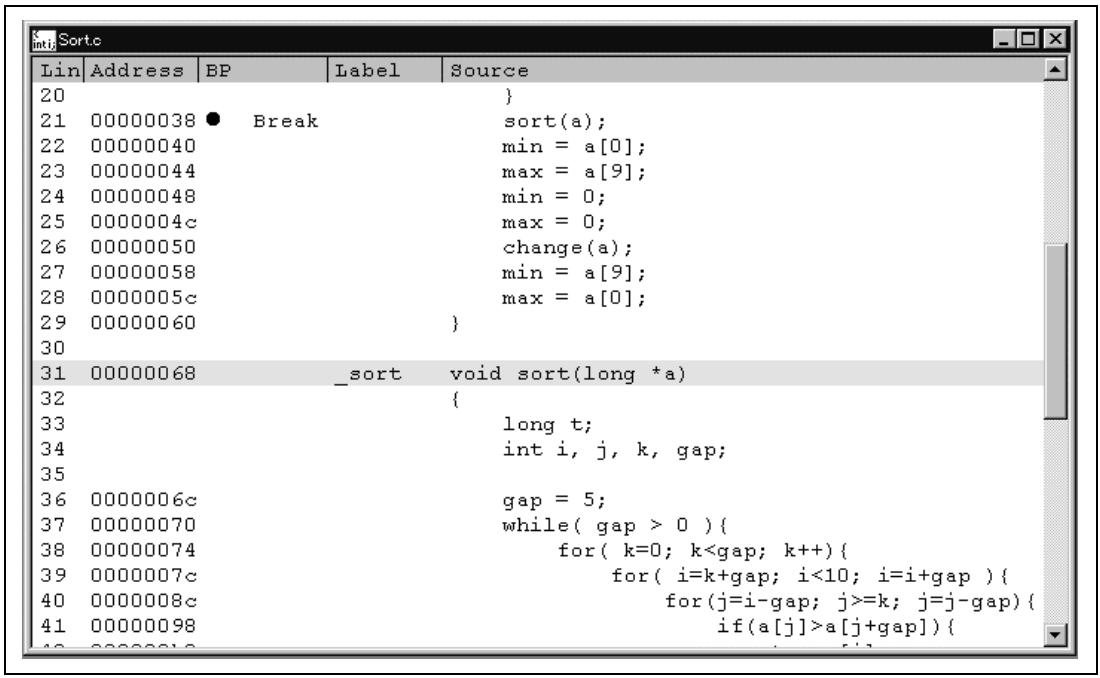


Figure 3.25 [Program] Window (Step In)

- The highlighted line moves to the first statement of the `sort` function in the [Program] window.

3.13.2 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.

Figure 3.26 [Step Out] Button

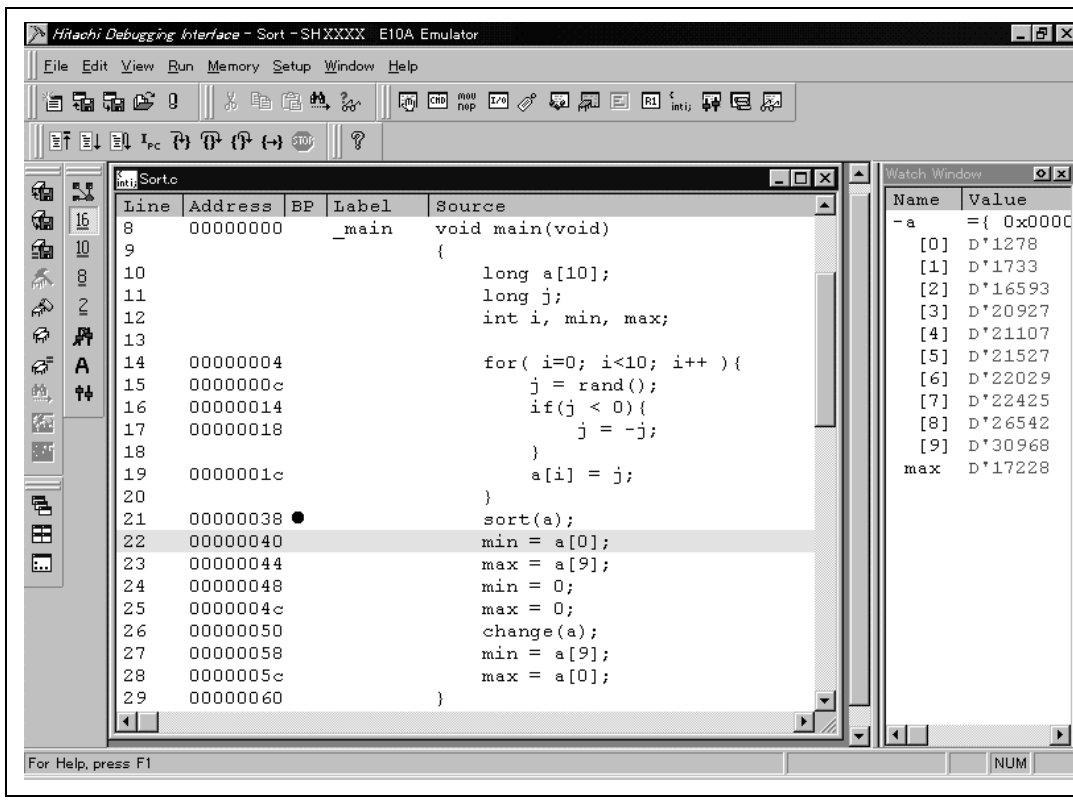


Figure 3.27 [Program] Window (Step Out)

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

- To execute two steps, use [Step In] twice.

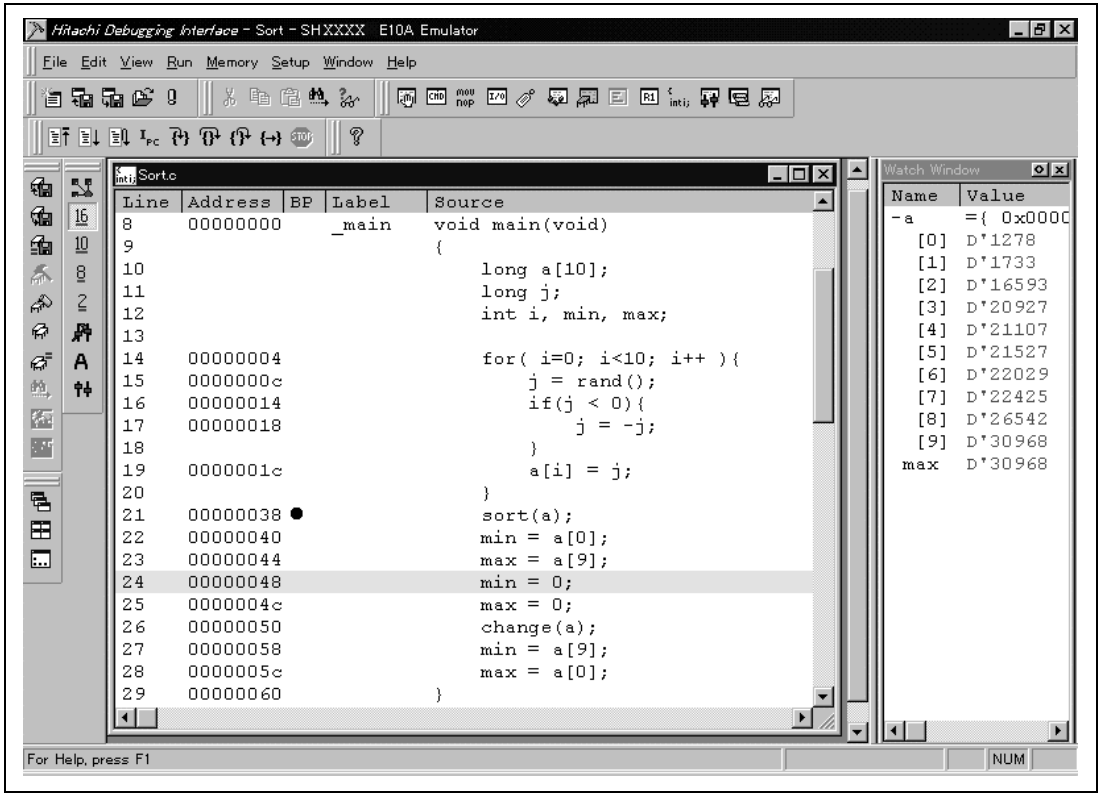


Figure 3.28 [Program] Window (Step In → Step In)

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

3.13.3 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.

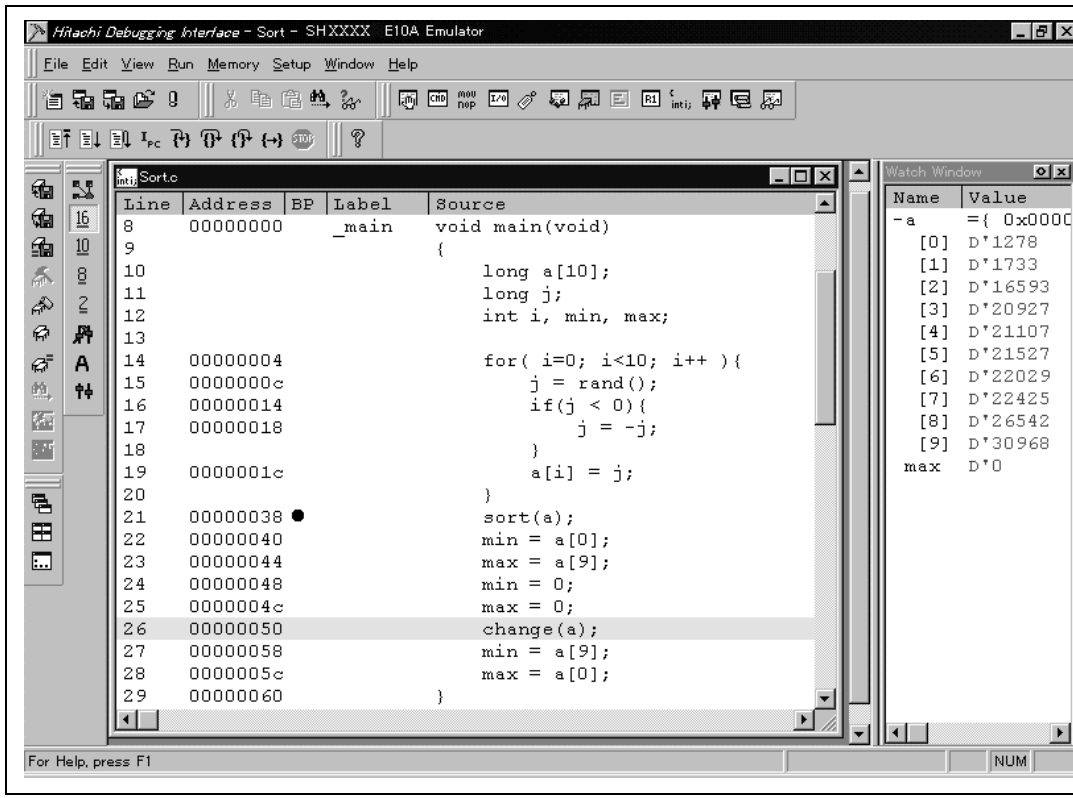


Figure 3.29 [Program] Window (Before Step Over Execution)

- 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

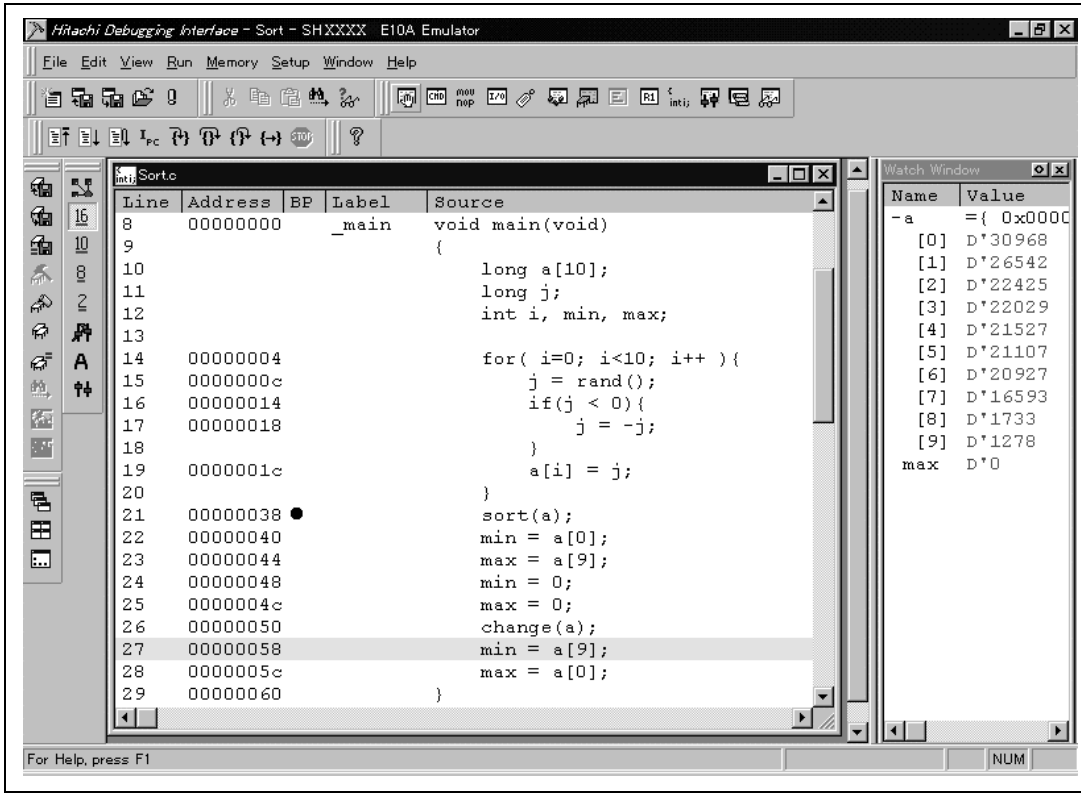


Figure 3.31 [Program] Window (Step Over)

When the last statement of the change function is executed, the data of variable a, which is displayed in the [Watch] window, is sorted in descending order.

3.14 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.

- Select [Step In] from the [Run] menu to execute a single step.

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

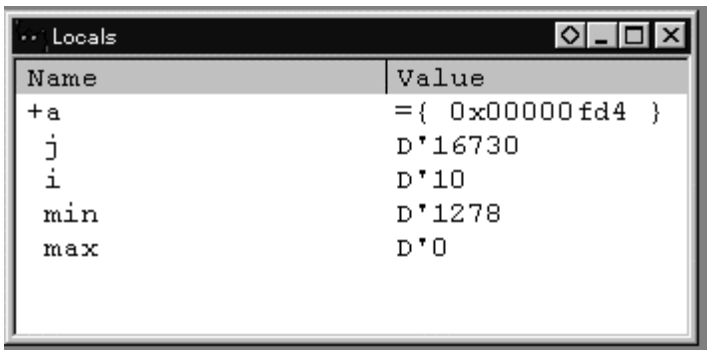


Figure 3.32 [Locals] Window

- 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.

3.15 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 1,2,3] dialog box.

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

3.15.1 Software Break Function

The emulator can set up to 255 software breakpoints. Setting a software breakpoint is 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.

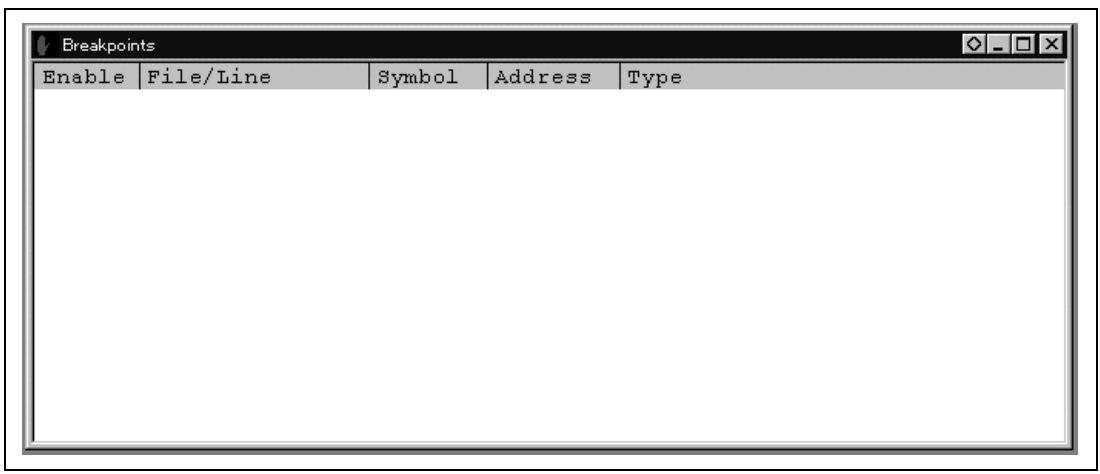


Figure 3.33 [Breakpoints] Window (Before Software Breakpoint Setting)

- 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.

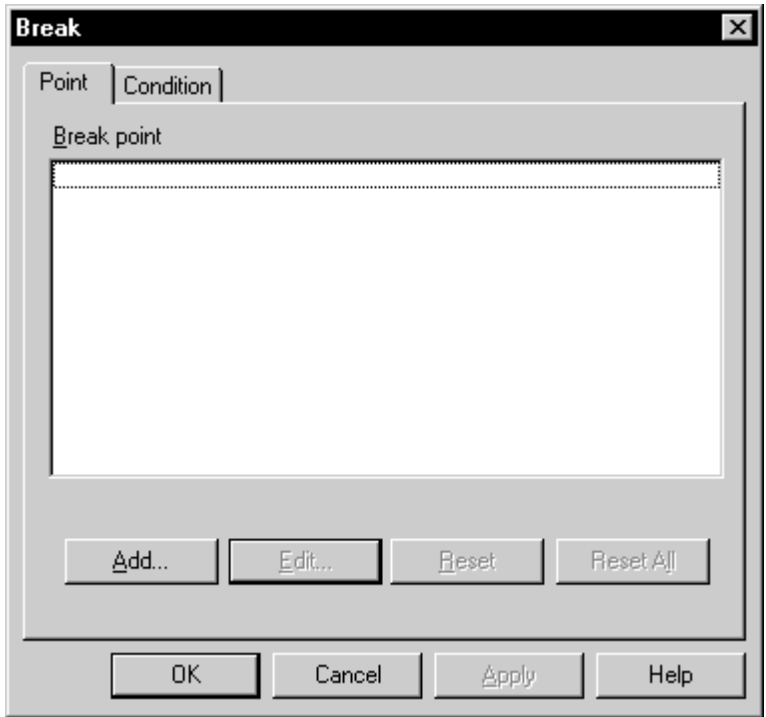


Figure 3.34 [Point] Page ([Break] Dialog Box)

- Click the [Add...] button to display the [Break Point] dialog box.
- Enter ***H'58*** to the [Value] edit box.

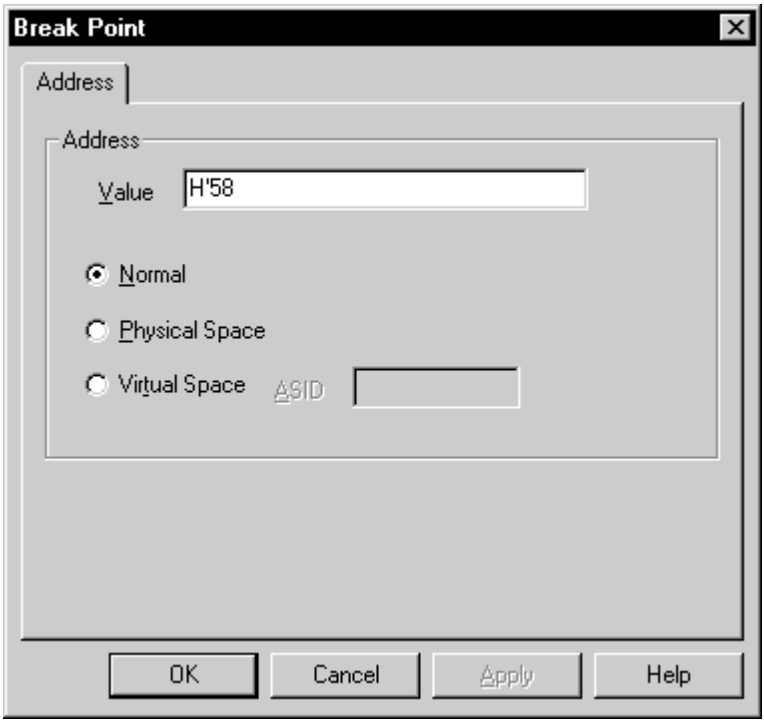


Figure 3.35 [Break Point] Dialog Box

- Click the [OK] button.

The [Break] dialog box is displayed. The address set in the value field of [Break Point] and the memory space are displayed.

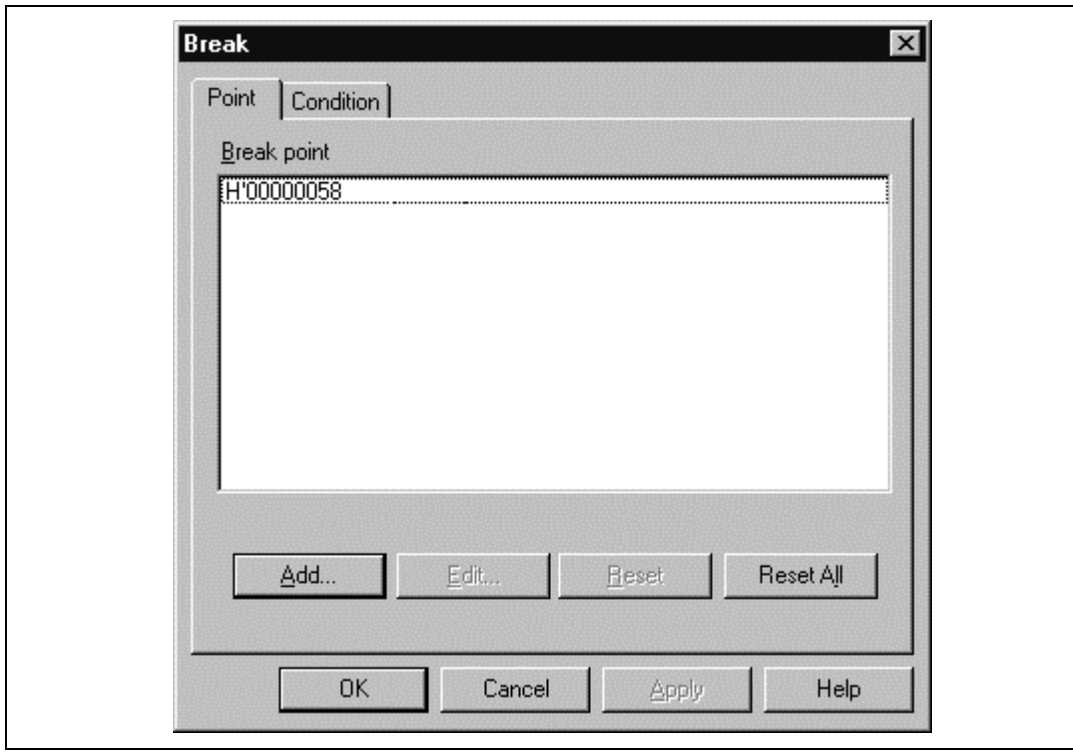


Figure 3.36 [Point] Page ([Break] Dialog Box) (After Software Breakpoint Setting)

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

The software breakpoint that has been set is displayed in the [Breakpoints] window.

Figure 3.37 [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'0 and R15 = H'1000) that have been set in section 3.8, Setting Registers, in the [Registers] window. Click the [Go] button.

The program runs, and stops at the set breakpoint.

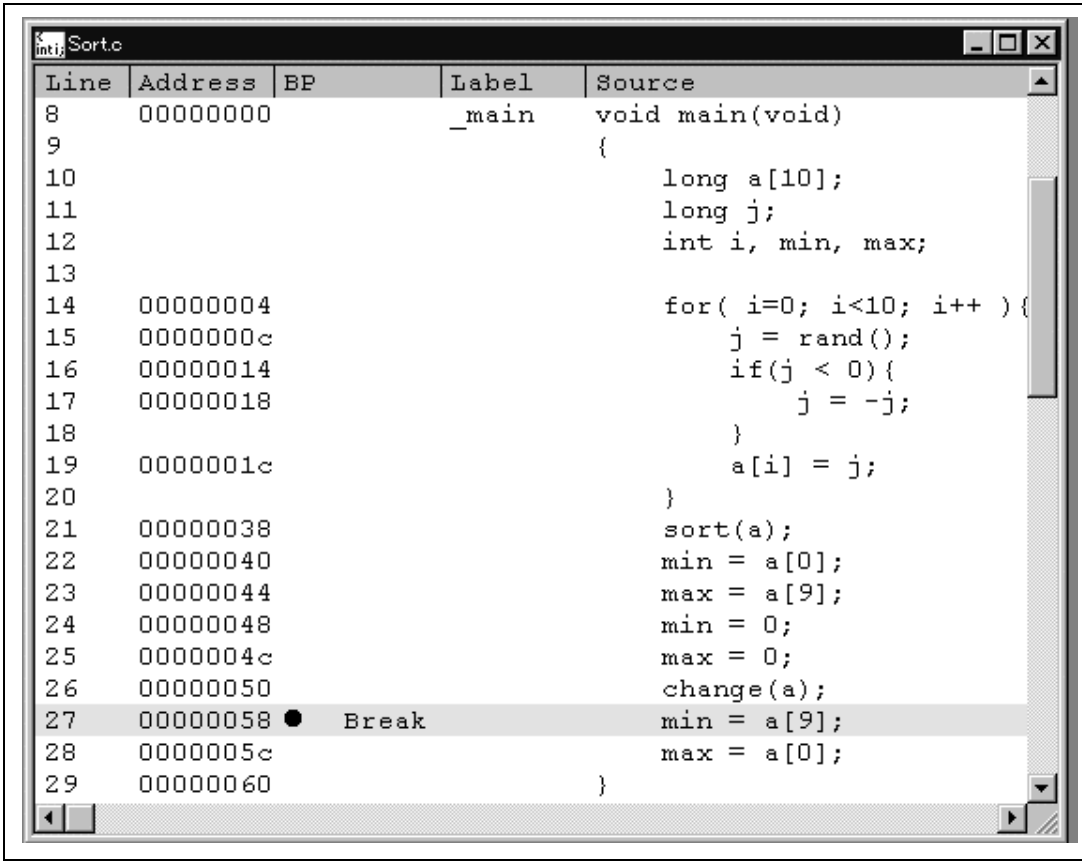


Figure 3.38 [Program] Window at Execution Stop (Software Break)

The [System Status] window displays the following contents.

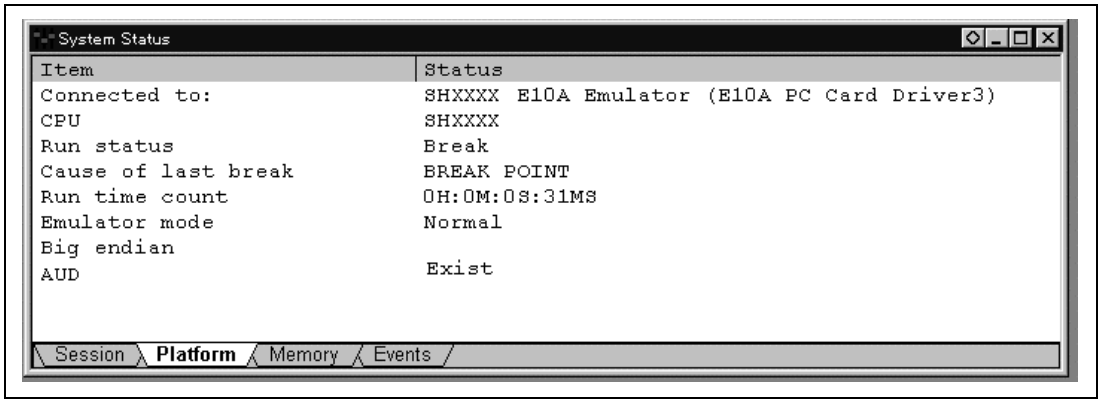


Figure 3.39 Displayed Contents of the [System Status] Window (Software Break)

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 on-line help.

3.16 Hardware Break Function

A method is given below in which the address bus condition and the read cycles for the state 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.

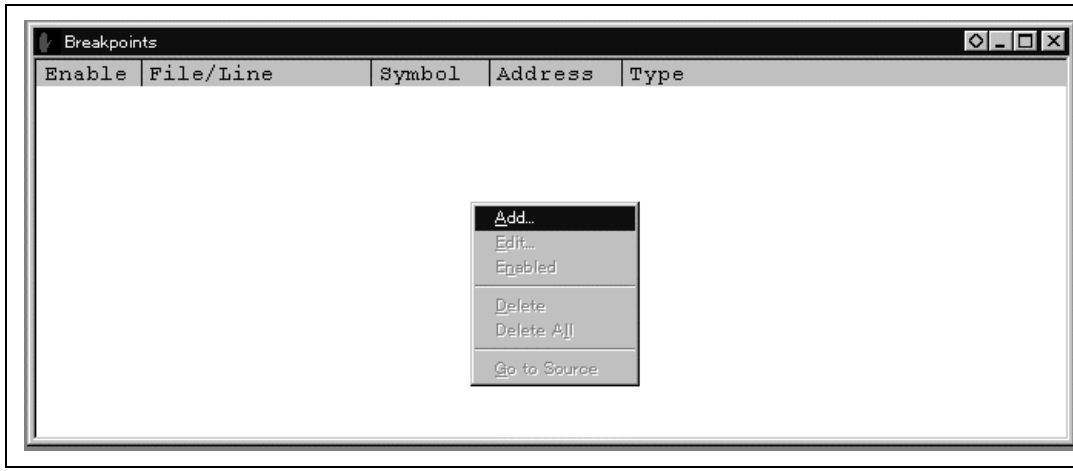


Figure 3.40 [Breakpoints] Window (Before Hardware Break Condition Setting)

The [Break] dialog box is displayed. To set hardware break conditions, select [Condition] in the [Break] dialog box to display the [Condition] page.

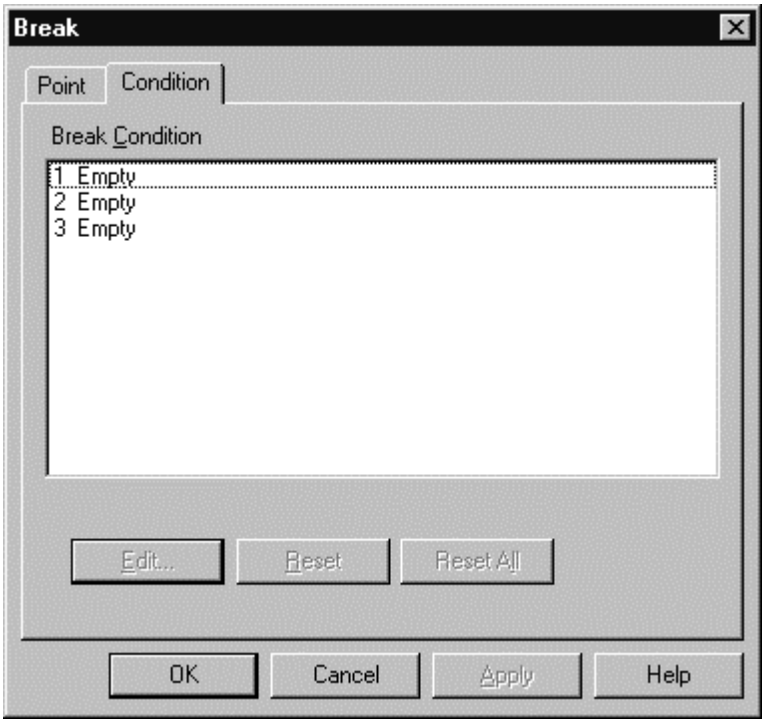


Figure 3.41 [Condition] Page ([Break] Dialog Box)

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

- Highlight the first point in the [Break Condition] list box.
- Click the [Edit...] button. The [Break Condition 1] dialog box is displayed.
- Clear the [Don't Care] check box in the [Address] page.
- Select the [Address] radio button and enter ***H'48*** as the value in the [Address] edit box.

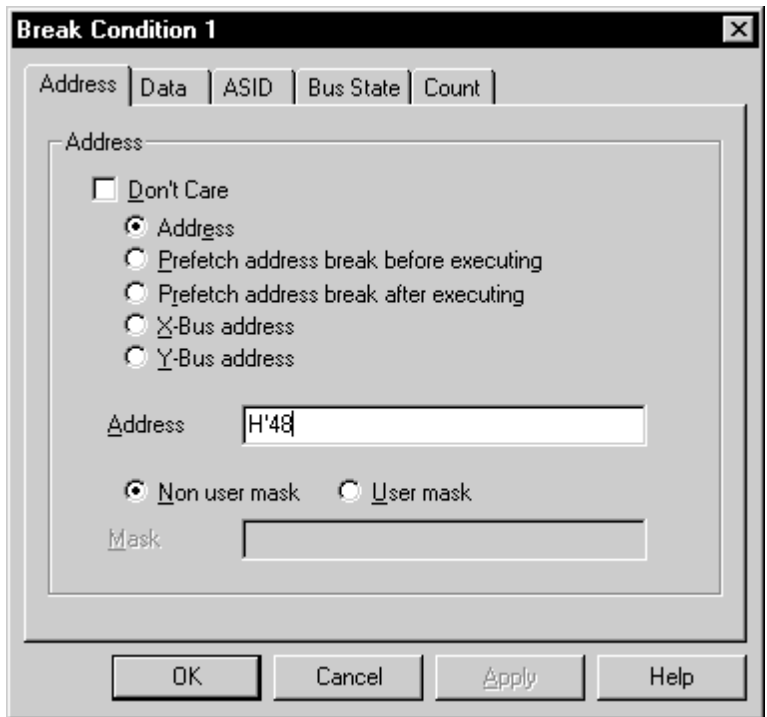


Figure 3.42 [Address] Page ([Break Condition 1] Dialog Box)

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

- Select [Bus State] to display the [Bus State] page.
- Select the [Read] radio button in the [Read/Write] group box.

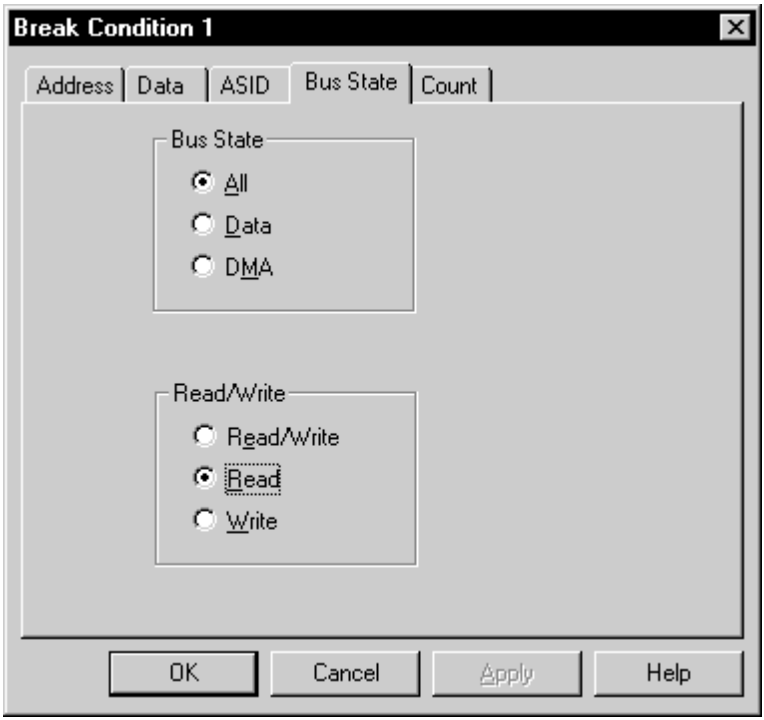


Figure 3.43 [Bus State] Page ([Break Condition 1] Dialog Box)

Note: The items that can be set in this window differ according to the product. For the settings for each product, refer to the on-line 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`.

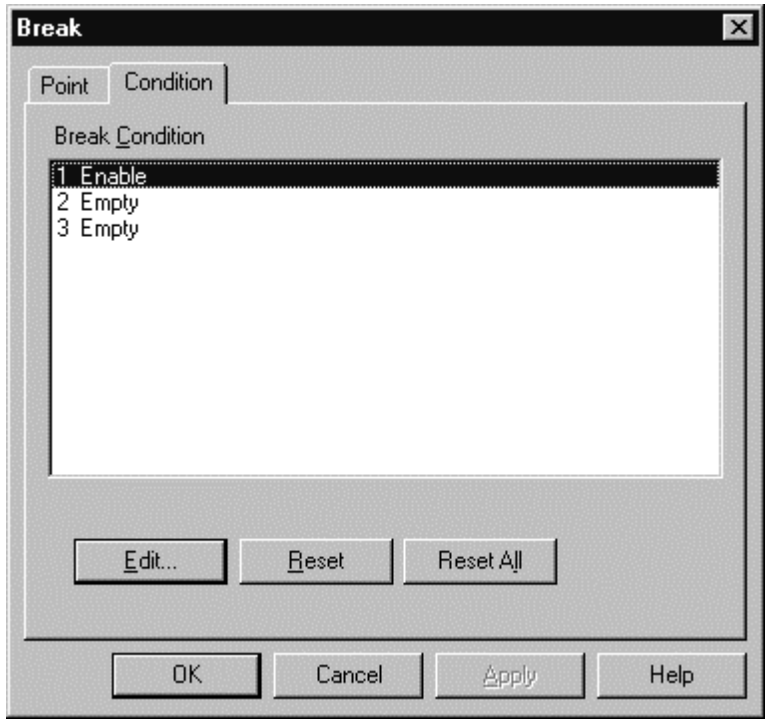


Figure 3.44 [Break] Dialog Box (After Hardware Break Condition Setting)

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

- Click the [OK] button.

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'48 is accessed in a read cycle.

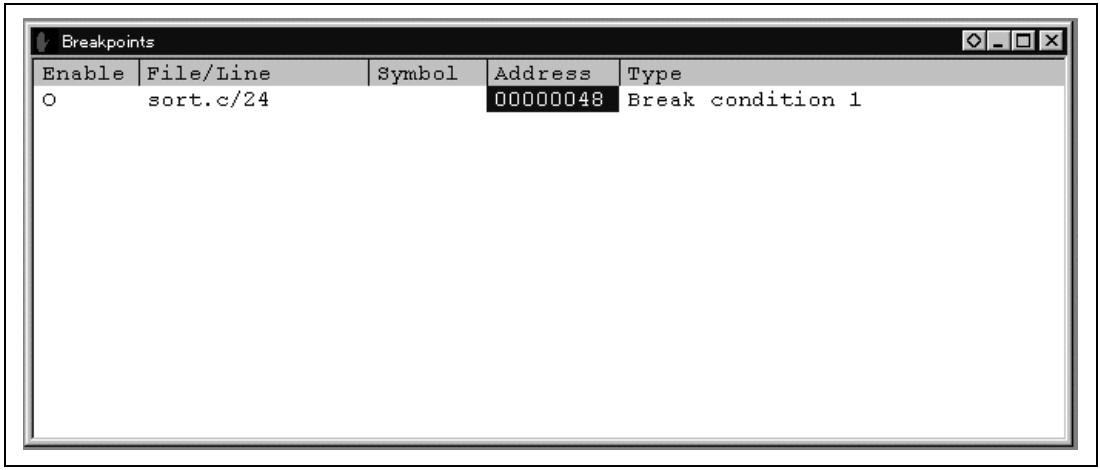


Figure 3.45 [Breakpoints] Window ([Break Condition 1] Setting)

- Close the [Breakpoints] window.
- Set the program counter and stack pointer values (PC = H’0 and R15 = H’1000) that have been set in section 3.8, Setting Registers, in the [Registers] window. Click the [Go] button.

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

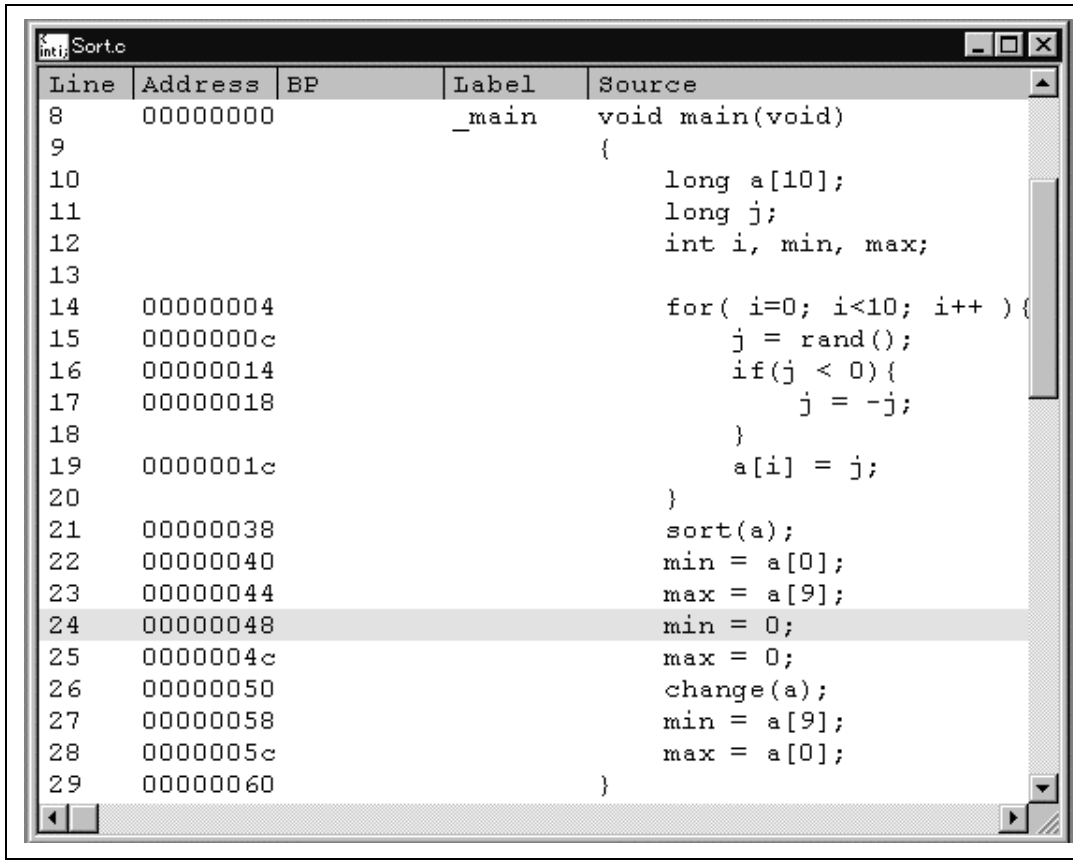


Figure 3.46 [Program] Window at Execution Stop (Break Condition 1)

The [System Status] window displays the following contents.

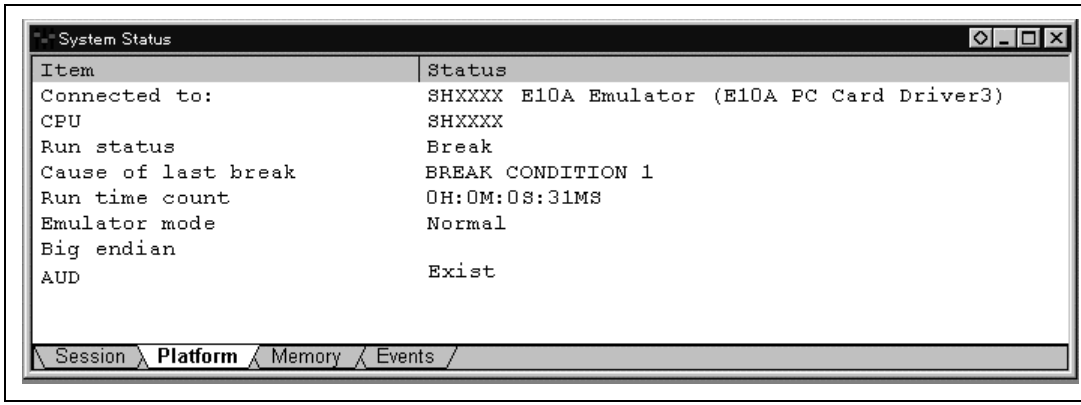


Figure 3.47 Displayed Contents of the [System Status] Window (Break Condition 1)

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 on-line help.

3.16.1 Setting the Sequential Break Condition

The emulator has sequential break functions. When the hardware break conditions listed in table 3.5 are satisfied, program execution is halted. This mode is called sequential break.

Table 3.5 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.

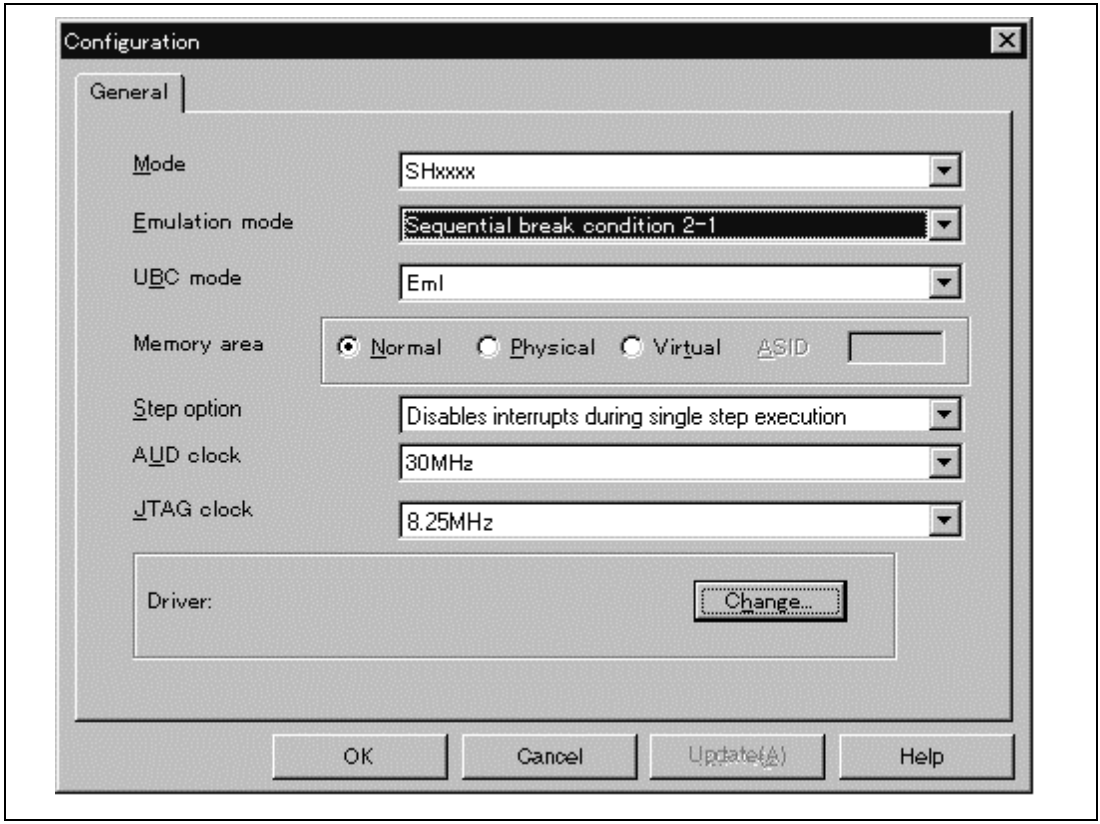


Figure 3.48 [Configuration] Dialog Box (Sequential Break Setting)

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

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

Set break conditions as follows:

Break condition 1: When address H'58 is accessed in a read cycle, a break condition is satisfied (Break Condition 1 is set).

Break condition 2: When address H'48 is accessed in a read cycle, a break condition is satisfied (Break Condition 2 is set).

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

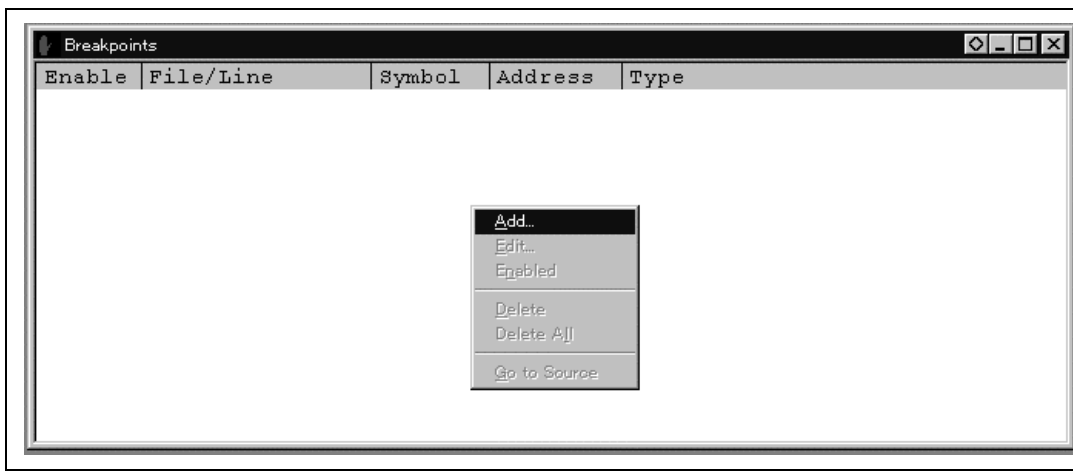


Figure 3.49 [Breakpoints] Window (Before Sequential Break Condition Setting)

The [Break] dialog box is displayed. To set sequential break conditions, select [Condition] in the [Break] dialog box to display the [Condition] page.

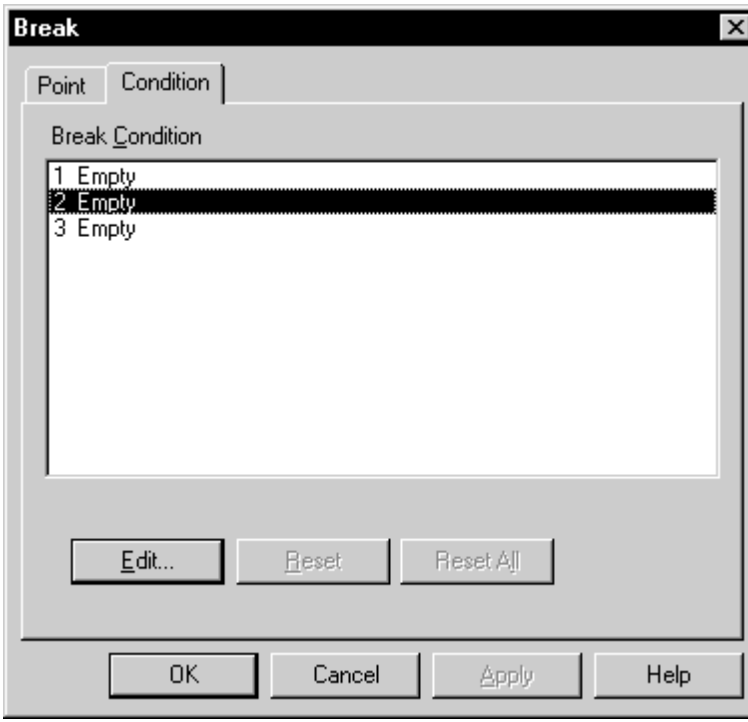


Figure 3.50 [Condition] Page ([Break] Dialog Box)

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 on-line help.

For the sequential break condition, set break condition 2 in the [Break Condition 2] dialog box, and set break condition 1 in the [Break Condition 1] dialog box.

- Highlight the second point in the [Break Condition] list box.
- Click the [Edit...] button. The [Break Condition 2] dialog box is displayed.
- Clear the [Don't Care] check box in the [Address] page.
- Select the [Address] radio button and input address ***H'48*** as the value in the [Address] edit box.

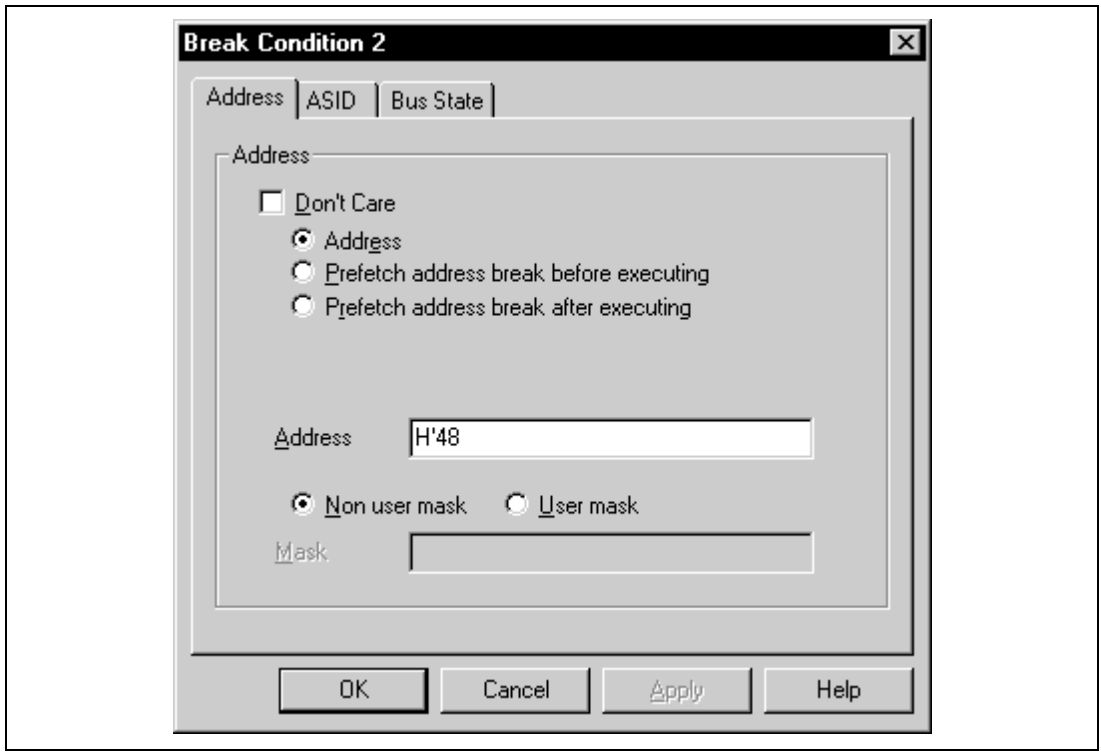


Figure 3.51 [Break Condition 2] Dialog Box (Condition 2 [Address] Page)

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

- Select [Bus State] to display the [Bus State] page.
- Select the [Read] radio button in the [Read/Write] group box.

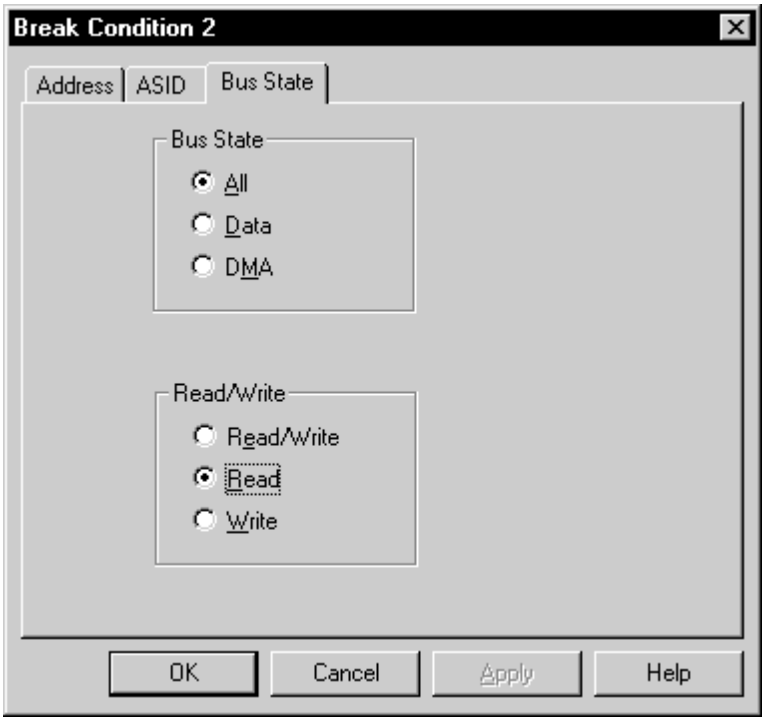


Figure 3.52 [Break Condition 2] Dialog Box (Condition 2 [Bus State] Page)

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

- Click the [OK] button.

- The [Break] dialog box is displayed, and the second point display in the [Break Condition] list box changes from `Empty` to `Enable`.

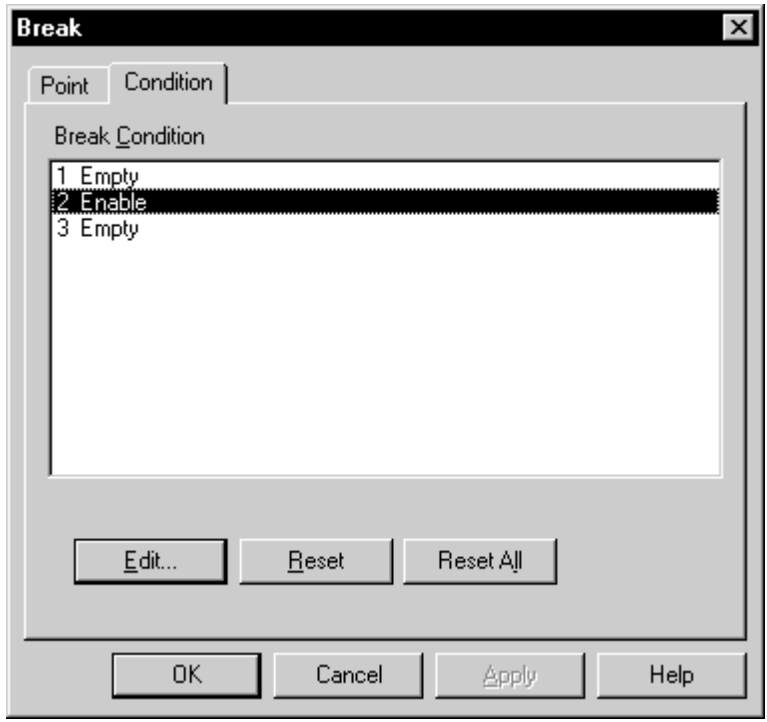


Figure 3.53 [Break] Dialog Box (After [Break Condition 2] Condition Setting)

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

This completes the setting of break condition 2. Next, set break condition 1 as follows:

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

The setting can then be made in the same way as for break condition 1.

- After setting break conditions 1 and 2, click the [OK] button.

`Break Condition 1` and `Break Condition 2` are displayed in [Type] in the [Breakpoints] window.

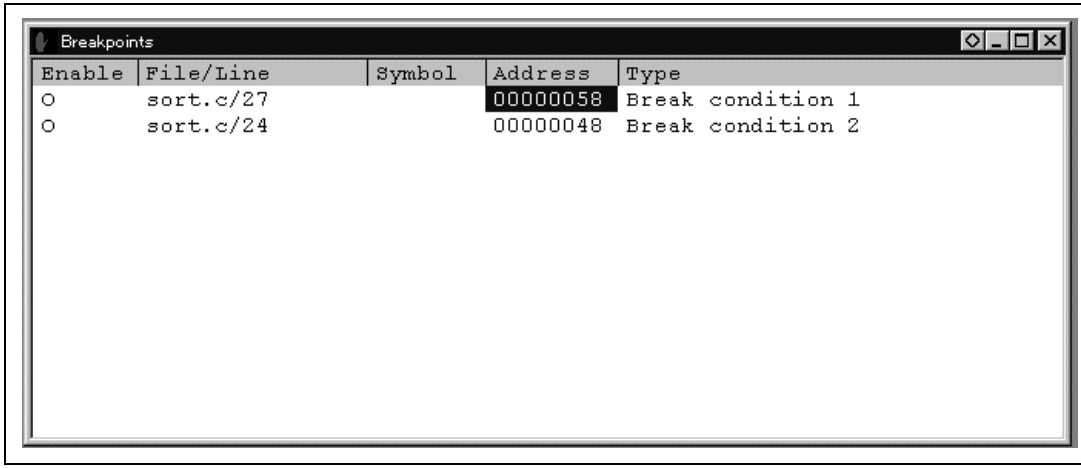


Figure 3.54 [Breakpoints] Window (After Sequential Break Condition Setting)

- Close the [Breakpoints] window.
- Set the program counter and stack pointer values (PC = H'0 and R15 = H'1000) that have been set in section 3.8, Setting Registers, in the [Registers] window. Click the [Go] button.

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

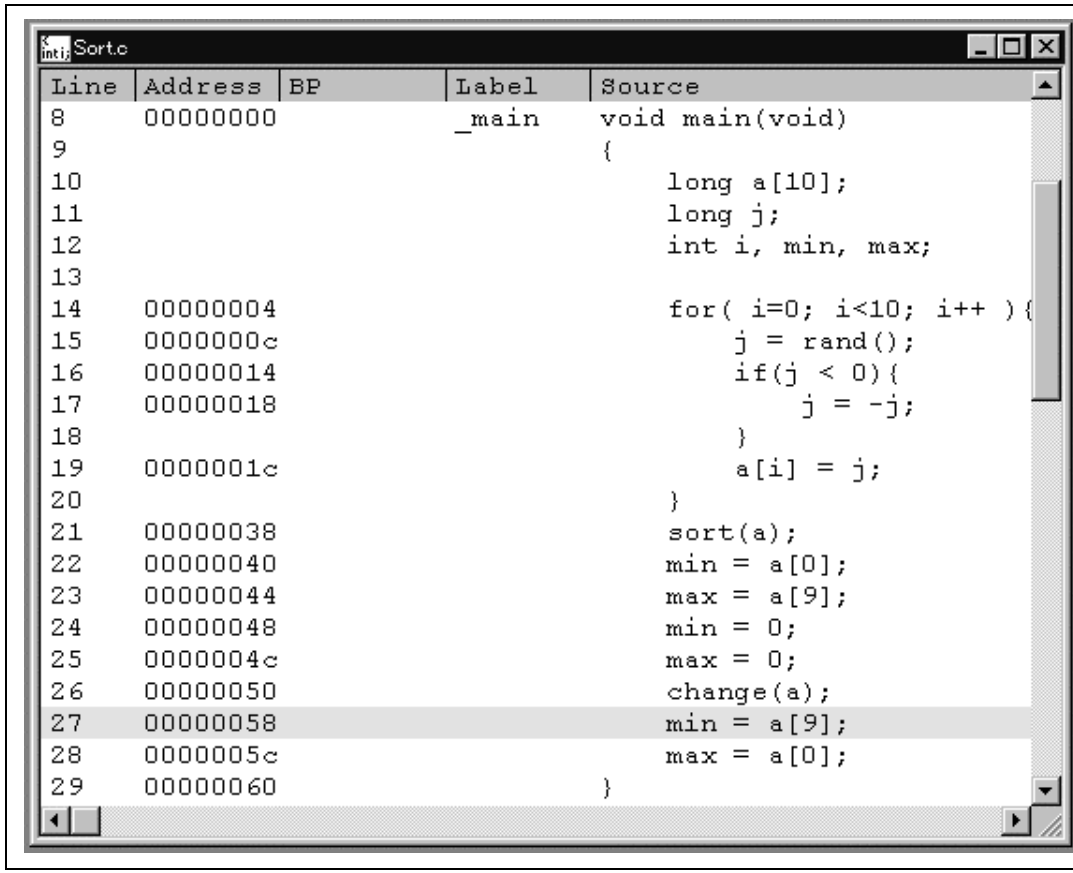


Figure 3.55 [Program] Window at Execution Stop (Sequential Break)

The [System Status] window displays the following contents.

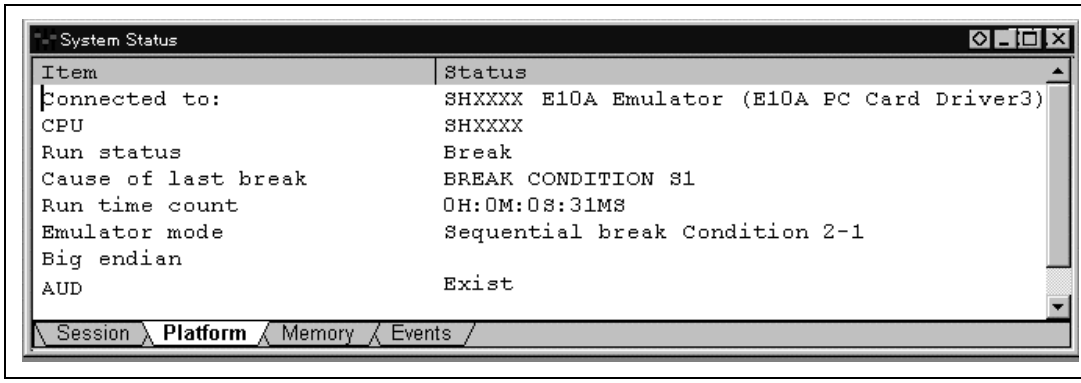


Figure 3.56 Displayed Contents of the [System Status] Window (Sequential Break)

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 on-line help.

3.17 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 on-line 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 on-line 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 on-line 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 4,096 in the PCMCIA-type emulator and 16,384 in the PCI-type emulator.

Table 3.6 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 on-line help.
2. The AUD 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 on-line help.

Table 3.6 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.

3.17.1 Internal Trace Function

The branch source and branch destination information for the latest 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.

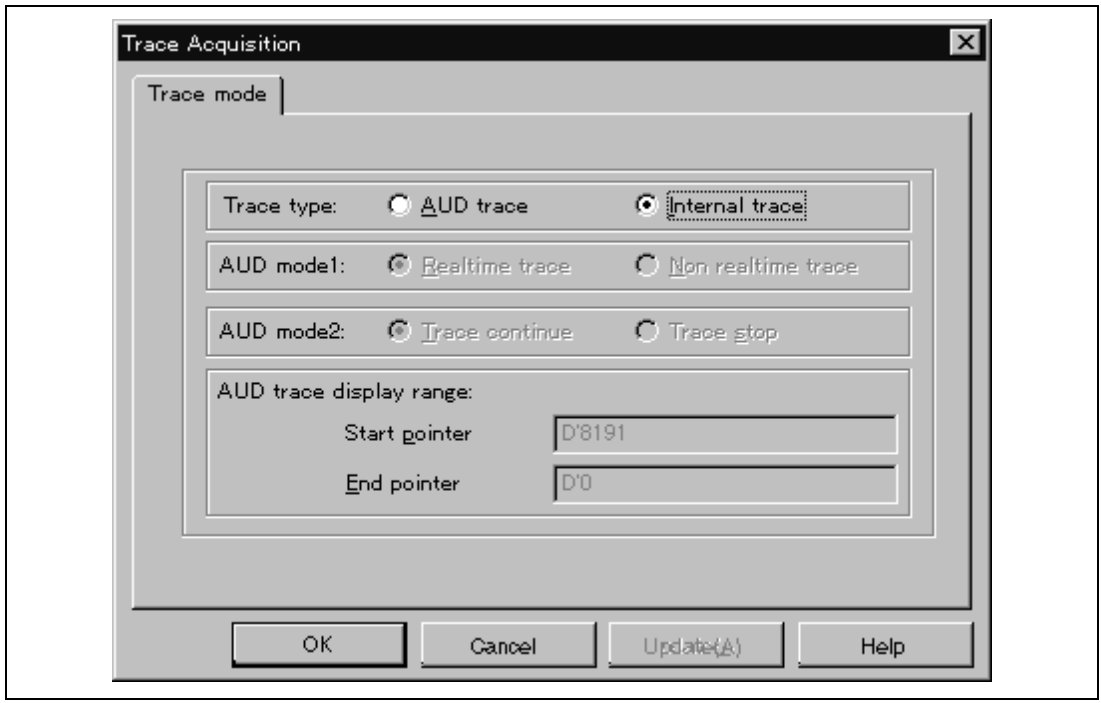


Figure 3.57 [Trace mode] Window

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

Run the program as shown in the example of section 3.15.1, Software Break Function. The trace results are displayed in the [Trace] window after the program execution is completed.

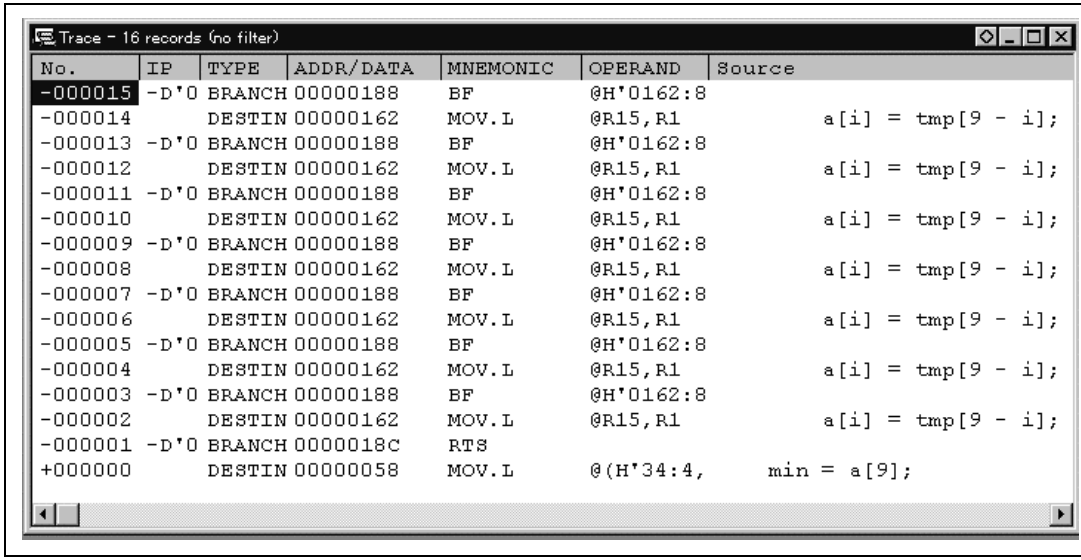


Figure 3.58 [Trace] Window

- 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 on-line help.

3.17.2 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.

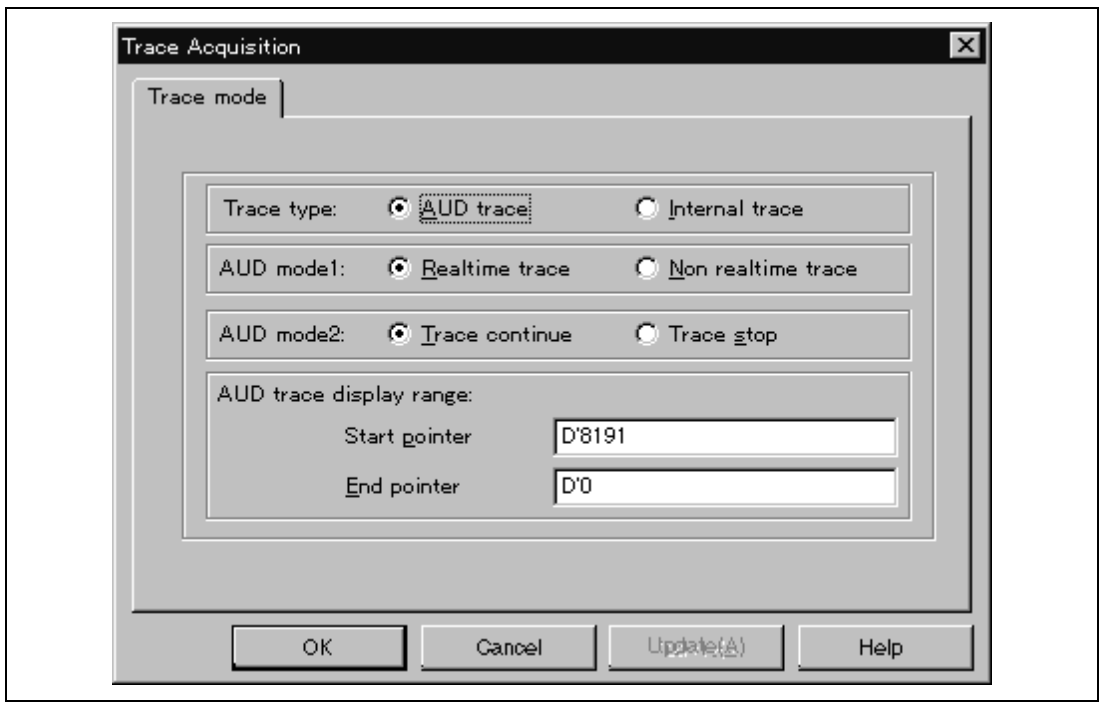


Figure 3.59 [Trace mode] Window

Note: For a description of each option, refer to table 3.6.

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

3.17.3 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 3.60.

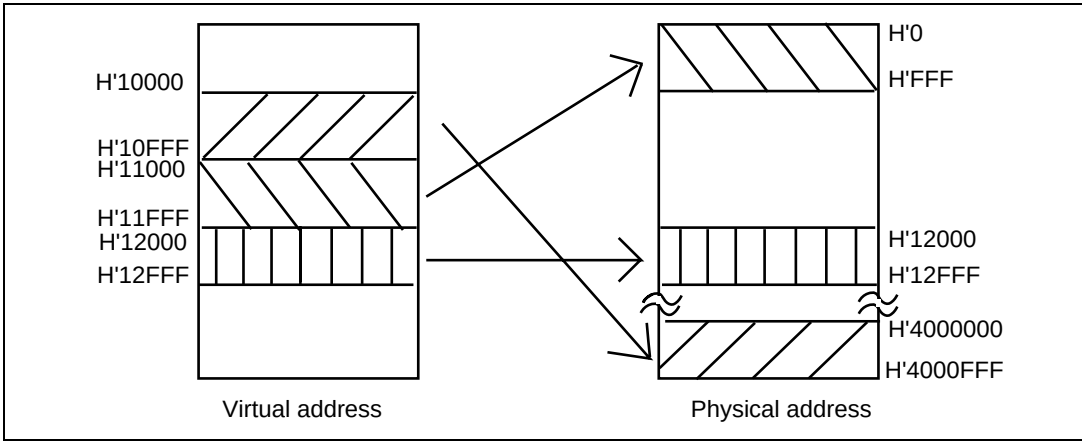


Figure 3.60 Address Translation according to VP_MAP Tables

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.

Table 3.7 Address Translation Tables

	VP_MAP		MMU		
Radio Button*	**Enabled/ Disabled**	**Within/ Outside the range**	**Enabled/ Disabled**	**Within/Outside the TLB Range**	**Table Used for Translation**
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 [Configuration] dialog box.

3.18 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	3.6.1
	Initialize	O	—
	Exit	O	—
Edit Menu	Cut	O	—
	Copy	O	—
	Paste	O	—
	Find…	O	—
	Evaluate…	O	—

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	3.10, 3.15.1, 4.2.3, 6.5.3
	Command Line	O	—
	Disassembly...	O	—
	I/O Area	O	—
	Labels	O	—
	Locals	O	3.14
	Memory...	O	3.11
	Performance Analysis	O	—
	Registers	O	3.8
	Source…	O	3.6.2
	Status	O	3.9, 3.15.1, 4.2.10
	Trace	O	4.2.8, 6.5.5, 6.5.6
	Watch	O	3.12
Run Menu	Reset CPU	O	—
	Go	O	3.9
	Reset Go	O	—
	Go to Cursor	O	—
	Set PC To Cursor	O	—
	Run…	O	—
	Step In	O	3.13.1
	Step Over	O	3.13.3
	Step Out	O	3.13.2
	Step…	O	—
	Halt	O	—

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	—

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:

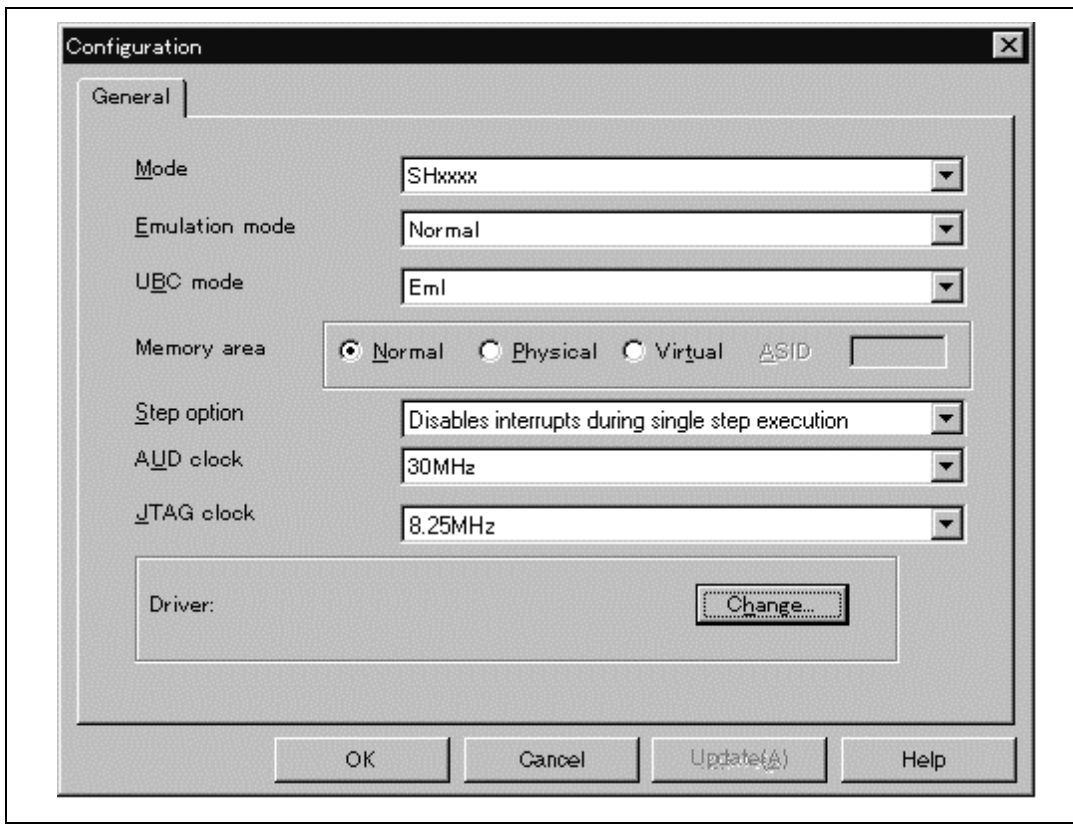


Figure 4.1 [Configuration] Dialog Box

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

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.

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

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

Function:

This page sets the emulator operation conditions, displays the MCU 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:

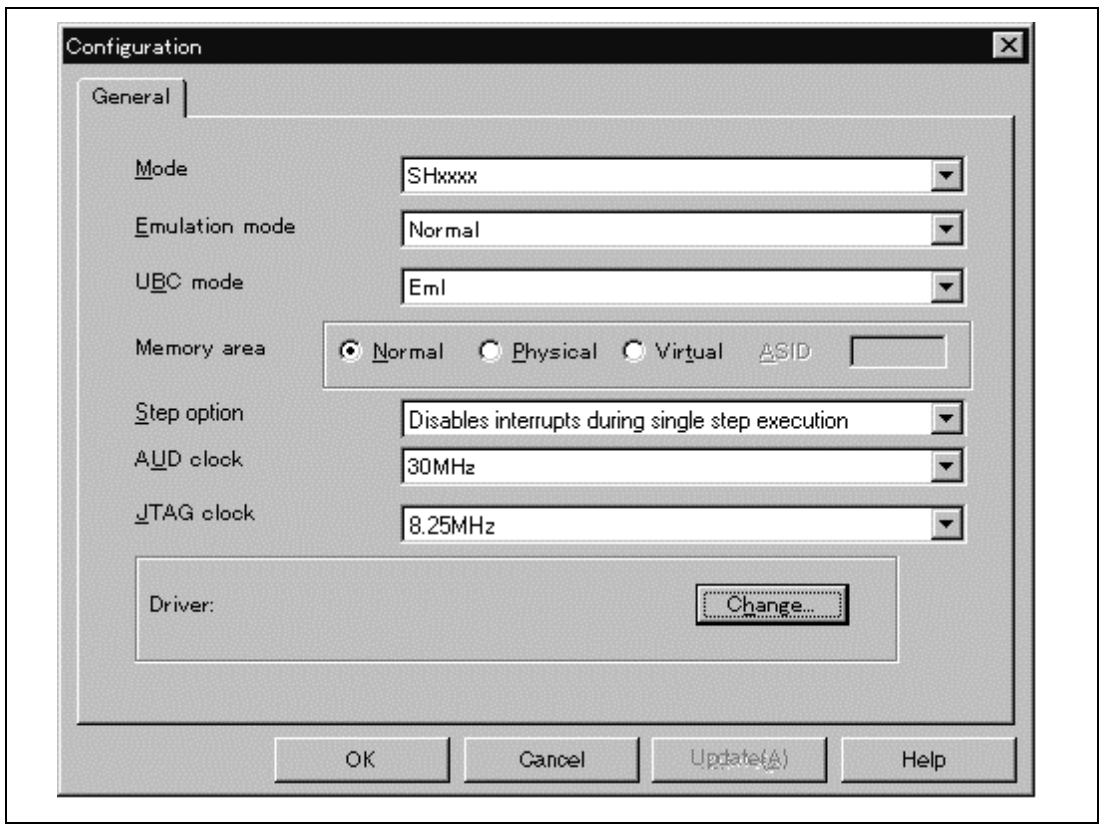


Figure 4.2 [General] Page ([Configuration] Dialog Box)

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

Description:

Table 4.3 [General] Page Options

Option	Description
[Mode] combo box	Displays the MCU 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.)
[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.
[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.
[AUD clock] combo box	Selects the AUD clock*[2]. By default, the maximum frequency of each card emulator is displayed.
[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 MCUs 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 MCUs used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).

When a driver is to be changed with the [Change..] button, the following message is displayed.

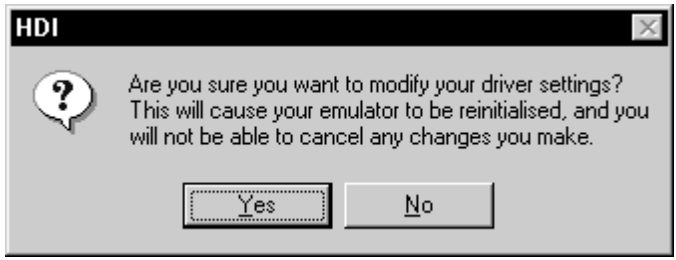


Figure 4.3 Warning Message Box

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

4.2.2 [E10A Driver Details] Dialog Box

The [E10A Driver Details] dialog box is displayed when the [OK] button is clicked. Select the correct driver.

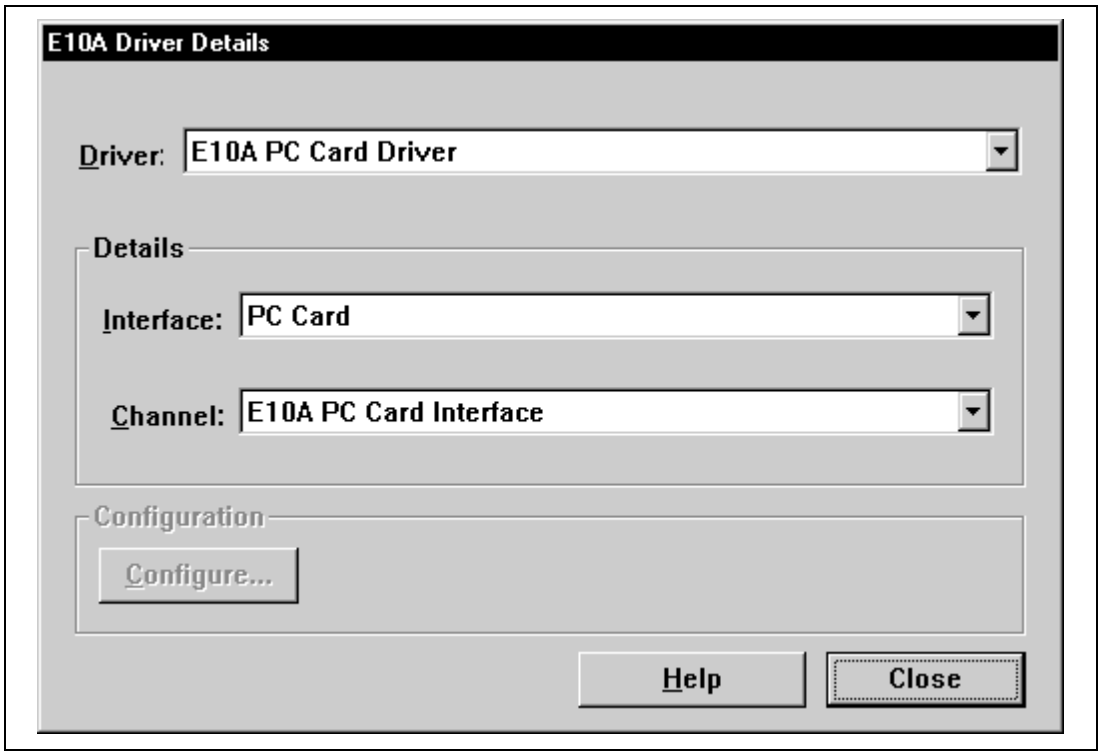


Figure 4.4 [E10A Driver Details] Dialog Box

- 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. 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.

  [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 section 6.5.1, Emulator Driver Selection.

  [Interface] combo box: Select [PC Card] to use the PCMCIA card emulator.
  Select [PCI] to use the PCI card emulator. (If the driver is not installed, the [PC Card] or [PCI] is not displayed.)

- Click the [Close] button.

4.2.3 [Breakpoints] Window

Function:

This window lists all break conditions that have been set.

Window:

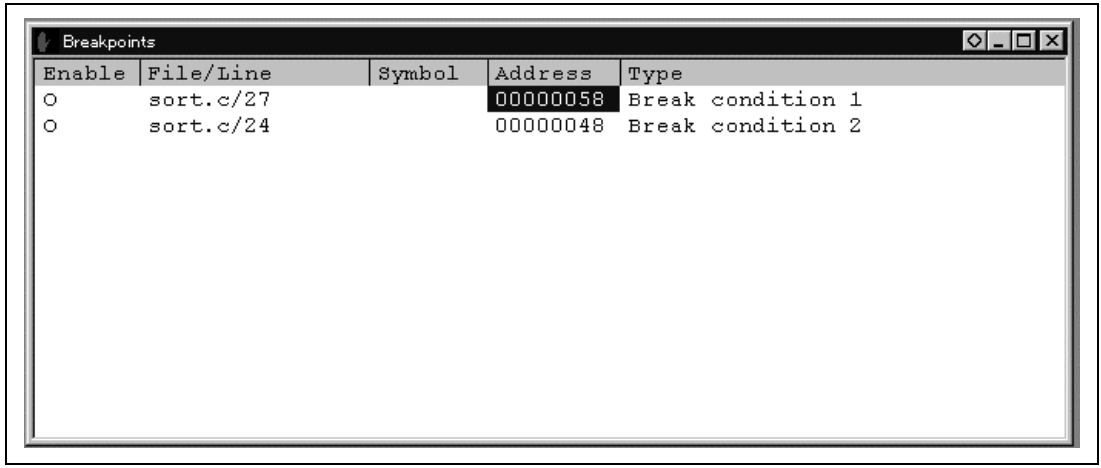


Figure 4.5 [Breakpoints] Window

Description:

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

Table 4.4 [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, a blank 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 MCU does not support the MMU.

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 4.5 [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.

4.2.4 [Break] Dialog Box

Function:

This dialog box displays the break condition settings.

Window:

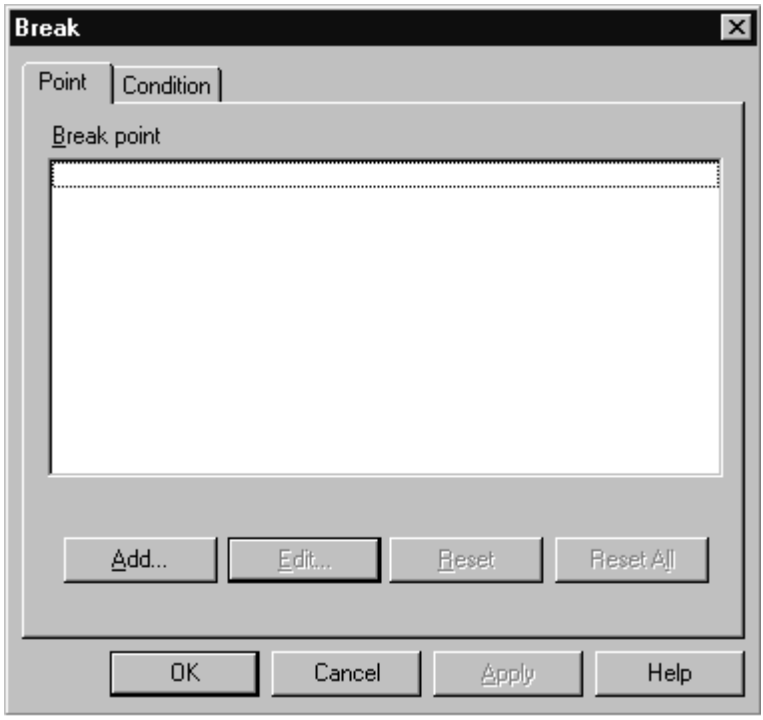


Figure 4.6 [Break] Dialog Box

Description:

The [Break] dialog box consists of the pages listed in table 4.6.

Table 4.6 [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.

(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:

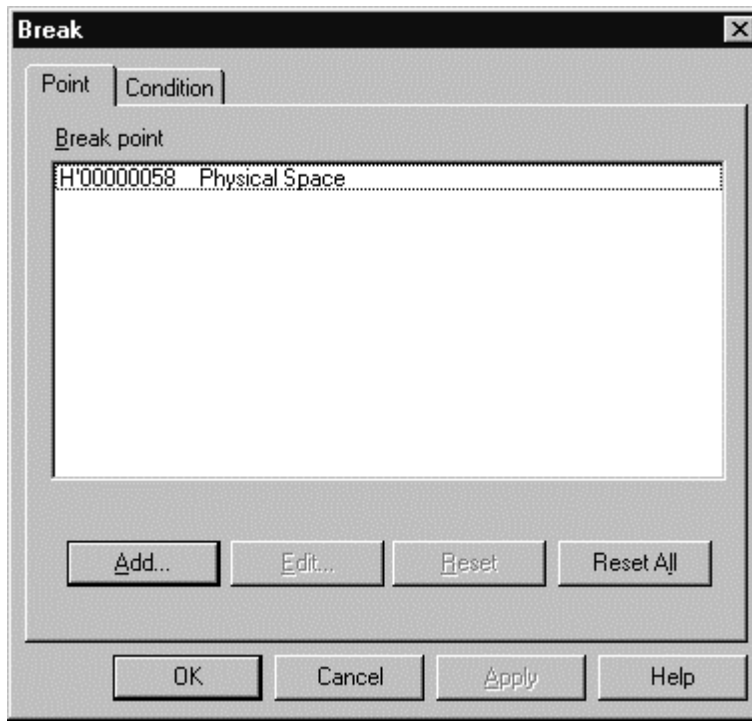


Figure 4.7 [Point] Page ([Break] Dialog Box)

Description:

Table 4.7 [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

(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.

Note: The function will be different according to the MCUs used.

Window:

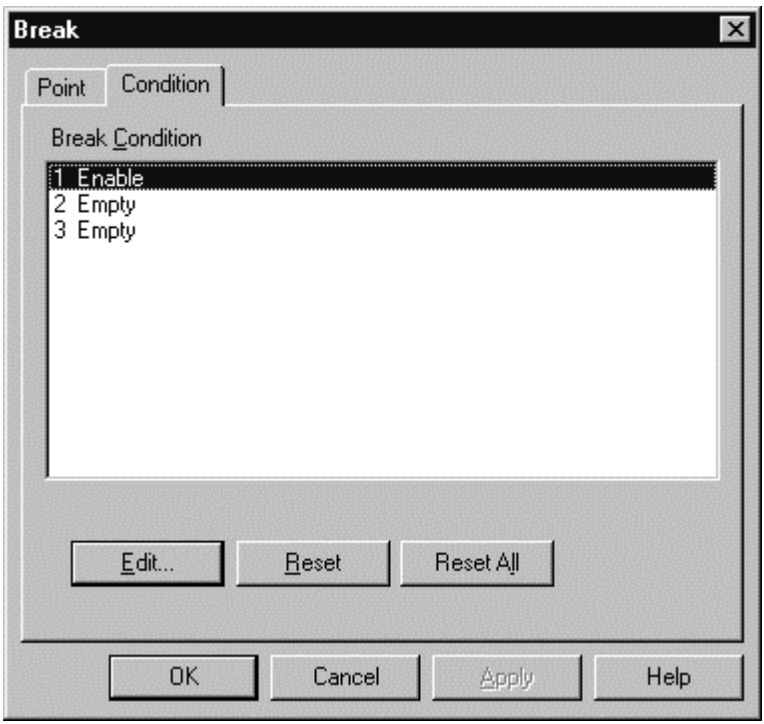


Figure 4.8 [Condition] Page ([Break] Dialog Box)

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

Description:

Table 4.8 [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

4.2.5 [Break Point] Dialog Box

Function:

This dialog box sets software breakpoints.

Window:

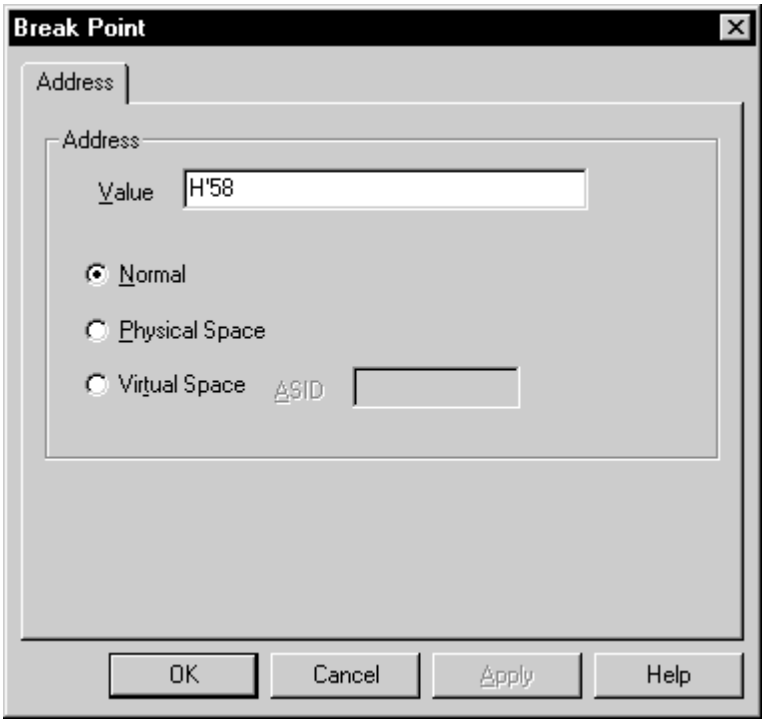


Figure 4.9 [Break Point] Dialog Box

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

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 4.9 [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 when the breakpoint address is in the virtual area. Nothing is set as default.*

Note: These options are not supported in products supporting an MCU 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

4.2.6 [Break Condition] Dialog Box

Function:

This dialog box sets hardware break conditions.

Note: The function will be different according to the MCUs used.

Window:

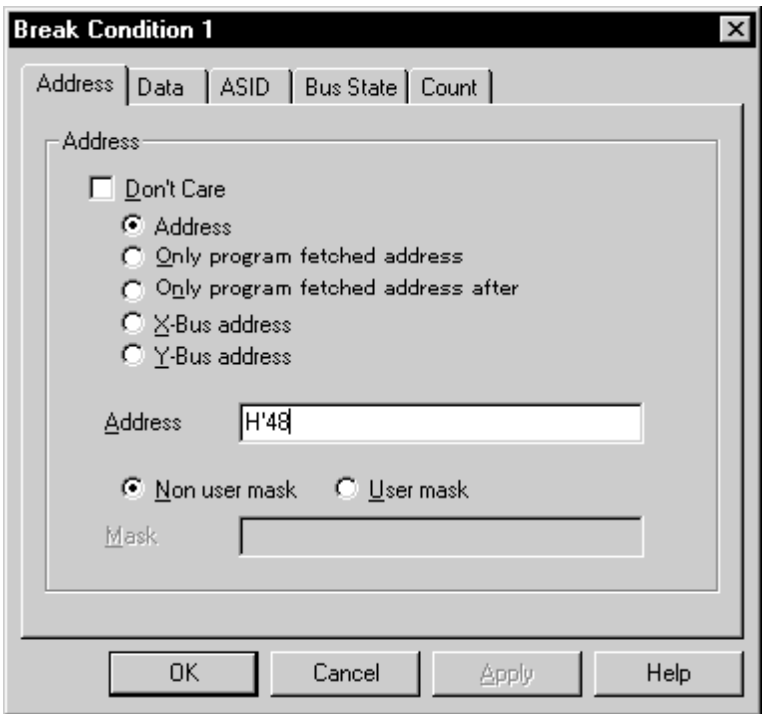


Figure 4.10 [Break Condition] Dialog Box

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

Description:

Contents to be set by each page are described in section 4.2.7, [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

4.2.7 [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 4.10. For details, refer to section 6.5.2, Hardware Break Functions.

Table 4.10 Setting Conditions in [Break Condition] Dialog Boxes

	Type				
Dialog Box	**Address Bus Condition**	**Data Bus Condition**	**Bus State and Read/Write Conditions**	**Count Condition**	**LDTLB Instruction Break, Internal I/O Access Break**
[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.

Table 4.11 shows all the [Break Condition] dialog box pages.

Note: The function will be different according to the MCUs used.

Table 4.11 [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 satisfuction count 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, Hardware Break Functions, or to the on-line help.

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

Function:

This page sets the address bus conditions.

Window:

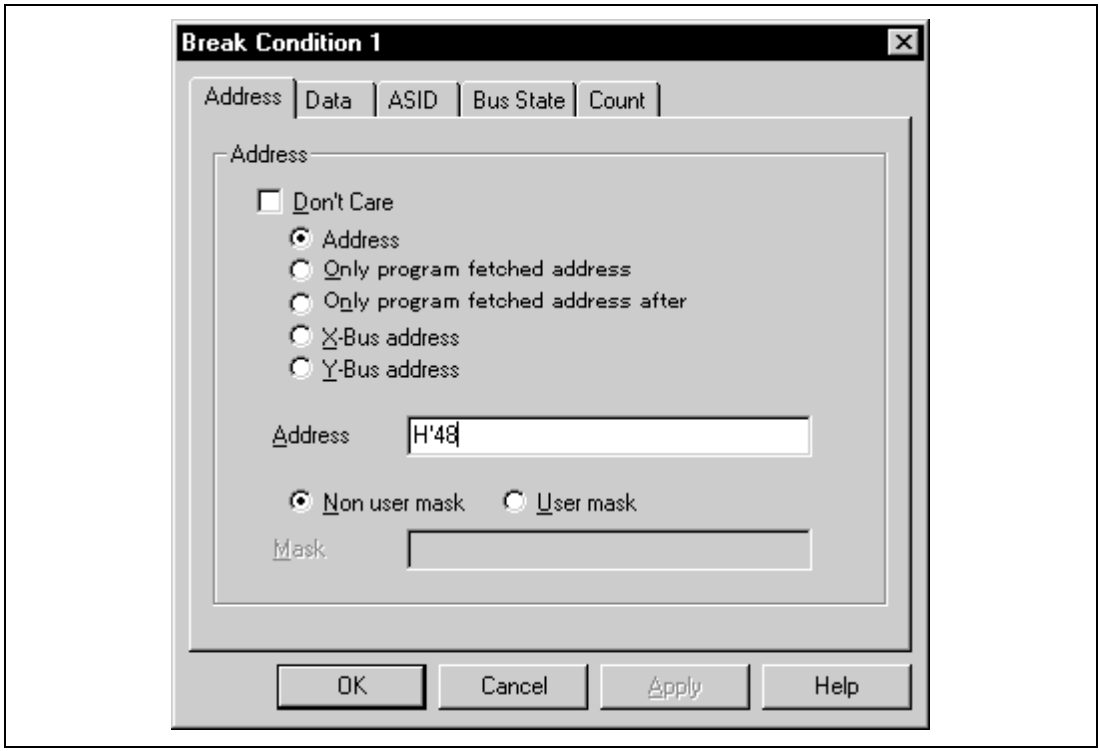


Figure 4.11 [Address] Page ([Break Condition 1] Dialog Box)

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

Description:

Table 4.12 [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 address bus as a break condition. Can be set only with Break Condition 1.
[Y-bus address] radio button	Sets the Y address bus 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 4.13 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, Hardware Break Functions, or to the on-line help.

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

Function:

This page sets the data bus conditions.

Window:

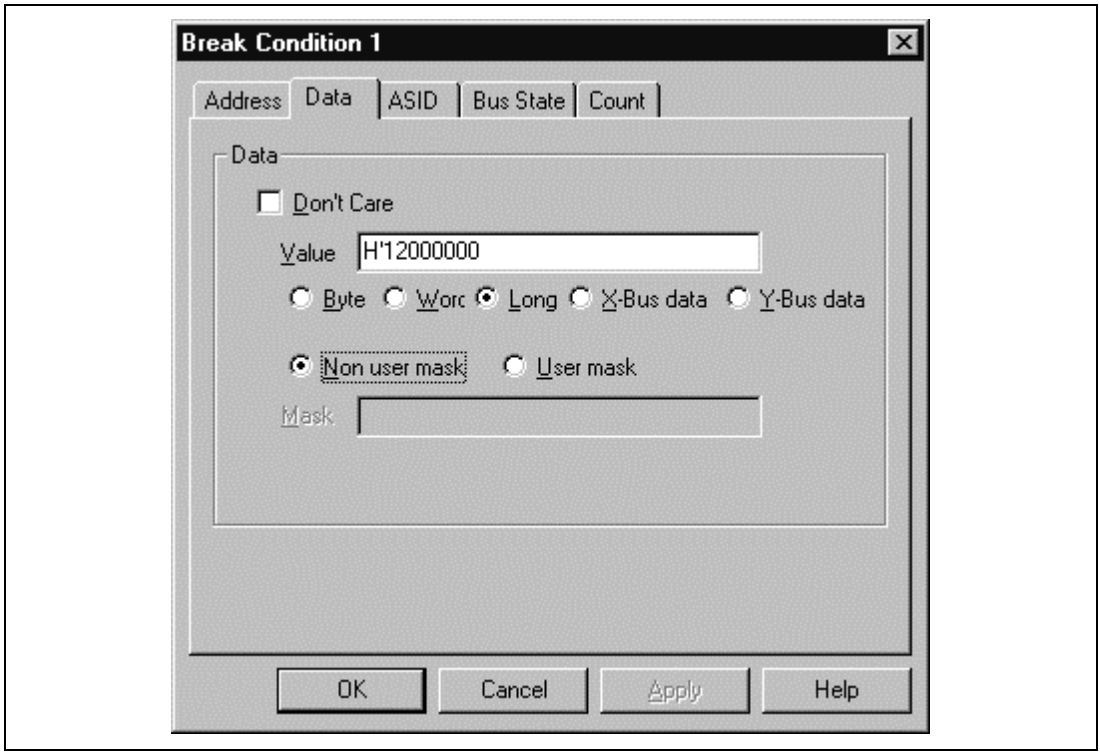


Figure 4.12 [Data] Page ([Break Condition 1] Dialog Box)

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

Description:

Table 4.14 [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.

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

Function:

This page sets the ASID conditions.

Window:

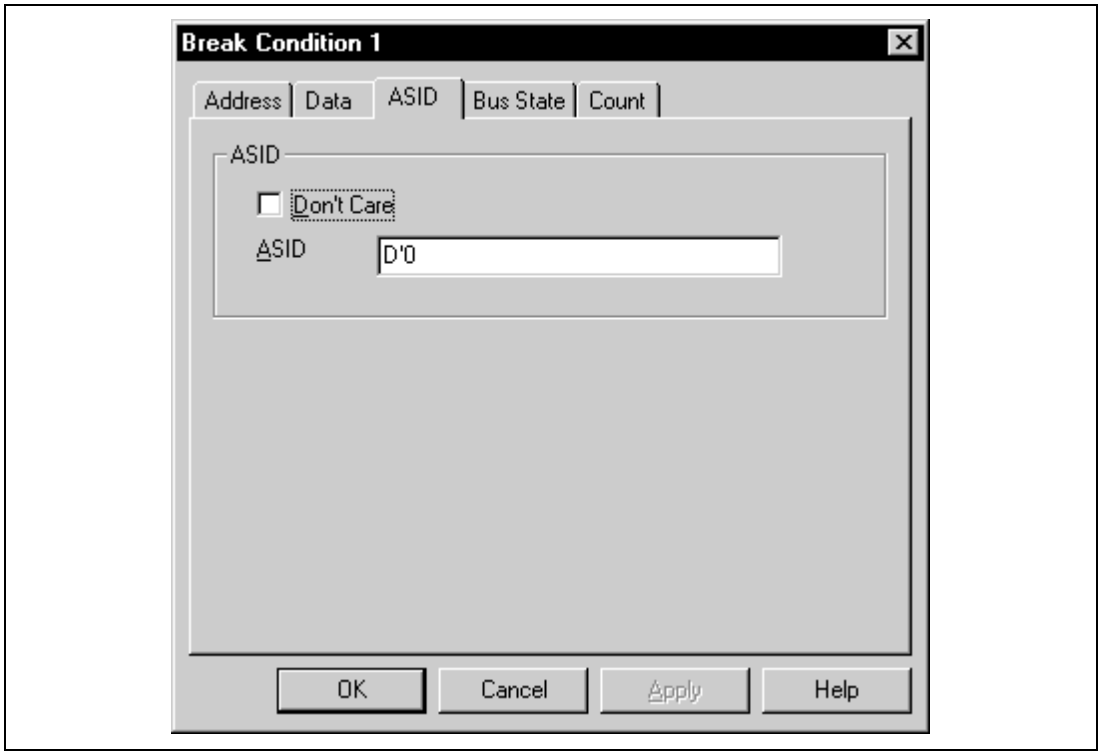


Figure 4.13 [ASID] Page ([Break Condition] Dialog Box)

Description:

Table 4.15 [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 products supporting an MCU 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:

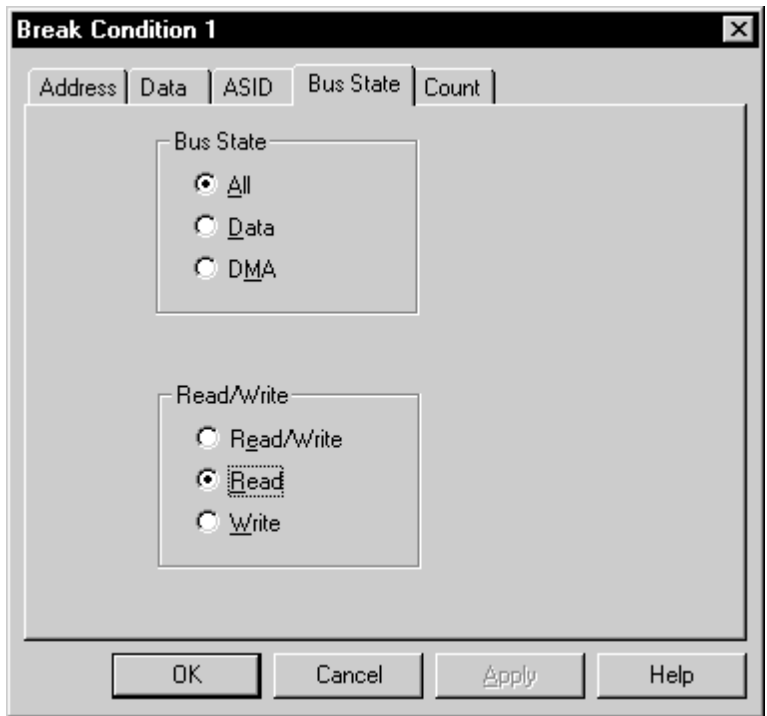


Figure 4.14 [Bus State] Page ([Break Condition] Dialog Box)

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

Description:

Table 4.16 [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.

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

Function:

This page sets the conditions for Break Condition 1.

Window:

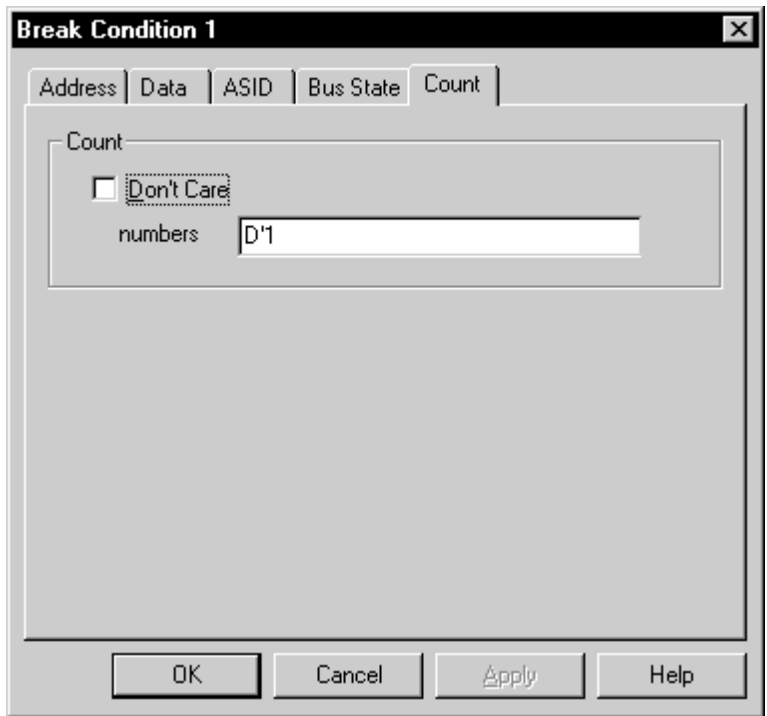


Figure 4.15 [Count] Page ([Break Condition] Dialog Box)

Table 4.17 [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 on-line help.

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

Function:

This page sets the conditions for Break Condition 3.

Window:

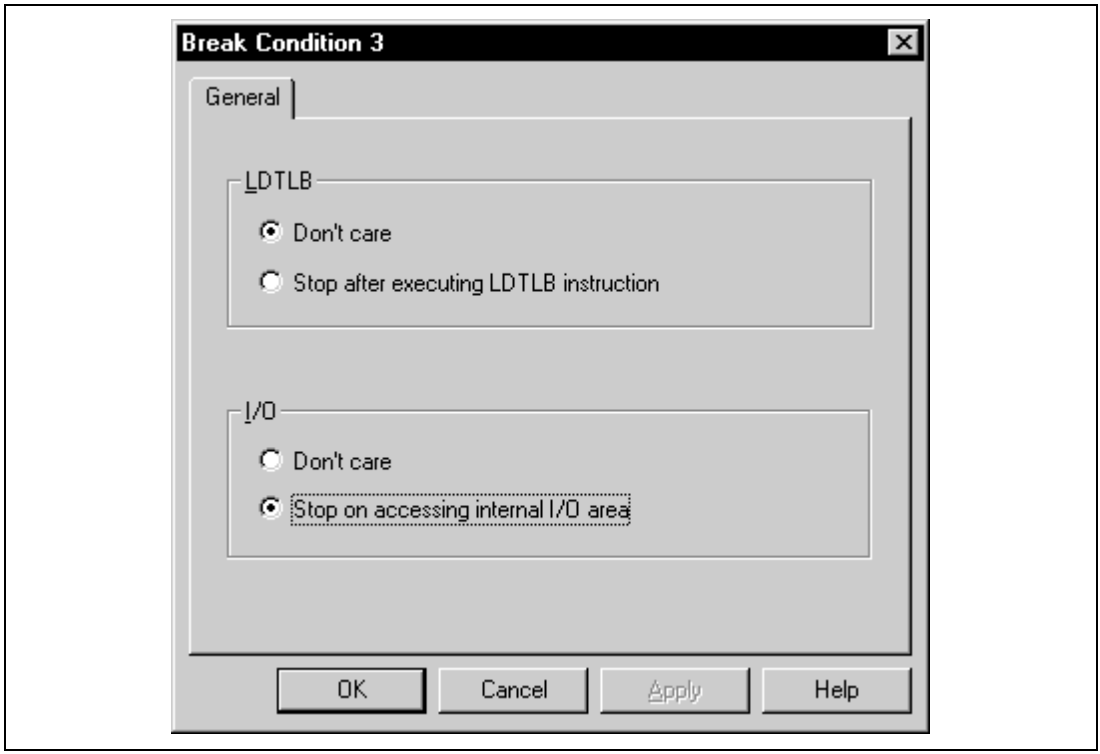


Figure 4.16 [General] Page ([Break Condition] Dialog Box)

Description:

Table 4.18 [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.
[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.

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

4.2.8 [Trace] Window

Function:

This window displays the trace buffer contents.

Window:

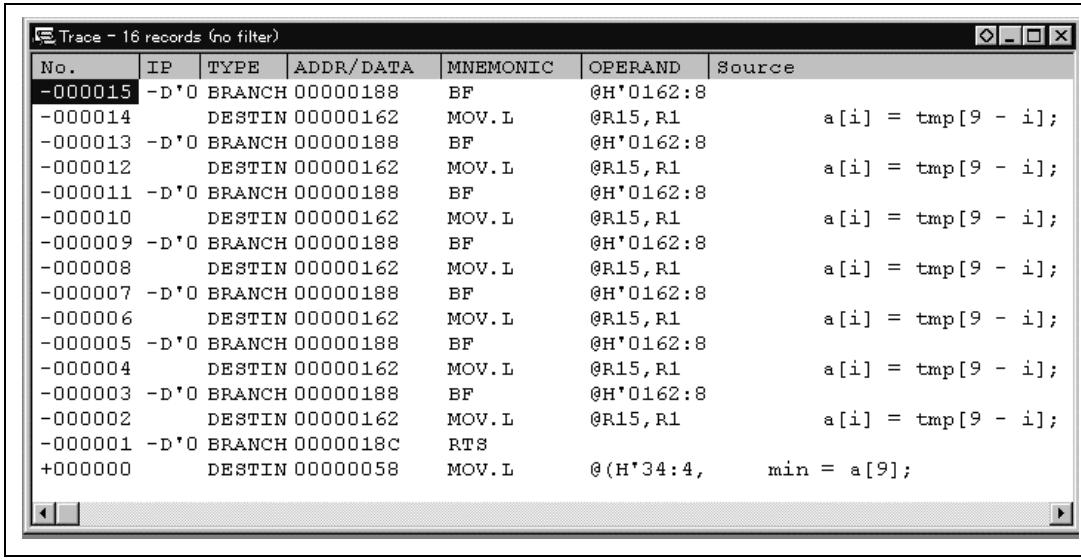


Figure 4.17 [Trace] Window

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

Description:

This window displays the trace buffer contents. The items listed in table 4.19 are displayed.

Table 4.19 [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]	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

4.2.9 [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 4.20 [Trace Acquisition] Page Options

Page Name	Function
[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:

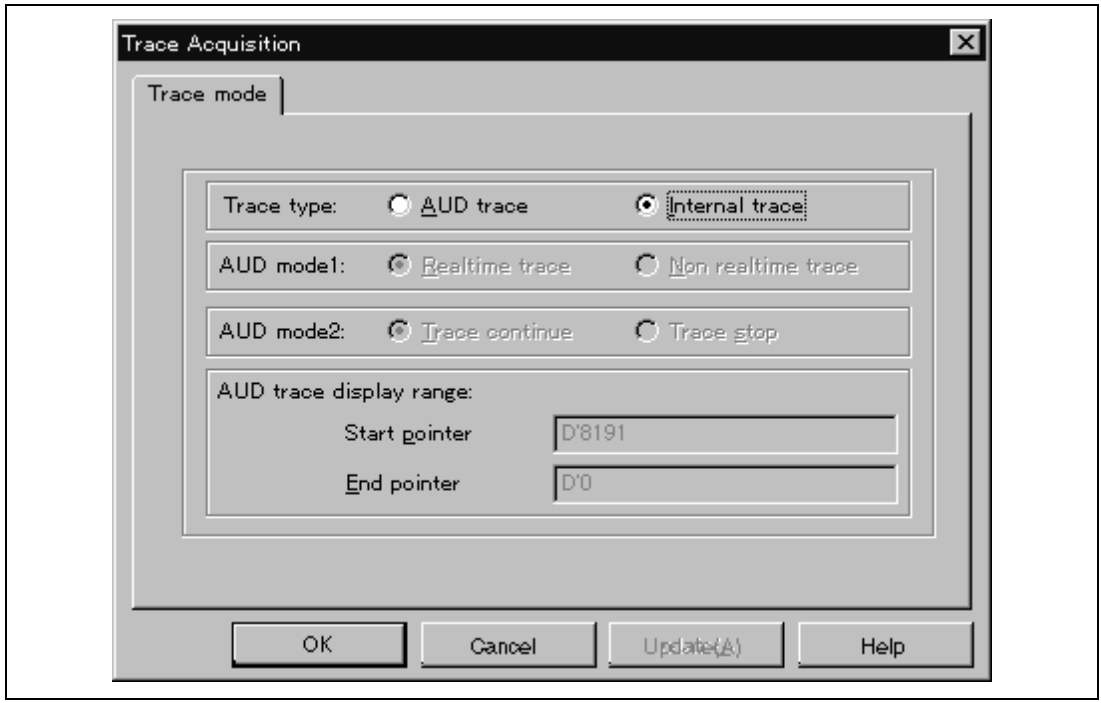


Figure 4.18 [Trace mode] Page ([Trace Acquisition] Dialog Box)

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 on-line help.

Description:

Table 4.21 [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

4.2.10 [System Status] Window

Function:

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

Window:

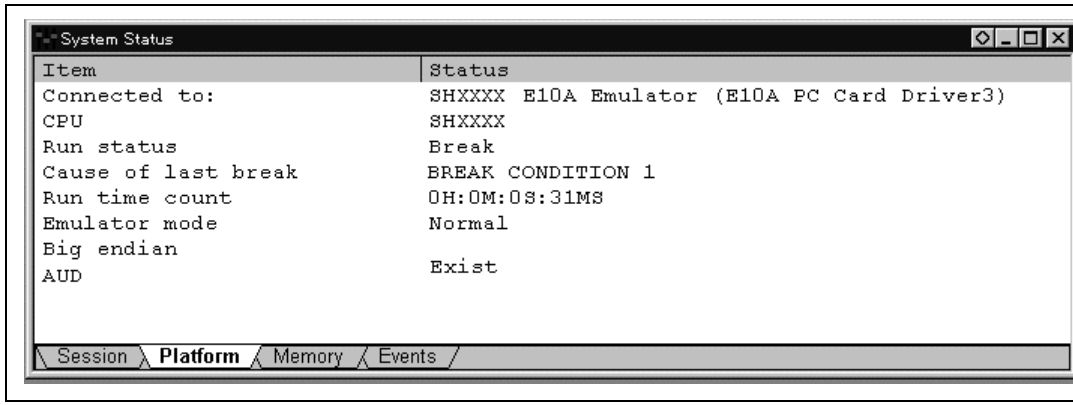


Figure 4.19 [System Status] Window

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 on-line help.

Description:

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

Table 4.22 [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 MCU 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, M: minutes, S: seconds, and MS: milliseconds. In this example, 0H:0M:0S:31MS 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.

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.

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 settings.
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 MCU type currently selected.
13	GO_OPTION	GP	Sets or displays the emulation mode during user program execution.
14	JTAG_CLOCK	JCK	Displays and sets a JTAG clock (TCK) frequency.
15	MEMORYAREA_SET	MAS	Displays and sets memory area at command input, such as load, verify, save, memory display, or memory change.

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	Sets or displays the enable or disable status of interrupts during step execution.
20	TRACE_DISPLAY	TD	Displays acquired trace buffer information.
21	UBC_MODE	UM	Sets or displays 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 on-line help.

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 is different according to the MCUs 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 be different 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 vary with the product. For the specifications of each product, refer to the on-line help.

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

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.

Notes: 1. When <option1> and <option2> are omitted, the current setting conditions are displayed.
2. Options may be different according to emulator products. For details, refer to section 6.5.5, Trace Functions.

Note: The parameters for this command vary with the product. For the specifications of each product, refer to the on-line help.

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

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.

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

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 canceled.

Examples:

To clear all hardware breakpoints:

```
>bcc (RET)
```

To clear all hardware breakpoints set at channel 2:

```
>bcc channel 2 (RET)
```

Related Items:

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

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 vary with the product. For the display specifications of each product, refer to the on-line help.

Related Items:

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

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: **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 be different according to the MCUs used. For functions of each emulator product, refer to section 6.5.2, Hardware Break 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>

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, Hardware Break 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. **x:** X-Bus address bus access **y:** Y-Bus address bus access
<maskdata>	Character string	Mask specification for desired address bus and data bus 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

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 1 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 break **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 vary with the product. For the specifications of each product, refer to the on-line help.

Examples:

To set the following conditions for channel 1 hardware breakpoints:

<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 breakpoints:

<addropt> item: Deletes 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 breakpoints:

<addropt> item: Deletes 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 breakpoints:

<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 breakpoints:

<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

5.2.8 BREAKPOINT: BP

Description:

Sets software breakpoints.

Note: The function will be different according to the MCUs 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: **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 vary with the product. For the specifications of each product, refer to the on-line help.

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

5.2.9 BREAKPOINT_CLEAR: BC

Description:

Cancels 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: **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 canceled.
2. When <address_space> and <asidopt> are not set, all software breakpoints that match the specified address are canceled.

Note: The parameters for this command vary with the product. For the specifications of each product, refer to the on-line help.

Examples:

To cancel all breakpoints:

```
>bc (RET)
```

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

```
>bc H'1000000 (RET)
```

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

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

To cancel 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 vary with the product. For the display specifications of each product, refer to the on-line help.

Related Items:

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

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: **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: **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 vary with the product. For the specifications of each product, refer to the on-line help.

2. The items displayed with this command vary with the product. For the display specifications of each product, refer to the on-line help.

Related Items:

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

5.2.12 DEVICE_TYPE: DE

Description:

Displays the currently selected MCU.

Format:

```
de
```

Table 5.13 DEVICE_TYPE Command Parameter

Parameter	Type	Description
None		

Example:

To display the currently selected MCU:

```
>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 is different according to emulator products. For details, refer to on-line help.
2. The parameters for this command vary with the product. For the specifications of each product, refer to the on-line help.

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 vary with the product. For the display specifications of each product, refer to the on-line help.

Related Items:

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

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).

```
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 is different according to the MCUs used. For details, refer to section 6.5.4, Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK).

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
```

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 1 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.

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)
```

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 big endian (E10A PCI
                       Card Driver2)
CPU                    SHxxxx
Run status             Break
Cause of last break    BREAK POINT
Run time count         0H:0M:0S:10MS
Emulator mode          Normal
Big endian
AUD                    Exist
```

Note: The items displayed with this command vary with the product. For the display specifications of each product, refer to the on-line help.

5.2.19 STEP_INTERRUPT: SI

Description:

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

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
```

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		

Note

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 ***

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

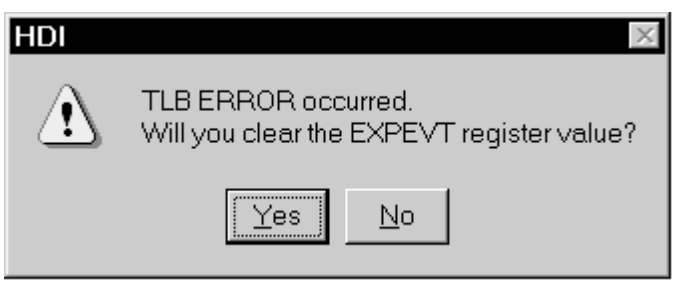


Figure 5.1 TLB Error Message Dialog

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:

BCS and TM commands
[Trace] window
[Break], [Break Condition], and [Trace Acquisition] dialog boxes

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 vary with the product. For the specifications of each product, refer to the on-line 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 vary with the product. For the display specifications of each product, refer to the on-line help.

Related Item:

[Configuration] dialog box

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 products supporting an MCU in which the MMU is not built-in.

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 products supporting an MCU in which the MMU is not built-in.

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 products supporting an MCU 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. Cancel 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 products supporting an MCU in which the MMU is not built-in.

Section 6 SH7055 E10A Emulator Specifications

6.1 Overview of the SH7055 E10A Emulator

In the SH7055 E10A emulator, the SH7055 MCM enables realtime emulation. In addition, realtime data tuning with the AUD function is also enabled.

The SH7055 MCM performs data calibration in a system that uses the SH7055 F-ZTAT™. It emulates the ROM parts through the SRAM unlike the SH7055 F-ZTAT™.

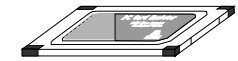
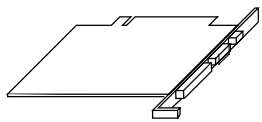
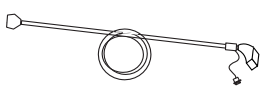
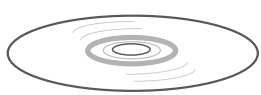
The differences in the memory map and operation mode between the SH7055 F-ZTAT™ and the SH7055 MCM are described below:

1. Extended internal ROM and RAM areas
   SH7055 F-ZTAT™: 512-kB internal ROM, 32-kB internal RAM
   SH7055 MCM: 1-MB internal ROM and 64-kB internal RAM
   Use the extended memory areas only for debugging. Do not access areas within H'FFFEE000 to H'FFFEFFFF of the extended internal RAM area because the emulator program uses this area.
2. Different operation modes
   SH7055 F-ZTAT™: MCU extension mode (valid or invalid internal ROM)
   MCU single-chip mode
   Boot mode
   User program mode
   Writer mode
   SH7055 MCM: MCU single-chip mode
   Boot mode (However, in the emulator, only the MCU single-chip mode can be used.)

When the emulator is activated, be sure to turn on the emulator in the MCU single-chip mode. For details, refer to section 7, Activation Sequence and Specific Commands of the SH7055 E10A Emulator.

Table 6.1 lists the components of the emulator.

Table 6.1 Components of the Emulator (HS7055KCM01H, HS7055KCM02H, HS7055KCI01H, or HS7055KCI02H)

Classi-fication	Component	Appearance	Quan-tity	Remarks
Hard-ware	Card emulator (MODEL name: HS0005KCM04H, HS0005KCM03H, HS0005KCI04H, or HS0005KCI03H)	 or 	1	HS0005KCM04H (PCMCIA: 36-pin type): Depth: 85.6 mm, Width: 54.0 mm, Height: 5.0 mm, Mass: 28.0 g HS0005KCM03H (PCMCIA: 14-pin type): Depth: 85.6 mm, Width: 54.0 mm, Height: 5.0 mm, Mass: 27.0 g HS0005KCI04H (PCI: 36-pin type): Depth: 122.0 mm, Width: 96.0 mm, Mass: 90.0 g HS0005KCI03H (PCI: 14-pin type): Depth: 144.0 mm, Width: 105.0 mm, Mass: 93.0 g
	User system interface cable		1	HS0005KCM04H (PCMCIA: 36-pin type): Length: 30 cm, Mass: 55.0 g HS0005KCM03H (PCMCIA: 14-pin type): Length: 50 cm, Mass: 33.0 g HS0005KCI04H (PCI: 36-pin type): Length: 80 cm, Mass: 69.0 g HS0005KCI03H (PCI: 14-pin type): Length: 150 cm, Mass: 86.0 g
Soft-ware	SH7055 E10A emulator setup program, SH7055 E10A Emulator User's Manual, and Hitachi Debugging Interface User's Manual		1	HS7055KCM01SR, HS7055KCM01HJ, HS7055KCM01HE, HS6400DIIW4SJ, and HS6400DIIW4SE (provided on a CD-R)

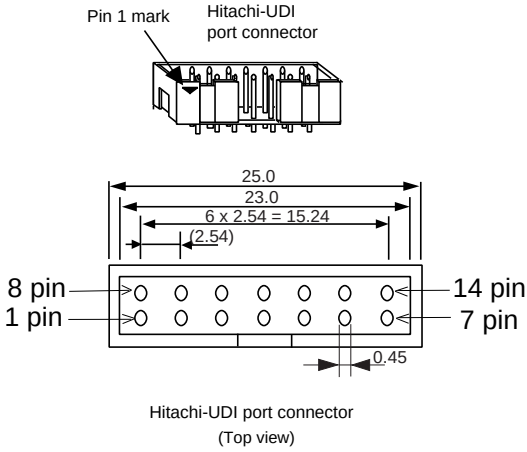
Note: The SH7055 E10A emulator does not operate on the actual chip. The SH7055 MCM is available. However, since the emulator does not operate on the chip according to the type number, ask Hitachi's sales department.

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	SH7055 MCM Pin No.
1	TCK	Input	236
2*2	/TRST	Input	233
3	TDO	Output	235
4	Not connected	—	—
5	TMS	Input	232
6	TDI	Input	234
7*2	/RESET	Output	58
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 through the UCONNECT and detects whether or not the user system is connected.

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

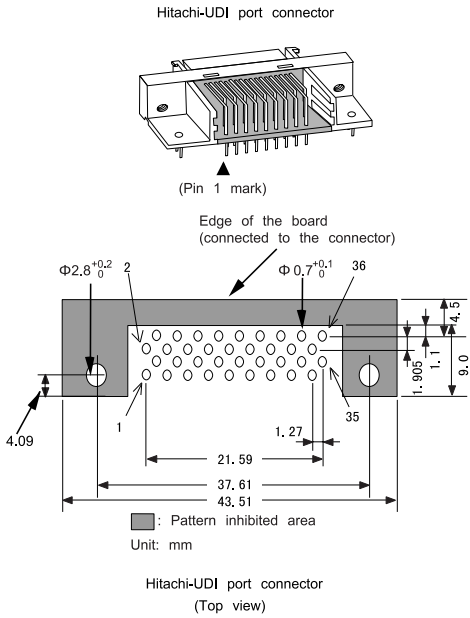
Notes: 1. Handling of the TCK, TMS, TDI, TDO, and /TRST pins depends on the use conditions of the Hitachi-UDI as follows:

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

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

2. The /RESET signal in the user side is input to the 58 pin of the SH7055. Connect this pin to the Hitachi-UDI port connector as the output from the user system.

Figure 6.2 shows the pin arrangement of the Hitachi-UDI port connector (36 pins).



Pin No.	Signal	Input/ Output*1	SH7055 MCM Pin No.	Note	Pin No.	Signal	Input/ Output*1	SH7055 MCM Pin No.	Note
1	AUDCK	I/O	245		19	TMS	Input	232	
2	GND	——			20	GND	——		
3	AUDATA0	I/O	241		21	/TRST*2	Input	233	
4	GND	——			22	GND	——		
5	AUDATA1	I/O	242		23	TDI	Input	234	
6	GND	——			24	GND	——		
7	AUDATA2	I/O	243		25	TDO	Output	235	
8	GND	——			26	GND	——		
9	AUDATA3	I/O	244		27	NC	——		
10	GND	——			28	GND	——		
11	/AUDSYNC*2	I/O	246		29	CK	Output	48	
12	GND	——			30	GND	——		
13	AUDRST	Input	238		31	/RESET*2	Output	58	User reset
14	GND	——			32	GND	——		
15	AUDMD	Input	240		33	GND	——		
16	GND	——			34	GND	——		
17	TCK	Input	236		35	NC	——		
18	GND	——			36	GND	——		

Notes: 1. Input to or output from the user system
2. The slash (/) means that the signal is low-active.

Figure 6.2 Pin Arrangement of the Hitachi-UDI Port Connector (36 Pins)

Notes: 1. Handling of the TCK, TMS, TDI, TDO, and /TRST pins depends 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, and /TRST pins must be pulled up by a resistance of several kilo-ohms.

The AUDATA3 to AUDATA0, /AUDSYNC, and AUDMD pins must be pulled up by a resistance of several kilo-ohms. The resistor of the AUDCK pin must be terminated (pulled up and pulled down by a resistance of several kilo-ohms).

The AUDRST pin must be pulled up by a resistance of several kilo-ohms (5 kilo-ohms or less).

(b) When the user system is used independently without using the emulator and Hitachi-UDI, the TCK, TMS, TDI, TDO, and /TRST pins must be pulled up by a resistance of several kilo-ohms. The AUDCK, AUDATA3 to AUDATA0, /AUDSYNC, and AUDMD pins must be pulled up by a resistance of several kilo-ohms.

The AUDRST pin must be pulled down by a resistance of several kilo-ohms.

2. **The /RESET signal in the user side is input to the 58 pin of the SH7055. Connect this pin to the Hitachi-UDI port connector as the output from the user system.**
3. **The CK pin is used by the emulator. When the emulator is used, connect the CK pin between the user interface connector (36 pins) and the SH7055 MCM via the buffer as shown in figure 6.3. The buffer voltage should be Vcc.**

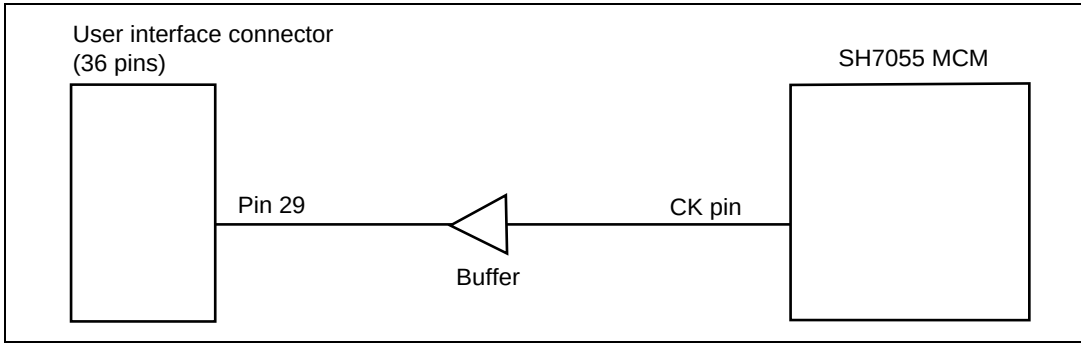


Figure 6.3 Circuit between User Interface Connector (36 Pins) and SH7055 MCM

6.3 User System Interface Circuit

The emulator is connected to the user system via the user interface cable. Figure 6.4 shows the user system interface circuit of the emulator (HS0005KCM03H).

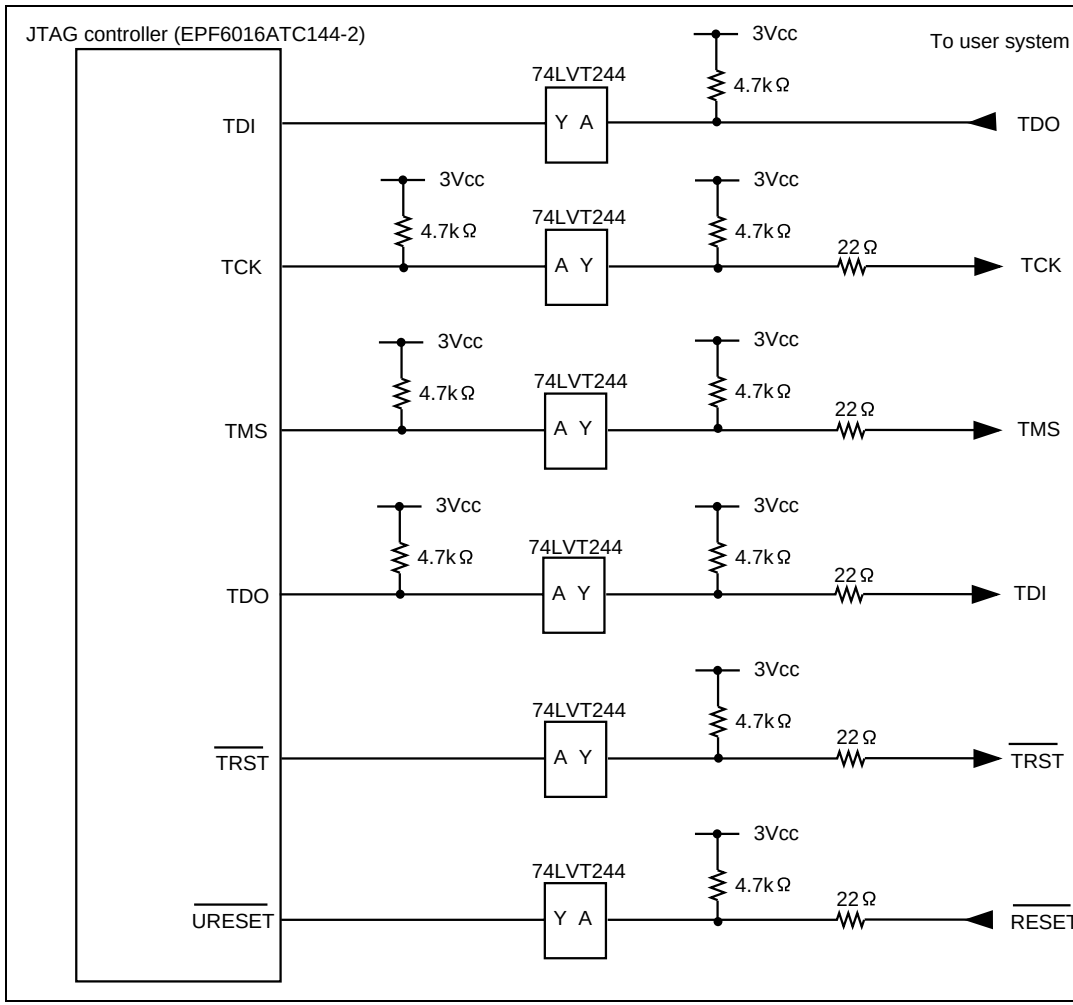


Figure 6.4 User System Interface Circuit (HS0005KCM03H)

The user system interface circuits of the emulator (HS0005KCM04H) are shown. Figures 6.5 and 6.6 show the circuits of the Hitachi-UDI pin and AUD pin, respectively.

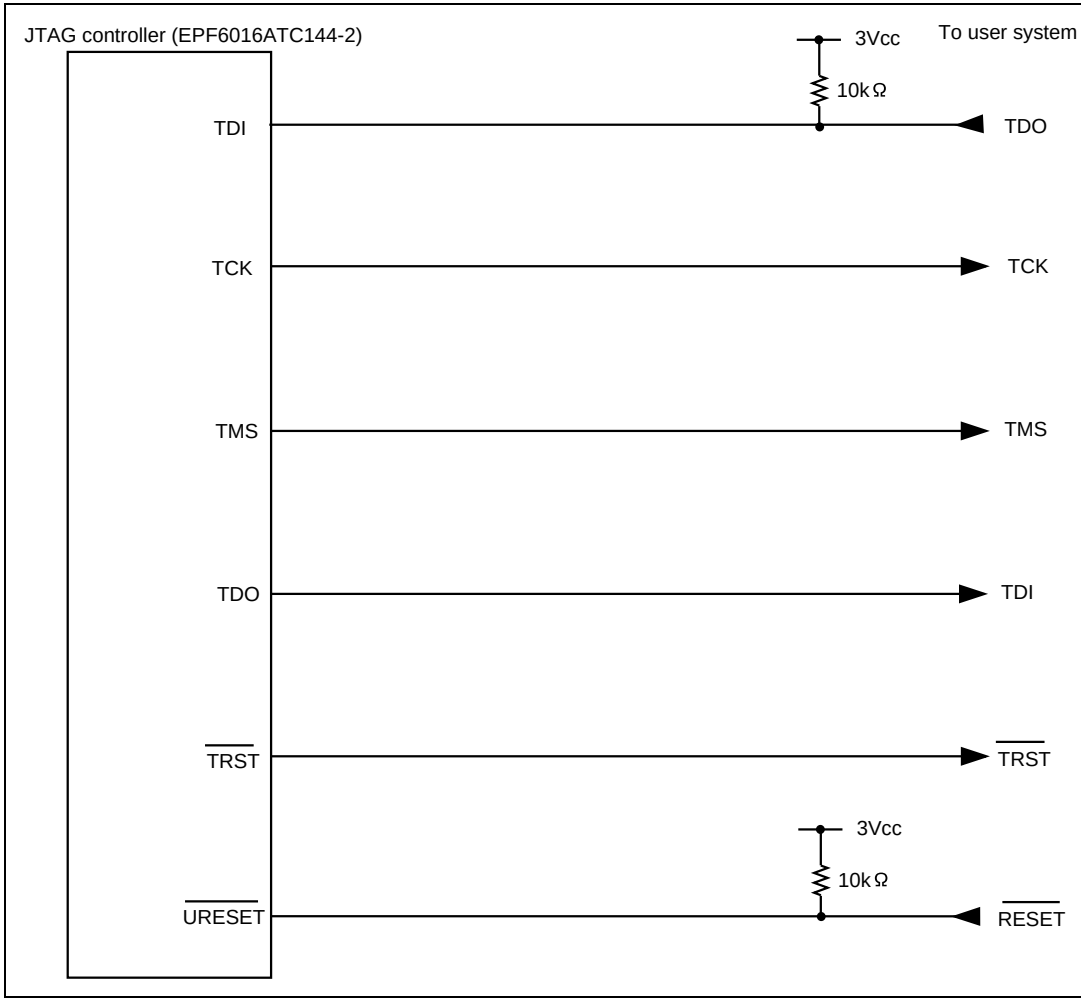


Figure 6.5 User System Interface Circuit of the Hitachi-UDI Pin (HS0005KCM04H)

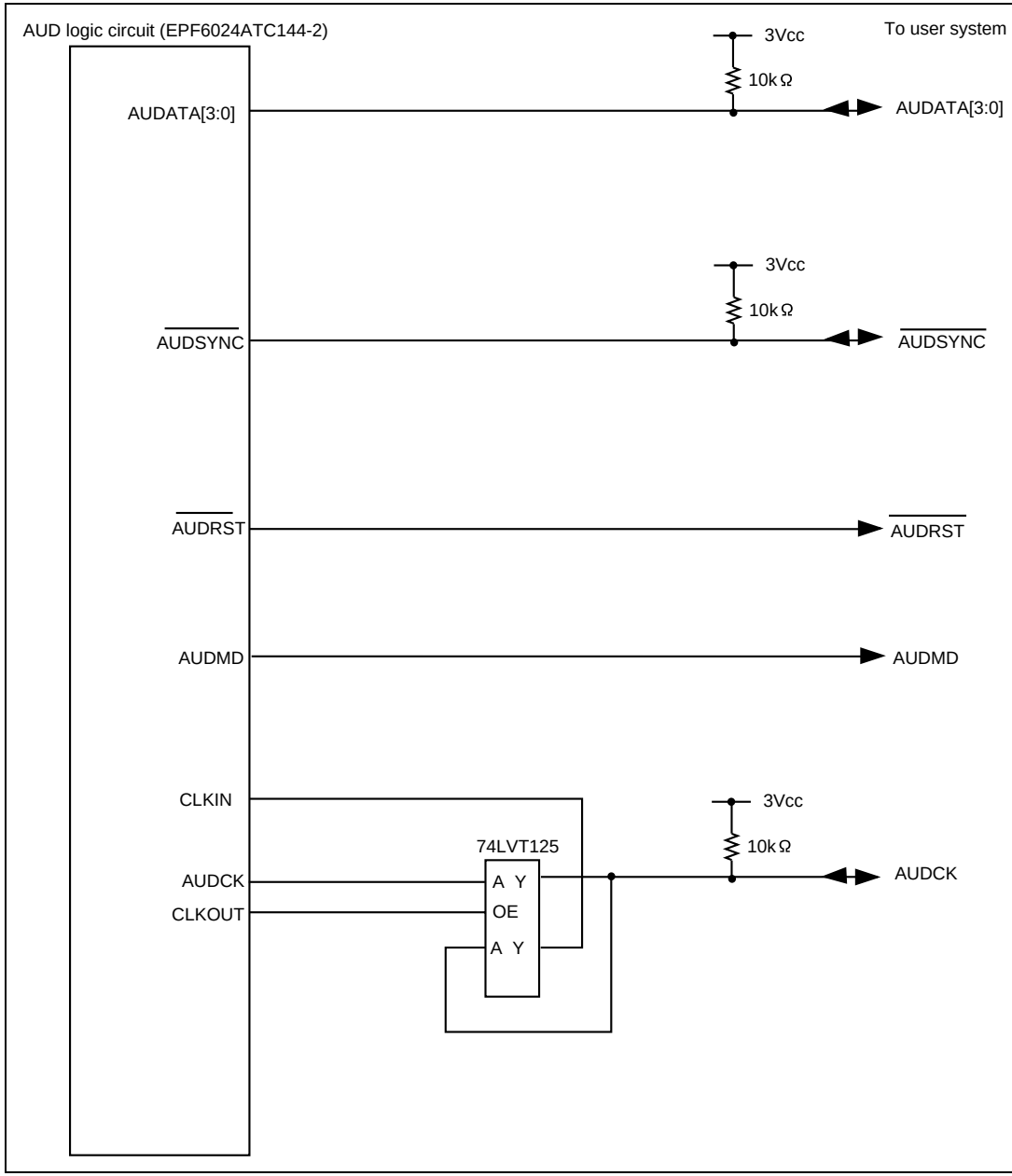


Figure 6.6 User System Interface Circuit of the AUD Pin (HS0005KCM04H)

Figure 6.7 shows the user system interface circuit of the emulator (HS0005KCI03H).

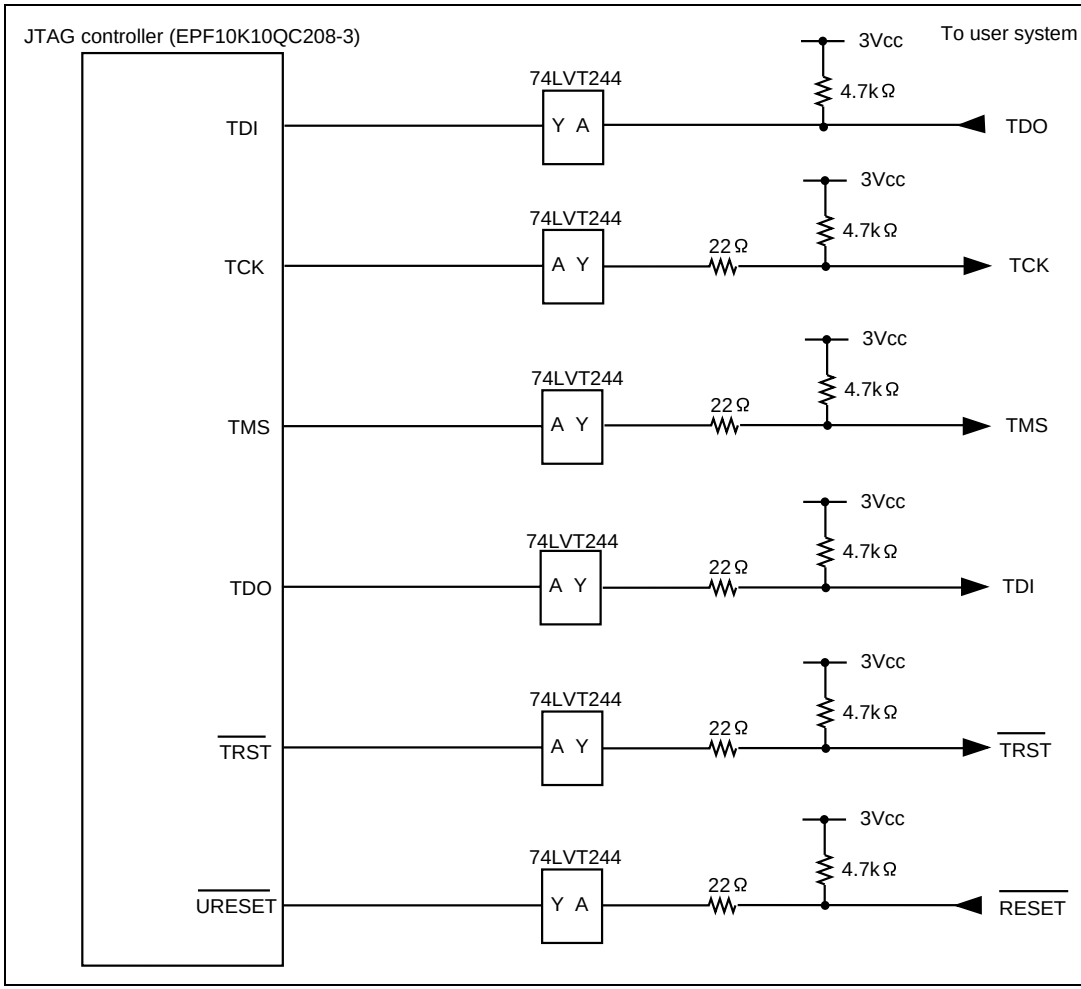


Figure 6.7 User System Interface Circuit (HS0005KCI03H)

The user system interface circuits of the emulator (HS0005KCI04H) are shown. Figures 6.8 and 6.9 show the circuits of the Hitachi-UDI pin and AUD pin, respectively.

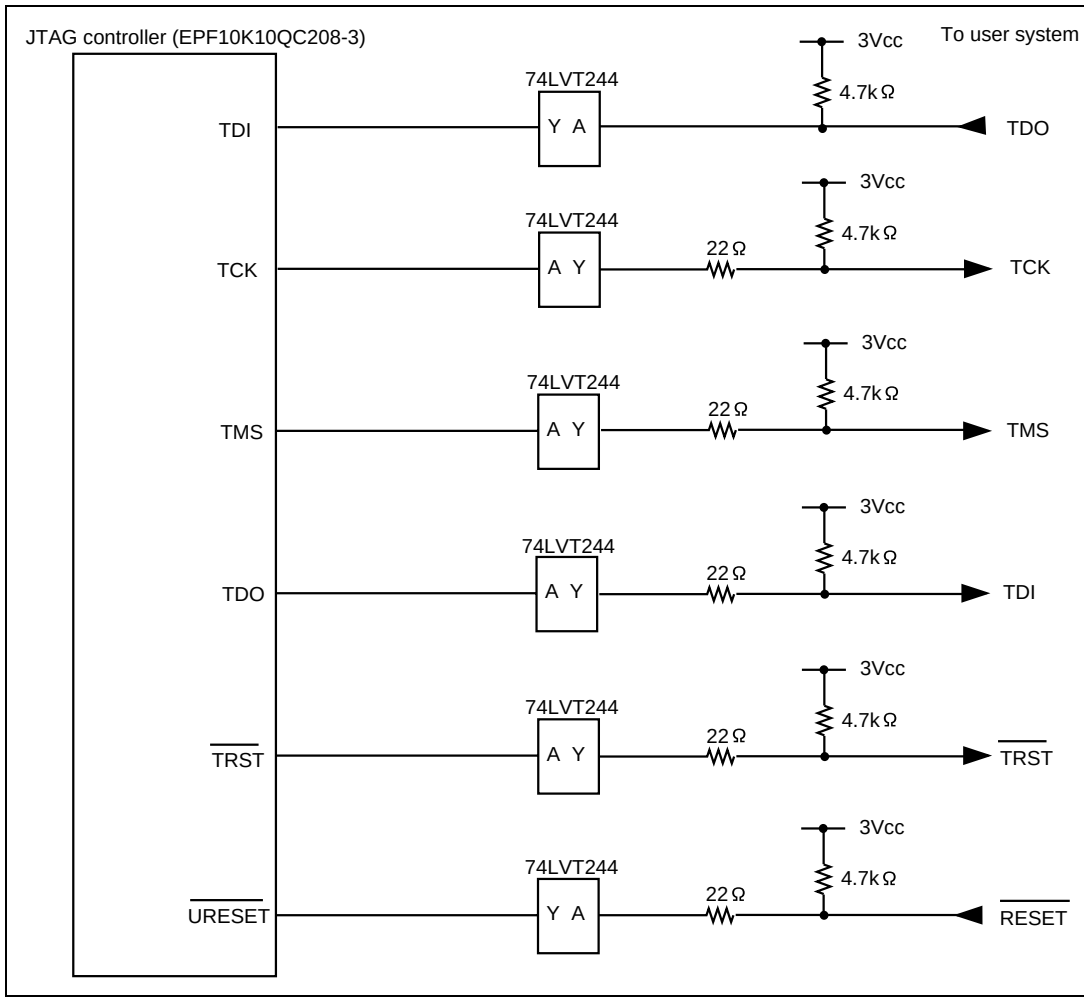


Figure 6.8 User System Interface Circuit of the Hitachi-UDI Pin (HS0005KCI04H)

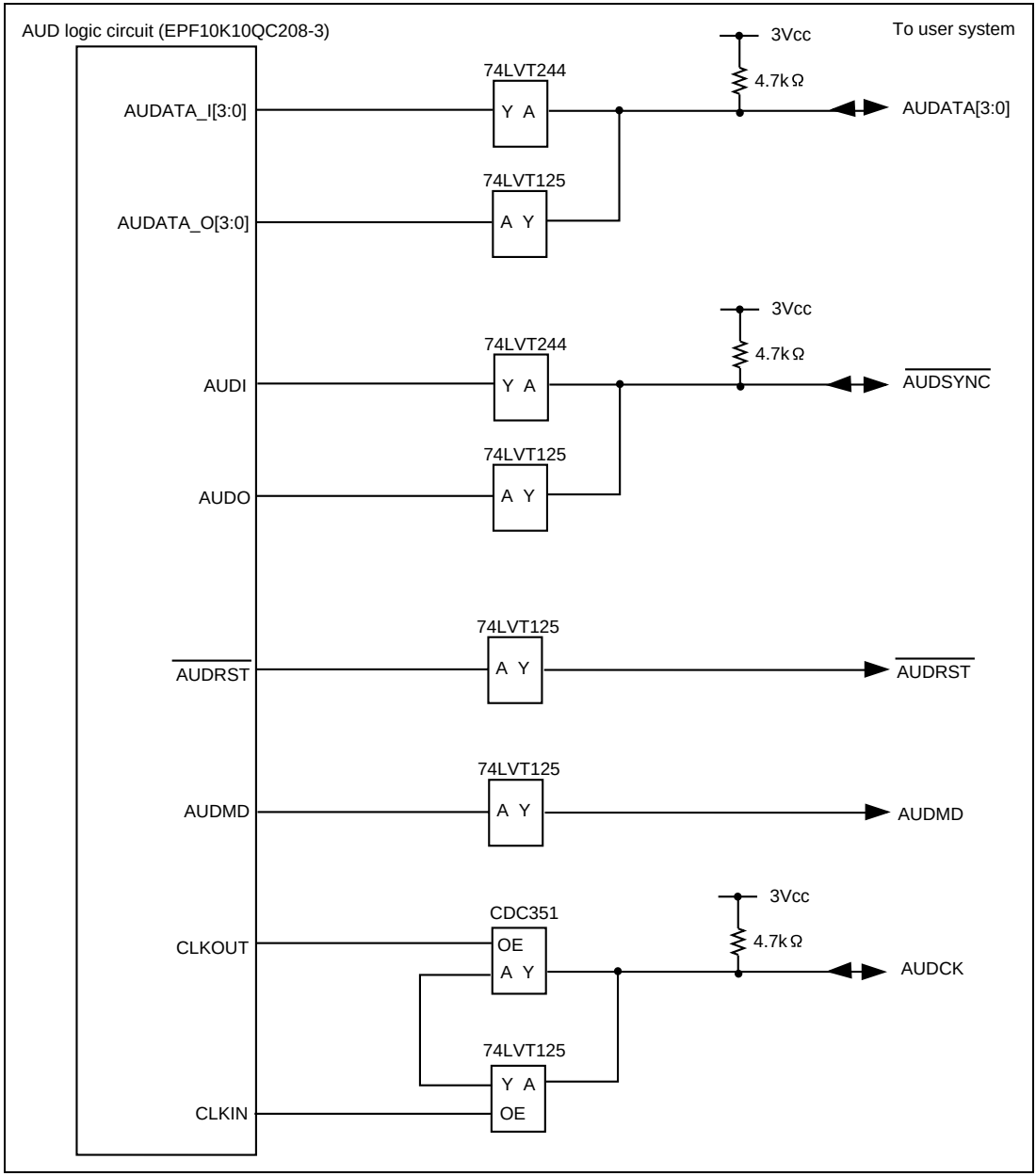


Figure 6.9 User System Interface Circuit of the AUD Pin (HS0005KCI04H)

6.4 Differences between the SH7055 F-ZTAT™ and the Emulator

- When the emulator system is initiated, it initializes the general registers and part of the control registers as shown in table 6.2. When the emulator is not used, the initial values of the actual SH7055 F-ZTAT™ general register is undefined.

Table 6.2 Register Initial Values at Emulator Power-On

Register	Initial value when the emulator is used
R0 to R14	H'00000000
R15 (SP)	Value of the SP in the vector address table
PC	Value of the PC in the vector address table
SR	H'000000E0
GBR	H'00000000
VBR	H'00000000
MACH	H'00000000
MACL	H'00000000
PR	H'00000000
FPUL	H'00000000
FPSCR	H'00040001
FR0 to FR15	H'00000000

- The emulator uses the Hitachi-UDI; do not access the Hitachi-UDI.

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

  For low-power consumption, the SH7055 F-ZTAT™ has sleep, standby, and module standby modes.

  When the emulator is used, the sleep mode can be cleared with either the normal clearing function or with the forced break. Note that, however, if a command has been entered in standby mode or module standby mode, no commands can be used from the emulator since the mode is cleared only with the normal clearing function.

Note: The memory must not be accessed or modified in software standby mode.

Do not set the MSTOP2 bit to 1 in the MSTCR register (address H'FFFFF70A) nor the RAME bit to 0 in the SYSCR register (address H'FFFFF708), since doing so may prevent the emulator from operating correctly.

When the HS7055KCM02H or HS7055KCI02H is used, do not set the following bits to 0: the MSTOP3 bit in the MSTCR register (address H'FFFFF70A) and the AUDSRST bit in the SYSCR register (address H'FFFFF708). The AUD function cannot be used.

- RESET Signal ($\overline{RES}$)

  The RESET signal can be accepted during user program break, when the peripheral modules are reset. Since the PC, SR, and SP registers are not initialized, reset them and execute the GO command when the user program is executed from the reset vector.

  When the RESET signal is input, the contents of the flash memory are always copied to the SRAM.

Notes: 1. Do not break the user program when the $\overline{RES}$, $\overline{BREQ}$, and $\overline{WAIT}$ pins are being low. A TIMEOUT error will occur. The TIMEOUT error will also occur at memory access when the $\overline{BREQ}$ and $\overline{WAIT}$ pins are fixed low during break.

2. There are limitations as follows after the RESET signal has been input:

— A break will not occur even if the break condition is matched.

— A break will not occur even if the breakpoint has been set in the SRAM area that the flash memory contents are copied.

- Direct Memory Access Controller (DMAC)

  The DMAC operates even in the command wait state. When a data transfer request is generated, the DMAC executes a DMA transfer. While the DREQ signal is being input, the CPU cannot operate. If a break occurs at this timing, a timeout error will occur. The timeout error will also occur when the DREQ signal is input when the emulator command is being issued.

- Memory Access during Emulation

  When the HS7055KCM01H and HS7055KCI01H are used, memory contents cannot be accessed during emulation.

- The emulator can rewrite the memory contents only to the RAM area. Therefore, an operation such as memory write, software break, or user program download should be performed only for the RAM area. Note that the flash memory write is not supported.

- Interrupt

  All interrupts except for NMI are masked during user program break.

- User Break Controller (UBC)

  The emulator uses the UBC to implement the Break Condition function. Therefore, if the UBC enters the module standby state, the Break Condition function cannot be used.

6.5 The SH7055 E10A Emulator Functions

In the SH7055 E10A emulator, the realtime memory access function by using the RAM monitor function has been added.

The following functions are not supported:

- MEMORYAREA_SET command
- VPMAP command
- Virtual or Physical specification in the [Configuration] window and command-line function
- Sequential break function
- STEP_INTERRUPT function
- RESET_CPU function and RESET command
- Internal trace function (The trace function for the SH7055 E10A emulator is AUD trace only.)

6.5.1 Emulator Driver Selection

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

Table 6.3 Type Name and Driver

Type Name	Driver
HS7055KCM01H	E10A PC Card Driver 3
HS7055KCM02H	E10A PC Card Driver 4
HS7055KCI01H	E10A PCI Card Driver 3
HS7055KCI02H	E10A PCI Card Driver 4

6.5.2 Hardware Break Functions

The emulator can set hardware break conditions. Table 6.4 lists these hardware break conditions.

Table 6.4 Hardware Break Condition Specification Items

Items	Description
Address bus condition (Address)	Breaks when the MCU address bus value or program counter value matches the specified value.
Data size condition (Size)	Breaks when the data size that has been accessed matches the specified value. Byte, word, or longword can be specified as the access data size.
Read or write condition (Read or Write)	Breaks in the read or write cycle.
Access type	Breaks when the bus cycle is the specified cycle.

Table 6.5 lists the combinations of conditions that can be set in the [Break Condition 1] dialog box.

Table 6.5 Conditions Set in the [Break Condition] Dialog Box

	Type	
Dialog Box	Address Bus Condition ([Address] page)	Access Type Condition, Read or Write Condition, Data Size Condition ([Bus state] page)
[Break Condition 1] dialog box	O	O

Note: O: Can be set by clicking the radio button in the dialog box.

Table 6.6 lists the combinations of conditions that can be set with the BREAKCONDITION_SET command.

Table 6.6 Conditions Set with the BREAKCONDITION_SET Command

	Type	
Channel	Address Bus Condition (<addropt> option)	Access Type Condition (<accessopt> option), Read or Write Condition (<r/wopt> option), Data Size Condition (<sizeopt> option)
Break Condition 1	O	O

Note: O: Can be set by the BREAKCONDITION_SET command.

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

1. When a hardware break condition is satisfied, emulation may stop after two or more instructions have been executed.
2. Note that the setting of the hardware break condition is cleared by power-on reset.
3. According to the Break Condition setting, a condition may be satisfied in the emulator program. In this case, a break occurs without executing any instruction when the GO command is executed the next time. This will occur when only the [Bus state] page is set. It is recommended that the condition of the [Address] page is also set.
4. When the elements that the break condition is satisfied exist in the successive addresses, there is a limitation:

   After a break occurs at the corresponding address, when the GO command is executed again, a break may occur without executing any instruction depending on the instruction location.
5. When the interrupt accept level is H'0F, normally set the level to H'0E or lower because a break does not occur when the condition of Break Condition is satisfied. When the condition of Break Condition is satisfied in the interrupt processing routine whose priority is H'0F or higher, a break does not occur until the SR.I3 to I0 bits are changed. A break occurs after the execution is returned from the interrupt processing routine.
6. If a software breakpoint has been set immediately after the condition that has been set with Break Condition, a break occurs in the emulator program. In this case, assume the place where a break occurred according to the trace result, and delete the corresponding software breakpoint.
7. The SH7055 F-ZTAT™ has limitations on the UBC. For details, refer to the SH7055 F-ZTAT™ Hardware Manual.

6.5.3 Notes on Setting the [Breakpoint] Dialog Box

1. When an odd address is set, the address is rounded down to an even address.
2. A software break is accomplished by replacing instructions. Accordingly, it can be set only to the RAM area. However, a software break cannot be set to the following addresses:
   - An area other than the CS0 to CS3 areas and the internal RAM area
   - An instruction in which Break Condition 1 is satisfied
   - A slot instruction of a delayed branch instruction
3. During step execution, software breakpoints are disabled.
4. A condition set at Break Condition 1 is disabled when an instruction at a software breakpoint is executed. Do not set a software breakpoint to an instruction in which Break Condition 1 is satisfied.
5. When execution resumes from the breakpoint address after the program execution stops at a software breakpoint, single-step execution is performed at the address before execution resumes. Therefore, realtime operation cannot be performed.
6. When a software breakpoint is set to the slot instruction of a delayed branch instruction, the PC value becomes an illegal value. Accordingly, do not set a software breakpoint to the slot instruction of a delayed branch instruction.

6.5.4 Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK)

1. When the JTAG clock (TCK) is used, set the frequency to lower than the frequency of quarter of the CPU clock.
2. The AUD clock (AUDCK) cannot be selected or changed.

   In the AUD branch trace function, the frequency of half of the system clock is automatically output. When the RAM monitor function is used, the frequency of quarter of the system clock is automatically input.

Note: The emulator can guarantee the AUD clock value of 10 MHz. Therefore, if the system clock is 20 MHz or higher, note that the contents acquired by AUD trace cannot be guaranteed.

6.5.5 AUD Functions

In the SH7055 E10A emulator, the following functions using the AUD function are added. These functions are operational when the AUD pin is connected to the emulator

Table 6.7 AUD Functions

Function	Description
Branch trace function	Displays the addresses and instruction words at the branch destination.
RAM monitor function	Enables realtime memory reading or writing during user program execution.

Note: When HS7055KCM01H and HS7055KCI01H are used, the AUD function cannot be used.

(1) Branch Trace Function

When a branch occurs during user program execution, the branch destination address is acquired. 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.

Note: After the trace buffer of the emulator becomes full, the trace information is not acquired. (The user program is continuously executed.)

(2) Realtime Memory Access Function

Realtime memory reading or writing is enabled during user program execution. The specified memory address contents (maximum three) can be displayed on the status bar. The memory contents can be changed by the command line, and can be referenced in the [Memory] window.

Memory can be read or written as follows:

1. When the [Memory] window is used:

   Memory can be read during user program execution but cannot be written. Open the address to be referenced in the [Memory] window. At reference, select Refresh from the [Memory] menu or issue the Refresh command in the command-line window.

2. When the command line is used:

   MEMORY_EDIT command: Memory can be read during user program execution but cannot be written.

   RAM_R command: Address and size displayed on the status bar are specified during user program execution.

RAM_W command: Memory can be changed during user program execution. One command can change a maximum of three addresses.

For command syntax of the RAM_R and RAM_W commands, refer to section 7.2, RAM Monitor Commands.

(3) Products Using the AUD Function and Note

Table 6.8 Type Name and AUD Function

Type Name	AUD Function
HS7055KCM01H	Not available
HS7055KCM02H	Available
HS7055KCI01H	Not available
HS7055KCI02H	Available

Note: Trace cannot be acquired while memory read/write is being performed by using the RAM monitor function during user program execution.

6.5.6 Notes on Displaying the [Trace] Window

1. The AUD trace outputs the differences between newly output branch destination addresses and the previously output branch destination addresses. If the previous branch destination address is the same as the upper 16 bits, the lower 16 bits are output. If it matches the upper 24 bits, the lower 8 bits are output. If it matches the upper 28 bits, the lower 4 bits are output. The emulator regenerates the 32-bit address from these differences and usually displays it in the [Trace] window. If the emulator cannot display the 32-bit address, it displays the difference from the previously displayed 32-bit address.
2. When a completion-type exception occurs during exception branch acquisition, the next address to the address in which an exception occurs is acquired.
3. When the [Halt] option is used from the popup menu in the [Trace] window, realtime emulation is retained.
4. In the SH7055 E10A emulator, the maximum number of trace display pointers is as follows:

   When HS7055KCM02H is used: D'16383 to -0

   When HS7055KCI02H is used: D'65535 to -0

   However, the maximum number of trace display pointers differs according to the AUD trace information to be output. Therefore, the above pointers cannot be always acquired.
5. When only one line is acquired by trace, the display data is not updated. In this case, open the [Trace] window again.

6.5.7 Notes on Executing the User Program

Table 6.9 shows notes on the use of the emulator.

Table 6.9 Notes on Executing the User Program

Item	Note
SR register	When the interrupt accept level is H'0F, set the level to H'0E or lower because the forced break is disabled. When the forced break (stop button) is issued in the interrupt processing routine whose priority is H'0F or higher, a break does not occur until the SR.I3 to I0 bits are changed. A break occurs after the execution is returned from the interrupt processing routine.
Stack area	Since the 12-byte user stack is used at a user program break, the stack pointer address should have an additional 12-byte area.
Vector	Be sure to write the following values in the vector table. If not written, the break function of the emulator might not operate correctly. H'FFFEEB0C for the address of VBR + H'0010 (illegal instruction interrupt) H'FFFEEB04 for the address of VBR + H'0030 (UBC interrupt) H'FFFEEB08 for the address of VBR + H'0038 (H-UDI interrupt) H'FFFEEB10 for the address of H'0000001C

A STEP command cannot be executed in the ROM area (data cannot be written to the ROM area).

6.5.8 Notes on Setting the UBC_MODE Command

In the [Configuration] window, if User is set while the UBC_MODE command has been set, the emulator initializes the UBC-related register to the default value. Note that the UBC enters the module standby mode.

6.5.9 Notes on HDI

1. Note on 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.

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, only its addresses are displayed in the program window. The file cannot be operated in the program window.

— 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 will be illegal.

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.

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: Link the program to be debugged with the Sdebug option and 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 should be used to specify the files in a command file so that the current directory alteration is not affected

   Example: FILE_LOAD C:\\HEW\\HDI4\\E10A\\TUTORIAL.ABS

9. About Hitachi Debugging Interface User's Manual

   This version of HDI does not support section 9, Selecting Functions, written in Hitachi Debugging Interface User's Manual.

10. Note on 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 HDI

— Automatic load of session files

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

If another HDI 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 software 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 verifying during user program execution.

14. [Performance Analysis] Window

    This HDI does not support the [Performance Analysis] window (described in section 13.7, 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.

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 in the [I/O Registers] window because the emulator uses the User Break Controller.
    - For each Watchdog Timer register and the MSTCR register related with the power-down mode, there are two registers to be separately used for write and read.

Table 6.10 Watchdog Timer and MSTCR Registers

Register Name	Usage	Register
TCSR (R)	Read	Watchdog timer control/status register
TCNT (R)	Read	Watchdog timer counter
RSTCSR (R)	Read	Reset control/status register
TCSRTCNT (W)	Write	Watchdog timer control/status register, Watchdog timer counter
RSTCSR (W)	Write	Reset control/status register
MSTCR (R)	Read	Module standby control register
MSTCR (W)	Write	Module standby control register

    - Note that the E10A emulator does not support the bit-field function described in the Hitachi Debugging Interface User's Manual.

    - Verify

      In the [I/O Registers] window, the verify function of the input value is invalid.

19. Note on [Registers] Window Operation During Program Execution

    Although a dialog box is displayed in which the register contetns can be changed by double-clicking the [Registers] window, do not change the register contetns during program execution.

20. 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.

21. Software Break

— Session file

When the software 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 software 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 software breakpoint settings with the [Del All] or [Disable] button in the [Breakpoints] window.

— [Run program] dialog box

If an invalid software breakpoint address is specified as a stop address in the [Run Program] dialog box, the invalid software breakpoint will become valid after the user program has stopped.

— [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.

22. Number of Software Breakpoint and [Stop At] Settings in the [Run...] Menu

The maximum number of software breakpoints and [Stop At] settings allowed in the [Run...] menu is 255. Therefore, when 255 software breakpoints are set, specification by [Stop At] in the [Run...] menu becomes invalid. Use the software breakpoints and [Stop At] in the [Run...] menu with 255 or less total settings.

23. Note on RUN-TIME Display

The execution time of the user program displayed in the [Status] window is not a correct value since the timer in the host computer has been used.

24. Note on Displaying COMMUNICATION TIMEOUT ERROR

If COMMUNICATION TIMEOUT ERROR is displayed, the emulator cannot communicate with the MCU. Select [Initialize] from the [File] menu to initialize the emulator. Before the emulator is restarted, turn off the user system and disconnect the interface cable that has been connected with the emulator.

25. 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.

26. 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.

27. 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.

28. 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
— MEMORY_FILL command

The [Format] specification in the [Copy Memory] dialog box is ignored. Memory is copied in a byte unit.

29. 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.

30. 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.

Section 7 Activation Sequence and Specific Commands of the SH7055 E10A Emulator

7.1 Activation Sequence

The following describes the activation sequence of the SH7055 E10A emulator. This sequence differs from those of other emulators:

1. Insert the card emulator into the host computer.
2. Connect the connector of the card emulator to the user interface cable.
3. Connect the user interface cable to the Hitachi-UDI port connector.
4. Turn on the host computer and the user system.

   When the emulator is activated, be sure to turn on the emulator in the MCU single-chip mode.
5. Select [HDI for SH7055 E10A Emulator] from the [Start] menu.
6. Select [SH7055 E10A Emulator] in the [Select Session] dialog box.
7. The [E10A Driver Details] dialog box is displayed. Select a driver that connects the HDI and the emulator in the [Driver] combo box (this dialog box is displayed only when the HDI is activated for the first time).

   The driver to be selected in the [E10A Driver Details] dialog box differs depending on the emulator products. The following shows the driver to be selected for each product:

   SH7055KCM01H: E10A PC Card Driver 3

   SH7055KCM02H: E10A PC Card Driver 4

   SH7055KCI01H: E10A PCI Card Driver 3

   SH7055KCI02H: E10A PCI Card Driver 4
8. When “Please, reset the user system” is displayed on the status bar, input the reset signal to the SH7055 MCM.
9. When “Link Up” is displayed on the status bar, the HDI activation is completed.

Notes: 1. The operation mode of the SH7055 MCM is set by pins FWE and MD2 to MD0. For operation in the MCU single-chip mode, set the pins as shown in table 7.1.

Table 7.1 Selection of the Operation Mode

	Pin Setting				
Operation Mode	**FWE**	**MD2**	**MD1**	**MD0**	**Mode Name**
Mode 3	0	1	1	1	MCU single-chip mode

2. After exiting the HDI, be sure to disconnect the interface cable of the user system from the emulator when the power is turned off. Failure to do so will cause the HDI being not activated at the next initiation.

7.2 RAM Monitor Commands

7.2.1 RAM_R: RR

Description:

Displays the specified address contents on the status bar during user program execution. The memory can be read by using the RAM monitor function (a maximum of three addresses are displayed).

Format:

```
RR   [[[<option1> [<option2>]] [<option1> [<option2>]] [<option1>
[<option2>]]][<init>]]
```

<option1> = address <address>
<option2> = size <size>
<init> = <init>

Table 7.2 RAM_R Command Parameter

Parameter	Type	Description
<address>	Numerical value	Sets the address to be displayed on the status bar.
<size>	Keyword	Sets the size to be read.
		B: Byte size
		W: Word size
		L: Longword size
<init>	Keyword	Clears the setting address.
		I: Clears the setting address.

Note: When only <option2> is omitted, the address is read in a byte size. When <option1>, <option2>, and <init> are omitted, the current setting address is displayed.

Notes: 1. While the memory is read by using the RAM monitor function at a fixed interval, no trace is acquired.
2. This command can be accepted during user program execution.

Examples:

To set the address to be displayed on the status bar:

```
>rr address 00000100 size B address 00000102 size L
```

To display the current setting address:

```
>rr (RET)
 address 00000100 size B address 00000102 size L
```

To clear settings:

```
>rr i (RET)
```

7.2.2 RAM_W: RW

Description:

Writes the specified contents in the specified address value to the memory. Memory can be written by using the RAM monitor function (a maximum of three addresses are written).

Format:

```
RW   <option1> <option2> [<option3>] [<option1> <option2> [<option3>]]
[<option1> <option2> [<option3>]]
```

<option1> = address <address>
<option2> = data <data>
<option3> = size <size>

Table 7.3 RAM_W Command Parameter

Parameter	Type	Description
<address>	Numerical value	Sets the address to be written.
<data>	Numerical value	Sets the data to be written.
<size>	Keyword	Sets the size to be written. **B**: Byte size **W**: Word size **L**: Longword size

Note: When <option3> is omitted, the address is written in a byte size.

Notes: 1. While the memory is written by using the RAM monitor function, no trace is acquired.
2. This command can be accepted during the user program execution.

Example:

To write memory by using the RAM monitor function:

```
>rr address 00000100 data 0009 size W address 00000200 data
FF000000 size L
```

SH7055MCM E10A User's Manual

Publication Date: 1st Edition, September 2000
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.