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.

User's Manual

SH-4A, SH4AL-DSP E200F Emulator

User's Manual

Renesas Microcomputer Development Environment System

SH-4A, SH4AL-DSP E200F R0E0200F0EMU00E

Renesas Electronics
www.renesas.com

Rev.6.00 2007.07

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, 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 products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

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 Renesas Technology Corp. and Renesas Solutions Corp. excluding all subsidiary products.

- Emulator
- Trace unit
- Expansion profiling unit
- Analyzer unit

The user system or a host machine 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 Renesas microcomputers. This emulator product must only be used for the above purpose.

Limited Applications:

This emulator product is not authorized for use in transportation, vehicular, medical (where human life is potentially at stake), aerospace, nuclear, or undersea repeater applications. Buyers of this emulator product must notify Renesas Technology Corporation, Renesas Solutions Corporation or an authorized Renesas Technology product distributor before planning to use the product in such applications.

Improvement Policy:

Renesas Technology Corp. (including its subsidiaries, hereafter collectively referred to as Renesas) pursues a policy of continuing improvement in design, performance, and safety of the emulator product. Renesas 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.

Users are required to be familiar with the basic knowledge for the electric circuits, logic circuits, and microcomputers.

Precautions to be Taken when Using This Product:

1. This emulator is a development supporting unit for use in your program development and evaluation stages. In mass-producing your program you have finished developing, be sure to make a judgment on your own risk that it can be put to practical use by performing integration test, evaluation, or some experiment else.

2. In no event shall Renesas Solutions Corporation be liable for any consequence arising from the use of this emulator.

3. Renesas Solutions Corporation strives to renovate or provide a workaround for product malfunction at some charge or without charge. However, this does not necessarily mean that Renesas Solutions Corporation guarantees the renovation or the provision under any circumstances.

4. This emulator has been developed by assuming its use for program development and evaluation in laboratories. Therefore, it does not fall under the application of Electrical Appliance and Material Safety Law and protection against electromagnetic interference when used in Japan.

5. This emulator does not conform to safety standards such as UL or IEC. Be careful when you take this emulator overseas.

6. Renesas 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.

LIMITED WARRANTY

Renesas warrants its emulator products to be manufactured in accordance with published specifications and free from defects in material and/or workmanship. Renesas, at its option, will replace any emulator products returned intact to the factory, transportation charges prepaid, which Renesas, upon inspection, shall determine to be defective in material and/or workmanship. The foregoing shall constitute the sole remedy for any breach of Renesas' warranty. See the Renesas 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. Renesas is not liable for any claim made by a third party or made by you for a third party.

DISCLAIMER

RENESAS 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 RENESAS 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:

Renesas 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 Renesas' prior written consent or any problems caused by the user system.

All Rights Reserved:

1. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Renesas' semiconductor products. Renesas 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 Renesas.

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

Figures:

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

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

Either in the user's manual or on the product, several icons are used to insure proper handling of this product and also to prevent injuries to you or other persons, or damage to your properties. Their graphic images and meanings are given in this safety page. Be sure to read this chapter before using the product.

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.

In addition to the four above, the following are also used as appropriate.

 means WARNING or CAUTION.

Example:  CAUTION AGAINST AN ELECTRIC SHOCK

 means PROHIBITION.

Example:  DISASSEMBLY PROHIBITED

 means A FORCIBLE ACTION.

Example:  UNPLUG THE POWER CABLE FROM THE RECEPTACLE.

⚠ WARNING

Warnings for AC Power Supply:

• If the attached AC power cable does not fit the receptacle, do not alter the AC power cable and do not plug it forcibly. Failure to comply may cause electric shock and/or fire.

• Use an AC power cable which complies with the safety standard of the country.

• Do not touch the plug of the AC power cable when your hands are wet. This may cause electric shock.

• This product is connected signal ground with frame ground. If your developing product is transformless (not having isolation transformer of AC power), this may cause electric shock. Also, this may give an unrepairable damage to this product and your developing one.

While developing, connect AC power of the product to commercial power through isolation transformer in order to avoid these dangers.

• If other equipment is connected to the same branch circuit care should be taken not to overload the circuit. Refer to nameplate for electrical ratings.

• When installing this equipment, insure that a reliable ground connection is maintained.

⚠ WARNING

• If you smell a strange odor, hear an unusual sound, or see smoke coming from this product, then disconnect power immediately by unplugging the AC power cable from the outlet.

Do not use this as it is because of the danger of electric shock and/or fire. In this case, contact your local distributor.

• When installing or connecting this product with other equipment, shut down AC power or disconnect the AC power cord from the equipment to prevent personal injury or damage to the equipment.

Warnings to Be Taken for This Product:

• Do not disassemble or modify this product. Personal injury due to electric shock may occur if this product is disassembled and modified.

• Make sure nothing falls into the cooling fan on the top panel, especially liquids, metal objects, or anything combustible.

Warning for Installation:

• Do not set this product in water or areas of high humidity. Make sure that the product does not get wet. Spilling water or some other liquid into the product may cause unrepairable damage.

Warning for Use Environment:

• This equipment is to be used in an environment with a maximum ambient temperature of 35°C. Care should be taken that this temperature is not exceeded.

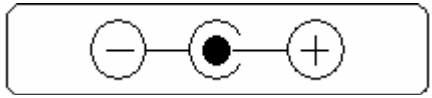
⚠ CAUTION

Cautions for AC Adapter:

• Use only the AC adapter included in this product package.

• The included AC adapter is for this emulator. Do not use it for other product.

• The DC plug on the included AC adapter has the below polarity.



• The included AC adapter has no power supply switch. The AC adapter is always active while connecting the AC power cable. Check if the power is supplied by the LED of AC adapter.

Cautions to Be Taken for This Product:

• Use caution when handling this product. Be careful not to apply a mechanical shock.

• Do not pull the main unit by the probe of the emulation probe or the flexible cable which are connected to this product. Excessive flexing or force of the flexible cable for connecting this product to the emulation probe may break connector.

Caution for Installation:

• When in use do not place the emulator on its side.

Introduction

The High-performance Embedded Workshop is a powerful development environment for embedded applications targeted at Renesas microcontrollers. The main features are:

- A configurable build engine that allows you to set-up compiler, assembler and linker options via an easy to use interface.
- An integrated text editor with user customizable syntax coloring to improve code readability.
- A configurable environment to run your own tools.
- An integrated debugger which allows you to build and debug in the same application.
- Version control support.

The High-performance Embedded Workshop has been designed with two key aims; firstly to provide you, the user, with a set of powerful development tools and, secondly, to unify and present them in a way that is easy to use.

About This Manual

This manual describes preparation before using the emulator, emulator functions, debugging functions specific to the emulator, tutorial, and emulator's hardware and software specifications.

Refer to the High-performance Embedded Workshop User's Manual for details on the information on the basic usage of the High-performance Embedded Workshop, customization of the environment, build functions, and debugging functions common to each High-performance Embedded Workshop product.

This manual does not intend to explain how to write C/C++ or assembly language programs, how to use any particular operating system or how best to tailor code for the individual devices. These issues are left to the respective manuals.

Document Conventions

This manual uses the following typographic conventions:

Table 1 Typographic Conventions

Convention	Meaning
[Menu->Menu Option]	Bold text with '->' is used to indicate menu options (for example, **[File->Save As...]**).
FILENAME.C	Uppercase names are used to indicate filenames.
"enter this string"	Used to indicate text that must be entered (excluding the "" quotes).
Key + Key	Used to indicate required key presses. For example, **CTRL+N** means press the **CTRL** key and then, whilst holding the **CTRL** key down, press the **N** key.
➲ (The "how to" symbol)	When this symbol is used, it is always located in the left hand margin. It indicates that the text to its immediate right is describing "how to" do something.

User Registration

When you have purchased the emulator represented in this user's manual, be sure to register it. As the H/W Tool Customer Registration Sheet is included with this product, fill it in and send the same contents to the following address by an email. Your registered information is used for only after-sale services, and not for any other purposes. Without user registration, you will not be able to receive maintenance services such as a notification of field changes or trouble information. So be sure to carry out the user registration.

For more information about user registration, send an email to the following address.

regist_tool@renesas.com

Contents

Section 1 Product Overview 1
1.1 Components 5
1.2 Emulator Hardware Configuration 6
1.3 Emulator Functions 20
1.3.1 Overview 20
1.3.2 Event Detection Functions 24
1.3.3 Trace Functions 27
1.3.4 Break Function 35
1.3.5 Probe Function 36
1.3.6 Peripheral I/O Analyzer Functions 36
1.3.7 Performance Measurement Function 37
1.3.8 Coverage Function 51
1.3.9 Memory Access Functions 52
1.3.9 Stack Trace Function 54
1.3.10 User-interrupt Open Function during User Program Break 54
1.3.11 Online Help 54
1.4 Environmental Conditions 55

Section 2 Setting Up the Emulator 57
2.1 Flow Chart before Using the Emulator 57
2.2 Installing Debugger 58
2.2.1 CD-R 58
2.3 Connecting the Optional Units to the Emulator Main Unit Case 59
2.3.1 Connecting the E200F Trace Unit to the User System 59
2.3.2 Connecting the E200F Peripheral I/O Analyzer Unit to the User System 62
2.3.3 Connecting the E200F Expansion Profiling Unit to the Main Unit Case 63
2.4 Connecting the AC Adapter to the Emulator Main Unit Case 69
2.5 Connecting the Emulator to the Host Machine 70
2.6 Connecting the Emulator to the User System 72
2.6.1 Connecting the E200F H-UDI/AUD Probe to the User System 73
2.6.2 Connecting System Ground 76
2.7 Changing the Settings 77
2.7.1 Changing the Functions when Using the E200F Main Unit 78
2.7.2 Changing the Functions when Using the External Bus Trace Unit 79
2.7.3 Changing the Functions when Using the Expansion Profiling Unit 80
2.7.4 Changing the Monitoring Function 81

Section 3 Hardware Specifications 83
3.1 List of Specifications 83
3.2 Interface Circuits in the Emulator 83
3.3 Reducing EMI Noise 88
3.4 Diagnostic Procedure 89
3.4.1 Installing the Diagnostic Program 89
3.4.2 Executing the Diagnostic Program 89
3.4.3 Creating a Log File 93

Section 4 Preparations for Debugging 95
4.1 System Check 95
4.2 Method for Activating High-performance Embedded Workshop 103
4.2.1 Creating the New Workspace (Toolchain Not Used) 104
4.2.2 Creating the New Workspace (Toolchain Used) 108
4.2.3 Selecting an Existing Workspace 113
4.3 Setting at Emulator Activation 115
4.4 Debug Sessions 117
4.4.1 Selecting a Session 117
4.4.2 Adding and Removing Sessions 118
4.4.3 Saving Session Information 121
4.5 Connecting the Emulator 122
4.6 Reconnecting the Emulator 123
4.7 Ending the Emulator 124
4.8 Uninstalling the Emulator's Software 125

Section 5 Debugging 131
5.1 Setting the Environment for Emulation 131
5.1.1 Opening the [Configuration] Dialog Box 131
5.1.2 [General] Page 132
5.1.3 [Main Board] Page 135
5.1.4 [Bus Board] Page 137
5.1.5 [Option Board] Page 141
5.1.6 Downloading to the Flash Memory 143
5.1.7 Opening the [Memory Mapping] Dialog Box 145
5.1.8 Changing the Memory Map Setting 147
5.2 Downloading a Program 148
5.2.1 Downloading a Program 148
5.2.2 Viewing the Source Code 149
5.2.3 Viewing the Assembly-Language Code 154
5.2.4 Modifying the Assembly-Language Code 155

5.2.5 Viewing a Specific Address 156
5.2.6 Viewing the Current Program Counter Address 156
5.3 Displaying Memory Contents in Realtime 157
5.3.1 Opening the [Monitor] Window 158
5.3.2 Changing the Monitor Settings 160
5.3.3 Temporarily Stopping Update of the Monitor 161
5.3.4 Deleting the Monitor Settings 161
5.3.5 Monitoring Variables 161
5.3.6 Hiding the [Monitor] Window 163
5.3.7 Managing the [Monitor] Window 164
5.4 Viewing the Current Status 165
5.5 Reading and Displaying the Emulator Information Regularly 166
5.5.1 Opening the [Extended Monitor] Window 166
5.5.2 Selecting Items to be Displayed 167
5.6 Using the Eventpoints 168
5.6.1 PC Breakpoints 168
5.6.2 Eventpoints 168
5.6.3 Opening the [Event] Window 170
5.6.4 Setting PC Breakpoints 171
5.6.5 Setting Onchip Eventpoints 173
5.6.6 Setting AUD Eventpoints 182
5.6.7 Setting BUS Eventpoints 199
5.6.8 Setting Other Eventpoints 209
5.6.9 Editing Breakpoint or Eventpoint 214
5.6.10 Enabling Breakpoint or Eventpoint 214
5.6.11 Disabling Breakpoint or Eventpoint 214
5.6.12 Deleting Breakpoint or Eventpoint 214
5.6.13 Deleting All Breakpoints or Eventpoints 214
5.6.14 Viewing the Source Line for Breakpoints or Eventpoints 214
5.7 Viewing the Trace Information 215
5.7.1 Opening the [Internal/AUD/Usermemory trace] Window 215
5.7.2 Opening the [BUS/MFI trace] Window 223
5.7.3 Hiding the [Trace] Column 226
5.7.4 Searching for a Trace Record 227
5.7.5 Clearing the Trace Information 234
5.7.6 Saving the Trace Information in a File 234
5.7.7 Viewing the [Source] Window 234
5.7.8 Trimming the Source 234
5.7.9 Temporarily Stopping Trace Acquisition 235
5.7.10 Extracting Records from the Acquired Information 235

5.7.11 Analyzing Statistical Information 242
5.7.12 Extracting Function Calls from the Acquired Trace Information 244
5.8 Analyzing Performance 245
5.8.1 Opening the [Onchip Performance Analysis] Window 245
5.8.2 Opening the [AUD Performance Analysis] Window 247
5.8.3 Opening the [BUS Performance Analysis] Window 250
5.8.4 Hiding the Column 254
5.8.5 Starting Performance Data Acquisition 254
5.8.6 Deleting a Measurement Condition 254
5.8.7 Deleting All Measurement Conditions 255
5.9 Viewing the Profile Information 255
5.9.1 Stack Information Files 255
5.9.2 Profile Information Files 257
5.9.3 Loading Stack Information Files 258
5.9.4 Enabling the Profile 259
5.9.5 Specifying Measuring Mode 259
5.9.6 Executing the Program and Checking the Results 259
5.9.7 [List] Sheet 260
5.9.8 [Tree] Sheet 261
5.9.9 [Profile-Chart] Window 264
5.9.10 Types and Purposes of Displayed Data 264
5.9.11 Creating Profile Information Files 265
5.9.12 Notes 266
5.10 Viewing Realtime Profile Information 268
5.10.1 Opening the [Realtime Profile] Window 274
5.10.2 Specifying the Measurement Range 275
5.10.3 Starting Measurement 276
5.10.4 Clearing Measurement Result 276
5.10.5 Deleting Measurement Range 276
5.10.6 Setting the Minimum Unit of the Measurement Time 276
5.11 Acquiring Code Coverage 278
5.11.1 Opening the [Code Coverage] Window 278
5.11.2 Displaying a Source File 283
5.11.3 Changing the Address to be Displayed 283
5.11.4 Changing the Coverage Range to be Displayed 284
5.11.5 Clearing the Coverage Information 286
5.11.6 Saving the Coverage Information to a File 286
5.11.7 Loading Coverage Information from a File 287
5.11.8 Updating Coverage Information 287
5.11.9 Preventing Update of Coverage Information 287

5.11.10 [Confirmation Request] Dialog Box 288
5.11.11 Displaying Code Coverage Information in the [Editor] Window 290
5.12 Synchronizing Multiple Debugging Platforms 291
5.12.1 Distinguishing Two Emulators 292

Section 6 Tutorial 295
6.1 Introduction 295
6.2 Running the High-performance Embedded Workshop 296
6.3 Setting up the Emulator 296
6.4 Setting the [Configuration] Dialog Box 297
6.5 Checking the Operation of the Target Memory for Downloading 298
6.6 Downloading the Tutorial Program 300
6.6.1 Downloading the Tutorial Program 300
6.6.2 Displaying the Source Program 301
6.7 Setting a PC Breakpoint 302
6.8 Setting Registers 303
6.9 Executing the Program 305
6.10 Reviewing Breakpoints 308
6.11 Referring to Symbols 309
6.12 Viewing Memory 310
6.13 Watching Variables 312
6.14 Displaying Local Variables 315
6.15 Stepping Through a Program 316
6.15.1 Executing [Step In] Command 316
6.15.2 Executing [Step Out] Command 318
6.15.3 Executing [Step Over] Command 319
6.16 Forced Breaking of Program Executions 320
6.17 Break Function 321
6.17.1 PC Break Function 321
6.18 Break Function by an Eventpoint 326
6.18.1 Setting the Break by an Onchip Eventpoint 326
6.18.2 Setting the Sequential Onchip Eventpoint 331
6.19 Trace Functions 335
6.19.1 Displaying the [Internal/AUD/Usermemory trace] Window 337
6.19.2 Displaying the [BUS/MFI trace] Window 352
6.20 MMU Support 356
6.21 Stack Trace Function 359
6.22 Download Function to the Flash Memory Area 361
6.23 What Next? 367

Section 7 Troubleshooting 369

Section 8 Maintenance and Guarantee 371
8.1 User Registration 371
8.2 Maintenance 371
8.3 Guarantee 371
8.4 Repair Provisions 372
8.4.1 Repair with Extra-Charge 372
8.4.2 Replacement with Extra-Charge 372
8.4.3 Expiration of the Repair Period 372
8.4.4 Transportation Fees at Sending Your Product for Repair 372
8.5 How to Make a Request for Repair 373

Appendix A Menus 375

Appendix B Command-Line Functions 379

Appendix C Notes on High-performance Embedded Workshop 381

Appendix D Repair Request Sheet 385

Section 1 Product Overview

The High-performance Embedded Workshop provides a graphical user interface that eases the development and debugging of applications written in the C/C++ programming languages or assembly language for Renesas microcomputers. Its aim is to provide a powerful yet intuitive way of accessing, observing and modifying the debugging platform on which the application is running.

The E200F emulator (hereafter referred to as the emulator) is a support tool for developing the hardware and software of application systems running on Renesas original microcomputers.

The main unit of the emulator is connected, through the dedicated debugging interface, to the user system. The user system can be debugged under the conditions similar to the actual application conditions.

Figures 1.1 and 1.2 show the system configuration using two types of emulators, R0E0200F0EMU00 and R0E0200F2EMU00, respectively.

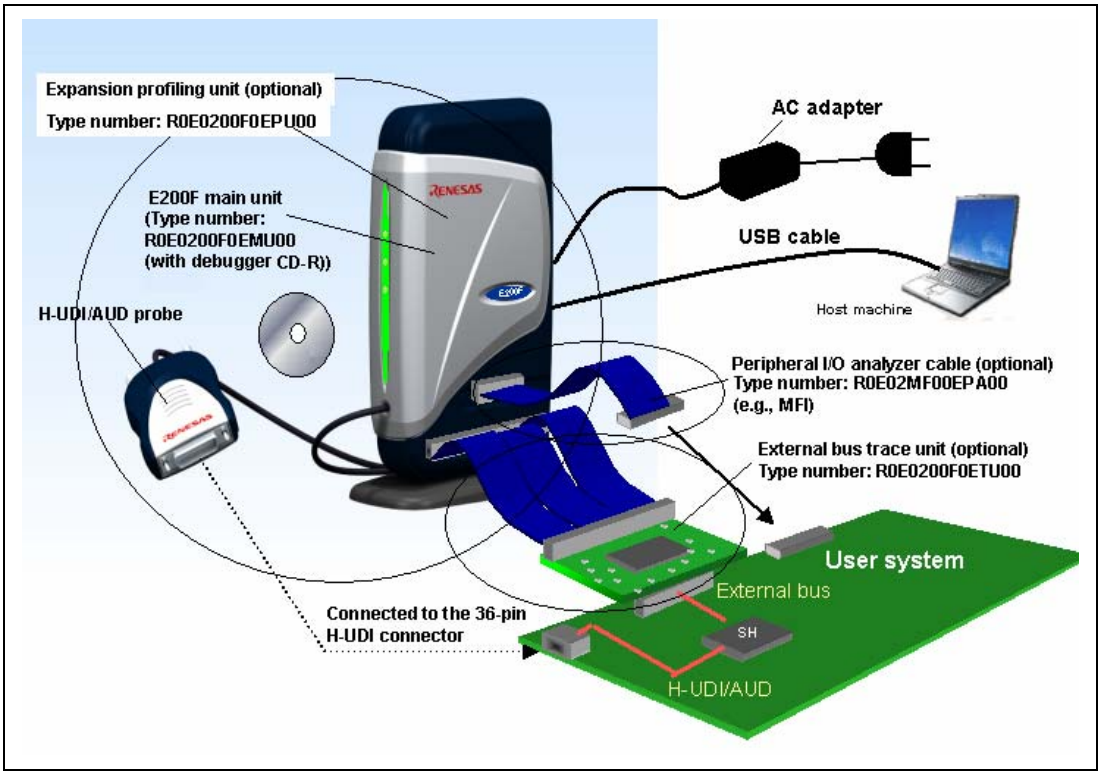


Figure 1.1 System Configuration with the Emulator (R0E0200F0EMU00)

Note: The H-UDI is an interface compatible with the Joint Test Action Group (JTAG) specifications.

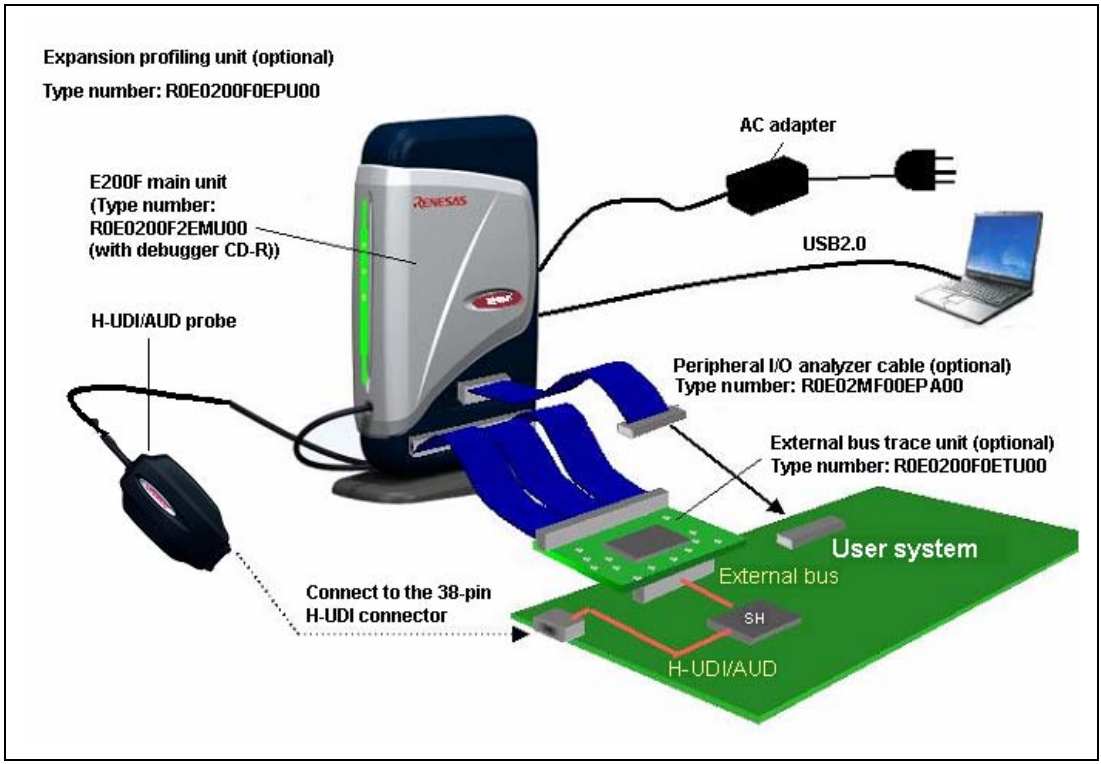


Figure 1.2 System Configuration with the Emulator (R0E0200F2EMU00)

The emulator provides the following features:

- Various debugging functions
  - Various break and trace functions enable efficient debugging. Breakpoints and break conditions can be set by the specific window, and trace information can be displayed on a window. In addition, various emulation functions, such as performance and profiling functions, are provided.
  - High-speed downloading is implemented by supporting USB 2.0.
  - An analyzer function is provided for the peripheral I/O modules. Such modules differ depending on the MPU.
  - Emulator functions can be changed by each debugging phase.
  - Various command-line functions can be used.
- Realtime emulation

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

  Using the High-performance Embedded Workshop on the Microsoft® Windows® 2000 and Microsoft® Windows® XP operating systems enables user program debugging using a pointing device such as a mouse. The High-performance Embedded Workshop enables high-speed downloading of load module files.
- 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 emulator is used, a laptop computer can be used as a host machine, creating a debugging environment in any place.

⚠ 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 1.4, Environmental Conditions.
4. When moving the host machine or user system, take care not to vibrate or damage it.
5. After connecting the cable, check that it is connected correctly. For details, refer to section 2, Setting Up the Emulator.
6. Supply power to the connected equipment after connecting all cables. Cables must not be connected or removed while the power is on.

1.1 Components

Check that all of the components are present when unpacking the product. For details on the emulator components, refer to section 1.1 in the additional document, Supplementary Information on Using the SHxxxx. If all of the components are not present, contact your nearest Renesas sales office.

1.2 Emulator Hardware Configuration

As shown in figure 1.3, the emulator (R0E0200F0EMU00) consists of a main unit case, a trace unit (optional), an expansion profiling unit, a USB cable, an AC adapter, an external probe, and an peripheral I/O analyzer unit (optional). The emulator is connected to the host machine via USB 2.0. Figure 1.4 shows the hardware configuration of the emulator (R0E0200F2EMU00).

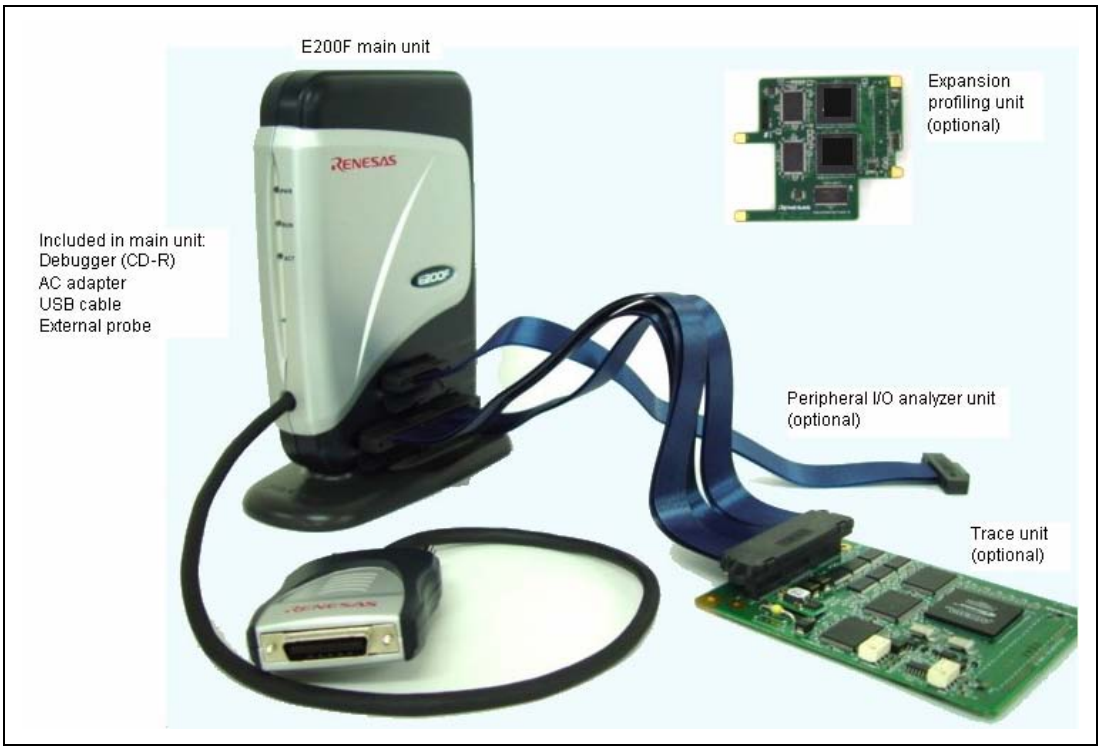


Figure 1.3 Emulator Hardware Configuration (R0E0200F0EMU00)

Figure 1.4 Emulator Hardware Configuration (R0E0200F2EMU00)

The names of each section of the emulator are explained next.

Emulator Front View:

Figure 1.5 Emulator Front View

(a) POWER LED:	Marked ‘PWR’. When this LED is lit, the emulator is supplied with power.
(b) RUN LED:	Marked ‘RUN’. When this LED is lit, the user program is in operation.
(c) ACTION LED:	Marked ‘ACT’. When this LED is lit, the emulator is in operation.

Emulator Rear-side View:

Figure 1.6 Emulator Rear-side View

(a) Power switch:	Marked ‘POWER’. Turning this switch to I (input) turns the emulator on and O (output) turns the emulator off.
(b) DC plug:	Marked ‘DC IN’. This is a connector to input DC (+12 V) of the AC adapter. Be sure to connect this plug to the provided AC adapter.
(c) External probe connector:	Marked ‘EXT’. This is a connector for the external probe. Be sure to connect this connector to the provided external probe.
(d) Host-side connector (USB connector):	Marked ‘  ’. A connector (USB connector) for the host machine is provided at the side of this mark. Be sure to connect this connector to the provided USB cable.

Emulator Right-side View:

Figure 1.7 Emulator Right-side View

(a) E200F logo plate:	A navy-blue plate (R0E0200F0EMU00) or a silver-and-red plate (R0E0200F2EMU00) dedicated for the emulator is provided to be easily distinguished from other E200F main unit case.
(b) Peripheral I/O analyzer unit connector:	Marked 'ANALYZER I/F'. This is a connector for the peripheral I/O analyzer unit. Be sure to connect this connector to the dedicated analyzer cable provided for the optional peripheral I/O analyzer unit.
(c) Trace unit connector:	Marked 'TRACE I/F'. This is a connector for the trace unit. Be sure to connect this connector to the dedicated trace cable provided for the optional trace unit.

Emulator Left-side View:

Figure 1.8 Emulator Left-side View

(a) Label for product management:	The serial number, revision, and safety standard, etc. of the emulator are written to. The contents differ depending on the time when you purchased the product.

36-pin Probe Head Upper View (R0E0200F0EMU00):

Figure 1.9 36-pin Probe Head Upper View (R0E0200F0EMU00)

(a) Screws for fastening the probe head: These screws fasten the user system and the probe head.

36-pin Probe Head Front View (R0E0200F0EMU00):

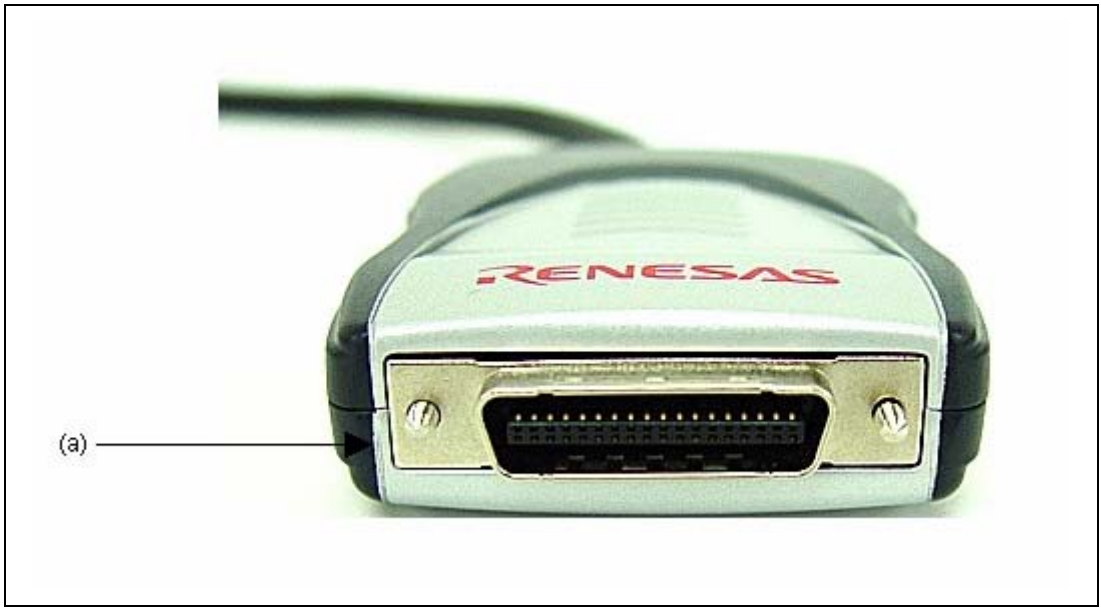


Figure 1.10 36-pin Probe Head Front View (R0E0200F0EMU00)

(a) H-UDI port connector (36 pins) for user system: This connector is used when the emulator is used as the on-chip debugger (the same as that of the 36-pin E10A-USB).

38-pin Probe Head Upper View (R0E0200F2EMU00):

Figure 1.11 38-pin Probe Head Upper View (R0E0200F2EMU00)

38-pin Probe Head Front View (R0E0200F2EMU00):

Figure 1.12 38-pin Probe Head Front View (R0E0200F2EMU00)

(a) H-UDI port connector (38 pins) for user system: This connector is used when the emulator is used as the on-chip debugger (the same as that of the 38-pin E10A-USB).

Storing the Probe Head:

Figure 1.13 Storing the Probe Head (R0E0200F0EMU00)

(a) Base unit for placing the emulator vertically:	A base unit when the emulator is placed vertically. The probe head can be stored when the emulator is not in use.

Trace Unit (Optional) Upper View:

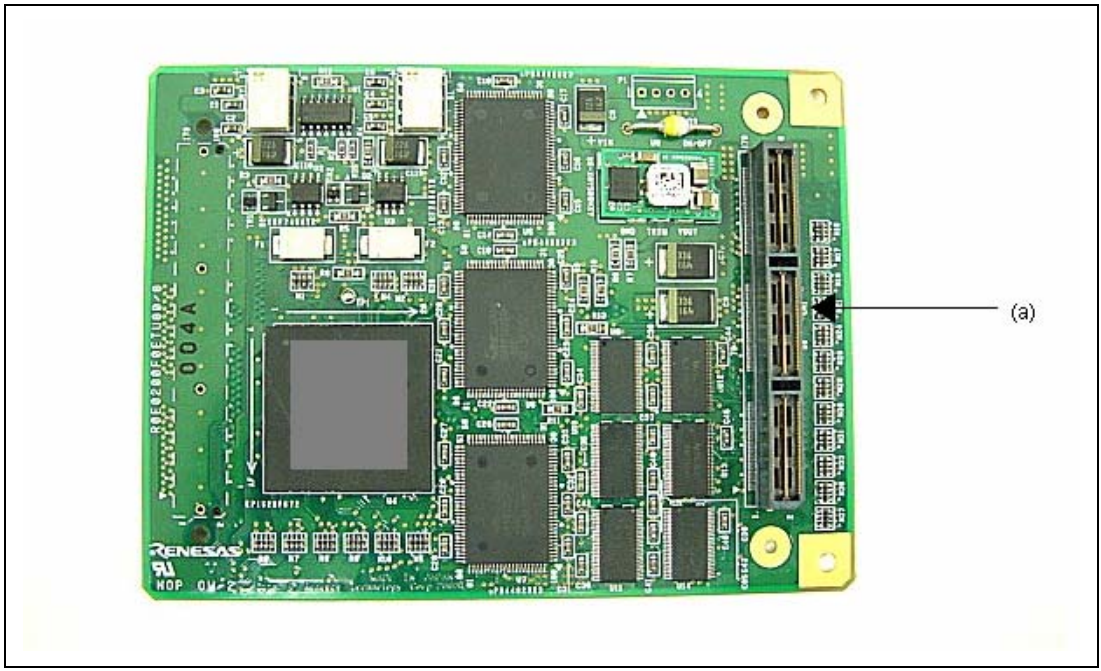


Figure 1.14 Trace Unit (Optional) Upper View

(a) Trace cable connectors: Connectors for connecting the trace cable in the trace board. Be sure to use the provided trace cable.

Trace Unit (Optional) Rear View:

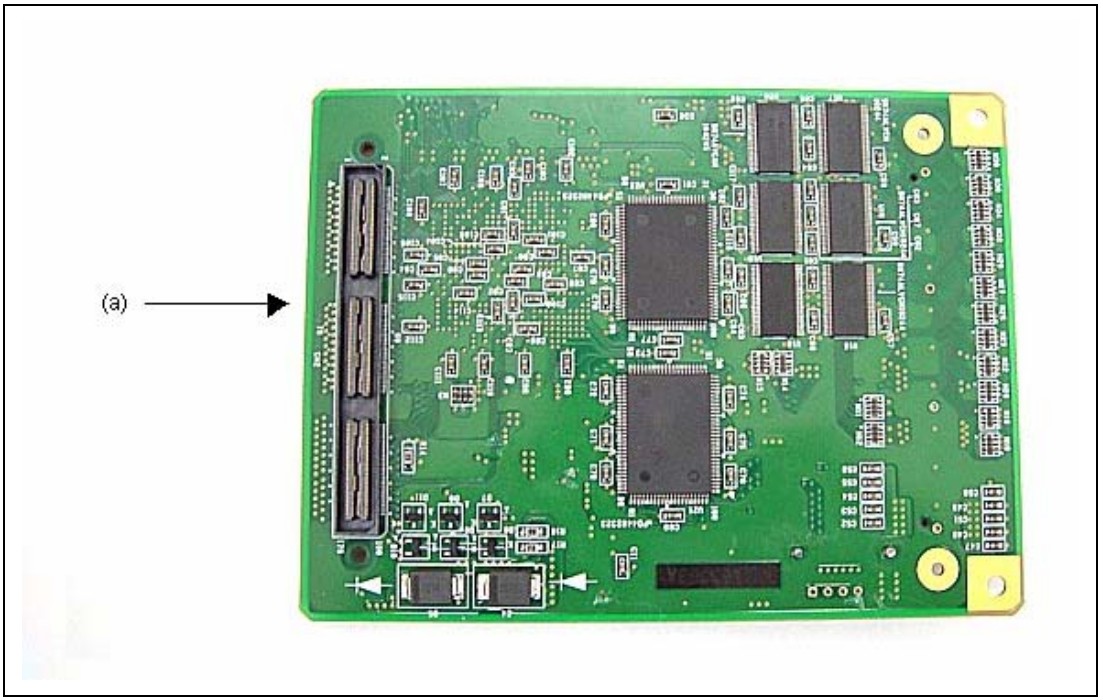


Figure 1.15 Trace Unit (Optional) Rear View

(a) User interface connectors for the external bus trace:	Connectors for the user interface of the external bus trace. They are connected to the dedicated connector on the user system.

Expansion Profiling Unit (Optional) Rear View:

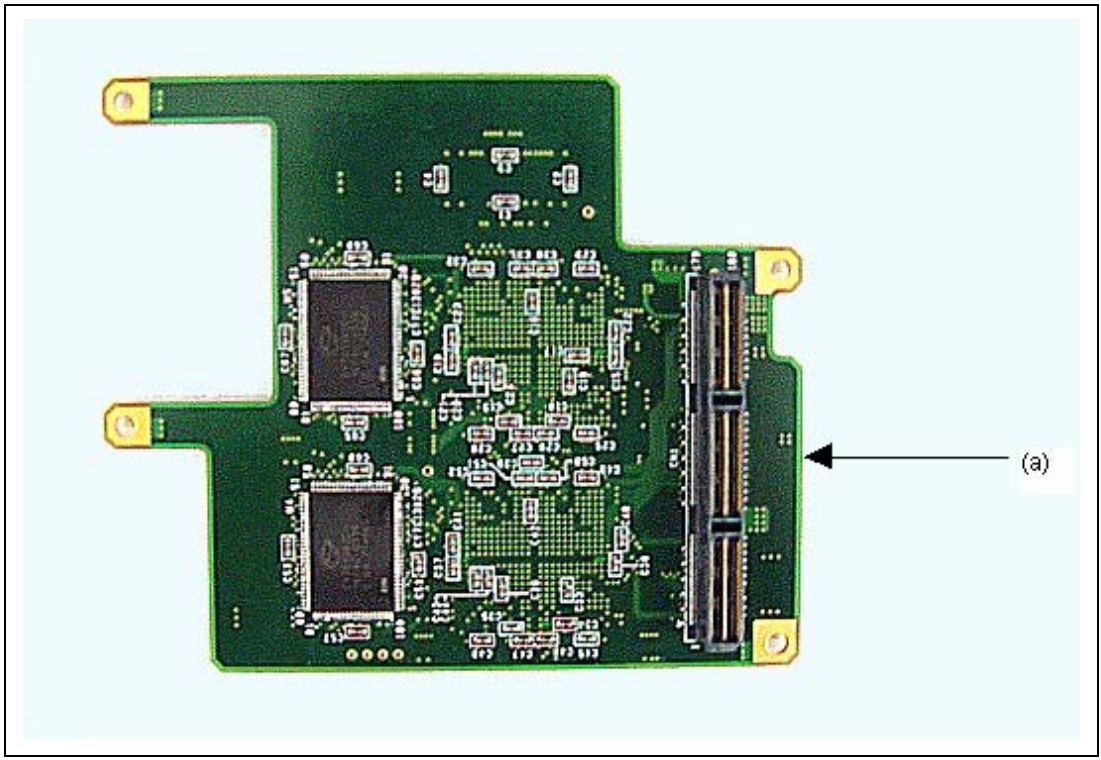


Figure 1.16 Expansion Profiling Unit (Optional) Rear View

(a) Optional connectors in the expansion profiling unit:	Connectors for the interface to connect the expansion profiling unit and the main unit case. They are connected to the optional connector on the main unit case.

1.3 Emulator Functions

This section describes the emulator functions. They differ according to the device supported by the emulator. For the usage of each function, refer to section 5, Debugging, and section 6, Tutorial.

1.3.1 Overview

Table 1.1 gives a functional overview of the emulator.

For details on the functions of each product, refer to the online help.

Table 1.1 Emulator Functions

No.	Item	Function
1	User program execution function	• Executes a program with the operating frequency within a range guaranteed by devices. • Reset emulation • Step functions: Single step (one step: one instruction) Source-level step (one step: one-line source) Step over (a break did not occur in a subroutine) Step out (when the PC points to a location within a subroutine, execution continues until it returns to the calling function)
2	Reset function	• Issues a power-on reset from the High-performance Embedded Workshop to the device during break.
3	Event detection functions	• 13 on-chip event detection channels • Eight AUD event detection channels • Six external-bus event detection channels • Other event detection One execution time event detection channel One external probe event detection channel • Other event detection functions on the basis of I/O-analyzed results
4	Trace functions	• Internal trace function at on-chip event detection • AUD trace function at AUD event detection • Outputs memory of the internal trace data • External bus trace function at external bus event detection • Other trace acquisition functions with the I/O analyzer

Table 1.1 Emulator Functions (cont)

No.	Item	Function
5	Break functions	• Breaks at satisfaction of a PC breakpoint condition (1000 points) • Breaks at on-chip event detection • Breaks at AUD event detection • Breaks at external-bus event detection • Breaks at other types of event detection • Breaks at overflow of a trace buffer • Forced break function
6	Performance measurement function	• Uses a counter in the device to measure the number of cycles that passes during point-to-point execution. • Uses an AUD event channel to measure the time or counts of point-to-point, point-to-range, or range-to-range execution. • Uses the performance function of an external bus to measure the time or counts of range, point-to-point, or range-to-range execution. • Measures the number of cycles that pass in executing individual functions and lists them at the end of execution from a 'Go' command. • Measures the execution time or execution counts of functions in the address range specified by the user and lists them at the end of execution from a 'Go' command.
7	Peripheral I/O analyzer function	• Monitors generated events of the specific peripheral I/O or acquires them by a trace. The supported peripheral I/Os differ depending on the MPU. For this function, refer to the additional document, Supplementary Information on Using the SHxxxx.

Table 1.1 Emulator Functions (cont)

No.	Item	Function
8	Memory access functions	• Downloading to RAM • Downloading to flash memory • Single-line assembly • Reverse assembly (disassembly) • Reading of memory • Writing to memory • Automatic updating of a display of selected variables during user program execution • Fill • Search • Move • Copy • Monitor (physical or virtual address) • Emulation memory
9	General/control register access function	Reads or writes the general/control registers.
10	Internal I/O register access function	Reads or writes the internal I/O registers.*
11	Source-level debugging function	Various source-level debugging functions.
12	Coverage function	Displays the information of executed instructions.
13	Command line function	Supports command input. Batch processing is enabled when a file is created by arranging commands in input order.
14	Help function	Describes the usage of each function or command syntax input from the command line window.

Note: The [IO] window displays the contents defined in [SHxxxx.io]. Editing those contents adds or deletes the registers to be displayed. For the contents to be described as [SHxxxx.io], refer to reference 5, I/O File Format, in the High-performance Embedded Workshop V.4.00 User's Manual.
The following directory contains [SHxxxx.io] (xxxx means the core name.):
<High-performance Embedded Workshop folder>:
\Tools\Renesas\DebugComp\Platform\E200F\xxxx\IOFiles

The specific functions of the emulator are described in the next section.

1.3.2 Event Detection Functions

The emulator has complex event detection functions in addition to the standard PC breakpoints of the High-performance Embedded Workshop.

(1) Events

In most practical debugging applications, the program or hardware errors that you are trying to debug occur under a certain restricted set of circumstances. For example, a hardware error may only occur after a specific area of memory has been accessed. Tracking down such problems using simple PC breakpoints can be very time-consuming.

With the emulator, the combination of the specified conditions such as address, data, or access condition can be defined as the eventpoint. When an eventpoint condition is satisfied, an event will occur. The event detection function of the emulator can be used to detect a generated event and control the operations of break, trace, and performance measurement start/end.

(2) Types of Events

The emulator has four types of events.

(a) On-chip event function (Onchip Event)

This is a function that uses an on-chip break controller and sets eventpoints according to the information in the MPU. There are 13 event detection channels. The eventpoints can be defined as a combination of one or more of the following:

- Address condition
- Data condition
- ASID condition
- Bus state condition
- Address condition for the destination of memory access
- System bus condition
- LDTLB instruction execution
- Branch condition
- Software trace condition*
- Event count condition

Note: The PC value when executing the Trace() function and the contents of the specified general register are acquired by trace; this function is hereafter called as the software trace function.

For an operation when an event is detected, break, internal trace acquisition/acquisition start/acquisition stop, or internal performance measurement start/end can be specified.

This function can be set in the [Onchip Event] sheet of the [Event] window.

(b) AUD event function (AUD Event)

This is a function that sets eventpoints according to the information output from the AUD interface. There are eight event detection channels. The eventpoints can be defined as a combination of one or more of the following:

- Branch trace data condition
- Window trace data condition
- System bus trace data condition
- Software trace data condition
- Event count condition
- Delay condition after an event has occurred

For an operation when an event is detected, break, AUD trace acquisition/acquisition start/acquisition stop, or AUD performance measurement start/end can be specified.

This function can be set in the [AUD Event] sheet of the [Event] window.

Notes: 1. When a break is generated by detecting the AUD event, there is a delay of several cycles from the time of detection to the break. If the delay from the generation of an event to the break in the user system becomes a problem, use the on-chip event function (Onchip event).
2. When the acquisition mode of the AUD trace is [Realtime trace], data that was not output cannot be compared.

(c) External bus event function (BUS Event)

This is a function that sets eventpoints according to the external bus of the MPU or pin information such as an interrupt pin. There are six event detection channels. The eventpoints can be defined as a combination of one or more of the following:

- External address bus condition
- External data bus condition
- Interrupt signal condition
- Event count condition
- Delay condition after an event has occurred

For an operation when an event has been detected, a break or external bus trace acquisition/acquisition start/acquisition stop can be specified.

This function can be set in the [BUS Event] sheet of the [Event] window.

Notes: 1. When a break is generated by detecting the external bus event, there is a delay of several cycles from the time of detection to the break.
2. This function is optional; it is available when a trace unit (optional) is used.

(d) Other events (Other Event)

This function can be set in the [Other Event] sheet of the [Event] window.

- Execution time event

  A break occurs after the specified execution time has passed. There is one event detection channel.

- External probe event

  There is one event detection channel. Eventpoints can be defined specifying four external probe signals via the probe cable as the condition.

  For an operation when an event has been detected, a break or AUD trace acquisition/acquisition start/acquisition stop can be specified.

Note: There is a delay of several cycles from the time of detection to the break.

- Events by the peripheral I/O analyzer

  Events can be detected with the peripheral I/O analyzer function.

Notes: 1. There is a delay of several cycles from the time of detection to the break.
2. This function is optional; it is available when a peripheral I/O unit (optional) is used.

1.3.3 Trace Functions

(1) Trace Functions

The emulator has mainly two trace functions:

- A function that acquires information within the MPU such as the CPU or system bus by a trace
- A function that acquires information outside the MPU such as the external bus by a trace

(a) Function to acquire information within the MPU by a trace

The acquired trace information is displayed in the [Internal/AUD/Usermemory trace] window.

The information to be acquired by a trace is an event that trace acquisition is selected as the action condition in the channel on the [Onchip Event] sheet of the [Event] window. To store such information, there are following three methods:

— Function to store the information in a dedicated small trace memory in the MPU

Eight types of information are stored.

This function is enabled when the AUD pin is not connected to the emulator or the memory cannot be used for tracing.

This function is hereafter called as the internal trace function or the Internal trace.

— Function to output the information in realtime from the AUD pins

This is the large-capacity trace function that is enabled when the AUD pins are connected to the emulator. The emulator generates a break on the basis of the information output from the AUD pins.

This function is hereafter called as the AUD trace function.

When a set of the branch source and branch destination instructions is one branch, the maximum amount of information acquired by a trace is 262,144.

— Function to store the information in the specified memory

The acquired trace information is stored within the specified memory range.

Do not specify the program area as the trace information is overwritten to the memory in the specified range.

This function is hereafter called as the memory output trace function or the Usermemory trace.

The AUD trace function is expanded in the emulator.

Expanded function 1: The timestamp information can be acquired.

Expanded function 2: The AUD trace acquisition, acquisition start, and acquisition end can be controlled depending on the specified AUD eventpoint condition and external-probe eventpoint condition. For setting eventpoints, refer to section 5.6, Using the Eventpoints.

Note: The contents that can be acquired by a trace differ according to the product. For details on the specifications of each product, refer to the online help.

The following four types of information in the MPU can be acquired by a trace:

- Branch information
- Range memory access information in the virtual address
- Software trace
- System bus access information

Branch Trace Functions:

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

[Setting Method]

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

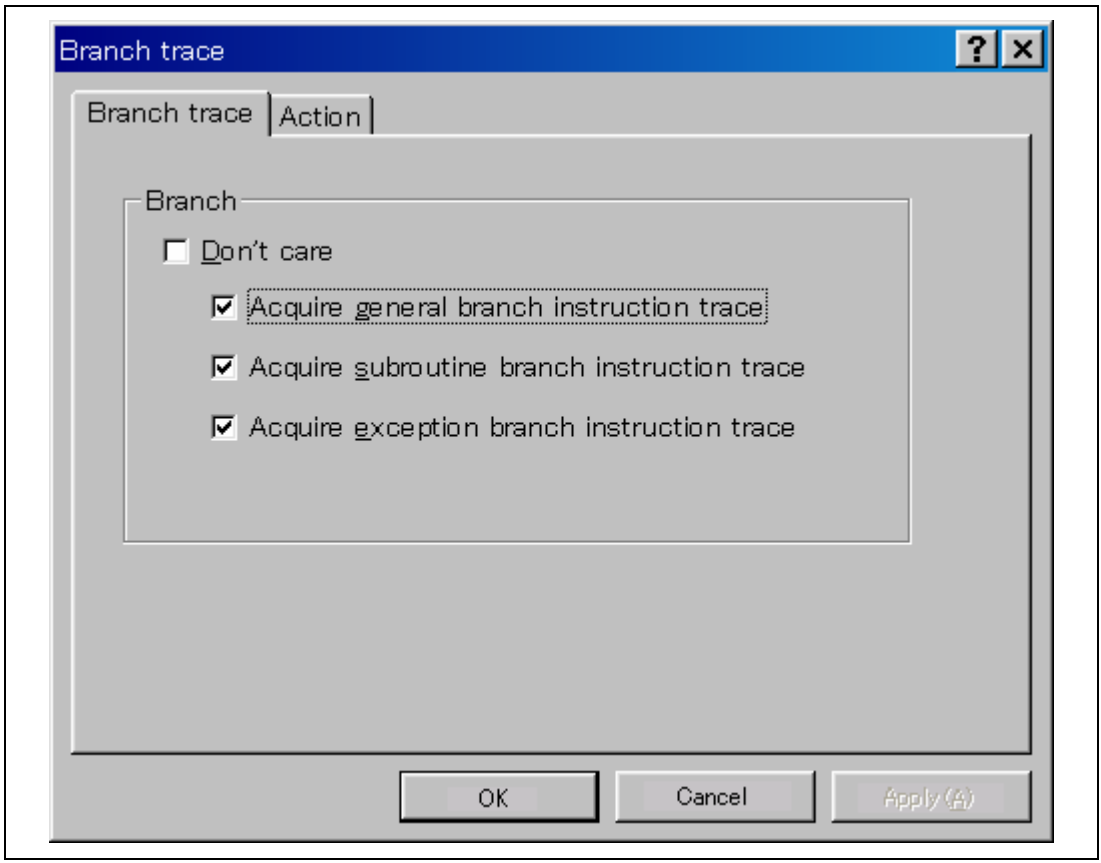


Figure 1.17 [Branch trace] Dialog Box

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

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

Range Memory Access Trace Functions:

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

[Setting Method]

To open the [Event condition 5] or [Event condition 6] dialog box, double-click on the Ch5 (OA) or Ch6 (OA) column of the [Event] window.

Remove the check mark of the [Don't care] check box in the [Window address] page and enter the memory range to be set.

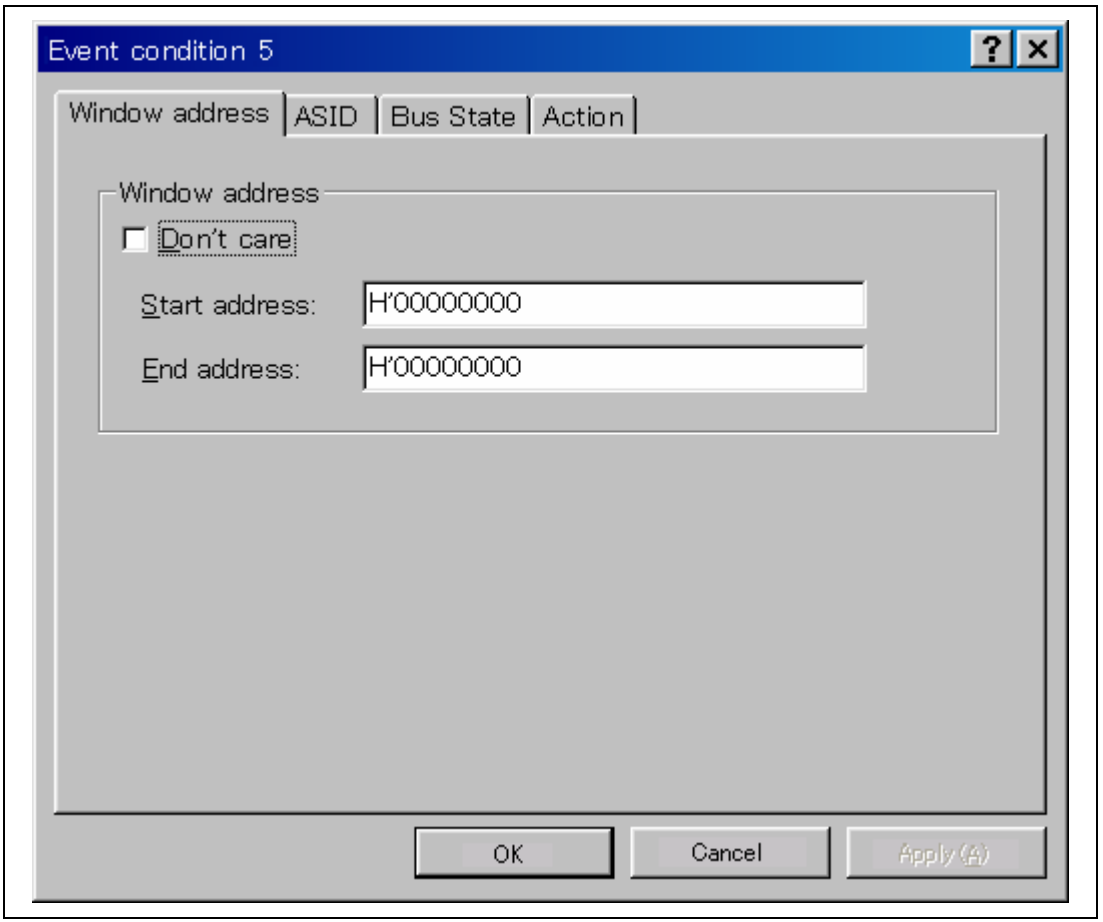


Figure 1.18 [Window address] Page

(i) Open the [ASID] page, remove the check mark of the [Don't care] check box, and enter the ASID value to be set.
When the ASID value is not set as a condition, do not remove the check mark of the [Don't care] check box.

(ii) Open the [Bus state] page and specify the bus type and bus cycle that are to be set.

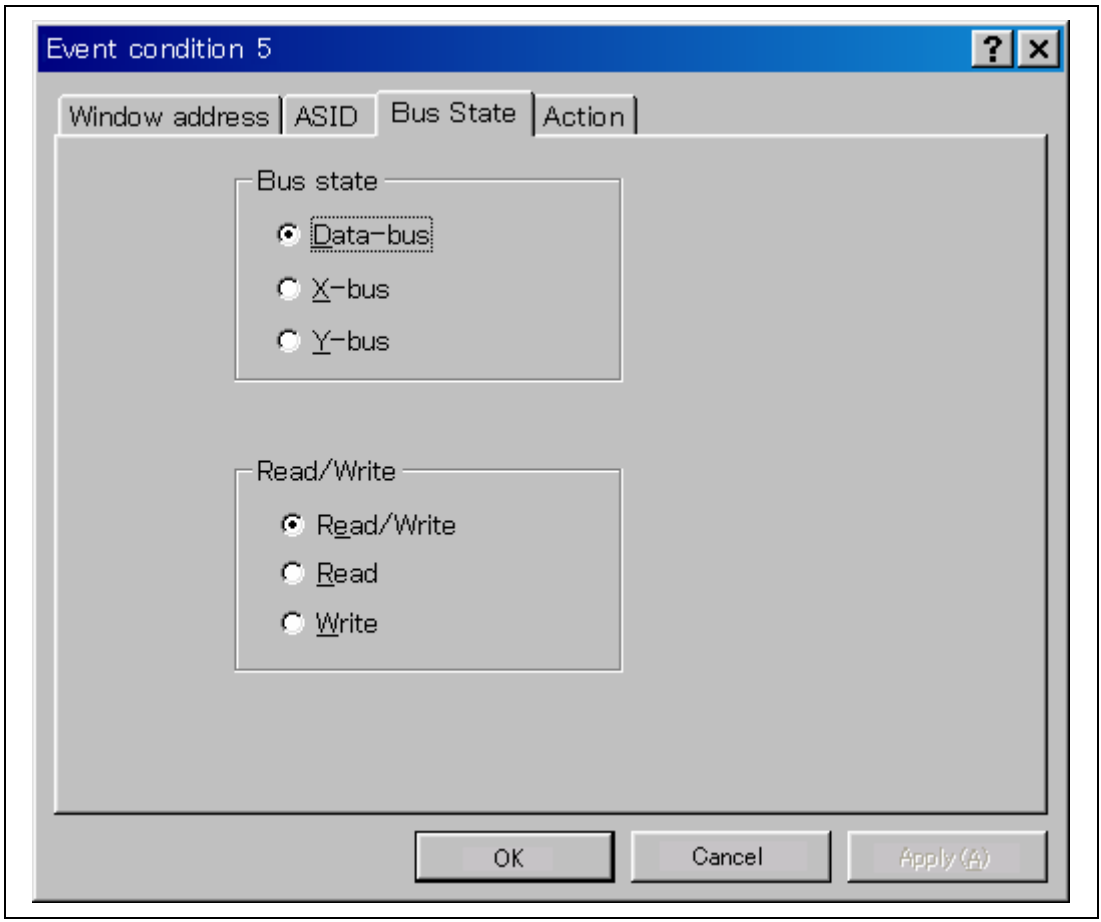


Figure 1.19 [Bus State] Page

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

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

Software Trace Function:

Note: This function can be supported with SHC/C++ compiler (manufactured by Renesas Technology Corp.; including OEM and bundle products) V6.0 or later. However, SHC/C++ compiler (including OEM and bundle products) V8.0 or later is needed when instructions other than those compatible with SH4 are output.

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

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

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

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

System Bus Access Trace Function: This function can be set with Ch8 or Ch9 of the [Event] window. The information on the system bus in the MPU can be acquired.

Display Contents: When the program breaks, the following trace results are displayed in the [Trace] window.

- PTR: The trace-buffer pointer (+0 from the last instruction to have been executed)
- IP: Indicates the number of cycles that have elapsed since the latest trace information was gathered. For branch instructions, the branch source and destination are counted together as one.
- Master: Type of bus master that accessed the memory.
- Type: Displays the type of trace acquisition information.
- Branch Type: Branch type (only displayed for a branch trace)
  For an AUD trace, this item is only displayed if the PPC option has been enabled.
- Bus: Displays which bus was accessed.
- R/W: Displays whether the access involved reading or writing.
- Address: Displays the addresses from which the trace data was acquired.
- Data: Displays the data acquired in the trace.

- PPC: Output from a performance counter
- Instruction, Source, Label: Displays the mnemonic of the instruction at the trace acquisition address, along with the corresponding source code and label information. Double-clicking on the [Source] column moves the cursor to the corresponding position in the [Editor] window.

The Type, BUS, R/W, Address, and Data columns have different meanings according to the type of trace that has been selected.

Table 1.2 [Trace Window] Display Contents

Trace Type	Type Column	BUS Column	R/W Column	Address Column	Data Column
Branch trace	BRANCH*1	No display	No display	Branch source address*1	No display
	DESTINATION	No display	No display	Branch destination address	No display
Memory-range access trace	MEMORY	No display	Read/write	Memory access address	Memory access data*1
Software trace	S_TRACE	No display	No display	Trace(x) function execution address	Variable x data
System bus trace	MEMORY	No display	Read/write	Memory access address	Memory access data (write only)*1
Data lost*2	LOST	No display	No display	No display	No display
CPU wait generation*2	CPU-WAIT	No display	No display	No display	No display

Notes: 1. Not displayed when the PPC option is in use.
2. According to the device being debugged, there may be no output for the [Lost] or [CPU-WAIT] type. In such a case, it is not possible to clarify whether the trace data was not output in time or the CPU generated a wait state for the output trace data.

(b) Function to acquire information outside the MPU by a trace

The acquired trace information is displayed in the [BUS/MFI trace] window.

External Bus Trace Function (BUS trace):

This is a large-capacity trace function that is enabled when the external bus pins of the MPU are connected to the emulator. If an external memory is read from or written to, trace information is output in realtime from the external bus pins.

The external bus trace acquisition, acquisition start, and acquisition end can be controlled depending on the specified external-bus eventpoint condition. For setting eventpoints, refer to section 5.6, Using the Eventpoints.

With this function, information of a maximum of 262,144 cycles can be acquired in each bus cycle.

Other Trace Functions:

When the optional peripheral I/O analyzer function is used for trace acquisition, the trace result can be displayed on the same window.

(2) Useful functions of the [Trace] window

The [Trace] window provides the following useful functions.

(a) Searches for the specified data.

(b) Extracts the specified data.

(c) Filters and displays again the specified data.

(d) Supplements the information from the branch destination address to the next branch source address.

For the usage of those functions, refer to section 5.7, Viewing the Trace Information.

(e) Changes the trace settings during user program execution.

Trace settings can be changed during user program execution.

1.3.4 Break Function

The emulator has the following seven break functions.

(1) PC break function (BREAKPOINT)

Breaks the program at the specified address by replacing the dedicated instruction onto the original instruction. This function cannot be set at a place other than RAM since a memory write occurs.

This function can be set in the [Breakpoint] sheet of the [Event] window. It can also be set in the [Source] or [Disassembly] window by double-clicking on the line to be set in the [S/W breakpoint] column.

(2) On-chip event break function

Breaks when the specified event has been detected by the on-chip event detection function.

This function is available when the operation after detecting an event is set as 'break' in the [Onchip Event] sheet of the [Event] window.

It can also be set in the [Source] or [Disassembly] window by double-clicking on the line to be set in the [Onchip event] column.

Note: On-chip event break settings can be changed during user program execution.

(3) AUD event break function

Breaks when the specified event has been detected by the AUD event detection function.

This function is available when the operation after detecting an event is set as 'break' in the [AUD Event] sheet of the [Event] window.

It can also be set in the [Source] or [Disassembly] window by double-clicking on the line to be set in the [AUD Event] column.

(4) External bus event break function

Breaks when the specified event has been detected by the external bus event detection function.

This function is available when the operation after detecting an event is set as 'break' in the [BUS Event] sheet of the [Event] window.

It can also be set in the [Source] or [Disassembly] window by double-clicking on the line to be set in the [BUS Event] column.

(5) Other event break functions

— Execution time event detection function

— External probe event detection function

Breaks when the specified event has been detected by other event detection functions.

This function is available when the operation after detecting an event is set as ‘break’ in the [Other Event] sheet of the [Event] window.

(6) Trace break function

Breaks when the AUD trace buffer and external bus trace buffer in the emulator become full.

This function can be set in the [Internal/AUD/Usermemory acquisition] dialog box and the [BUS acquisition] dialog box.

(7) Forced break function

Forcibly breaks the user program.

1.3.5 Probe Function

Six external probes can be placed on the emulator.

(1) Four input probes

Monitor input signals, break the user program, and start or end AUD tracing by satisfying the specified condition. This can be set on the [Other Event] sheet of the [Event] window.

(2) One event output probe

Outputs the event signal when the AUD event is detected. This can be set on the [Action] page of the dialog box for setting the AUD event.

(3) One GND

A probe for connecting the ground.

1.3.6 Peripheral I/O Analyzer Functions

When the pin signals of peripheral I/Os are connected to the analyzer cable, the emulator sets a break and starts or ends tracing the pin signals with the specified condition by analyzing the protocols of peripheral IPs.

Note: The analyzer functions to be supported differ depending on the product. For details on the specifications of each product, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help.

1.3.7 Performance Measurement Function

The emulator has five types of performance measurement functions.

(1) On-chip Performance Analysis Function (Onchip Performance Analysis)

This function applies a counter in the device to measure the number of cycles from one specified condition being satisfied until a next specified condition is satisfied.

Not only the number of cycles but also various items such as the number of cache misses or of TLB misses can be measured according to the supported devices.

Note: Items to be measured will differ according to the product. For details on the specifications of each product, refer to the online help.

(a) Setting the performance measurement conditions

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

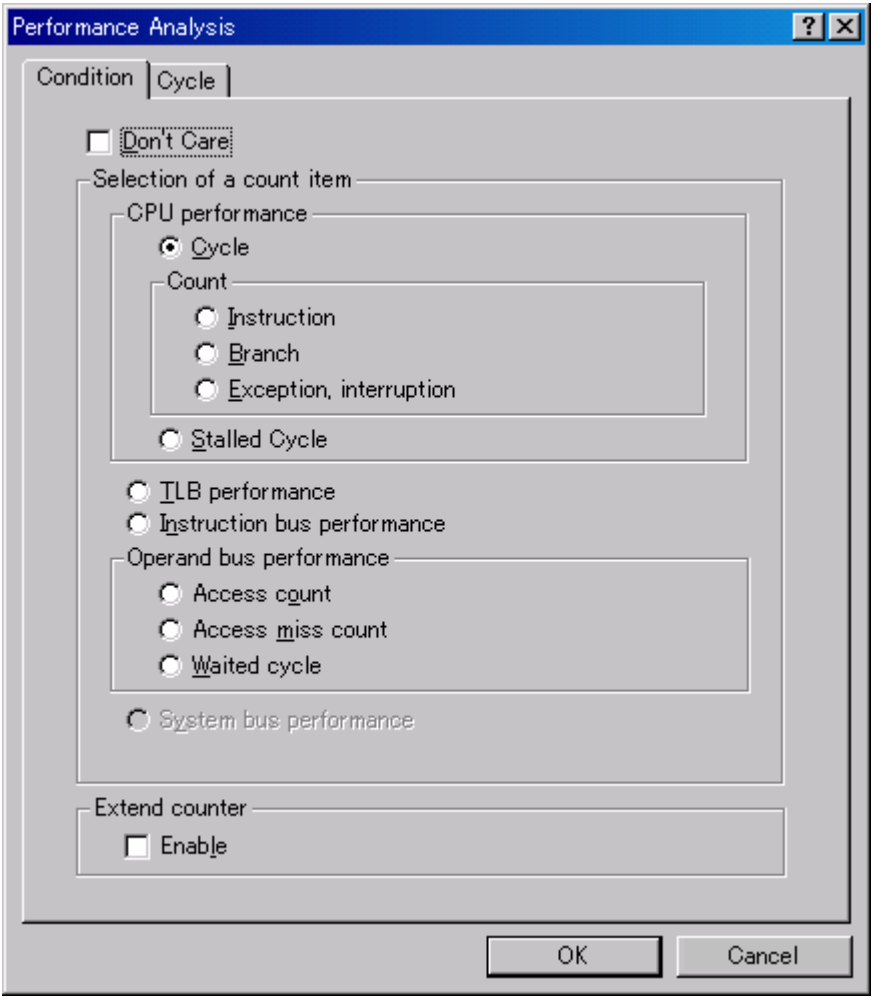


Figure 1.20 [Performance Analysis] Dialog Box

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

- Specifying the measurement start/end conditions

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

Notes: 1. When no measurement start/end conditions are specified, measurement is started by executing a program and ended when an event condition is satisfied.

2. When only the measurement start or end condition is specified, performance cannot be measured. Be sure to specify both of the measurement start and end conditions.

3. When the measurement start/end conditions are specified, step operation cannot be performed. In addition, when execution is restarted from the address where step operation has been stopped by the break conditions of BREAKPOINT or Event Condition, step functions are used and operation is disabled. Restart execution after the settings of the break conditions of BREAKPOINT or Event Condition have been canceled.

4. It is not possible to use the break conditions and the start/end conditions for performance measurement at the same time with one channel. If the start/end conditions for performance measurement are set, the settings of the break conditions will be disabled.

Table 1.3 Conditions Specified in the [Action] Page

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

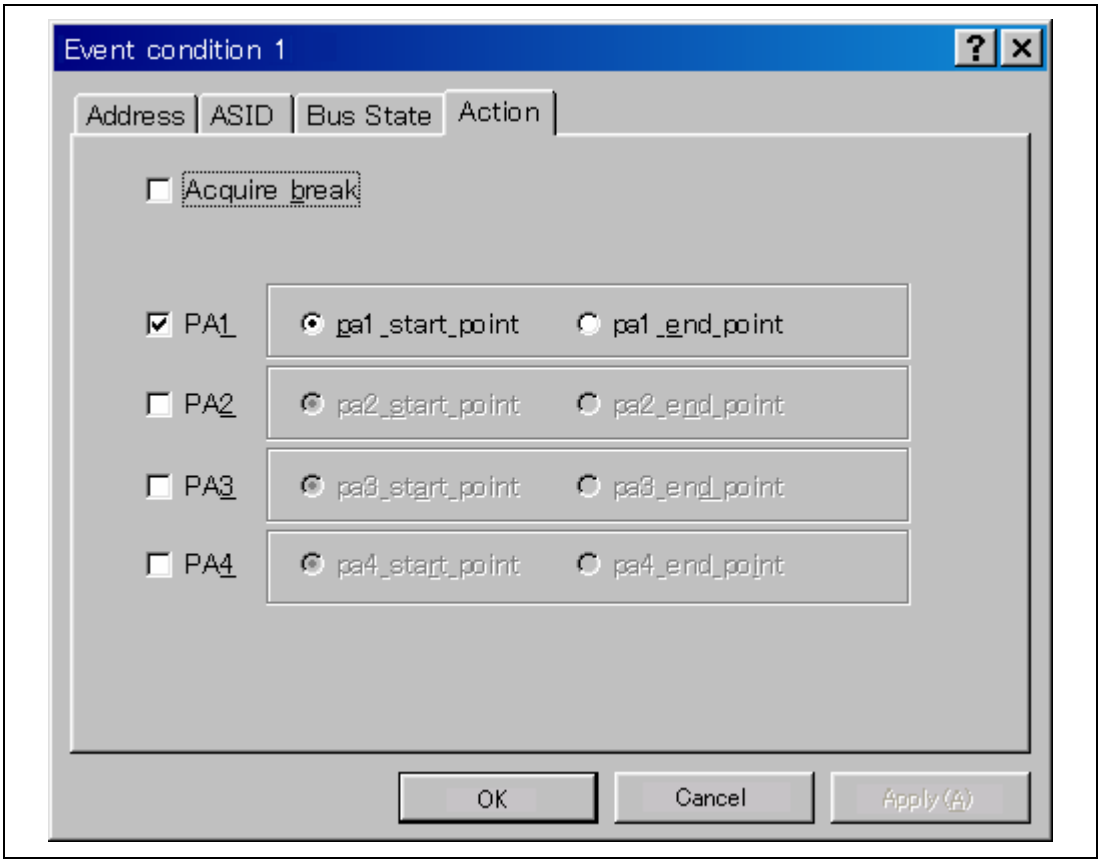


Figure 1.21 [Action] Page

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

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

  For details, see table 1.6.

- Measurement items

  Items are measured in the [Performance Analysis] dialog box for each channel from Ch1 to Ch4. A maximum of four conditions can be specified at the same time. Table 1.4 shows the measurement items. (Options in table 1.4 are parameters for <mode> of the PERFORMANCE_SET command. They are displayed in CONDITION of the [Performance Analysis] window.)

Table 1.4 Measurement Items

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

Table 1.4 Measurement Items (cont)

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

Table 1.4 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
Instruction bus performance (cont)	Instruction (cont)	Number of instruction cache miss	ICM	The number of cache misses by an instruction cache access (the number of accesses to the outside of the CPU core due to a cache miss).
		Number of internal-RAM access for instruction fetch (XY-RAM or L memory (O-L memory))	XL	The number of accesses for the XY memory in the MPU during memory accesses of the opcode.
		Number of I-L memory access for instruction fetch	ILIF	The number of accesses for the I-L memory in the MPU during memory accesses of the opcode.
		Number of U memory access for instruction fetch	ULF	The number of accesses for the U memory in the MPU during memory accesses of the opcode.
Operand bus performance	Access count	Number of memory access for operand fetch (READ)	MR	The number of memory accesses by an operand read (equal to loading on the operand bus). Accesses by the PREF instruction or canceled accesses are not included.
		Number of memory access for operand fetch (WRITE)	MW	The number of memory accesses by an operand write (equal to storing memory on the operand bus). Canceled accesses are not included.
		Number of operand cache access (READ)	CR	The number of operand-cache reads during memory access (read) of an operand.
		Number of operand cache access (WRITE)	CW	The number of operand-cache reads during memory access (write) of an operand.

Table 1.4 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
Operand bus performance (cont)	Access count (cont)	Number of internal-RAM access for operand fetch (READ) (XY-RAM or L memory (O-L memory))	XLR	The number of accesses to XY memory in the MPU during memory access (read) of an operand. (Accesses via the XY bus and the operand bus are included. When MOVX and MOVY are executed simultaneously, it increments one count regardless of the read or write.)
		Number of internal-RAM access for operand fetch (WRITE) (XY-RAM or L memory (O-L memory))	XLW	The number of accesses to XY memory in the MPU during memory access (write) of an operand. (Accesses via the XY bus and the operand bus are included. When MOVX and MOVY are executed simultaneously, it increments one count regardless of the read or write.)
		Number of I-L memory access for operand fetch (READ/WRITE)	ILRW	The number of accesses to I-L memory in the MPU during memory access (read/write) of an operand.
		Number of U-RAM access (READ)	UR	The number of U-memory accesses during memory access (read) of an operand. (Accesses via the cache are not included.)
		Number of U-RAM access (WRITE)	UW	The number of U-memory accesses during memory access (write) of an operand. (Accesses via the cache are not included.)
	Access miss count	Number of operand cache miss (READ)	CMR	The number of cache misses by an operand cache access (read) (number of accesses to the outside of the CPU core due to a cache miss). Cache misses are not counted by the PREF instruction.

Table 1.4 Measurement Items (cont)

Classification	Type	Measurement Item	Option	Note
Operand bus performance (cont)	Access miss count (cont)	Number of operand cache miss (WRITE)	CMW	The number of cache misses by an operand cache access (write) (number of accesses to the outside of the CPU core due to a cache miss). Write-through accesses are not counted. Cache misses are not counted by the PREF instruction.
		Number of U-RAM read-buffer miss	UBM	This function is disabled for the MPU that the U memory is not incorporated.
	Waited cycle	Waited cycles for operand fetch (READ)	WOR	The number of wait cycles by a memory access (read) of an operand.
		Waited cycles for operand fetch (WRITE)	WOW	The number of wait cycles by a memory access (write) of an operand.
		Waited cycles for operand cache miss (READ)	WCMR	The number of wait cycles by an operand cache miss (read) (however, the number of wait cycles of cache FIII is included due to contention).
		Waited cycles for operand cache miss (WRITE)	WCMW	The number of wait cycles by an operand cache miss (write).
		Number of waited cycles by an I-L memory access for operand fetch (READ)	WILR	The number of waited cycles by an I-L memory access (read) of an operand.
		Number of waited cycles by an I-L memory access for operand fetch (WRITE)	WILW	The number of waited cycles by an I-L memory access (write) of an operand.

Table 1.4 Measurement Items (cont)

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

Table 1.5 shows the measurement items and methods that are mainly used.

Table 1.5 Main Measurement Items

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

Each measurement condition is also counted when conditions in table 1.6 are generated.

Table 1.6 Performance Measurement Conditions to be Counted

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

Notes: 1. In the non-realtime trace mode of the AUD trace and memory output trace, normal counting cannot be performed because the generation state of the stall or the execution cycle is changed.

2. Since the clock source of the counter is the CPU clock, counting also stops when the clock halts in the sleep mode.

- Extension setting of the performance-result storing counter

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

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

(b) Displaying the result of performance

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

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

Note: If a performance counter overflows as a result of measurement, "********" will be displayed.

(c) Initializing the measured result

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

(2) AUD Performance Analysis Function (AUD Performance Analysis)

The emulator allows you to measure the time or count of execution between specified events in the AUD event detection system. It is possible to set the resolution of the timer to any of the following values:

20 ns, 40 ns, 100 ns, or 400 ns.

At 20 ns the maximum time that can be measured is about six hours, and at 400 ns the maximum time is about five days.

(3) External Bus Performance Analysis Function (BUS Performance Analysis)

The emulator allows you to measure the time or count of execution between specified start and end conditions. It is possible to set the resolution of the timer to any of the following values:

20 ns, 50 ns, or 400 ns.

At 20 ns the maximum time that can be measured is about six hours, and at 400 ns the maximum time is about five days.

(4) Profiling Function (Profile)

The profiling function is used to measure the performance of each function.

A function having low performance can be easily found if the statistics of the time for each function are maintained.

Items that can be measured are the same as those for the on-chip performance measurement function.

Notes: 1. Use of the profiling and on-chip performance analysis functions at the same time is not possible. The [Can not use this function] error message dialog box will be displayed if simultaneous use is attempted.

2. In this function, a break occurs whenever a branch is generated, and information is collected to execute the user program again. Therefore, realtime emulation of the user program will not be performed.

(5) Realtime Profiling Function (Realtime Profile)

The performance of each function can be measured within the specified address range.

A function having low performance can be easily found if the statistics of the time for each function are monitored.

In this function, performance information is collected without break. Therefore, realtime emulation of the user program will be performed.

The specifiable address ranges are as follows:

- When the expansion profiling unit is not connected: 512 kbytes to 4 Mbytes (8 blocks at 512 kbytes)
- When the expansion profiling unit is connected: 512 kbytes to 12 Mbytes (24 blocks at 512 kbytes)

1.3.8 Coverage Function

The emulator displays the information on the executed instructions in the C/C++ and assembly-language levels to measure the C0 coverage.

1.3.9 Memory Access Functions

The emulator has the following memory access functions.

(1) Memory read/write function

[Memory] window: The memory contents are displayed in the window. Only the amount specified when the [Memory] window is opened can be read. Since there is no cache in the emulator, read cycles are always generated. If the memory is written in the [Memory] window, read cycles in the range displayed in the [Memory] window will occur for updating the window. When the [Memory] window is not to be updated, change the setting in [Lock Refresh] from the popup menu.

me command: A command line function that reads or writes the specified amount of memory at the specified address.

(2) User program downloading function

A load module registered in the workspace can be downloaded. Such module can be selected from [Download Module] in the [Debug] menu. Downloading is also possible by a popup menu that is opened by right-clicking on the mouse at the load module in the workspace. The user program is downloaded to the RAM or flash memory.

When downloading to the flash memory, select [Emulator] from the [Options] menu, open the [Configuration] window, and perform required settings on the [Loading flash memory] page.

This function also downloads information required for source-level debugging such as debugging information.

(3) Memory data uploading function

The specified amount of memory from the specified address can be saved in a file.

(4) Memory data downloading function

The memory contents saved in a file can be downloaded. Select [Load] from the popup menu in the [Memory] window.

(5) Displaying the variable contents

The variable contents specified in the user program are displayed.

(6) Monitoring function

The emulator monitors a value in the area that has been accessed without suspending the execution of program and displays it on the window. The virtual address that has been accessed by the CPU can be only monitored.

(7) Emulation memory function

The emulator allocates the memory for emulation in the CS0 area.

(8) Other memory operation functions

Other functions are as follows:

- Memory fill
- Memory copy
- Memory save
- Memory verify
- Memory search
- Internal I/O display
- Cache table display and edit (only for devices incorporating caches)
- TLB table display or edit (only for devices incorporating MMU)
- Displaying label and variable names and their contents

For details, refer to the online help.

Notes: 1. Memory access during user program execution:
When a memory is accessed from the memory window, etc. during execution of the user program, execution stops for the memory access and is then resumed. Therefore, realtime emulation cannot be performed. The stopping time of the user program depends on the performance of the host machine in use or the JTAG clock; that time under the following environment is shown below:

Environment:

Host machine: 800 MHz (Pentium® III)
SH7323: 96 MHz (CPU clock)
JTAG clock: 30 MHz

When a one-byte memory is read in the command-line window, the stopping time will be about 96 ms.

2. Memory access during user program break:
The program can also be downloaded for the flash memory area by the emulator. Other memory write operations are enabled for the RAM area. Therefore, an operation such as memory write or BREAKPOINT should be set only for the RAM

area. When the memory area can be read by the MMU, do not perform memory write, BREAKPOINT setting, or downloading.

3. Cache operation during user program break:
   When cache is enabled in the device incorporating a cache, the emulator accesses the memory by the following methods:
   - At memory write: Writes to the cache, then issues an external single write. The LRU is not updated.
   - At memory read: Reads memory from the cache. The LRU is not updated.

1.3.9 Stack Trace Function

The emulator uses the stack's information to display the name of the calling function for a function at the current program counter. This function can be used only when the load module that has the Dwarf2-type debugging information is loaded. For the usage of this function, refer to section 6.21, Stack Trace Function.

1.3.10 User-interrupt Open Function during User Program Break

Some devices to be debugged enable all interrupts while executing emulation to the user. During a user program break, it is possible to specify the mode whether or not the interrupt processing is executed.

1.3.11 Online Help

An online help explains the usage of each function or the command syntax that can be entered from the command line window.

Select [Emulator Help] from the [Help] menu to view the emulator help.

1.4 Environmental Conditions

⚠ CAUTION

Observe the conditions listed in tables 1.7 and 1.8 when using the emulator. Failure to do so will cause illegal operation in the user system, the emulator product, and the user program.

Table 1.7 Environmental Conditions

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

Table 1.8 lists the acceptable operating environments.

Table 1.8 Operating Environments

Item	Description
Host machine	Built-in Pentium® III or higher-performance CPU (1 GHz or higher recommended); IBM PC or compatible machine with USB 1.1/2.0 (Full-Speed).
Operating system	Windows® 2000 or Windows® XP
Minimum memory capacity	128 Mbytes or more (512 Mbytes recommended)
Hard-disk capacity	Installation disk capacity: 250 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 machine; compatible with Windows® 2000 or Windows® XP.
Display	Monitor resolution: 1024 x 768 or higher
AC-input power supply	5.0 ± 0.25 V (USB-bus power type)
Current consumption	Voltage: AC100 V ± 10% Frequency: 50/60 Hz Consumption power: 48 W
CD-ROM drive	Required to install the High-performance Embedded Workshop for the emulator or refer to the emulator user's manual.

Section 2 Setting Up the Emulator

2.1 Flow Chart before Using the Emulator

Unpack the emulator and prepare it for use as follows:

⚠ 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.

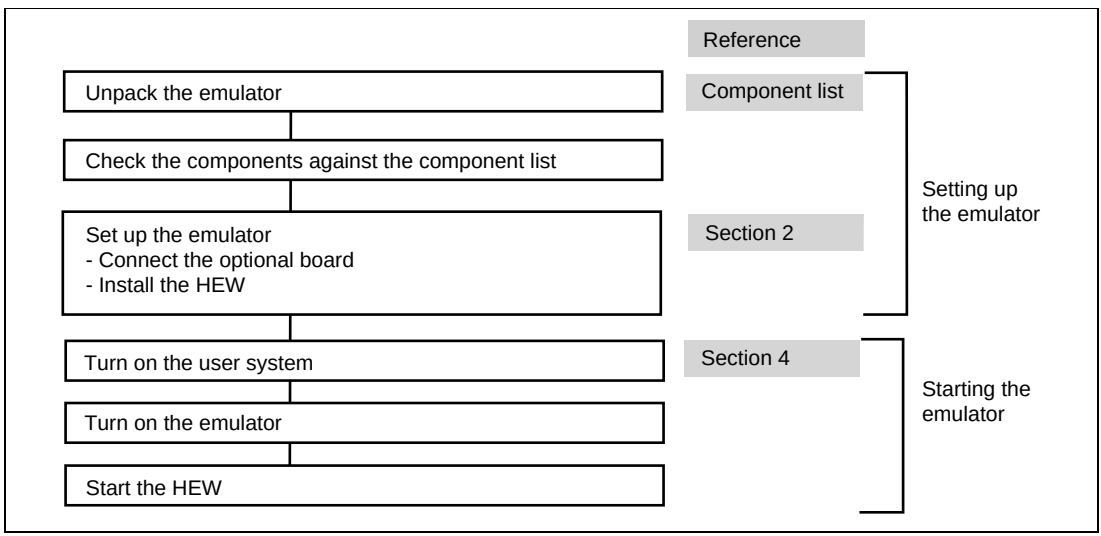


Figure 2.1 Emulator Preparation Flow Chart

2.2 Installing Debugger

2.2.1 CD-R

The root directory of the CD-R contains a setup program for installing the emulator's software. The folders contain the files and programs listed below.

Table 2.1 Contents of the CD-R Directories

Directory Name	Contents	Description
Dlls	Microsoft® runtime library	A runtime library for the High-performance Embedded Workshop. The version is checked at installation and this library is copied to the hard disk as part of the installation process.
Drivers	E200F emulator driver	USB drivers for the E200F emulator.
Help	Online help for the E200F emulator	An online help file. This is copied to the hard disk as part of the installation process.
Manual	E200F emulator manuals	E200F emulator user's manuals. They are provided as PDF files.
sot	Diagnostic program for the E200F emulator	The usage is described in section 3.4, Diagnostic Procedure.

Execute HewInstMan.exe from the root directory of the CD-R to start the installation manager.

Follow the cues given by the installation manager to install the software.

Note: When a driver is installed in Windows® XP, a warning message on the Windows® logo test may be displayed, but it is not a problem. Select [Continue Anyway] to proceed with driver installation.

2.3 Connecting the Optional Units to the Emulator Main Unit Case

Optional units are included in the emulator. This section describes how to connect them to the E200F main unit case.

2.3.1 Connecting the E200F Trace Unit to the User System

- Open the cover of TRACE I/F on the side of the main unit case.
- Connect the trace cable provided for the trace unit to the emulator as shown in figure 2.2.

Figure 2.2 Connecting the Trace Cable to E200F

- Connect the trace unit to the trace cable (CN1 side).

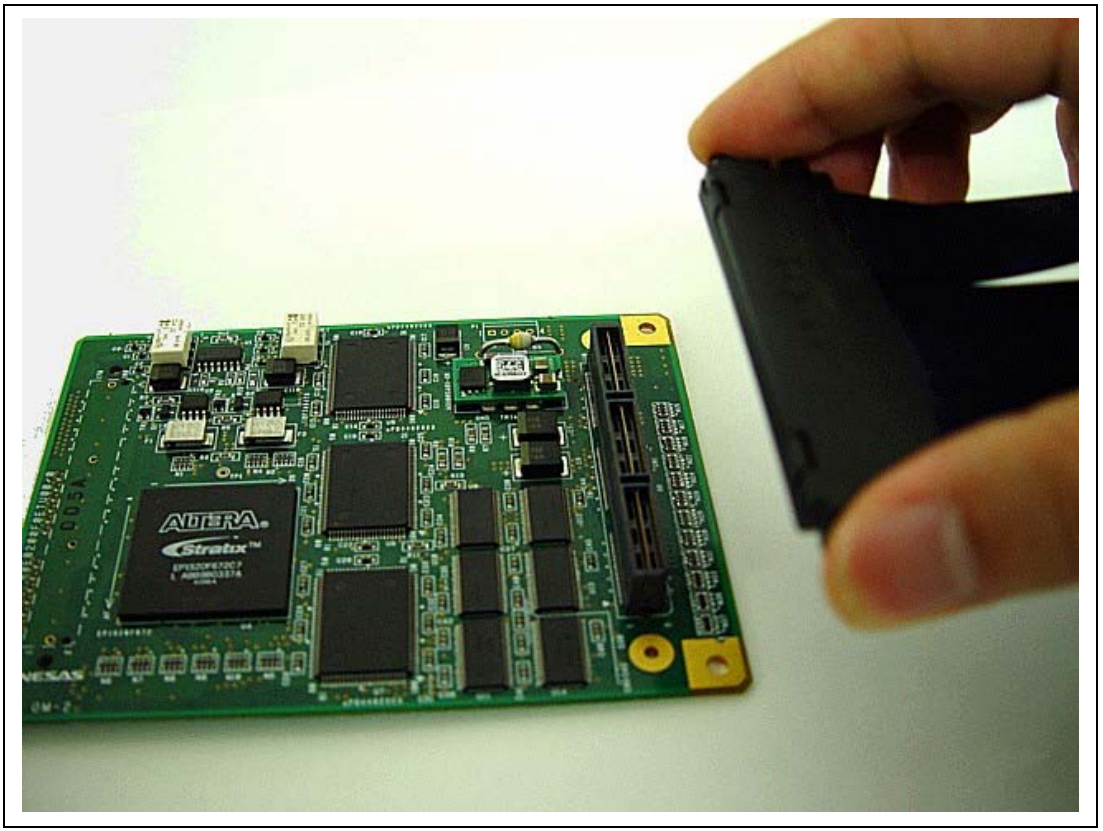


Figure 2.3 Connecting the Trace Cable to the Trace Unit

- After checking the location of pin 1, connect the user system to the trace unit.

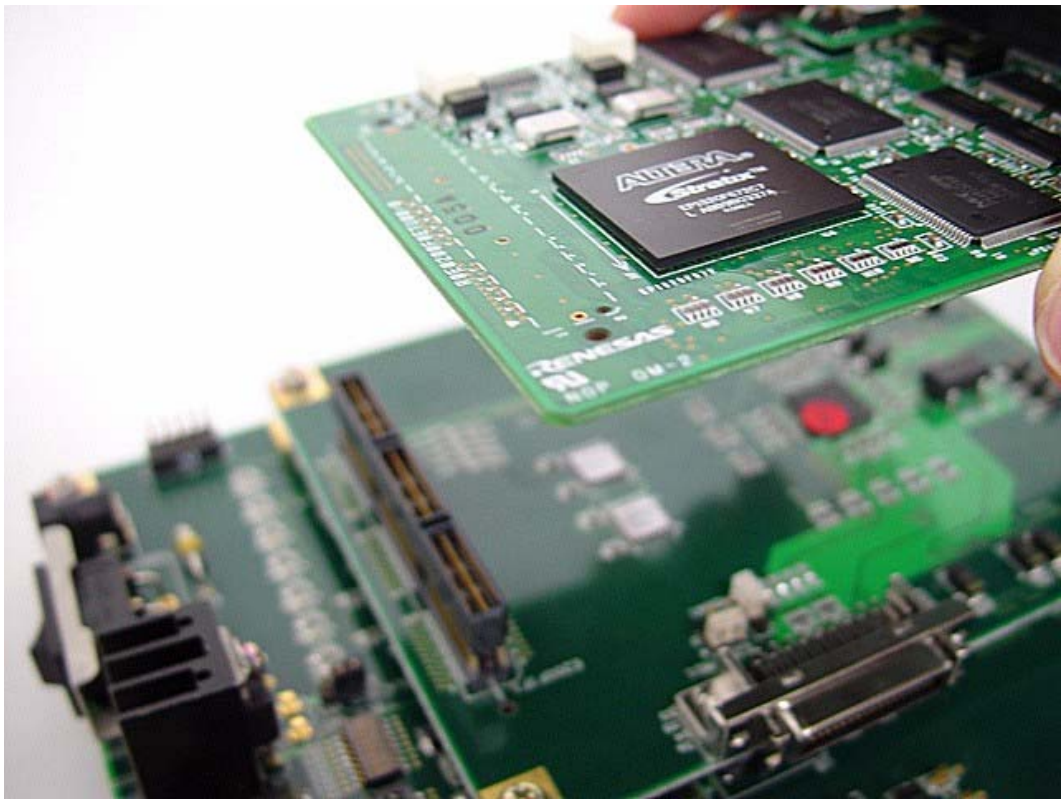


Figure 2.4 Connecting the User System to the Trace Unit

⚠ CAUTION
Check the location of pin 1 before connecting.

Notes: 1. Connection of the signals differs depending on the MPU used.
2. To connect the signals to the user system, refer to the additional document, Supplementary Information on Using the SHxxxx.

2.3.2 Connecting the E200F Peripheral I/O Analyzer Unit to the User System

Figure 2.5 Connecting the Analyzer Cable to E200F when Using the Peripheral I/O Analyzer Unit

⚠ CAUTION
Check the location of pin 1 before connecting.

Notes: 1. Connection of the signals differs depending on the peripheral I/O analyzer unit.

2. To connect the signals to the user system, refer to the additional document, Supplementary Information on Using the SHxxxx.

2.3.3 Connecting the E200F Expansion Profiling Unit to the Main Unit Case

- Remove the screw for fastening the base unit.

Figure 2.6 Screw for the Base Unit for Placing the Emulator Vertically

Figure 2.7 Removing the Base Unit for Placing the Emulator Vertically

- Remove two screws on the rear side of the main unit case.

Figure 2.8 Screws for the Main Unit Case

- Remove the main unit case as shown in figure 2.9.

Figure 2.9 Removing the Main Unit Case

- After checking the location of pin 1, connect the expansion profiling unit to the main unit case.

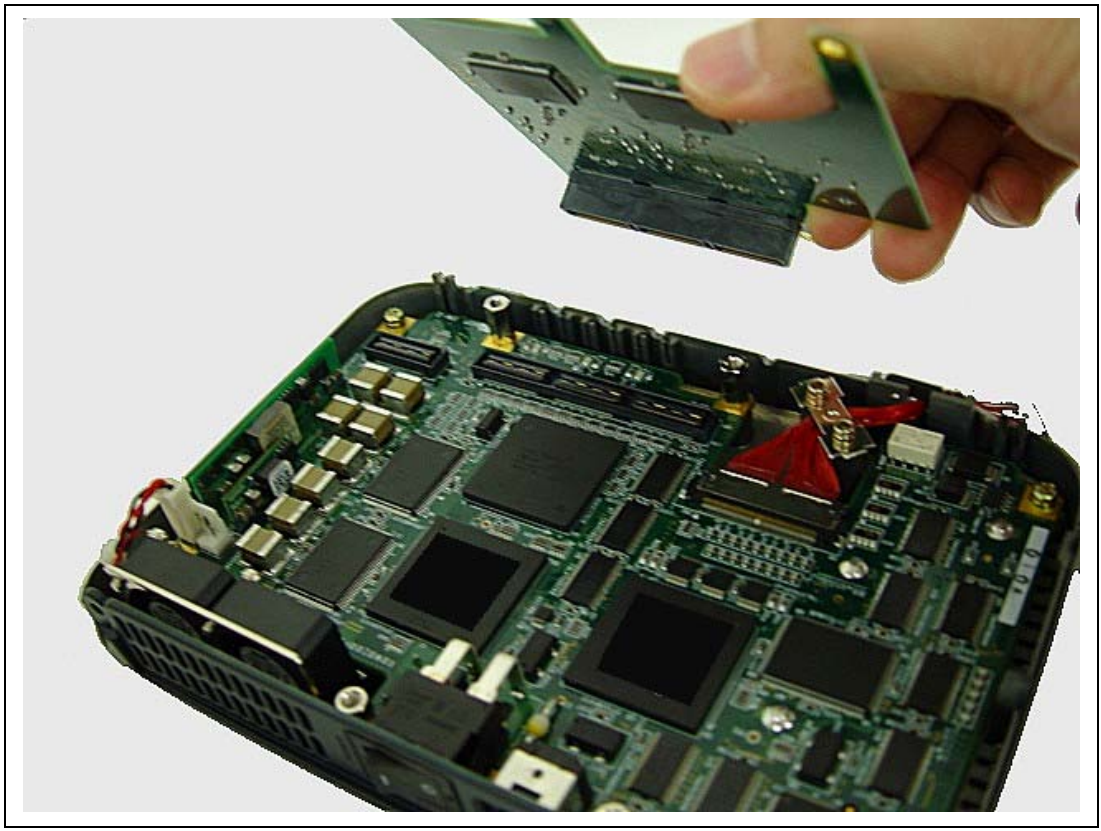


Figure 2.10 Connecting the Expansion Profiling Unit to the Main Unit Case

- Fasten the expansion profiling unit with screws provided.

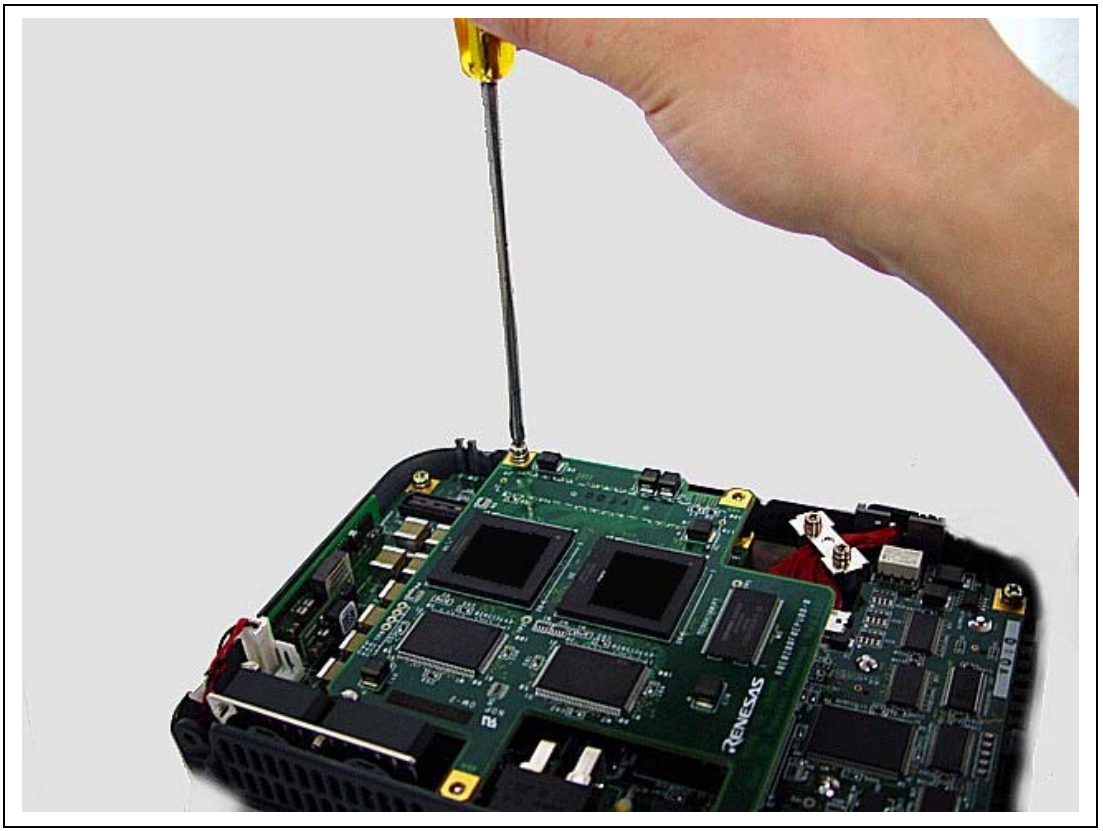


Figure 2.11 Fastening the Expansion Profiling Unit to the Main Unit Case

- Close the cover of the main unit case to be fastened with two screws.
- Fasten the screw of the base unit.

2.4 Connecting the AC Adapter to the Emulator Main Unit Case

Connect the provided AC adapter to the main unit case.

Figure 2.12 Connecting the AC Adapter to the Main Unit Case

Connect the provided AC adapter to the connector to input DC (+12 V) of the AC adapter marked ‘DC IN’.

⚠ WARNING

Be sure to use the AC adapter dedicated for E200F. Failure to do so will result in a FIRE HAZARD and will damage the user system or the emulator product.

2.5 Connecting the Emulator to the Host Machine

This section describes how to connect the emulator to the host machine. For the position of each connector of the emulator, refer to section 1.2, Emulator Hardware Configuration.

Notes: 1. When [Add New Hardware Wizard] is displayed, select the [Search for the best driver for your device. (Recommended)] radio button and then the [Specify a location] check box to select the path to be searched for drivers. The location must be specified as <Drive>:\DRIVERS. (<Drive> is the CD drive letter.)

2. Be sure to install the software for the emulator before putting the emulator in place.

⚠ WARNING

Always switch OFF the emulator product and the user system before connecting or disconnecting any CABLES except for the USB interface cable. 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.

The emulator is connected to the host machine via the USB 1.1/2.0. Figure 2.13 shows the system configuration.

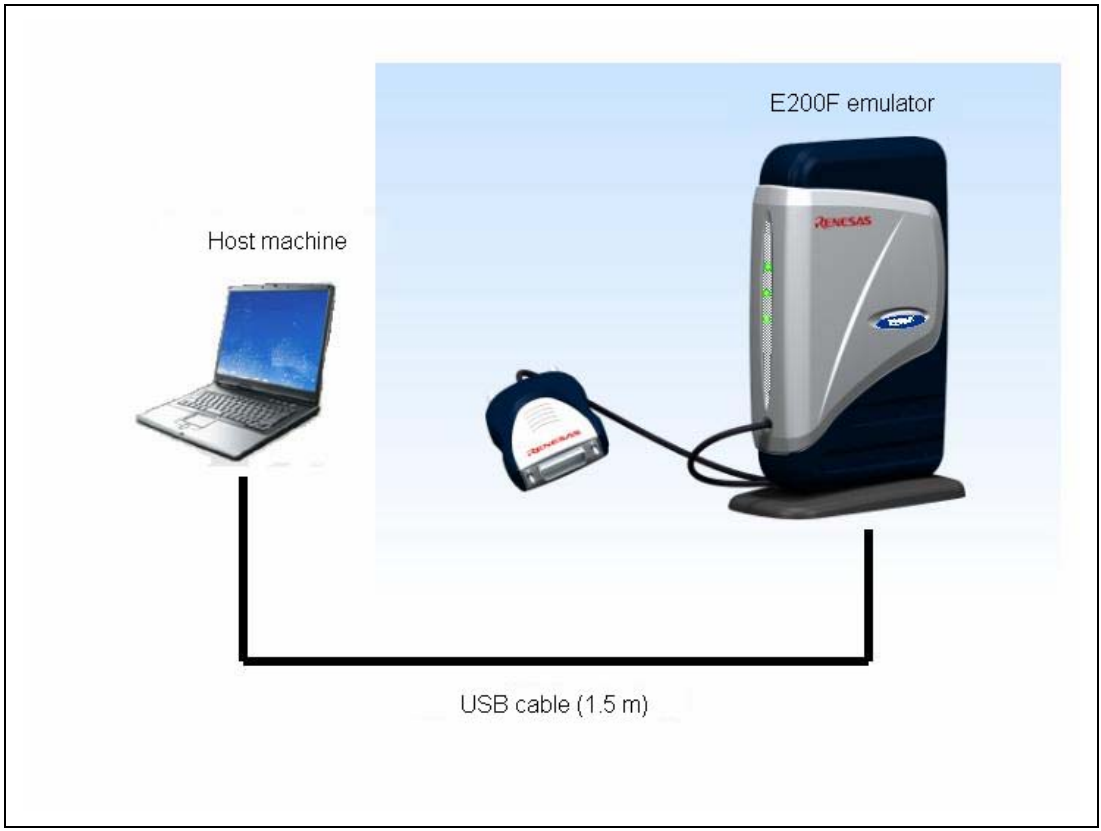


Figure 2.13 System Configuration when Connecting the Emulator to the Host Machine

2.6 Connecting the Emulator to the User System

Use the procedure below 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.

1. Check that the emulator is turned off.
2. Connect the probe head connector to the user system.
3. Fasten the probe head to the user system with the screws (for the 36-pin connector only).

(1) The connector must be installed to the user system. Table 2.2 shows the recommended H-UDI port connectors for the emulator. The connector type of the probe head will differ depending on the emulator in use.

Table 2.2 Recommended Connectors

Connector	Type Number	Manufacturer	Specifications	Emulator Type
36-pin connector	2514-6002	3M Limited	36-pin straight type	R0E0200F0EMU00
38-pin connector	2-5767004-2	Tyco Electronics AMP K.K.	38-pin Mictor type	R0E0200F2EMU00

(2) The pin assignments of the connector are shown in section 2 in the additional document, Supplementary Information on Using the SHxxxx.

(3) When using the 36-pin user system interface cable, connect pins 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 33, 34, and 36 of the H-UDI port connector to GND on the PCB. When using the 38-pin user system interface cable, connect pin 5 of the H-UDI port connector and ground bus leads of the connector to GND. These pins are used as electrical GND and to monitor the connection of the H-UDI port connector. Note the pin assignments of the H-UDI port connector.

(4) For the recommended connectors and pin assignments of the optional products, refer to the additional document, Supplementary Information on Using the SHxxxx.

2.6.1 Connecting the E200F H-UDI/AUD Probe to the User System

Connect the H-UDI/AUD probe to the user system as shown in figure 2.14.

Figure 2.14 Connecting the H-UDI/AUD Probe to the User System

Fasten the user system and the H-UDI/AUD probe with screws as shown in figure 2.15.

Figure 2.15 Fastening the User System and the H-UDI/AUD Probe

⚠ CAUTION

Note that the pin number assignments of the connector differ from those of the connector manufacturer.

Notes:
1. Connection of the signals differs depending on the package. For details, refer to the MPU's pin assignments.
2. The range of communication that the emulator operates at is different depending on the MPU used.
3. To connect the signals from the connector, refer to section 1 in the additional document, Supplementary Information on Using the SHxxxx.

4. When developing user systems, do not connect the TDI and TDO signals of the device to the boundary scan loop, or separate them by using a switch (figure 2.16).

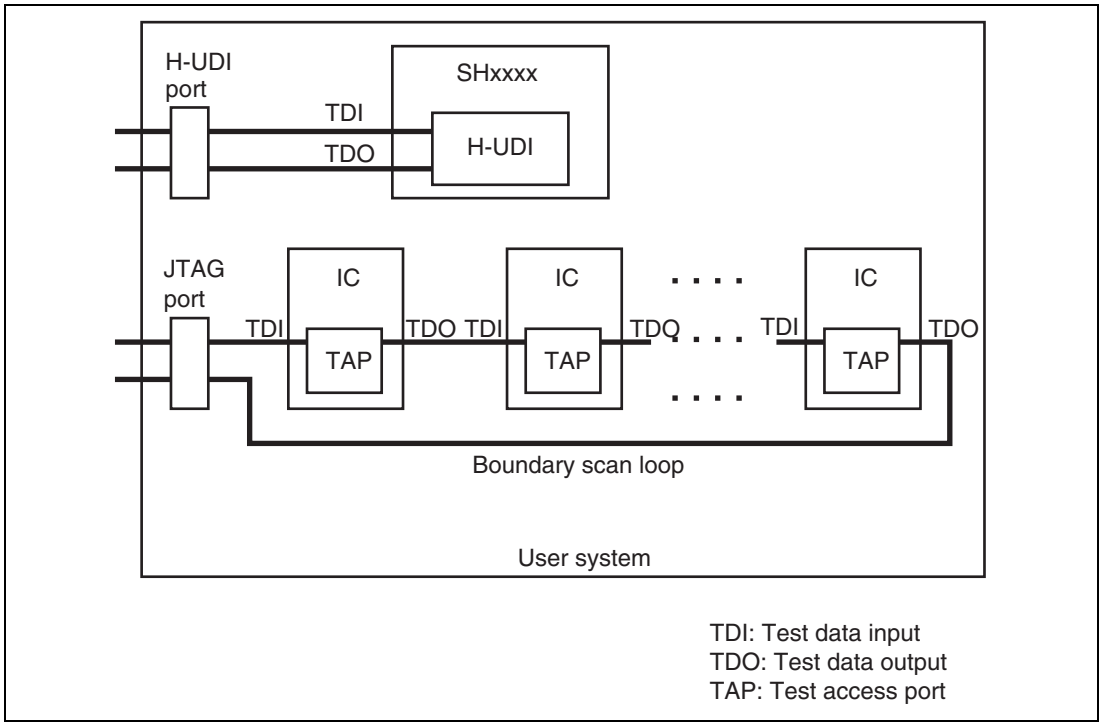


Figure 2.16 User System Example

2.6.2 Connecting System Ground

⚠ WARNING

Separate the frame ground from the signal ground at the user system. 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 emulator's signal ground is connected to the user system's signal ground. In the emulator, the signal ground and frame ground are connected. In the user system, connect the frame ground only; do not connect the signal ground to the frame ground (figure 2.17).

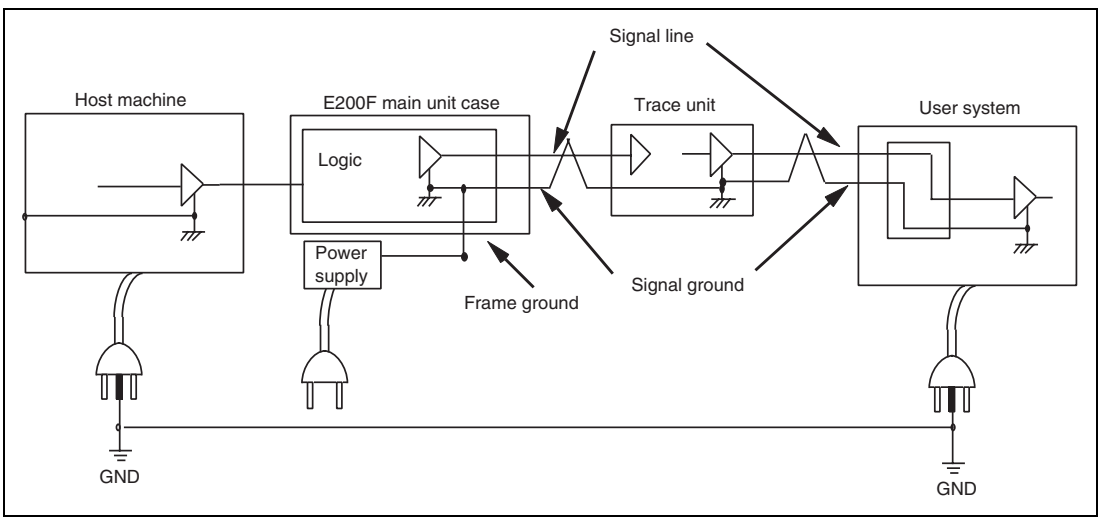


Figure 2.17 Connecting System Ground

2.7 Changing the Settings

In the emulator, it is possible to change the emulator function flexibly depending on the user's request for debugging.

When the High-performance Embedded Workshop is activated and connected to the emulator, the required functions can be selected. At this time, the following dialog box will be displayed.

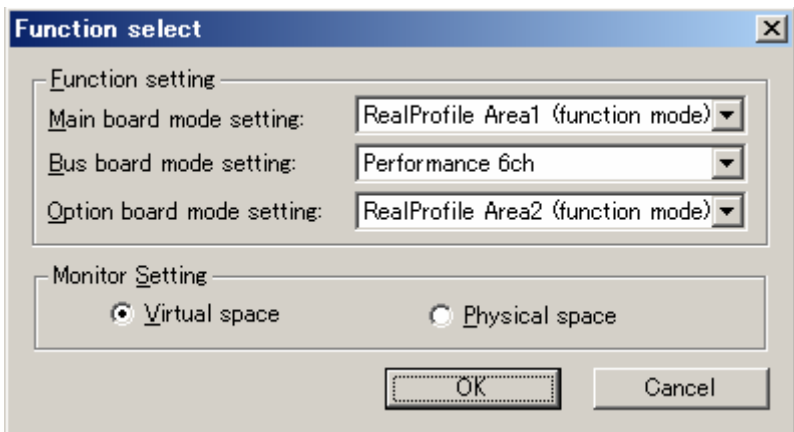


Figure 2.18 [Function select] Dialog Box

The following sections describe the contents that can be selected in this dialog box.

Note: For details on each function, refer to section 1.3, Emulator Functions.

2.7.1 Changing the Functions when Using the E200F Main Unit

Select the item in the [Main board mode setting] combo box of the [Function select] dialog box.

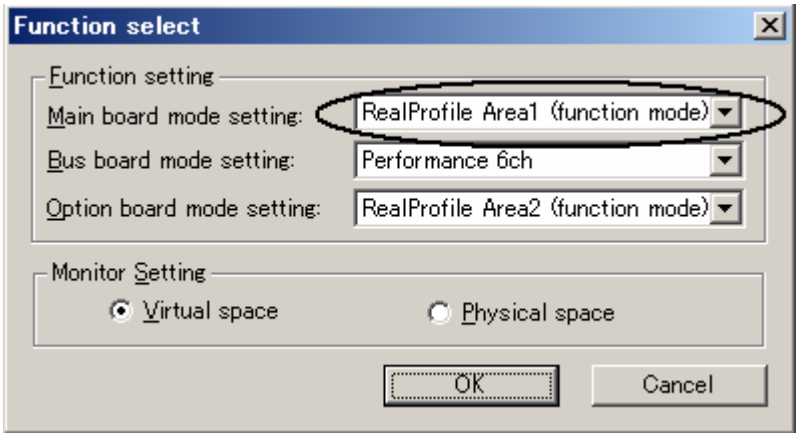


Figure 2.19 [Function select] Dialog Box

- [RealProfile Area1 (function mode)]: Measures accumulation of execution time of each function; the subroutine execution time is not included.
- [RealProfile Area1 (nest mode)]: Measures accumulation of execution time of each function; the subroutine execution time is also included.
- [Coverage (4M)]: Acquires the information on the C0 coverage in the 4-Mbyte areas.
- Others: The peripheral I/O analyzer function can be selected depending on the product.

Note: Using the realtime profiling and coverage functions increases the ranges that can be set with the expansion profiling unit.

2.7.2 Changing the Functions when Using the External Bus Trace Unit

Select the item in the [Bus board mode setting] combo box of the [Function select] dialog box.

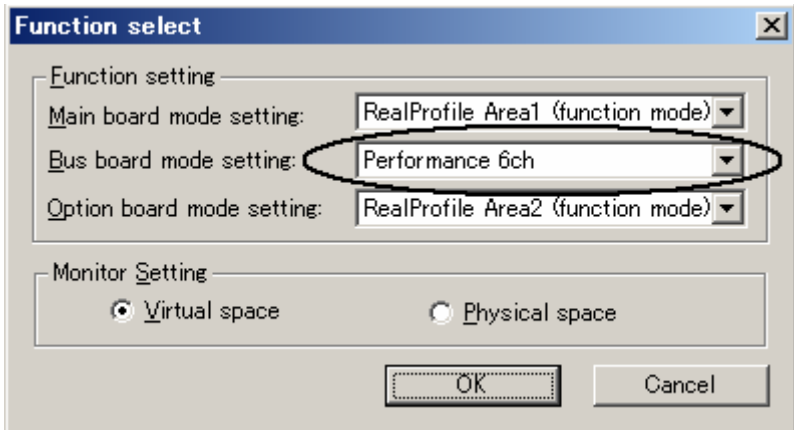


Figure 2.20 [Function select] Dialog Box

- [Trace/break 6ch (Trace 262144 cycles)]: Uses the channel for detecting the external bus event as a break.
- [Performance 6ch]: Uses the channel for detecting the external bus event as a channel for measuring the execution time.
- [Emulation memory (4M, Trace 8192 cycles)]: Uses the external emulation memory function (4 Mbytes x 1 block).

Note: When the trace unit is not connected to the emulator, this combo box is displayed in gray.

2.7.3 Changing the Functions when Using the Expansion Profiling Unit

Select the item in the [Option board mode setting] combo box of the [Function select] dialog box.

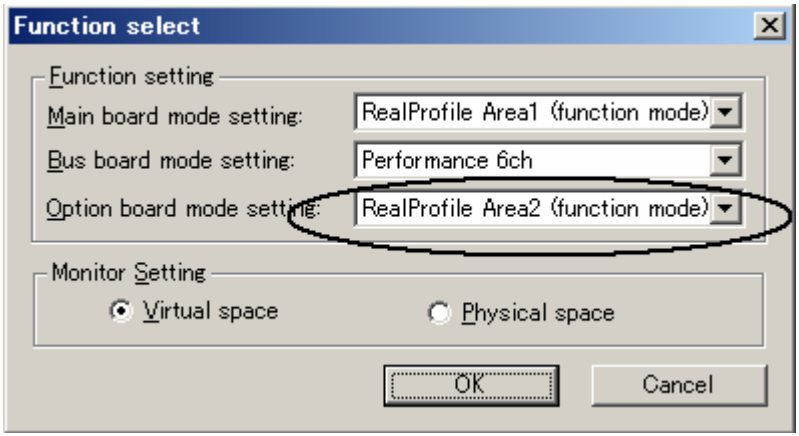


Figure 2.21 [Function select] Dialog Box

- [RealProfile Area2 (function mode)]: Measures accumulation of execution time of each function; the subroutine execution time is not included.
- [RealProfile Area2 (nest mode)]: Measures accumulation of execution time of each function; the subroutine execution time is also included.
- [Coverage (8M)]: Acquires the information on the C0 coverage in the 8-Mbyte areas.

Note: When the expansion profiling unit is not connected to the emulator, this combo box is displayed in gray.

2.7.4 Changing the Monitoring Function

There are two radio buttons in the [Monitor Setting] group box of the [Function select] dialog box.

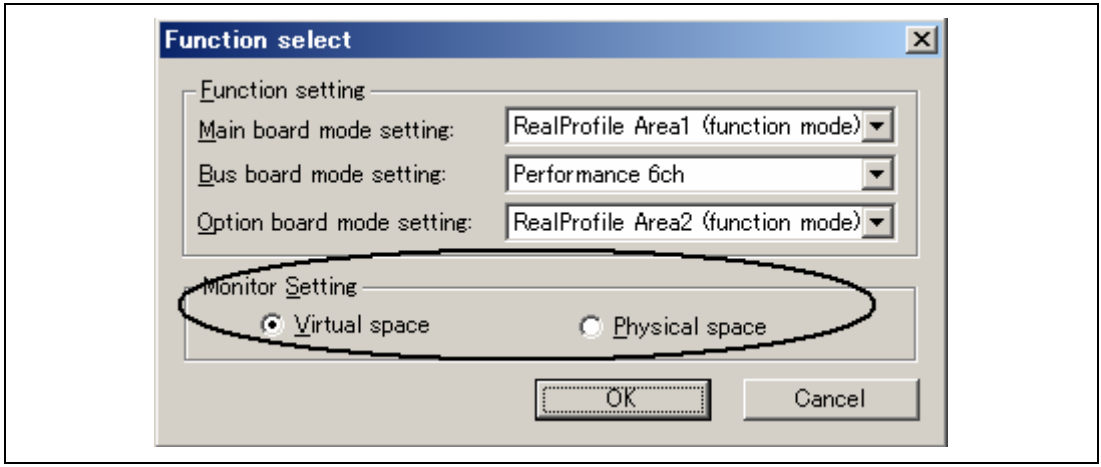


Figure 2.22 [Function select] Dialog Box

- [Virtual space] radio button: Displays the monitoring function as a virtual address. If the value is changed without accessing the CPU, that value cannot be displayed.
- [Physical space] radio button: Displays the monitoring function as a physical address.

Section 3 Hardware Specifications

3.1 List of Specifications

Table 3.1 lists the external dimensions and mass of the emulator.

Table 3.1 External Dimensions and Mass

No.	Item	Specification
1	External dimensions of E200F main unit case	185 x 130 x 45 (mm)
2	Mass of E200F main unit case	321 (g)
3	External dimensions of E200F trace unit	120 x 90 x 1.6 (mm)
4	Mass of E200F trace unit	104 (g)
5	External dimensions of E200F profiling unit	115 x 89 x 1.6 (mm)
6	Mass of E200F profiling unit	95 (g)

3.2 Interface Circuits in the Emulator

Figures 3.1 and 3.2 show 36-pin interface circuits in the emulator. Use them as a reference to determine the value of the pull-up resistance.

Figures 3.3 and 3.4 show 38-pin interface circuits in the emulator.

Note: The IC with UVCC power supply operates at 3.3 V or VCC (1.8 to 3.3 V) from the H-UDI port connector.

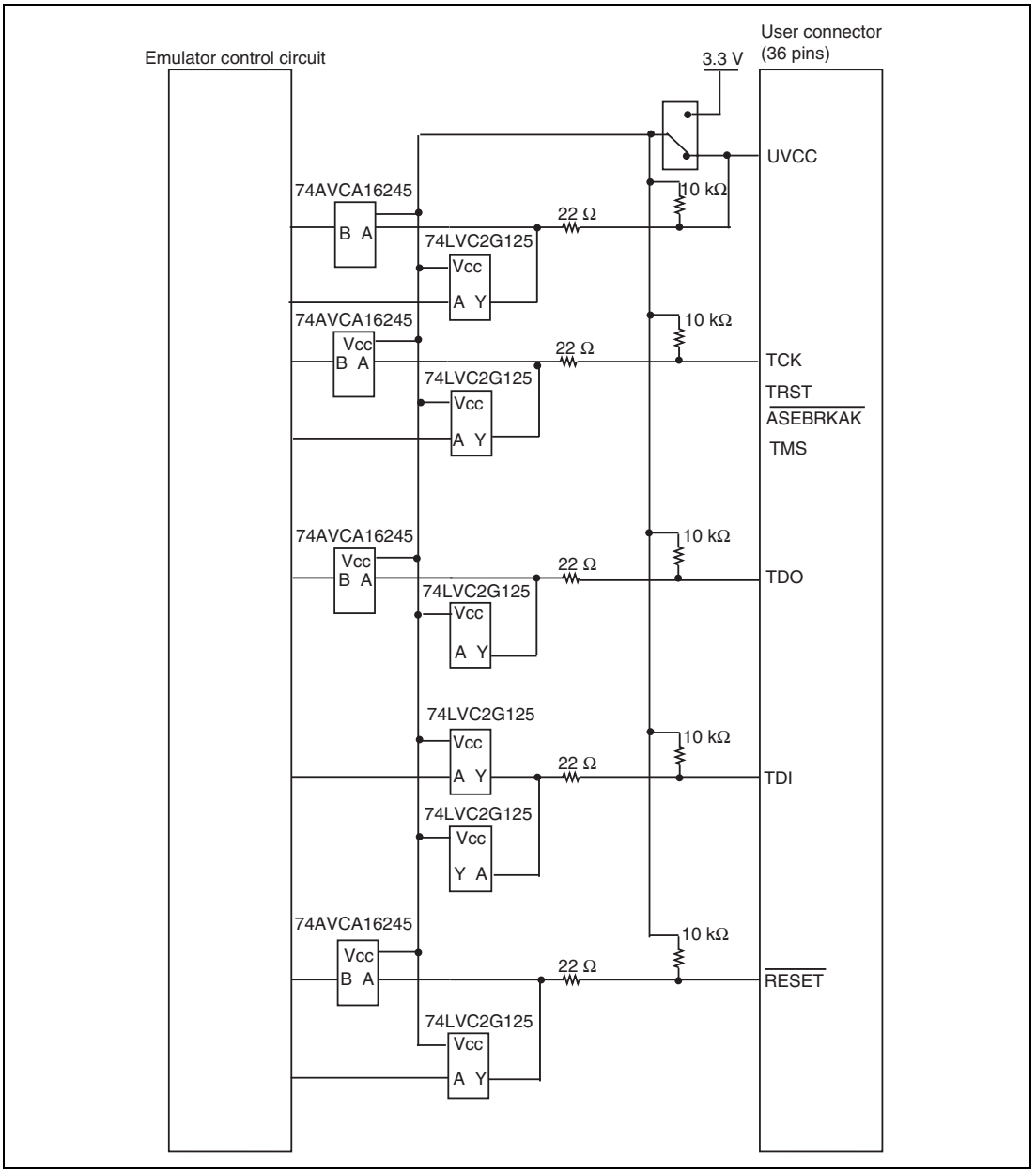


Figure 3.1 Interface Circuits in the Emulator (36-pin Interface for H-UDI)

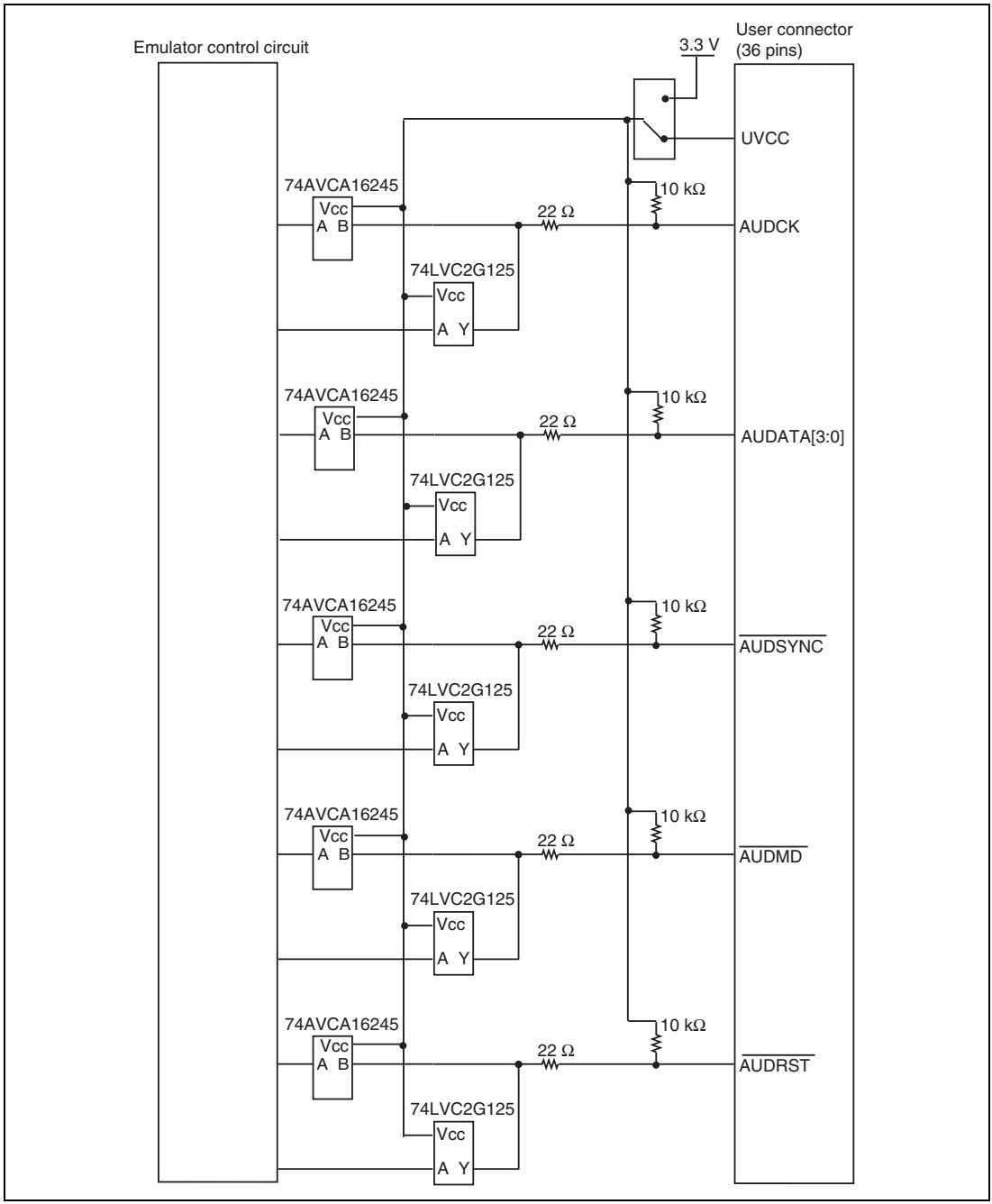


Figure 3.2 Interface Circuits in the Emulator (36-pin Interface for AUD)

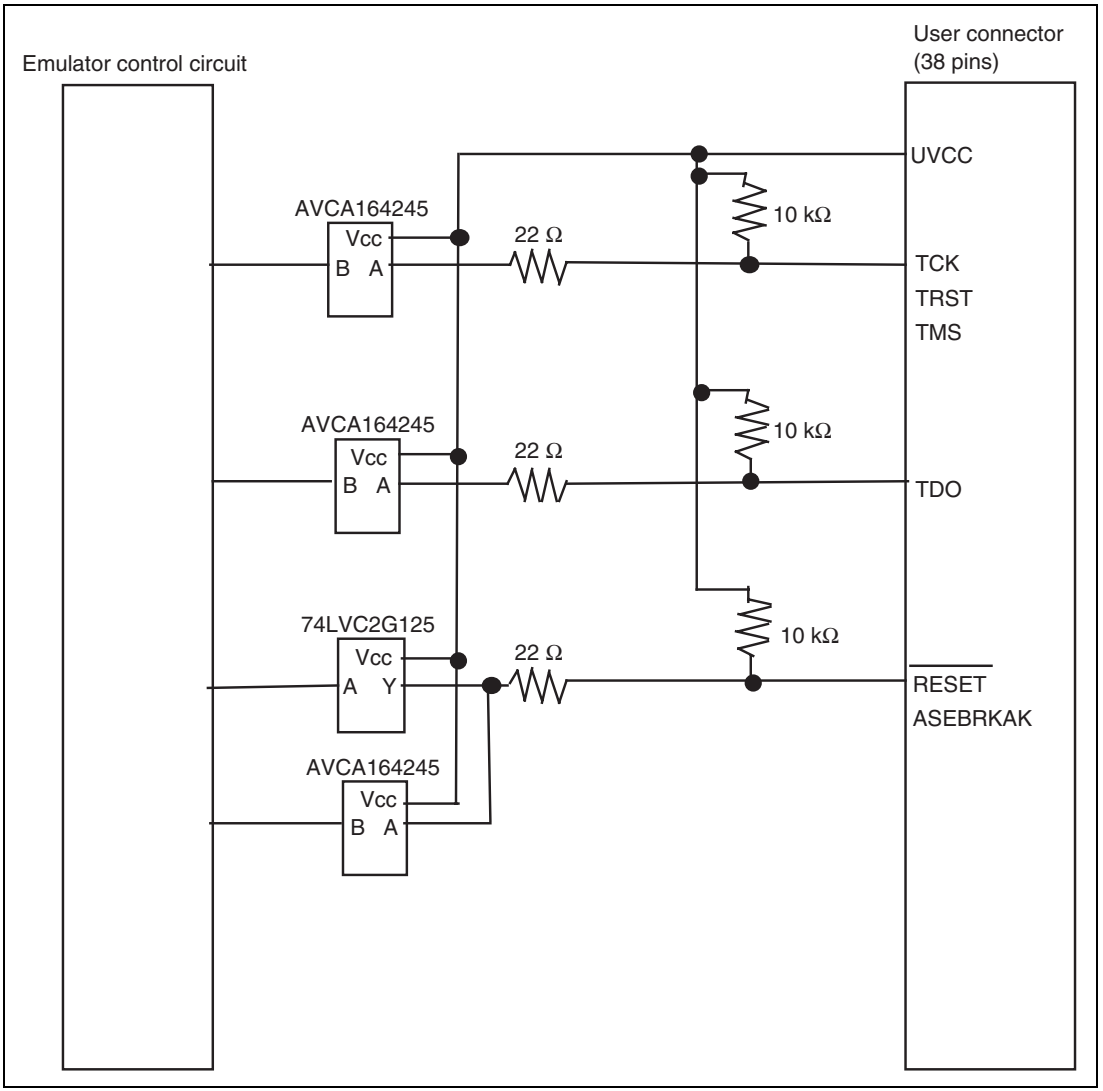


Figure 3.3 Interface Circuits in the Emulator (38-pin Interface for H-UDI)

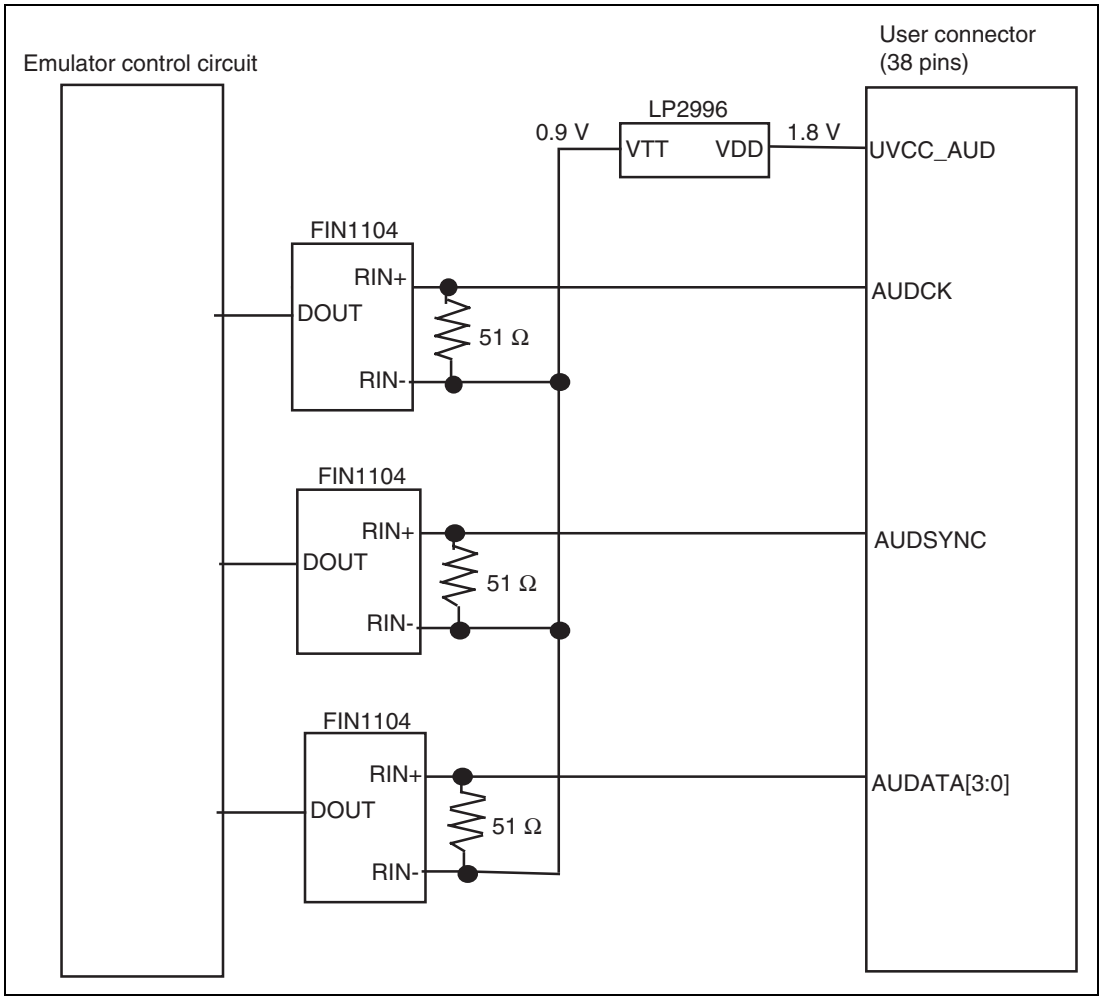


Figure 3.4 Interface Circuits in the Emulator (38-pin Interface for AUD)

3.3 Reducing EMI Noise

To prevent EMI noise, enclose the trace unit in a case at usage, as shown in figure 3.5.

The recommended material of the case is iron plated with nickel or resin internally plated with nickel.

The case must be large enough to include the trace unit, the probe head, and the user system.

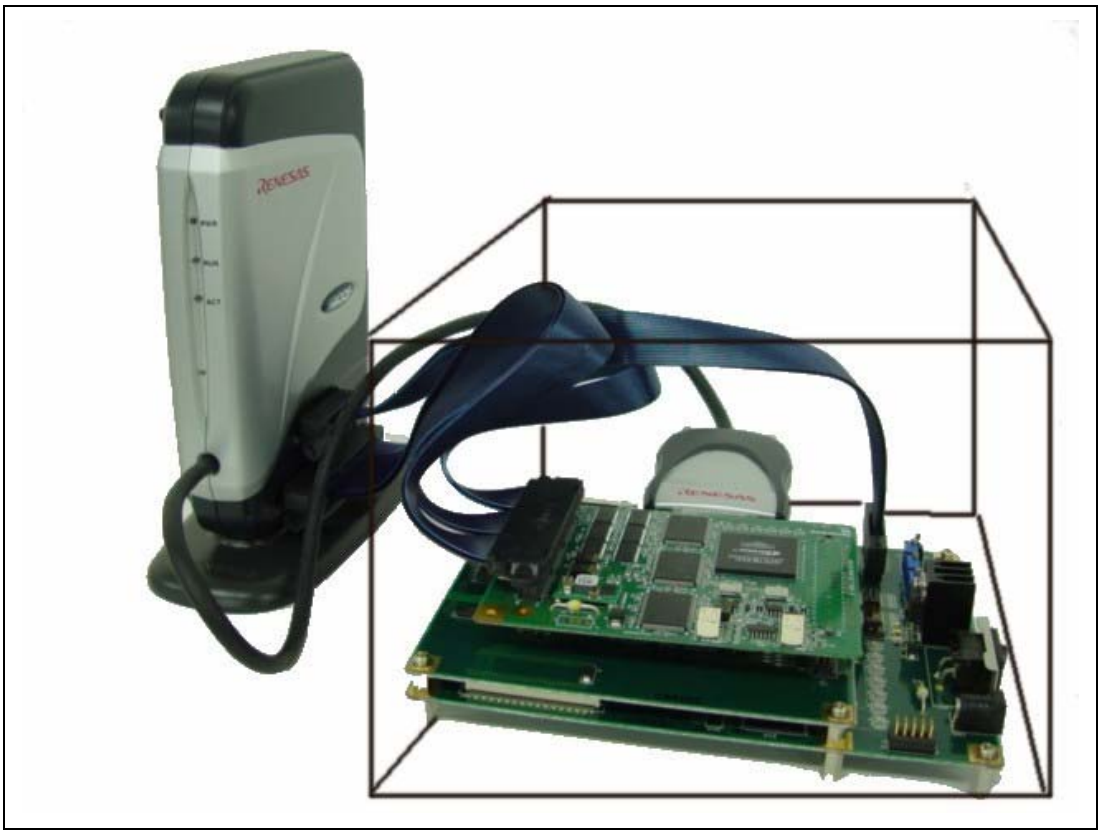


Figure 3.5 Countermeasure for EMI Noise

Note: EMI stands for Electrical Magnetic Interference.

3.4 Diagnostic Procedure

This section describes how to set up and execute the diagnostic program and output the result.

3.4.1 Installing the Diagnostic Program

(1) Opening \SOT\R0E0200FxEMU00\Setup.exe

- Execute Setup.exe from the \SOT\R0E0200FxEMU00 directory of the CD-R. Open \SOT\R0E0200F0EMU00\Setup.exe and \SOT\R0E0200F2EMU00\Setup.exe for R0E0200F0EMU00 and R0E0200F2EMU00, respectively.
- Follow the cues given by the installation wizard to install the software.

3.4.2 Executing the Diagnostic Program

The following describes how to execute the diagnostic program.

1. Connect the emulator to the host machine; not to the user system.
2. Turn on the emulator.
3. Select [E200F TM] -> [E200F Fx TM] (x: 0 or 2) from [Programs] in the [Start] menu.
4. The diagnostic program of the emulator is started and the initial screen (figure 3.6) is displayed.
5. Select [COMPONENT]. Do not select [TARGET].
6. Click the [USER TEST MODE] button. Do not select [QA TEST MODE].

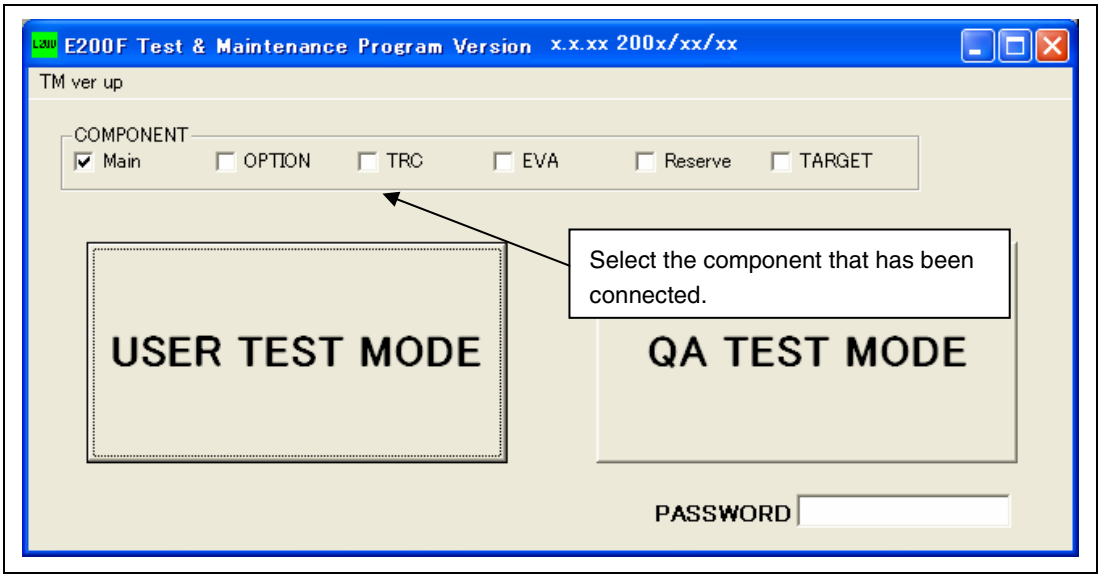


Figure 3.6 Initial Screen of the E200F System Operation Test

7. When the emulator enters USER TEST MODE, the screen shown in figure 3.7 is displayed.
8. When clicking [UNIT ONLY], items that can be tested are selected.
9. Click the [START] button.
10. When loading FPGA is completed, the test will begin.

Note: When [COMPONENT] is selected, do not select [TARGET] ([TARGET] is used for jigs for the shipment test).

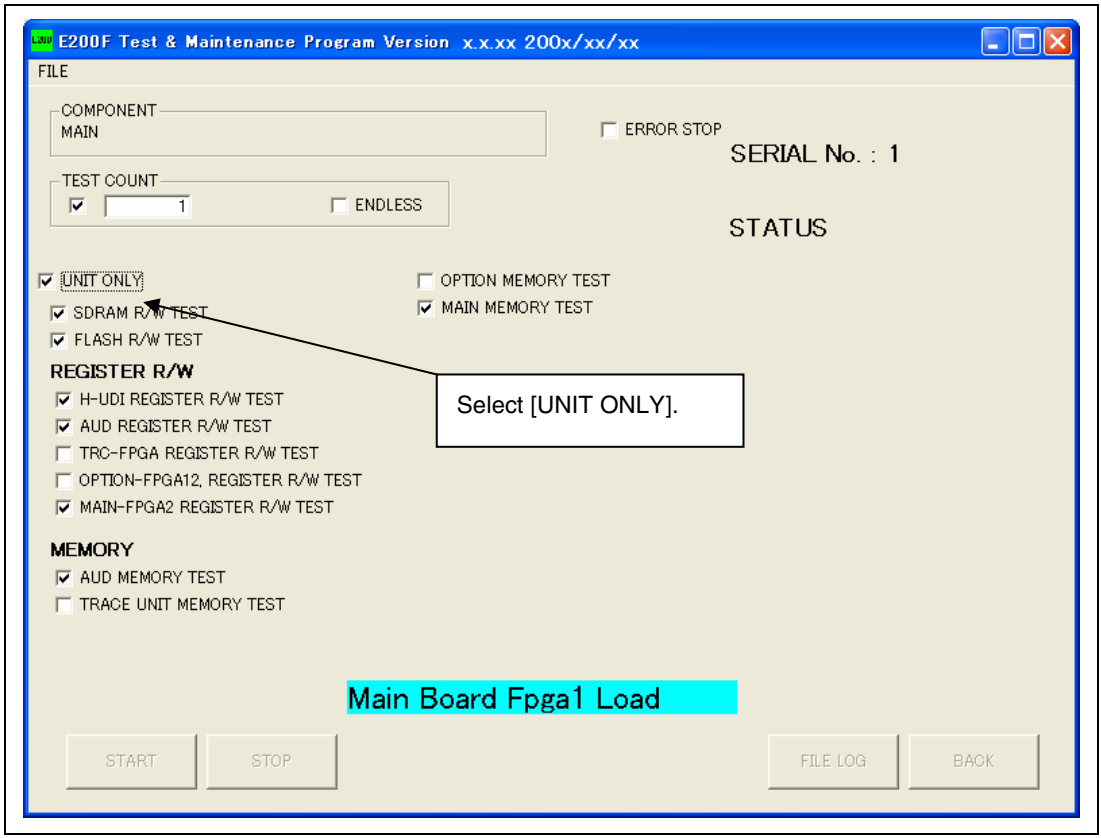


Figure 3.7 Screen for Selecting Test

Note: Do not disconnect the USB cable during the test.

11. When the test is executed, ‘Testing’ is shown at the left of [STATUS].
12. When the test is successfully completed, ‘Test OK’ is displayed.

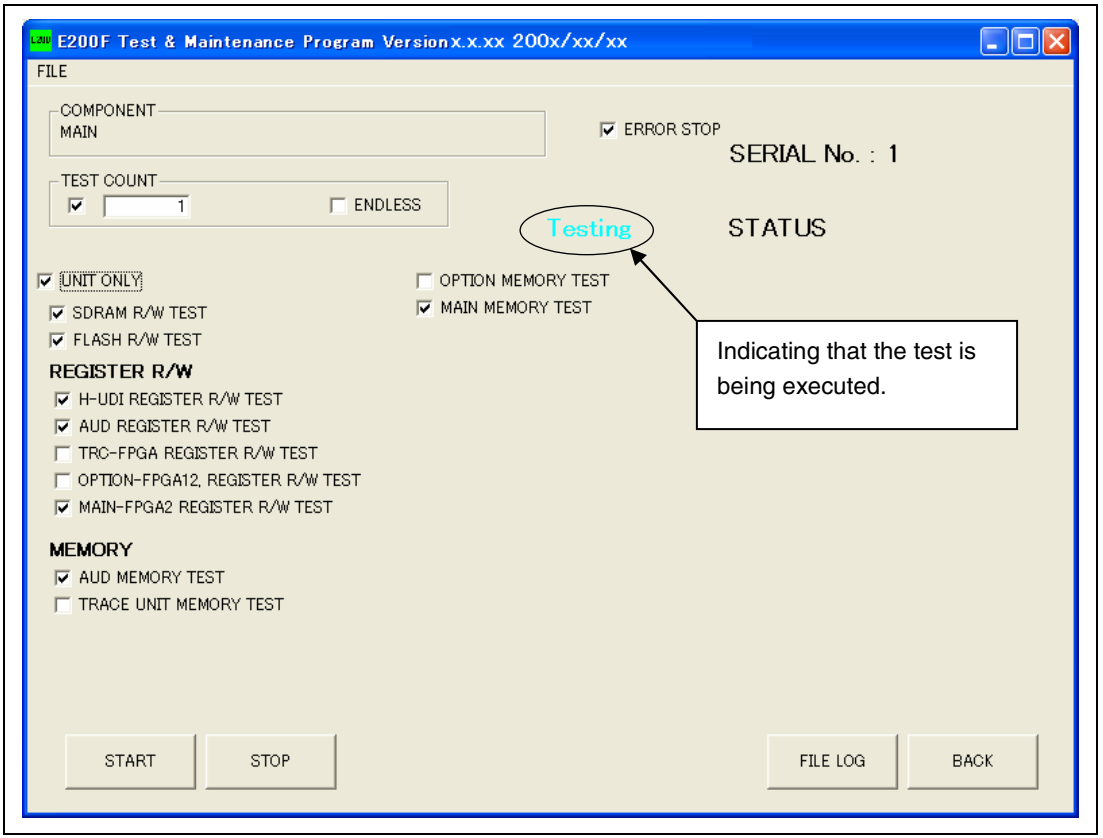


Figure 3.8 Screen Showing ‘Testing’

3.4.3 Creating a Log File

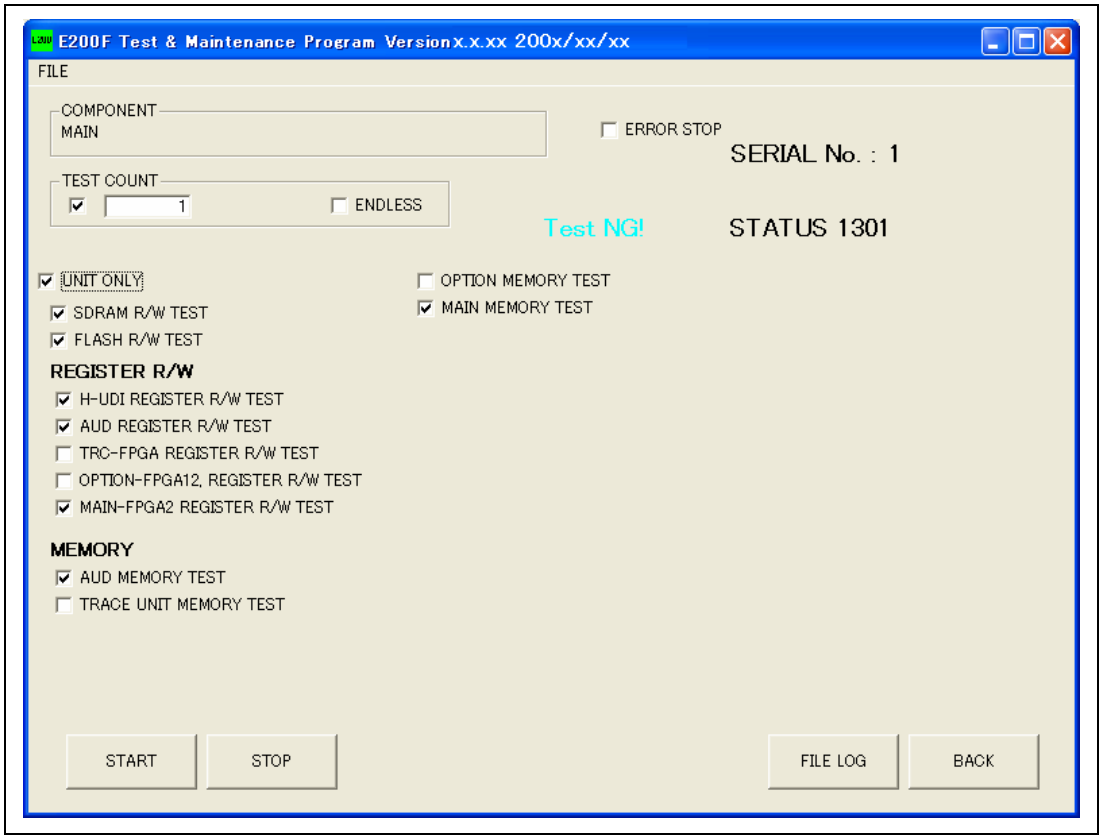


Figure 3.9 Screen Showing 'Test NG'

If the test is failed, 'Test NG!' is shown (figure 3.9). In this case, the emulator may be malfunctioned. Create a log file according to the following procedures and send it to the sales office.

- How to create a log file

1. Click the [FILE LOG] button.
2. The E200FTM.LOG file is created under the \SOT\R0E0200FxEMU00 directory.
3. Send the log file to the sales office.

Example of the E200F TM error log

```
*** E200F Emulator T/M ERROR LOG ***
    TM Version x.x.xx
    Serial Number 000x
    Date 200x/xx/xx
    No.    STATUS    NG Address  NG Data
    02     1301      A0000000    7EC0A5F      NG
    03     0304      A8000108    FFFFFFFF     NG
```

Section 4 Preparations for Debugging

4.1 System Check

When the software is executed, use the procedure below to check that the emulator is connected correctly. Here, use the workspace for a tutorial provided on the product.

1. Connect the emulator to the host machine.
2. Connect the user system interface cable to the connector of the emulator.
3. Connect the user system interface cable to the connector in the user system.
4. Turn on the user system.
5. Turn on the emulator.
6. Select [Renesas] -> [High-performance Embedded Workshop] -> [High-performance Embedded Workshop] from [Programs] in the [Start] menu.

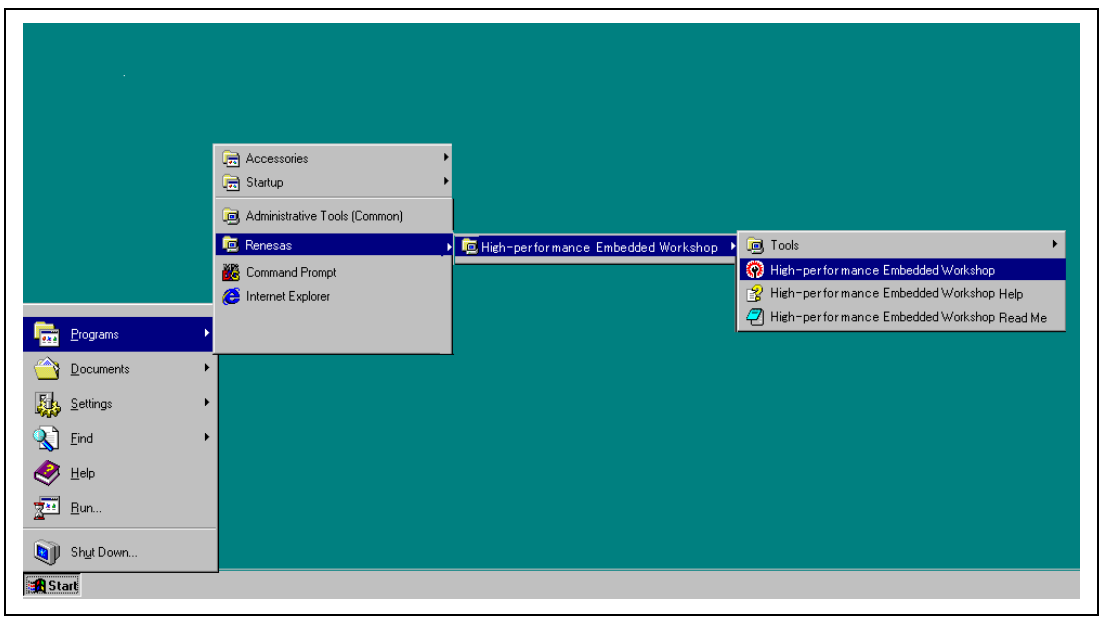


Figure 4.1 [Start] Menu

Note: The [High-performance Embedded Workshop] -> [Tools] is not displayed depending on the user's environment.

7. The [Welcome!] dialog box is displayed.

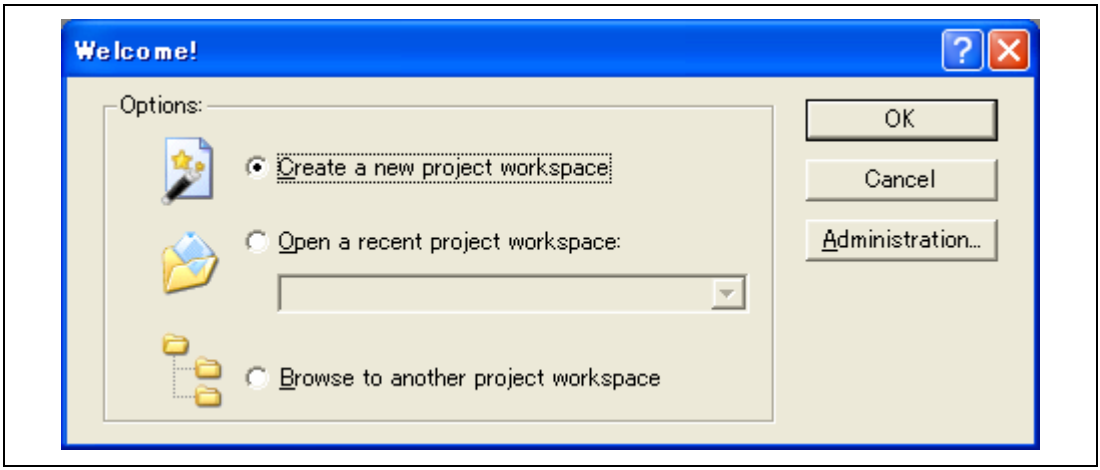


Figure 4.2 [Welcome!] Dialog Box

[Create a new project workspace] radio button:	Creates a new workspace.
[Open a recent project workspace] radio button:	Uses an existing workspace and displays the history of the opened workspace.
[Browse to another project workspace] radio button:	Uses an existing workspace; this radio button is used when the history of the opened workspace does not remain

To use a workspace for the tutorial, select the [Browse to another project workspace] radio button and click the [OK] button.

When the [Open workspace] dialog box is opened, specify the following directory:
<Drive where the OS has been installed>: \Workspace\Tutorial\E200F\xxxx\Tutorial

Here, ‘xxxx’ means the name of the target MPU.

After the directory has been specified, select the following file and click the [Open] button.

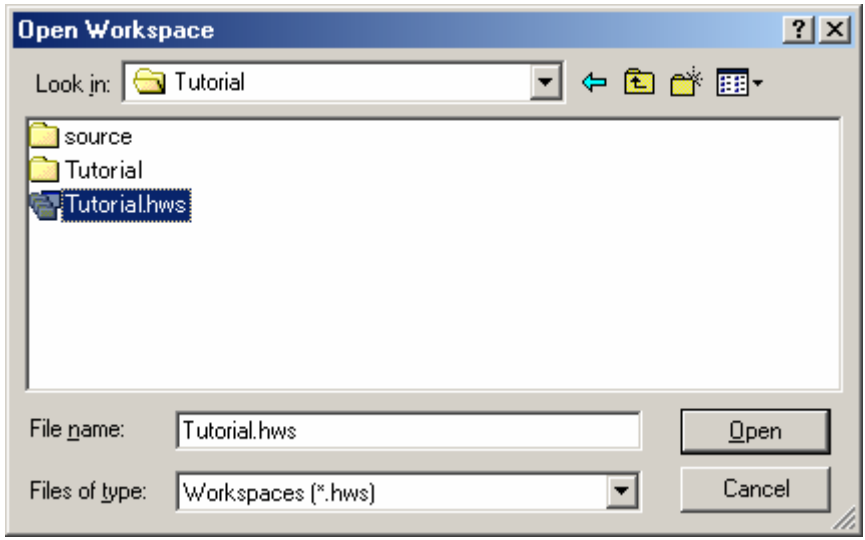


Figure 4.3 [Open Workspace] Dialog Box

8. The [CPU Select] dialog box is displayed.

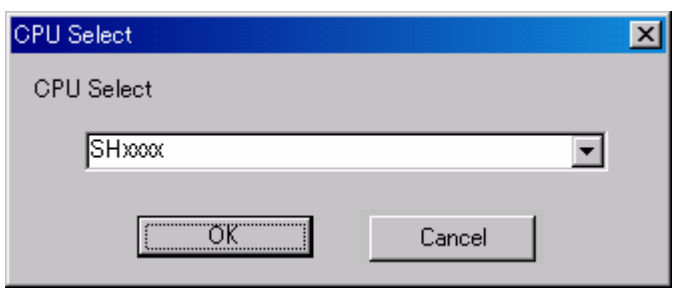


Figure 4.4 [CPU Select] Dialog Box

Select the CPU from the drop-down list and click the [OK] button.

If the following dialog box is displayed, the emulator will not support the CPU in use. Check the type number of the main unit referring to section 1.1, Components of the Emulator, in the additional document, Supplementary Information on Using the SHxxxx.

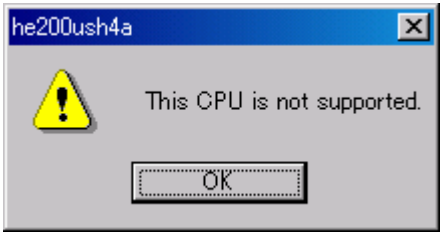


Figure 4.5 [This CPU is not supported] Dialog Box

9. The [Function select] dialog box is displayed.

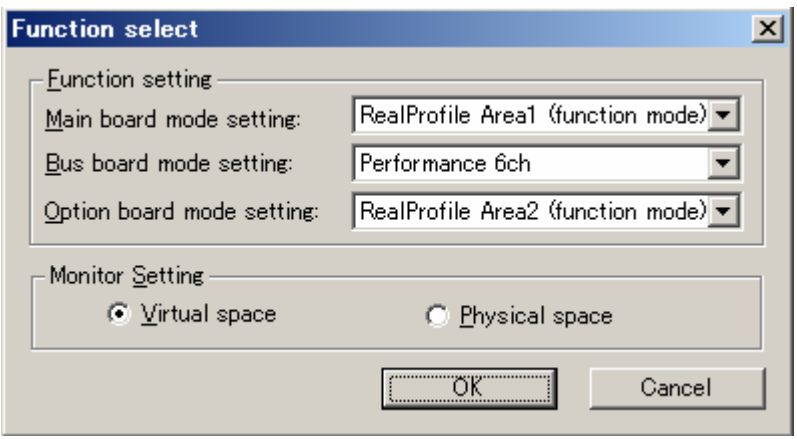


Figure 4.6 [Function select] Dialog Box

Select the emulator function to be used. For the items to be selected, refer to section 2.7, Changing the Settings.

10. The [Connecting] dialog box is displayed and the emulator connection is started.

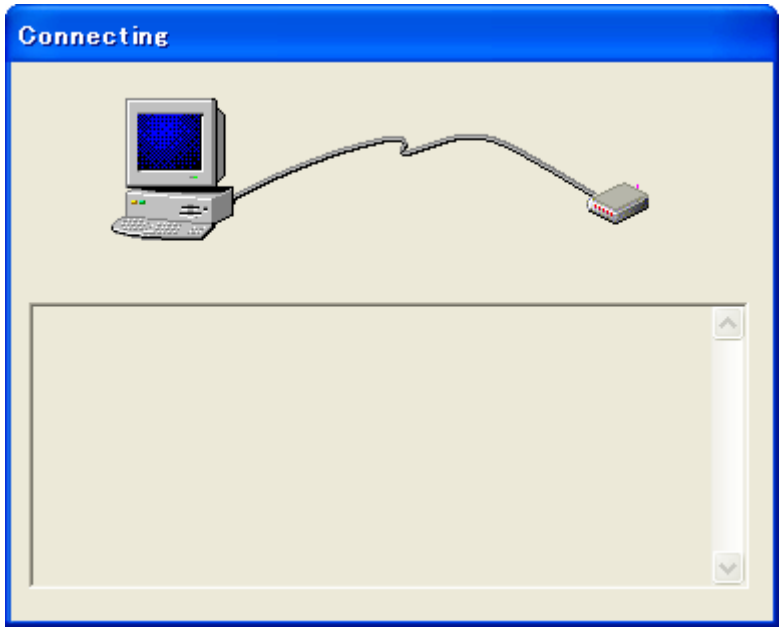


Figure 4.7 [Connecting] Dialog Box

11. The dialog box is displayed as shown in figure 4.8.

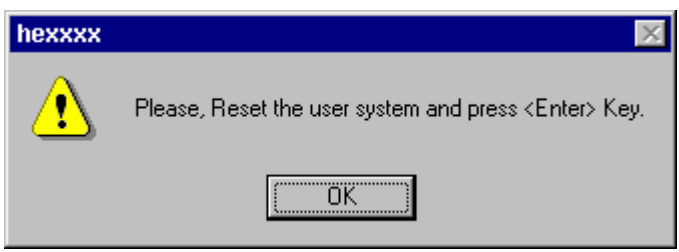


Figure 4.8 Dialog Box of the RESET Signal Input Request Message

12. Input the reset signal from the user system, and click the [OK] button.

13. If no reset signal is detected, the following dialog box is displayed.

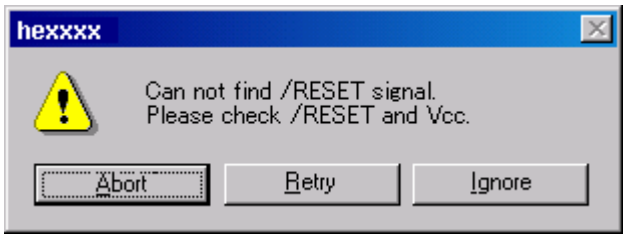


Figure 4.9 [Can not find /RESET signal] Dialog Box

When the [Ignore] button is clicked, the emulator issues a reset in the CPU for initiation. However, this method is unavailable for some products. For details, refer to section 2.2, Specific Functions for the Emulator when Using the SHxxxx, in the additional document, Supplementary Information on Using the SHxxxx.

14. When "Connected" is displayed in the [Output] window of the High-performance Embedded Workshop, the emulator initiation is completed.

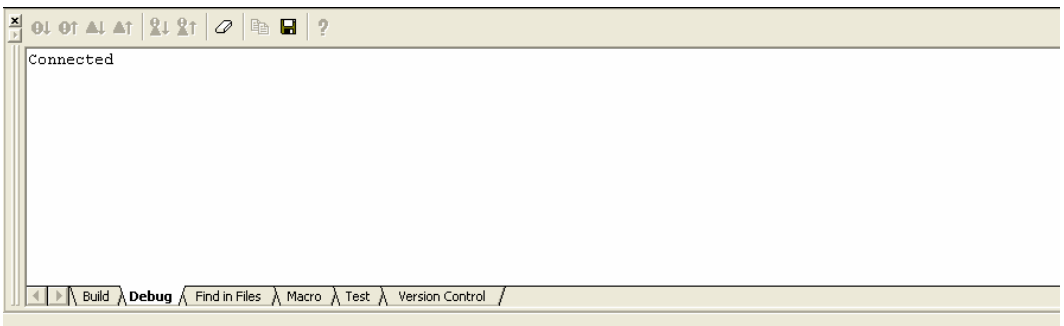


Figure 4.10 [Output] Window

Notes: 1. If the emulator is not initiated, the following dialog boxes shown in figures 4.11 through 4.15 will be displayed.

(a) If the following dialog box is displayed and the method 12 above is unavailable, the power of the user system may not be input or the RESET signal may not be input to the device. Check the input circuits for the power of the user system and the reset pin.

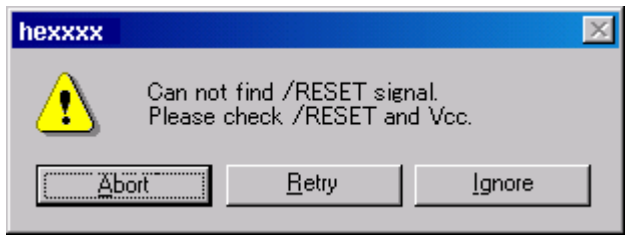


Figure 4.11 [Can not find /RESET signal] Dialog Box

(b) If the following dialog box is displayed, the user system may be turned off or the H-UDI port connector may not be correctly connected. Check that the user system is turned on and the H-UDI port connector is connected.

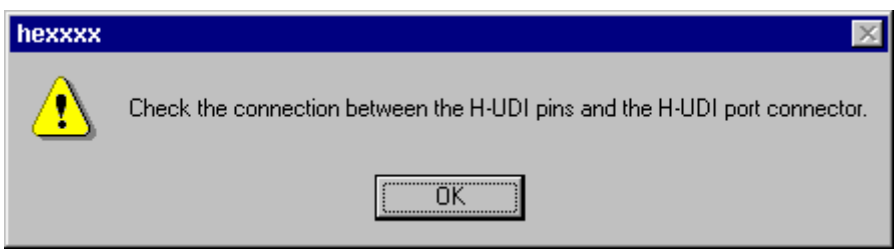


Figure 4.12 [Check the connection] Dialog Box

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

Figure 4.13 [COMMUNICATION TIMEOUT ERROR] Dialog Box

Figure 4.14 [INVALID ASERAM FIRMWARE!] Dialog Box

Figure 4.15 [Error JTAG boot] Dialog Box

2. If the emulator is not activated due to other reasons, a message box corresponding to the status is displayed. Use the message as a reference to check the wiring on the board.

4.2 Method for Activating High-performance Embedded Workshop

To activate the High-performance Embedded Workshop, follow the procedure listed below.

1. Connect the emulator to the host machine and the user system, then turn on the user system.
2. Select [High-performance Embedded Workshop] from [Renesas High-performance Embedded Workshop] of [Programs] in the [Start] menu.
3. The [Welcome!] dialog box is displayed.

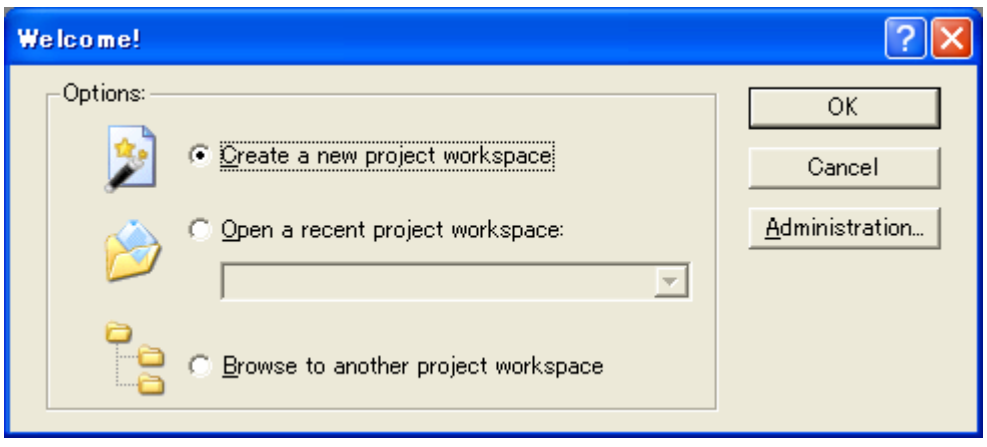


Figure 4.16 [Welcome!] Dialog Box

[Create a new project workspace] radio button:	Creates a new workspace.
[Open a recent project workspace] radio button:	Uses an existing workspace and displays the history of the opened workspace.
[Browse to another project workspace] radio button:	Uses an existing workspace; this radio button is used when the history of the opened workspace does not remain.

In this section, we describe the following three ways to start up the High-performance Embedded Workshop:

- [Create a new project workspace] - a toolchain is not in use
- [Create a new project workspace] - a toolchain is in use
- [Browse to another project workspace]

The following describes how to activate the High-performance Embedded Workshop when selecting [Create a new project workspace] without any toolchain, [Create a new project workspace] with a toolchain, and [Browse to another project workspace]. The [Open a recent project workspace] radio button is used to omit the operation for specifying the workspace file when [Browse to another project workspace] is selected.

4.2.1 Creating the New Workspace (Toolchain Not Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Create a new project workspace] radio button and click the [OK] button.

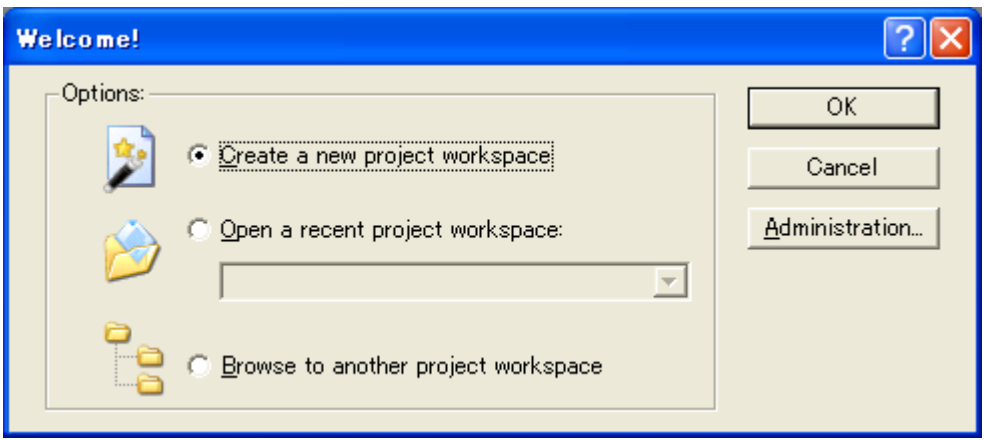


Figure 4.17 [Welcome!] Dialog Box

2. The Project Generator is started. In this section, we omit description of the settings for the toolchain.
   If you have not purchased the toolchain, the following dialog box is displayed.

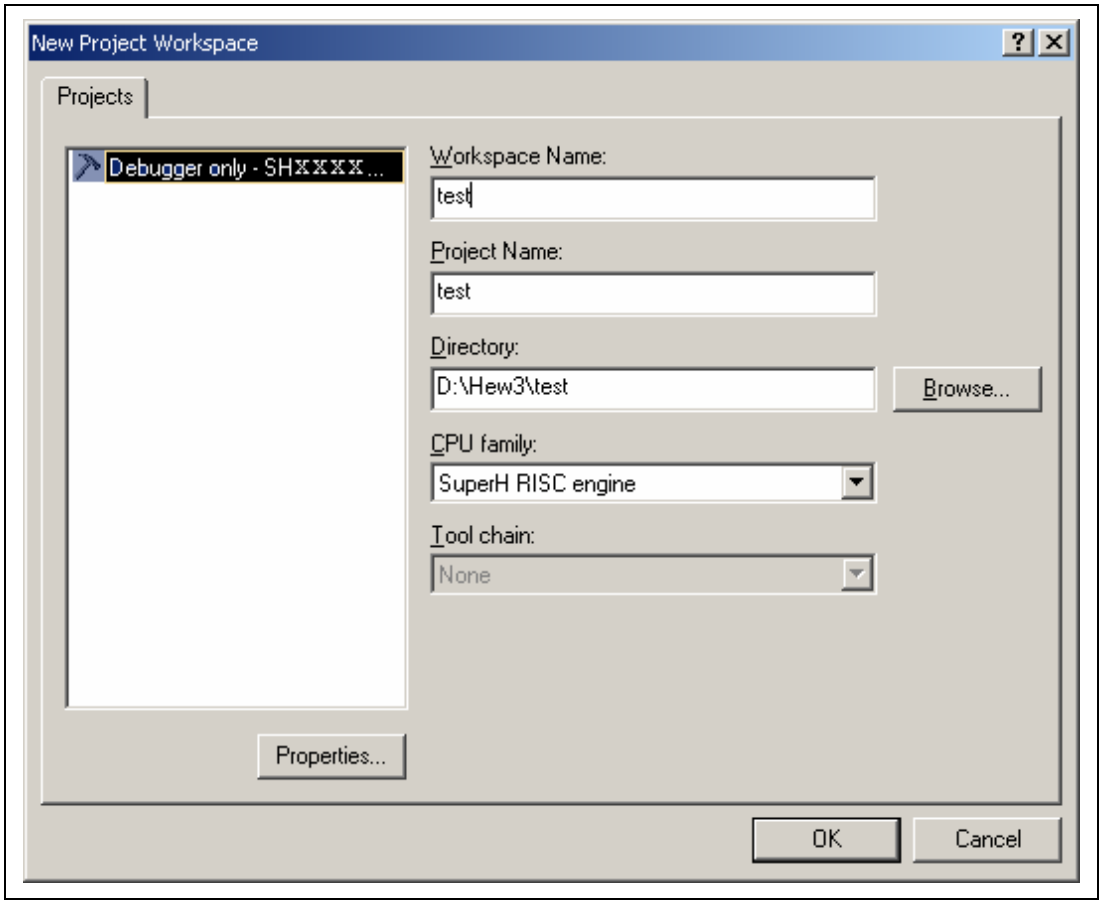


Figure 4.18 [New Project Workspace] Dialog Box

[Workspace Name] edit box: Enter the new workspace name. Here, for example, enter 'test'.

[Project Name] edit box: Enter the project name. When the project name is the same as the workspace name, it needs not be entered.

Other list boxes are used for setting the toolchain; the fixed information is displayed when the toolchain has not been installed.

3. The following dialog box is displayed.

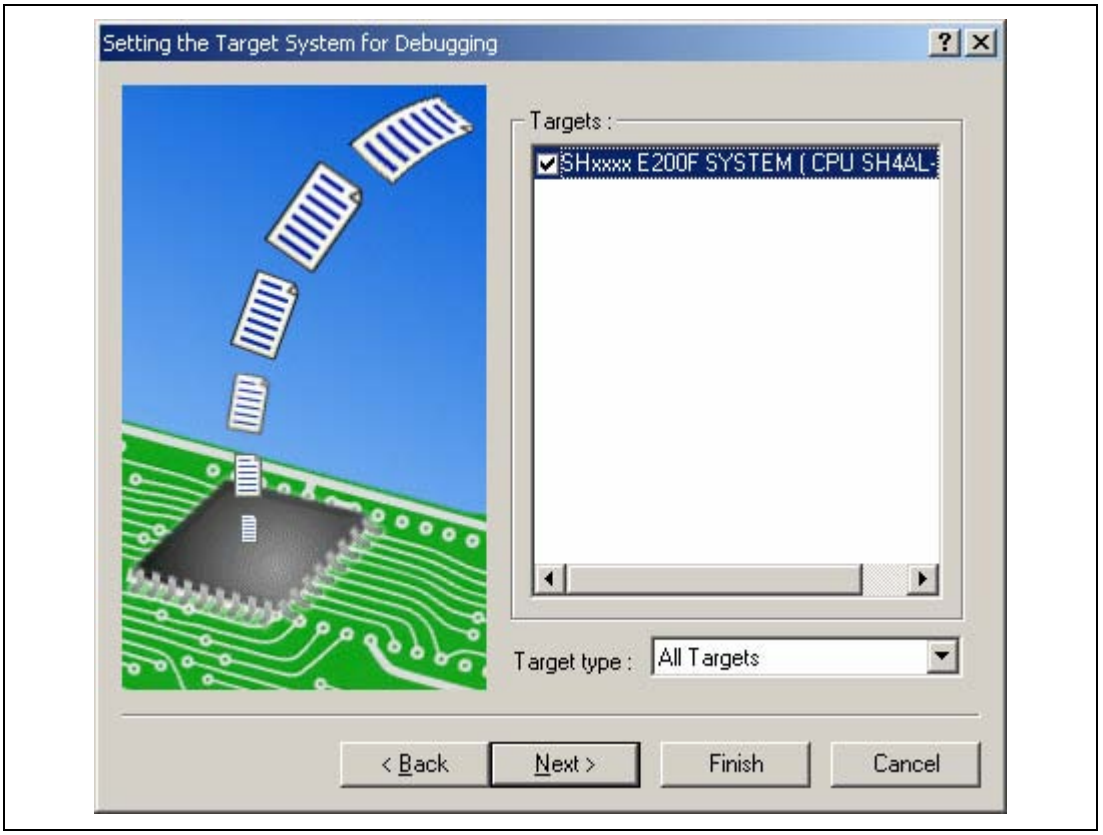


Figure 4.19 [Setting the Target System for Debugging] Dialog Box

Check the target emulator and click the [Next] button.

4. Set the configuration file name. The configuration file saves the state of High-performance Embedded Workshop except for the emulator.

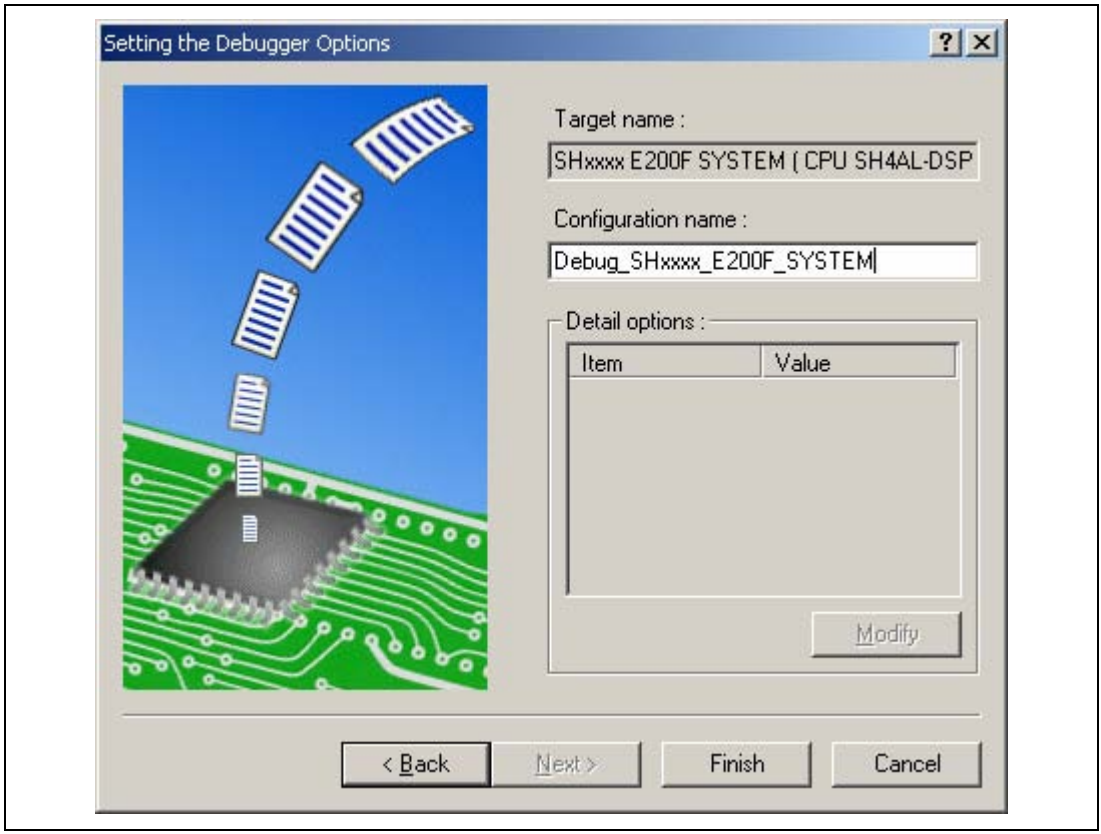


Figure 4.20 [Setting the Debugger Options] Dialog Box

This is the end of the emulator setting.

Click the [Finish] button to exit the Project Generator. The High-performance Embedded Workshop is activated.

5. After the High-performance Embedded Workshop has been activated, the emulator is automatically connected. For operation during connection, refer to section 4.1, System Check.

4.2.2 Creating the New Workspace (Toolchain Used)

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Create a new project workspace] radio button and click the [OK] button.

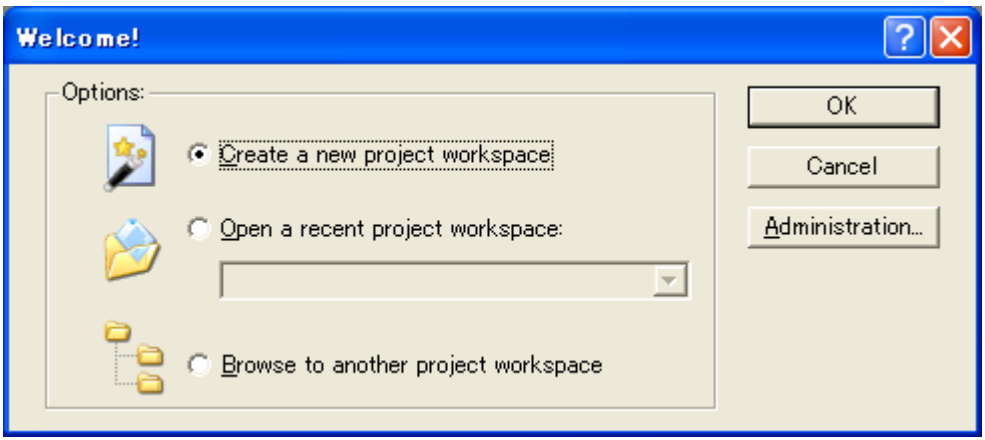


Figure 4.21 [Welcome!] Dialog Box

2. The Project Generator is started.
   If you have purchased the toolchain, the following dialog box is displayed.

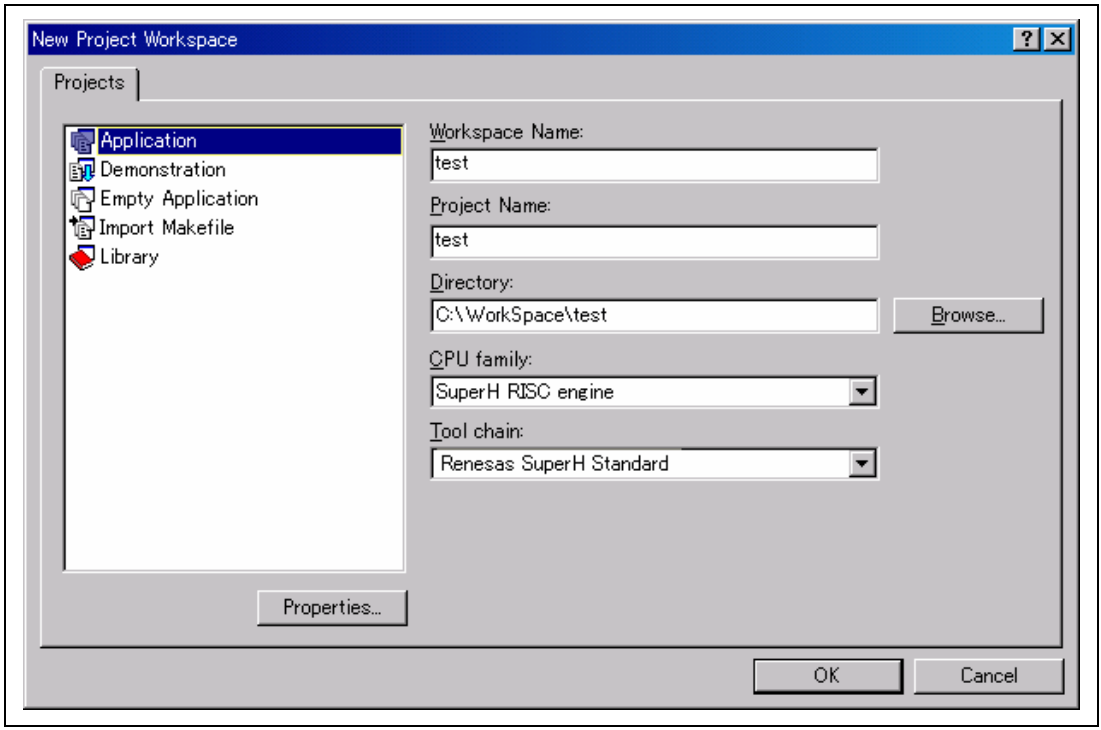


Figure 4.22 [New Project Workspace] Dialog Box

[Workspace Name] edit box:	Enter the new workspace name. Here, for example, enter 'test'.
[Project Name] edit box:	Enter the project name. When the project name is the same as the workspace name, it needs not be entered.
[CPU family] drop-down list box:	Select the target CPU family.
[Tool chain] drop-down list box:	Select the target toolchain name when using the toolchain. Otherwise, select [None].
[Project type] list box:	Select the project type to be used.

Note: When [Demonstration] is selected in the emulator, note the following:
The [Demonstration] is a program for the simulator. When using a program to be generated, delete the Printf statement.

3. Make the required setting for the toolchain. When the setting has been completed, the following dialog box is displayed.

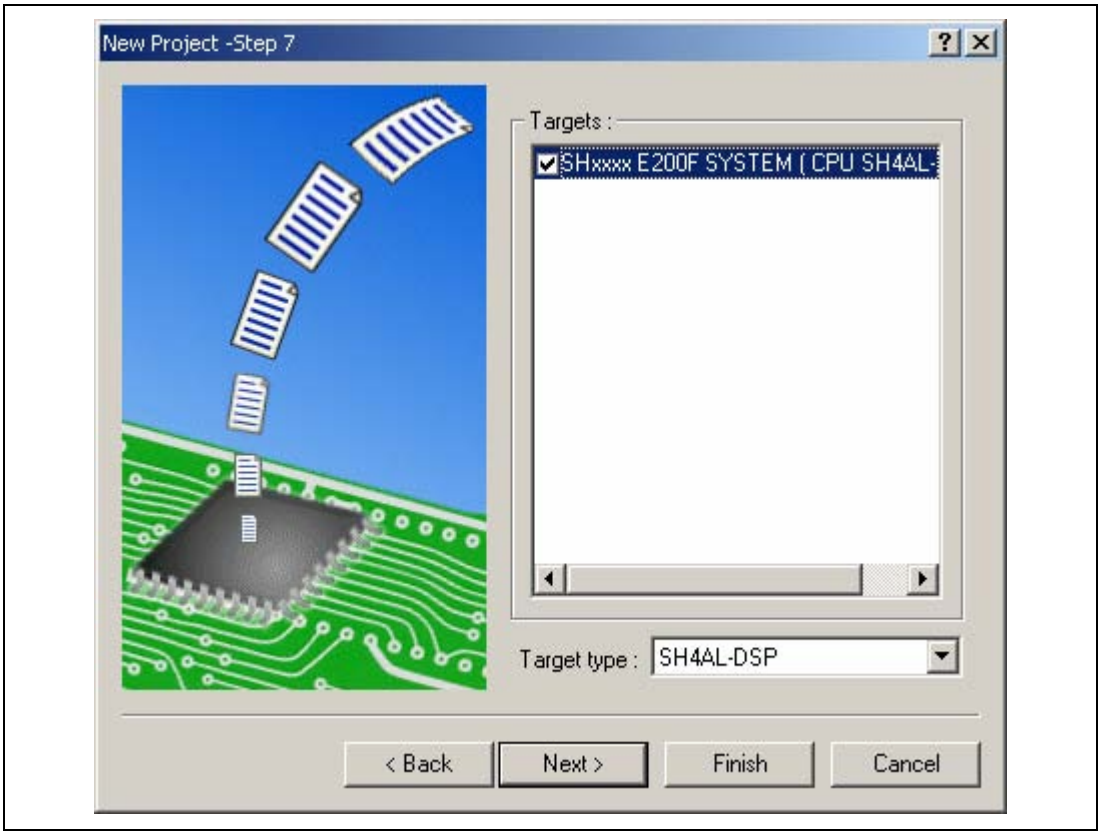


Figure 4.23 [New Project – Step 7] Dialog Box

Check the target emulator and click the [Next] button. Mark other products as required.

4. Set the configuration file name. The configuration file saves the state of High-performance Embedded Workshop except for the emulator.

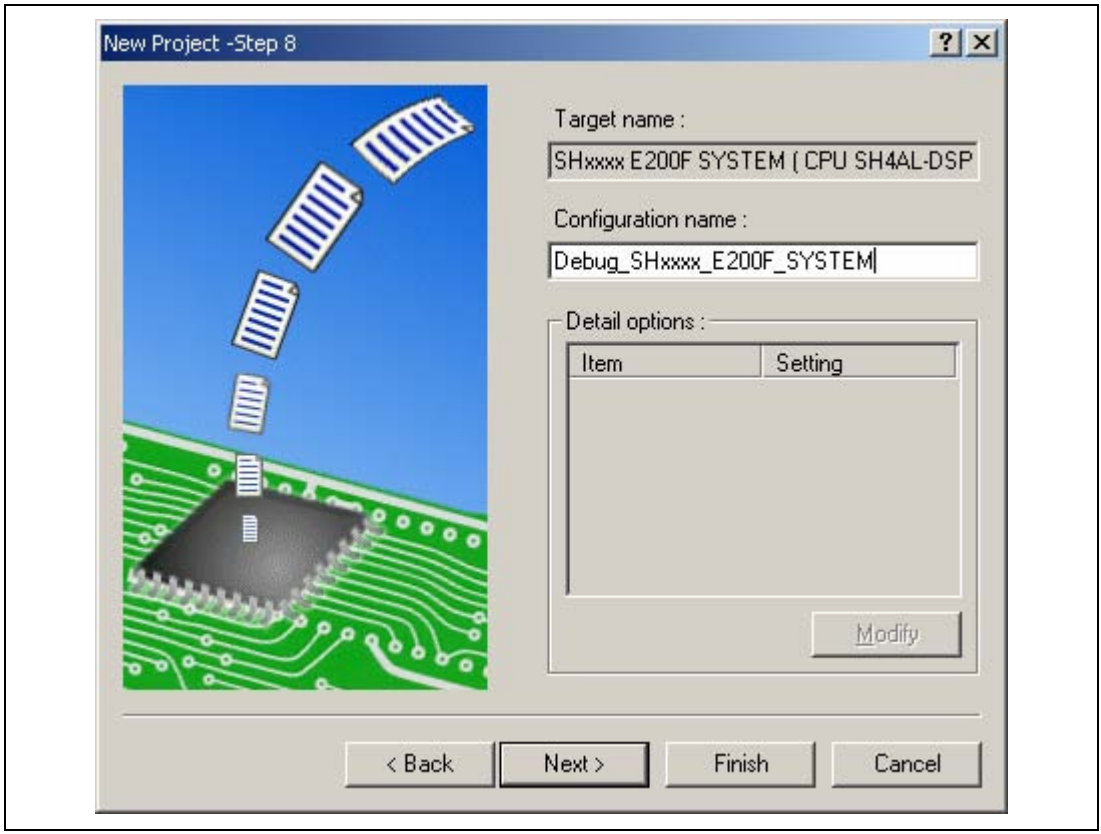


Figure 4.24 [New Project – Step 8] Dialog Box

This is the end of the emulator setting.

Exit the Project Generator according to the instructions on the screen. The High-performance Embedded Workshop is activated.

5. After the High-performance Embedded Workshop has been activated, connect the emulator. However, it is not needed to connect the emulator immediately after the High-performance Embedded Workshop has been activated.
   To connect the emulator, use one of the methods (a) and (b) below. For operation during connection, refer to section 4.1, System Check.

(a) Connecting the emulator after the setting at emulator activation

Select [Debug settings] from the [Options] menu to open the [Debug Settings] dialog box. It is possible to register the download module or the command chain that is automatically executed at activation. For details on the [Debug Settings] dialog box, refer to section 4.3, Setting at Emulator Activation.

After the [Debug Settings] dialog box has been set, when the dialog box is closed, the emulator is connected.

(b) Connecting the emulator without the setting at emulator activation

The emulator can be easily connected by switching the session file that the setting for the emulator use has been registered.

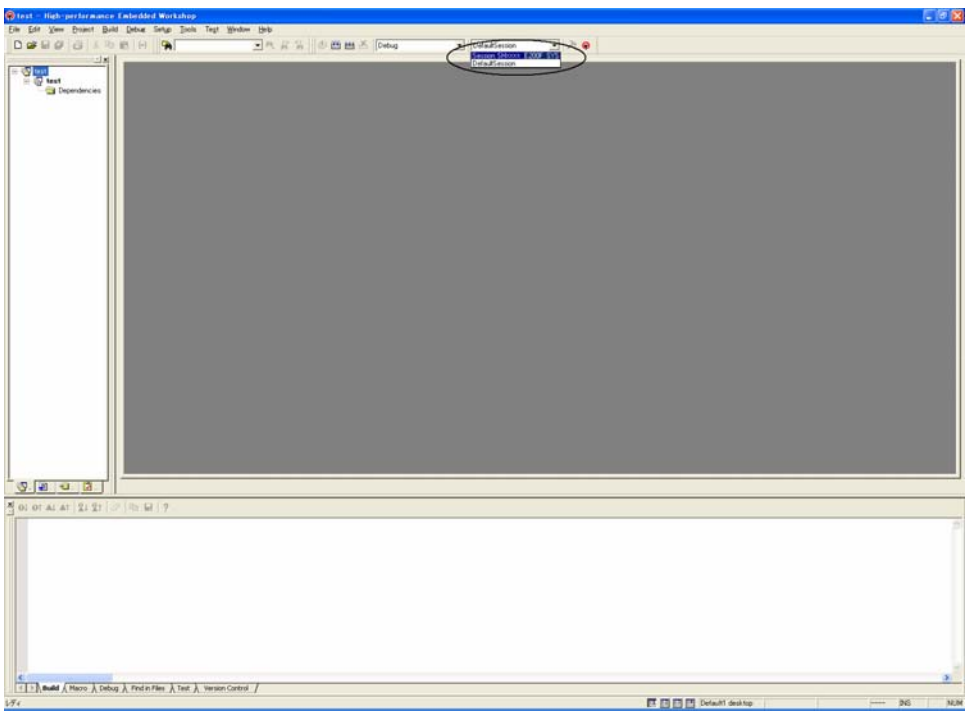


Figure 4.25 Selecting the Session File

In the list box that is circled in figure 4.25, select the session file name including the character string that has been set in the [Target name] text box in figure 4.24, [New Project – Step 8] dialog box. The setting for using the emulator has been registered in this session file.

After selected, the emulator is automatically connected.

4.2.3 Selecting an Existing Workspace

1. In the [Welcome!] dialog box that is displayed when the High-performance Embedded Workshop is activated, select [Browse to another project workspace] radio button and click the [OK] button.

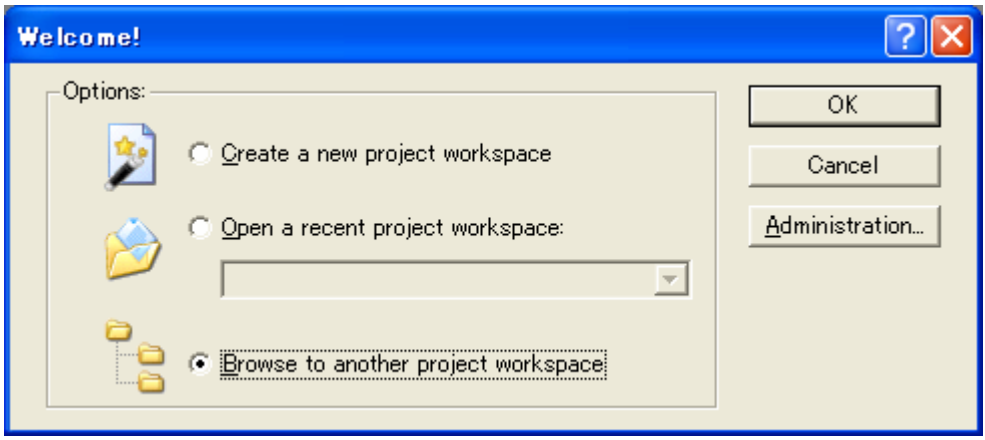


Figure 4.26 [Welcome!] Dialog Box

2. The [Open Workspace] dialog box is displayed. Select a directory in which you have created a workspace.
   After that, select the workspace file (.hws) and press the [Open] button.

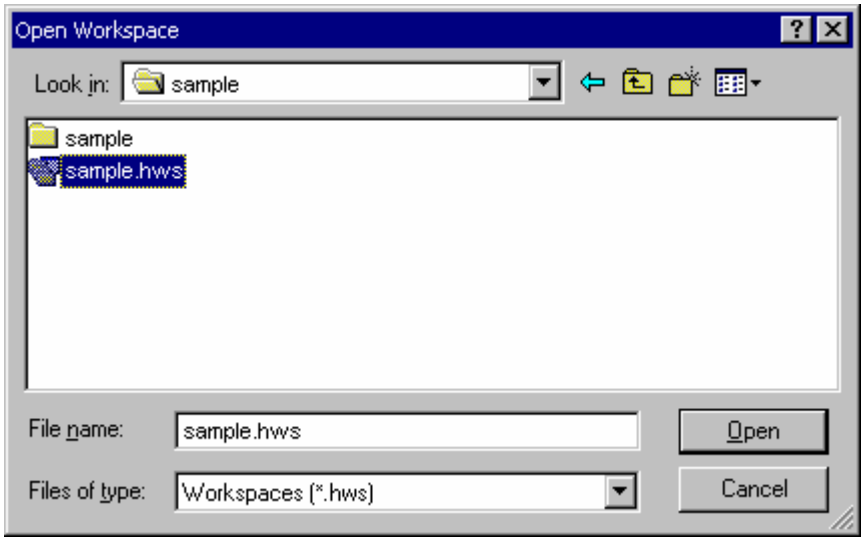


Figure 4.27 [Open Workspace] Dialog Box

3. This activates the High-performance Embedded Workshop and recovers the state of the selected workspace at the time it was saved.
   When the saved state information of the selected workspace includes connection to the emulator, the emulator will automatically be connected. To connect the emulator when the saved state information does not include connection to the emulator, refer to section 4.5, Connecting the Emulator.

4.3 Setting at Emulator Activation

When the emulator is activated, the command chain can be automatically executed. It is also possible to register multiple load modules to be downloaded. The registered load modules are displayed on the workspace window.

1. Select [Debug settings] from the [Options] menu to open the [Debug Settings] dialog box.

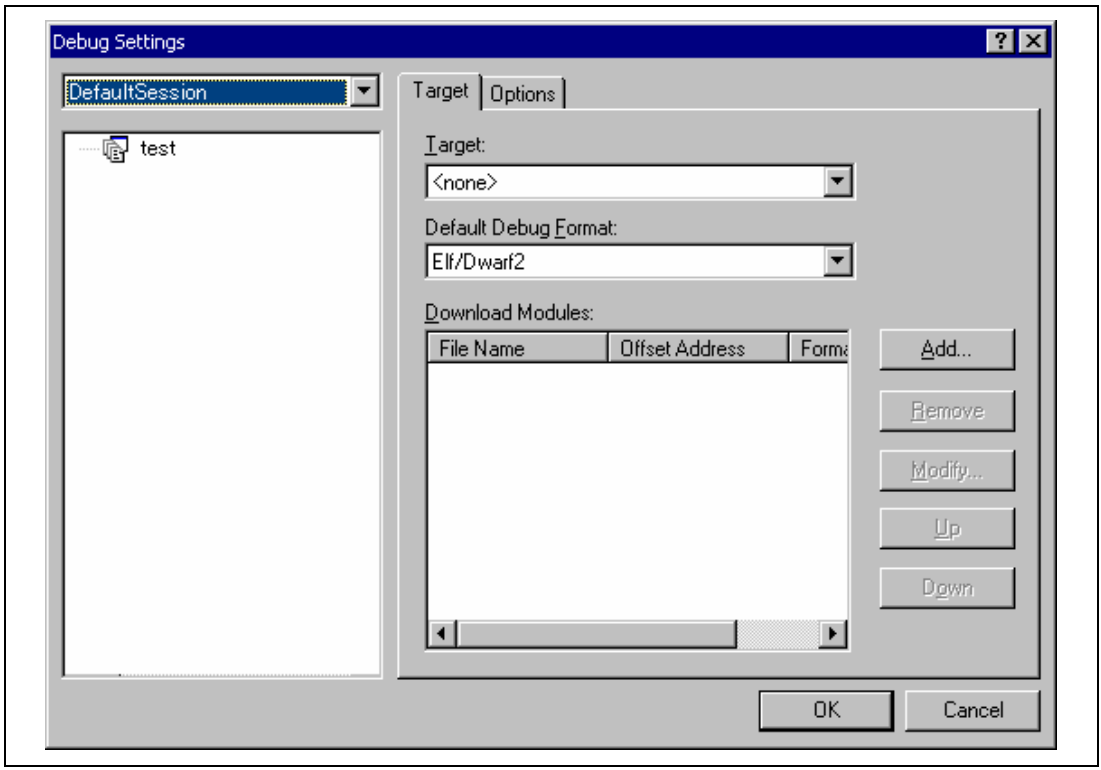


Figure 4.28 [Debug Settings] Dialog Box ([Target] Page)

2. Select the product name to be connected in the [Target] drop-down list box.
3. Select the format of the load module to be downloaded in the [Default Debug Format] drop-down list box, then register the corresponding download module in the [Download Modules] list box.

   Note: Here, no program has been downloaded. For downloading, refer to section 5.2, Downloading a Program.
4. Click the [Options] tab.

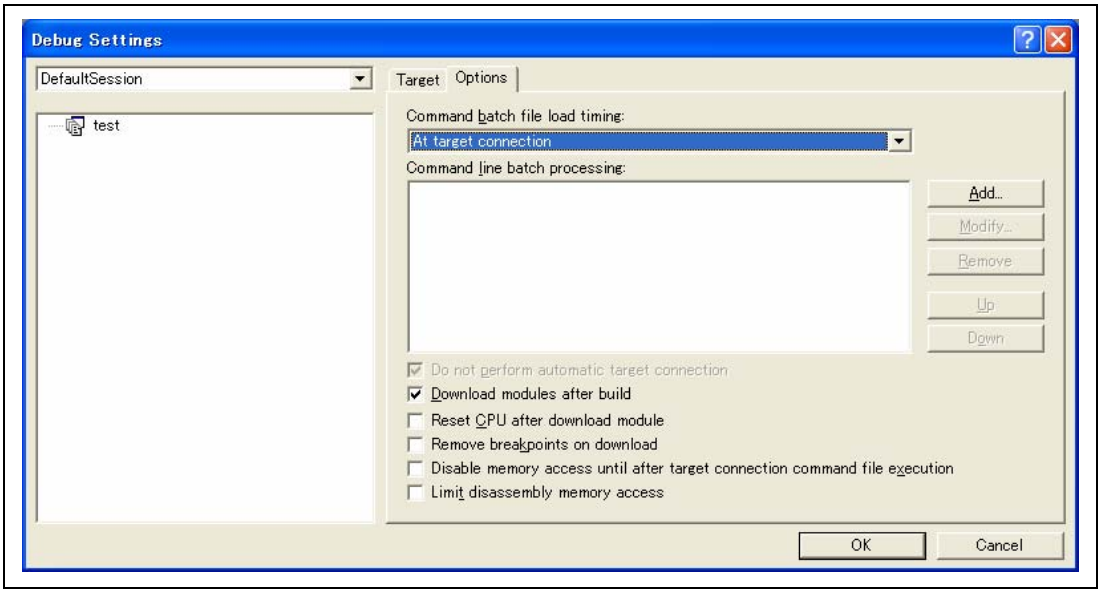


Figure 4.29 [Debug Settings] Dialog Box ([Options] Page)

The command chain that is automatically executed at the specified timing is registered. The following three timings can be specified:

- At connecting the emulator
- Immediately before downloading
- Immediately after downloading

Specify the timing for executing the command chain in the [Command batch file load timing] drop-down list box. In addition, register the command-chain file that is executed at the specified timing in the [Command Line Batch Processing] list box.

4.4 Debug Sessions

The High-performance Embedded Workshop stores all of your builder options into a configuration. In a similar way, the High-performance Embedded Workshop stores your debugger options in a session. The debugging platforms, the programs to be downloaded, and each debugging platform's options can be stored in a session.

Sessions are not directly related to a configuration. This means that multiple sessions can share the same download module and avoid unnecessary program rebuilds.

Each session's data should be stored in a separate file in the High-performance Embedded Workshop project. Debug sessions are described in detail below.

4.4.1 Selecting a Session

The current session can be selected in the following two ways:

- From the toolbar
  Select a session from the drop-down list box (figure 4.30) in the toolbar.

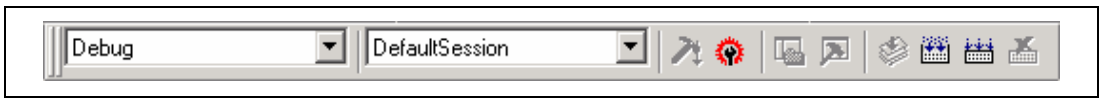


Figure 4.30 Toolbar Selection

- From the dialog box
  1. Select [Options -> Debug Sessions…]. This will open the [Debug Sessions] dialog box (figure 4.31).

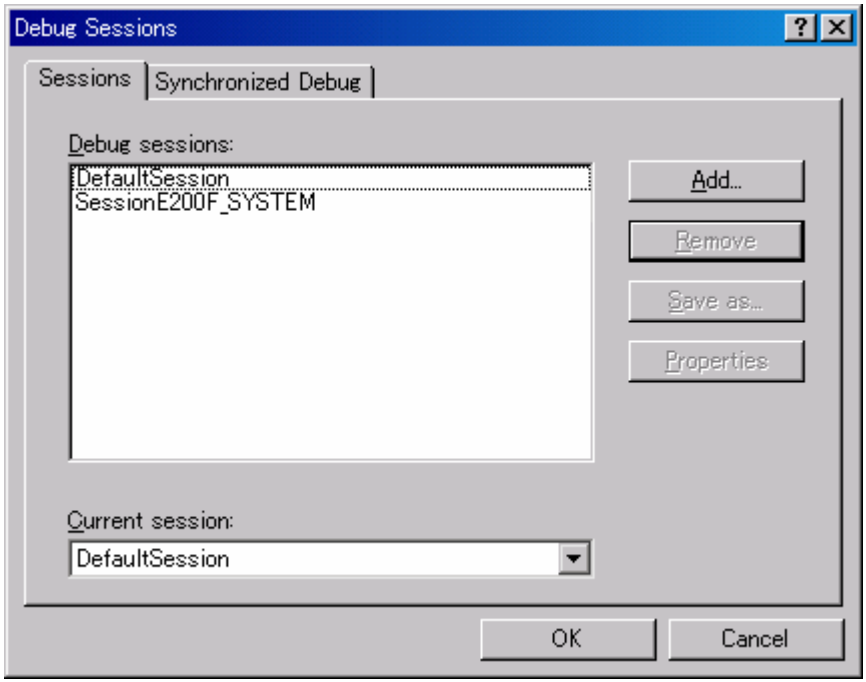


Figure 4.31 [Debug Sessions] Dialog Box

  2. Select the session you want to use from the [Current session] drop-down list.
  3. Click the [OK] button to set the session.

4.4.2 Adding and Removing Sessions

A new session can be added by copying settings from another session or removing a session.

- To add a new empty session
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).
  2. Click the [Add…] button to display the [Add new session] dialog box (figure 4.32).
  3. Check the [Add new session] radio button.
  4. Enter a name for the session.

5. Click the [OK] button to close the [Debug Sessions] dialog box.
6. This creates a file with the name entered in step 4. If a file with this name already exists, an error is displayed.

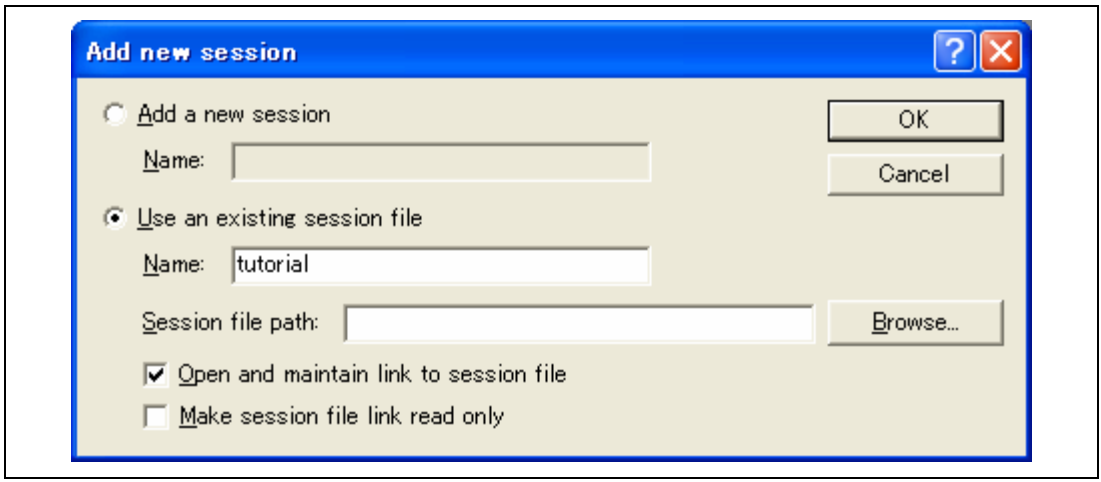


Figure 4.32 [Add new session] Dialog Box

- To import an existing session into a new session file
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).
  2. Click the [Add…] button to display the [Add new session] dialog box (figure 4.32).
  3. Check the [Use an existing session file] radio button.
  4. Enter a name for the session.
  5. Enter the name of an existing session file that you would like to import into the existing project or click the [Browse] button to select the file location.

     If the [Open and maintain link to session file] check box is not checked, the imported new session file is generated in the project directory.

     If the [Open and maintain link to session file] check box is checked, a new session file is not generated in the project directory but is linked to the current session file.

     If the [Make session file link read only] check box is checked, the linked session file is used as read-only.
  6. Click the [OK] button to close the [Debug Sessions] dialog box.

- To remove a session
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).

2. Select the session you would like to remove.
3. Click the [Remove] button.

   Note that the current session cannot be removed.
4. Click the [OK] button to close the [Debug Sessions] dialog box.

- To view the session properties
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).
  2. Select the session you would like to view the properties for.
  3. Click the [Properties] button to display the [Session Properties] dialog box (figure 4.33).

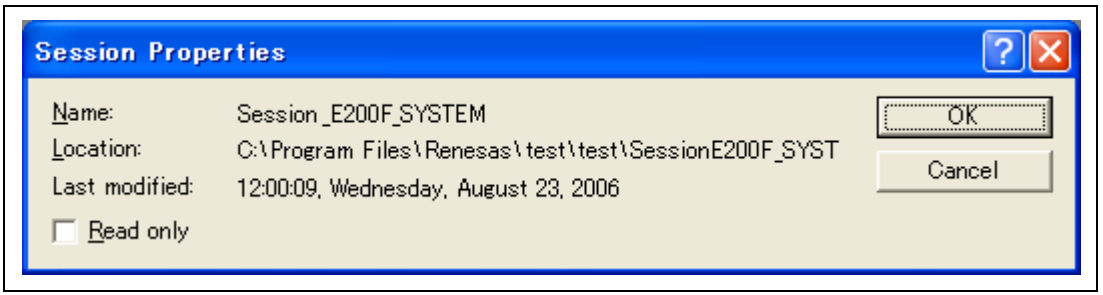


Figure 4.33 [Session Properties] Dialog Box

- To make a session read-only
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).
  2. Select the session you would like to make read-only.
  3. Click the [Properties] button to display the [Session Properties] dialog box (figure 4.33).
  4. Check the [Read only] check box to make the link read-only. This is useful if you are sharing debugger-setting files and you do not want data to be modified accidentally.
  5. Click the [OK] button.

- To save a session with a different name
  1. Select [Options -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.31).
  2. Select the session you would like to save.
  3. Click the [Save as…] button to display the [Save Session] dialog box (figure 4.34).
  4. Browse to the new file location.

5. If you want to export the session file to another location, leave the [Maintain link] check box unchecked. If you would like the High-performance Embedded Workshop to use this location instead of the current session location, check the [Maintain link] check box.
6. Click the [Save] button.

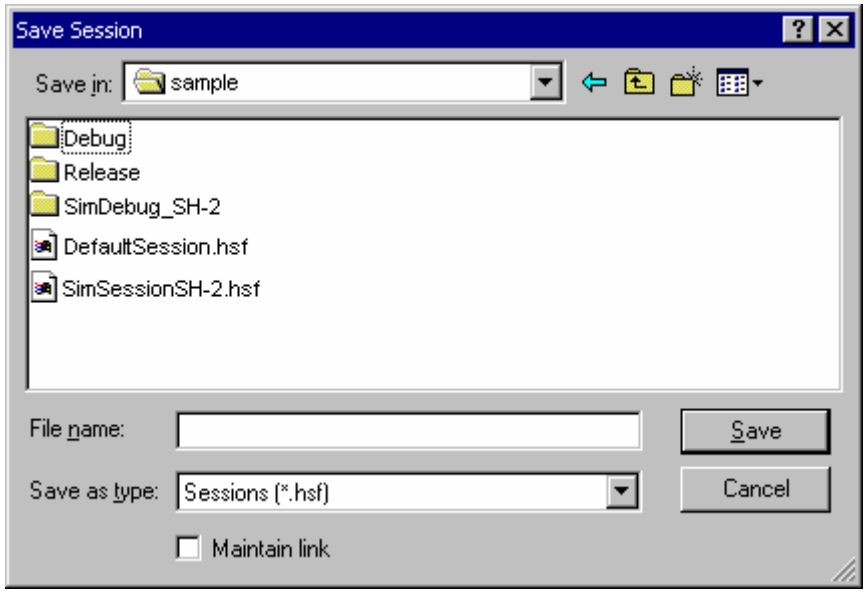


Figure 4.34 [Save Session] Dialog Box

4.4.3 Saving Session Information

- To save a session
  Select [File -> Save Session].

4.5 Connecting the Emulator

Select either of the following two ways to connect the emulator:

(a) Connecting the emulator after the setting at emulator activation

Select [Debug settings] from the [Options] menu to open the [Debug Settings] dialog box. It is possible to register the download module or the command chain that is automatically executed at activation. For details on the [Debug Settings] dialog box, refer to section 4.3, Setting at Emulator Activation.

When the dialog box is closed after setting the [Debug Settings] dialog box, the emulator will automatically be connected.

(b) Connecting the emulator without the setting at emulator activation

Connect the emulator by simply switching the session file to one in which the setting for the emulator use has been registered.

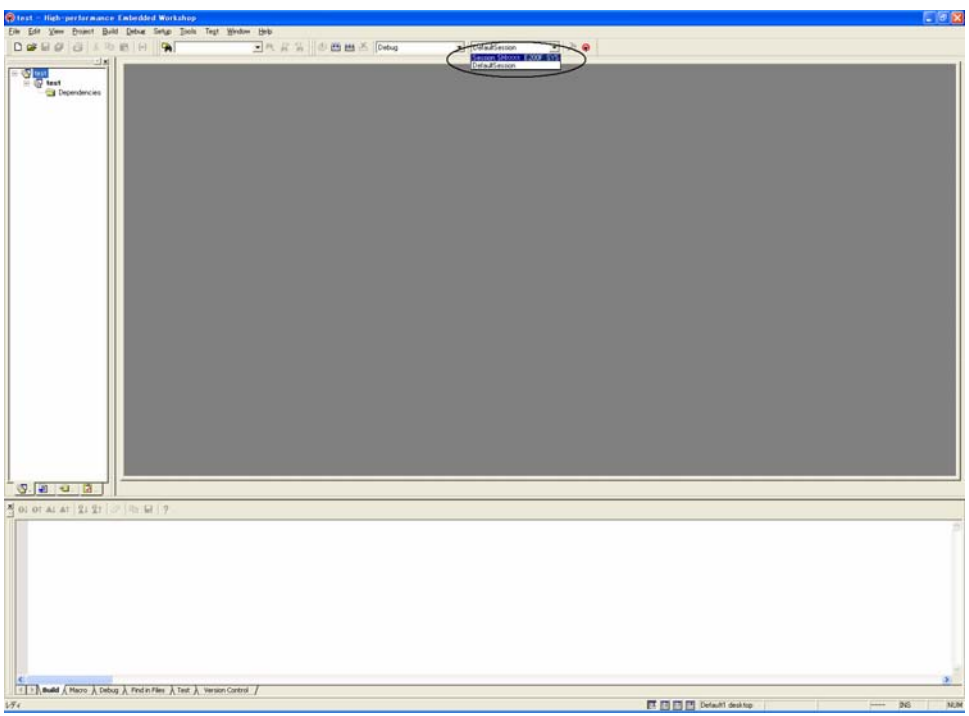


Figure 4.35 Selecting the Session File

In the list box that is circled in figure 4.35, select the session file name including the character string that has been set in the [Target name] text box in figure 4.24, [New Project –8/9– Debugger Option] dialog box. The setting for using the emulator has been registered in this session file.

After the session file name is selected, the emulator will automatically be connected. For details on the session file, refer to section 4.4, Debug Sessions.

4.6 Reconnecting the Emulator

When the emulator is disconnected, use the following way for reconnection:

Select [Build -> Debug -> Connect] or click the [Connect] toolbar button (). The emulator is connected.

Note: The emulator must be selected in the [Target] drop-down list box of the [Debug Settings] dialog box (see figure 4.28, [Debug Settings] Dialog Box ([Target] Page)) that is opened by selecting [Debug settings] from the [Options] menu.

4.7 Ending the Emulator

When using the toolchain, the emulator can be exited by using the following two methods:

- Canceling the connection of the emulator being activated
- Exiting the High-performance Embedded Workshop

(1) Canceling the connection of the emulator being activated

Select [Disconnect] from the [Debug] menu or click the [Disconnect] toolbar button ( ).

(2) Exiting the High-performance Embedded Workshop

Select [Exit] from the [File] menu.

A message box is displayed. If necessary, click the [Yes] button to save a session. After saving a session, the High-performance Embedded Workshop exits. If not necessary, click the [No] button to exit the High-performance Embedded Workshop.

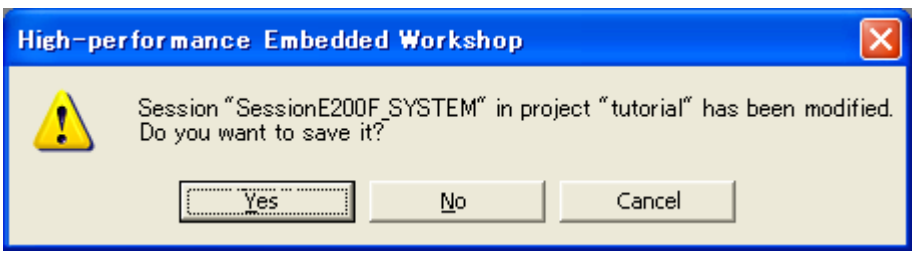


Figure 4.36 [Session has been modified] Message Box

4.8 Uninstalling the Emulator's Software

Follow this procedure to remove the installed emulator's software from the user's host machine. As the installed product is registered with the High-performance Embedded Workshop, uninstall the product on the High-performance Embedded Workshop screen.

It is also possible to uninstall the emulator's software by using [Add/Remove Programs] in the control panel. In this case, however, note that all the tools (including the compiler) in the High-performance Embedded Workshop will be removed.

1. Activate the High-performance Embedded Workshop.
2. Click the [Administration…] button in the [Welcome!] dialog box.

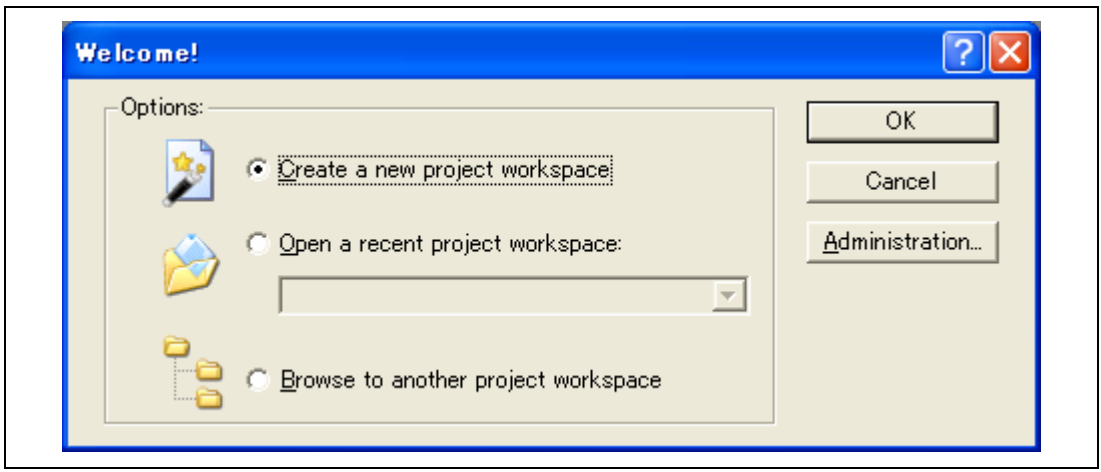


Figure 4.37 [Welcome!] Dialog Box

3. The [Tools Administration] dialog box is opened.

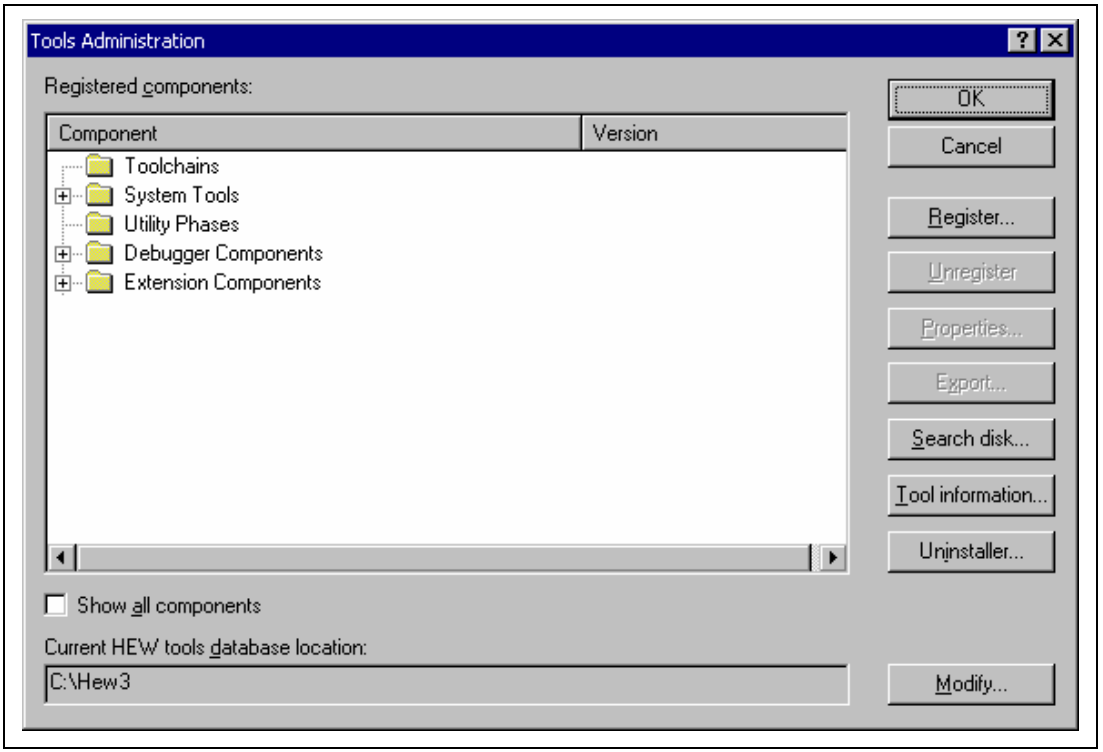


Figure 4.38 [Tools Administration] Dialog Box

4. Click the [+] mark at the left of [Debugger Components] in the [Registered components] list box to list the installed components. Then, highlight the product name to be uninstalled.

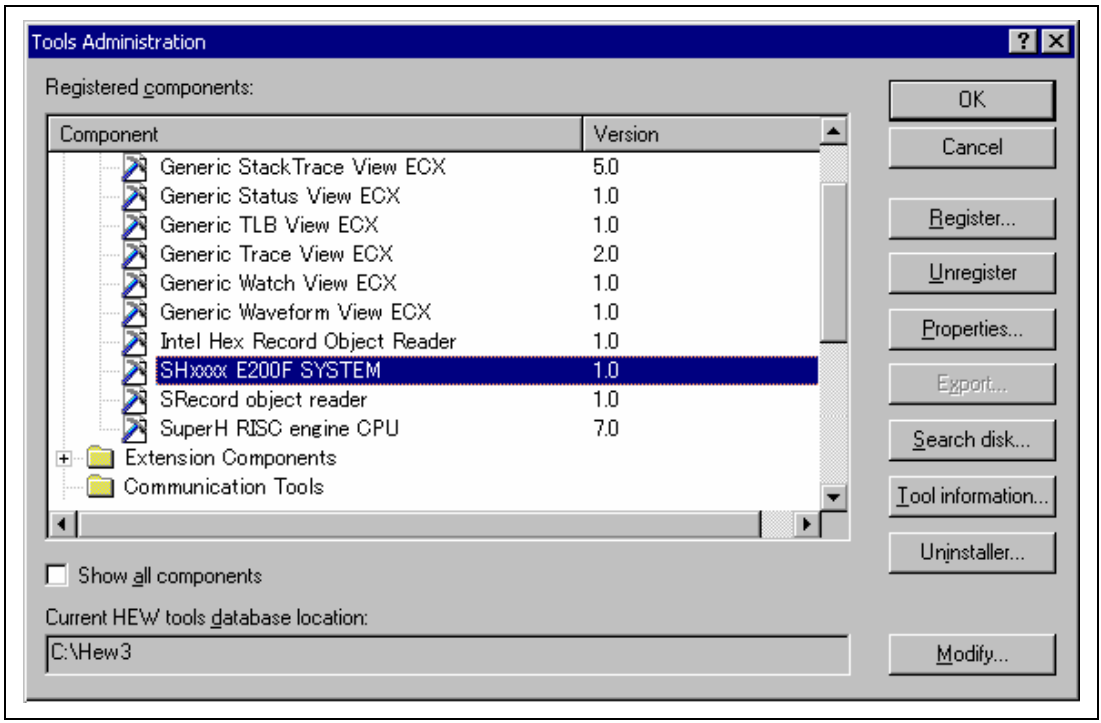


Figure 4.39 Highlighting the Product to be Uninstalled

5. Click the [Unregister] button. After the following message box is displayed, click the [Yes] button.

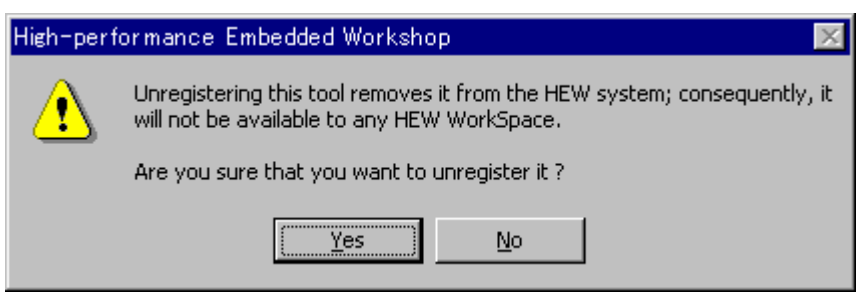


Figure 4.40 [Unregistering this tool] Message Box

This is the end of canceling the High-performance Embedded Workshop registration. Then, remove the file for the emulator from the host machine.

6. Click the [Uninstaller…] button in the [Tools Administration] dialog box to open the [Uninstall HEW Tool] dialog box.

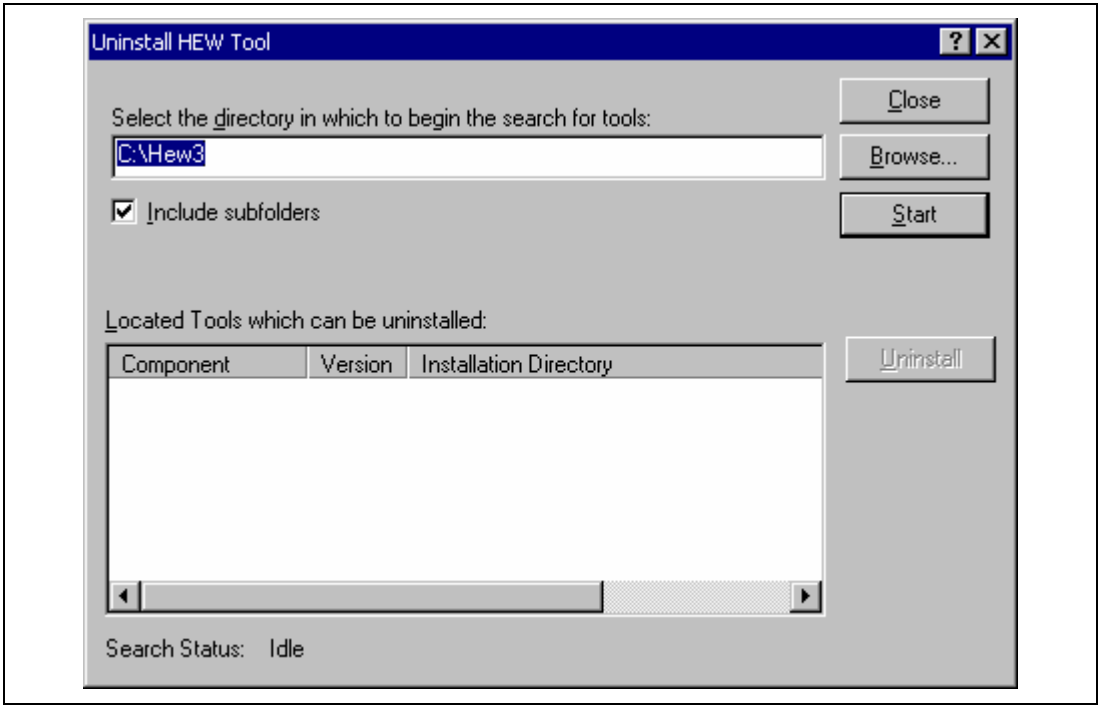


Figure 4.41 [Uninstall HEW Tool] Dialog Box

7. Click the [Start] button to list the installed components.

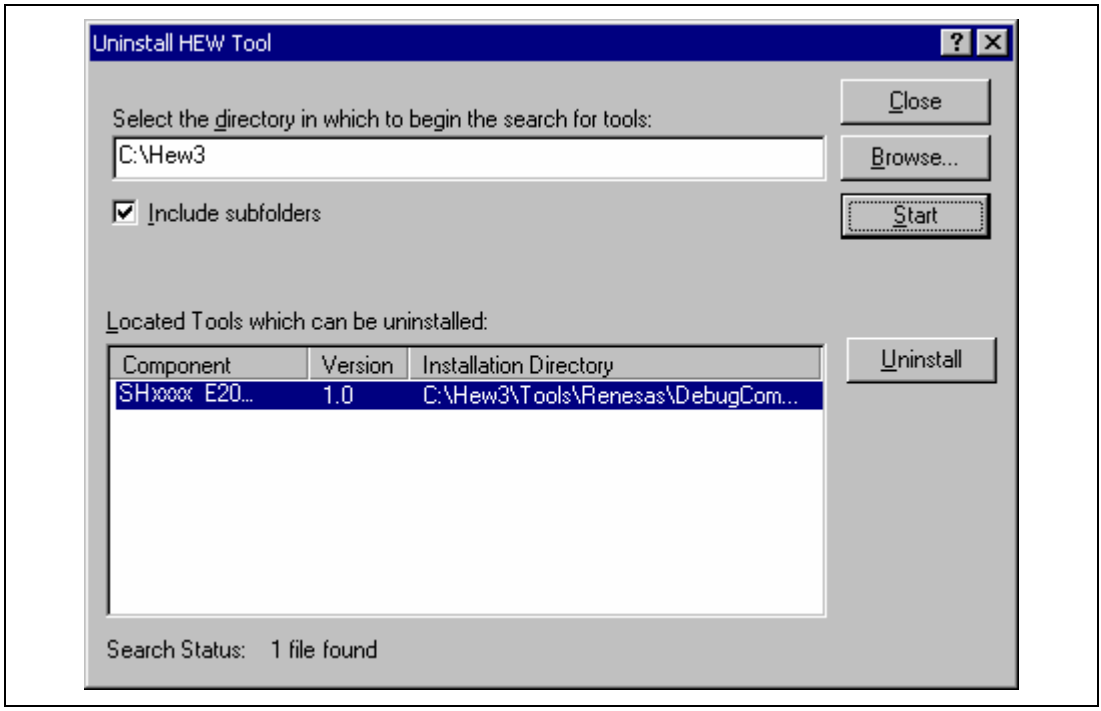


Figure 4.42 Highlighting the Product to be Uninstalled

Highlight the product name to be uninstalled and click the [Uninstall] button. This is the end of uninstallation.

CAUTION

A shared file may be detected while the program is being removed. If another product may be using the shared file, do not remove the file. If another product does not start up after the removal process, re-install that product.

Section 5 Debugging

This section explains the functions specific to this product that have not been described in section 15, Using the Debugger, of the High-performance Embedded Workshop User's Manual.

5.1 Setting the Environment for Emulation

5.1.1 Opening the [Configuration] Dialog Box

Selecting [Setup -> Emulator -> System…] or clicking the [Emulator System] toolbar button () opens the [Configuration] dialog box.

5.1.2 [General] Page

Sets the emulator operation conditions.

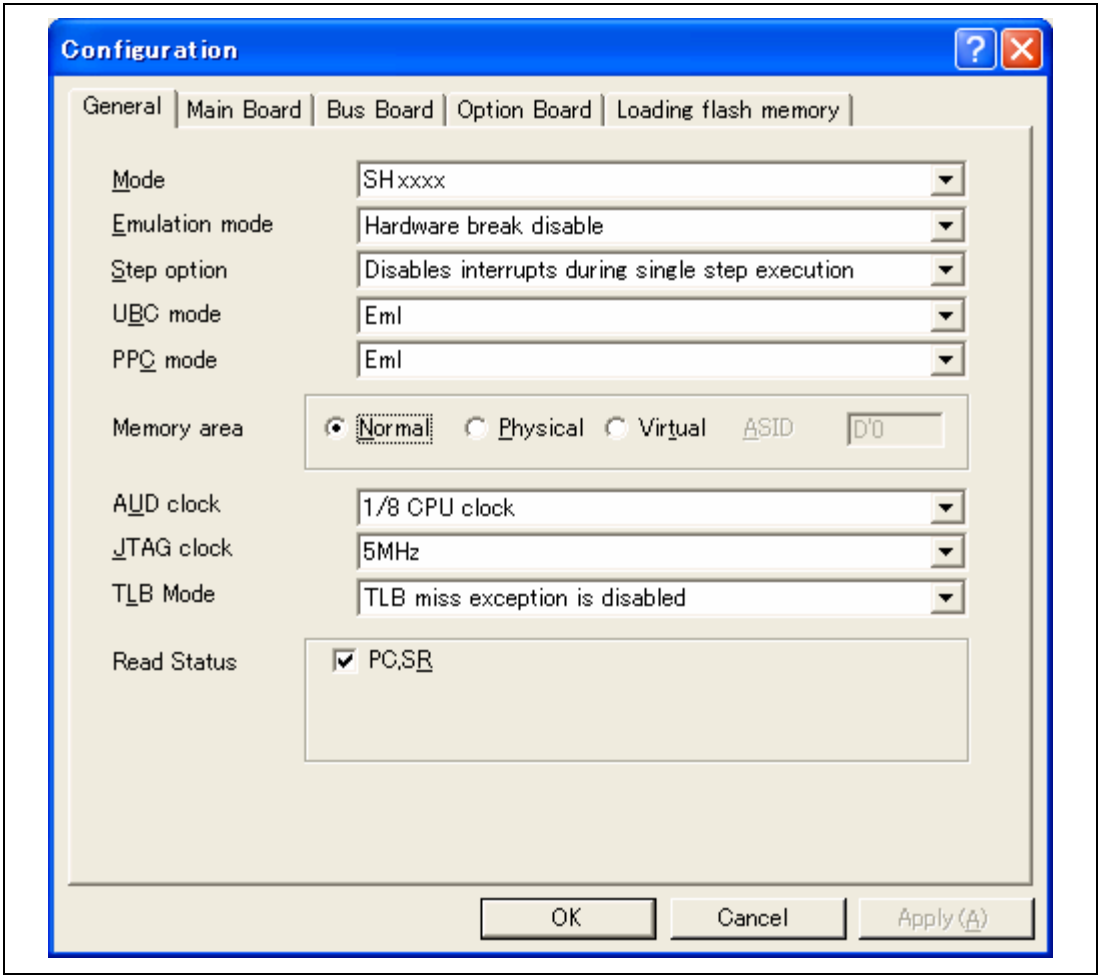


Figure 5.1 [Configuration] Dialog Box ([General] Page)

Items that can be displayed in this page are listed below.

[Mode] Displays the MPU name.

[Emulation mode] Selects the emulation mode at user program execution.

[Hardware break disable]: Disables hardware breaks generated in the emulator main unit case. Functions to be disabled are AUD Event break, BUS Event break, Other Event break, and trace overflow break.

[Hardware break enable]: Enables hardware breaks generated in the emulator main unit case. In this case, the Onchip Event function and the monitoring function of the physical address during user program execution will be disabled.

[No break]: Temporarily disables PC breakpoint or break condition settings during emulation.

[Step option] Sets the step interrupt option.

[Disable interrupts during single step execution]:
Disables interrupts[*1] during step execution.

[Enable interrupts during single step execution]:
Enables interrupts[*1] during step execution.

[UBC mode] Sets the UBC mode.

[EML]: The UBC is used as an Onchip event function by the emulator.

[User]: Releases the UBC to the users.[*2]

[PPC mode] Sets the PPC mode.

[EML]: The PPC is used as a performance counter by the emulator.

[User]: Releases the PPC to the users.[*2]

[Memory area] Sets the method to specify the address in the memory range.

[Normal]: Specifies the address depending on the MMU state.

[Physical]: Specifies the physical address.

[Virtual]: Specifies the virtual address.

[AUD clock]	A clock used in acquiring AUD traces. If its frequency is set too low, complete data may not be acquired during realtime tracing. Set the frequency not to exceed the upper limit for the MPU's AUD clock. The AUD clock is only needed for using emulators that have an AUD trace function. For the upper limit for the AUD clock, refer to section 2.2.2, Notes on Using the JTAG (H-UDI) Clock (TCK) and AUD Clock (AUDCK), in the additional document, Supplementary Information on Using the SHxxxx.
[JTAG clock]	A communication clock used except for acquiring AUD trace. If its frequency is set too low, the speed of downloading will be lowered. Set the frequency not to exceed the upper limit for the MPU's guaranteed TCK range. For the upper limit for TCK, refer to section 2.2.2, Notes on Using the JTAG (H-UDI) Clock (TCK) and AUD Clock (AUDCK), in the additional document, Supplementary Information on Using the SHxxxx.
[TLB Mode]	Enables or disables memory-access exception during a break. [TLB miss exception is disabled]: Disables memory-access exception during a break. [TLB miss exception is enabled]: Enables memory-access exception during a break.
[Read status]	Specifies whether or not the value of PC_SR is displayed on the status bar during user program execution.

Notes: 1. Includes interrupts in a break.

2. When ‘User’ is selected for [UBC mode], Ch10(IA_OA_R) and Ch11(IA_OA_CT_R) that can be set on the [Onchip] sheet of the [Event] window cannot be used. When ‘User’ is selected for [PPC mode], Ch1 and Ch2 of the performance analysis function and options 1 and 2 of the profiling function cannot be used.

Note: The items that can be set in this dialog box vary according to the emulator in use. For details, refer to the online help.

5.1.3 [Main Board] Page

Sets the main unit case operation conditions.

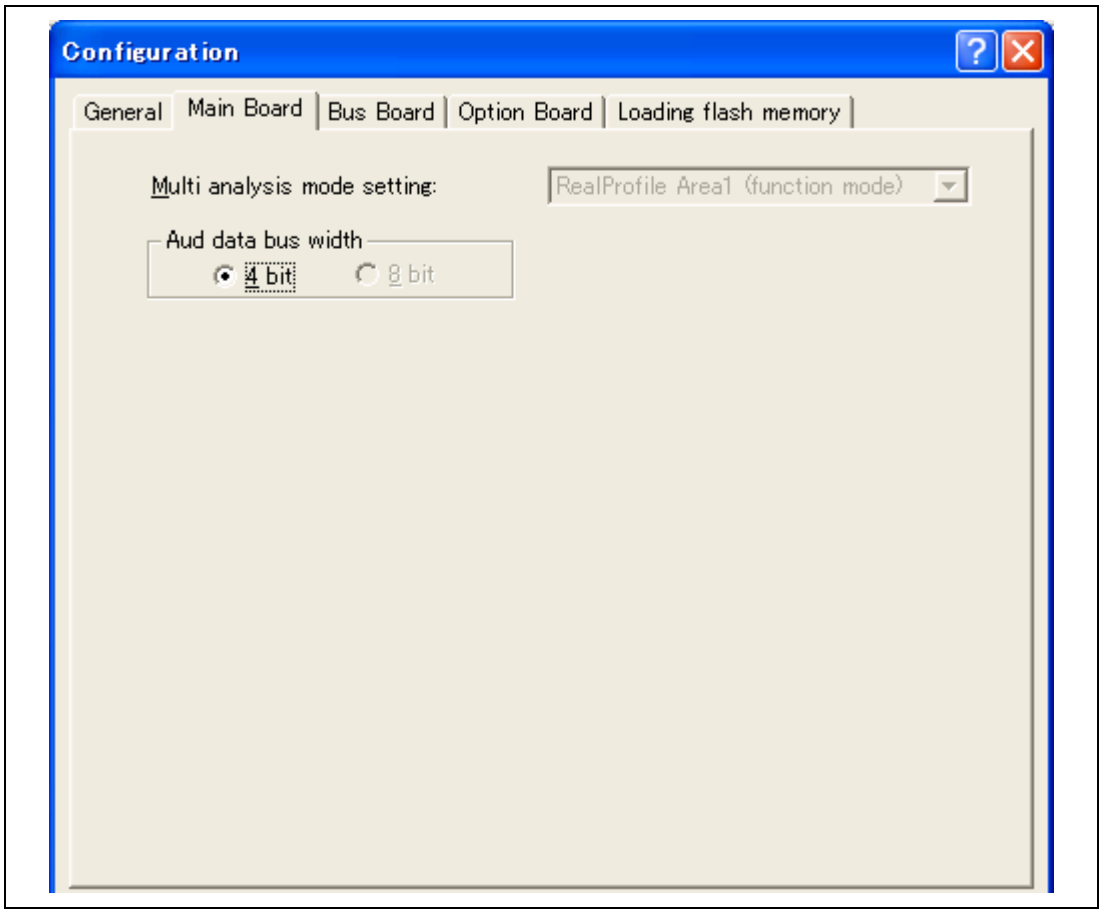


Figure 5.2 [Configuration] Dialog Box ([Main Board] Page)

Items that can be displayed in this page are listed below.

[Multi analysis mode setting]	Displays the functions of the E200F main unit. Here, this item displays which function of realtime profiling, coverage, or I/O analyzer has been selected. [RealProfile Area 1 (function mode)]: The realtime profiling function (measurement mode: function mode) is selected. [RealProfile Area1 (nest mode)]: The realtime profiling function (measurement mode: nest mode) is selected. [Coverage (4M)]: The coverage function is selected. <Depending on the product>: Uses the I/O analyzer function.
[AUD data bus width]	Sets the AUD data bus width.* The bus width can be selected as 4 bits or 8 bits.

Note: In some products, the bus width will be fixed. For the specifications of each product, refer to the online help.

5.1.4 [Bus Board] Page

Sets the trace unit operation conditions.

Note: When the trace unit is not connected to the emulator, this page is not displayed.

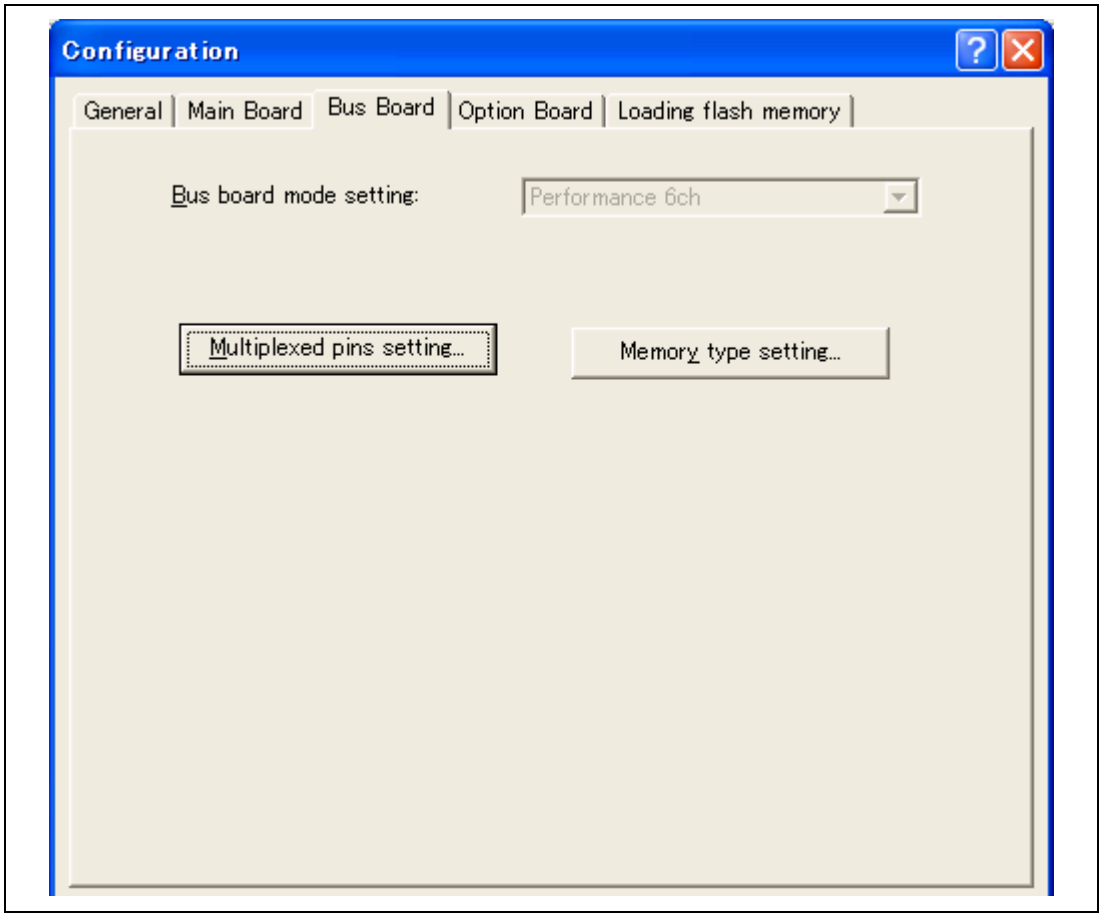


Figure 5.3 [Configuration] Dialog Box ([Bus Board] Page)

Items that can be displayed in this page are listed below.

[Bus board mode setting]	Displays the function of the trace unit. [Trace/break 6ch (Trace 262144 cycles)]: The external bus trace or break function has been selected (event detection channels: six). The 262144 bys cycles can be acquired by a trace. [Performance 6ch]: The external bus trace function has been selected (measurement channels: six). [Emulation memory (4M, Trace 8192 cycles)]: The external emulation memory function (4 Mbytes x 1 block) has been selected. The 8192 bys cycles can be acquired by a trace.
[Multiplexed pins setting...]	Selects the state of the multiplexed pins.
[Memory type setting...]	Selects the type of memory for each area or the content of connection. Select [Normal] for other than SRAM with byte control.

Clicking the [Multiplexed pins setting...] button opens the [Multiplexed pins setting] dialog box. Select the pin name according to the state of the multiplexed pin.

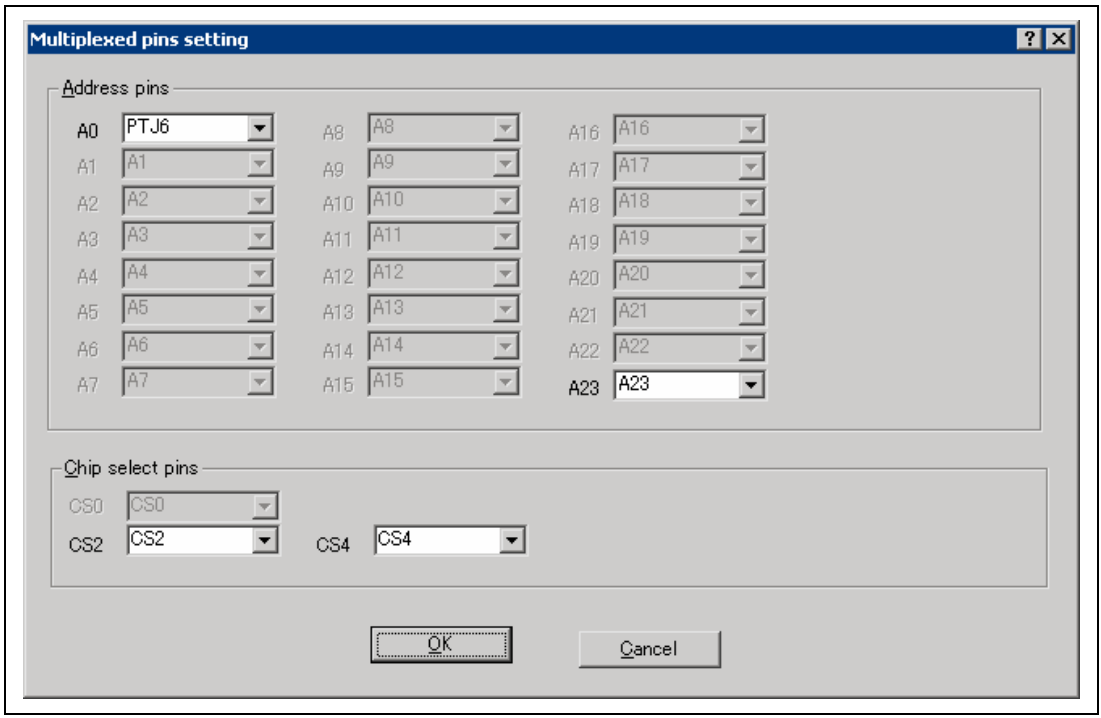


Figure 5.4 [Multiplexed pins setting] Dialog Box

Note: This dialog box differs depending on the product.

Clicking the [Memory type setting...] button opens the [Memory type setting] dialog box. Set the memory type on the board.

When SDRAM is to be acquired by trace, the number of bits of the row and column addresses and the CAS latency must be set in this dialog box. Note that the data bus width will also need to be set.

If those items are set incorrectly, they may be illegally displayed on the [BUS/MFI trace] window.

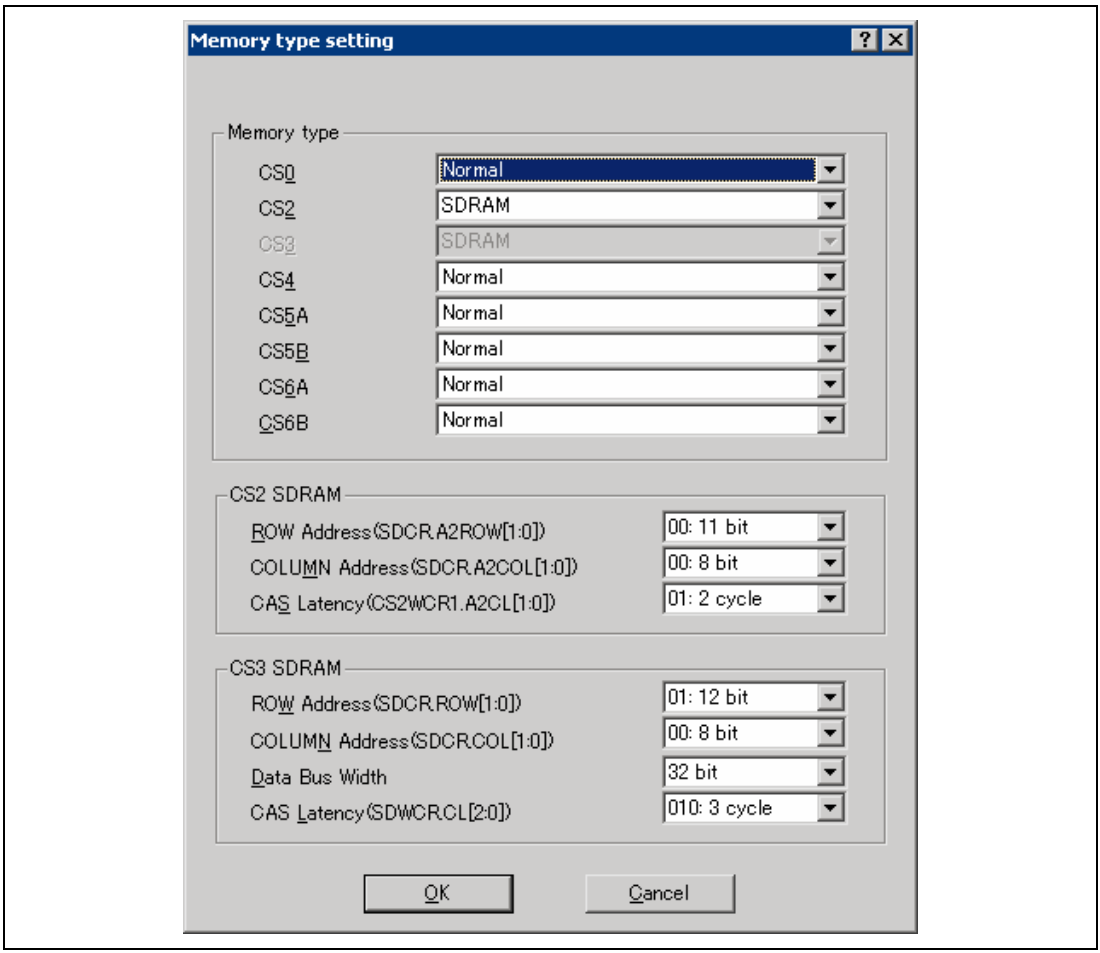


Figure 5.5 [Memory type setting] Dialog Box

Note: This dialog box differs depending on the product.

5.1.5 [Option Board] Page

Sets the expansion profiling unit operation conditions.

Note: When the expansion profiling unit is not connected to the emulator, this page is not displayed.

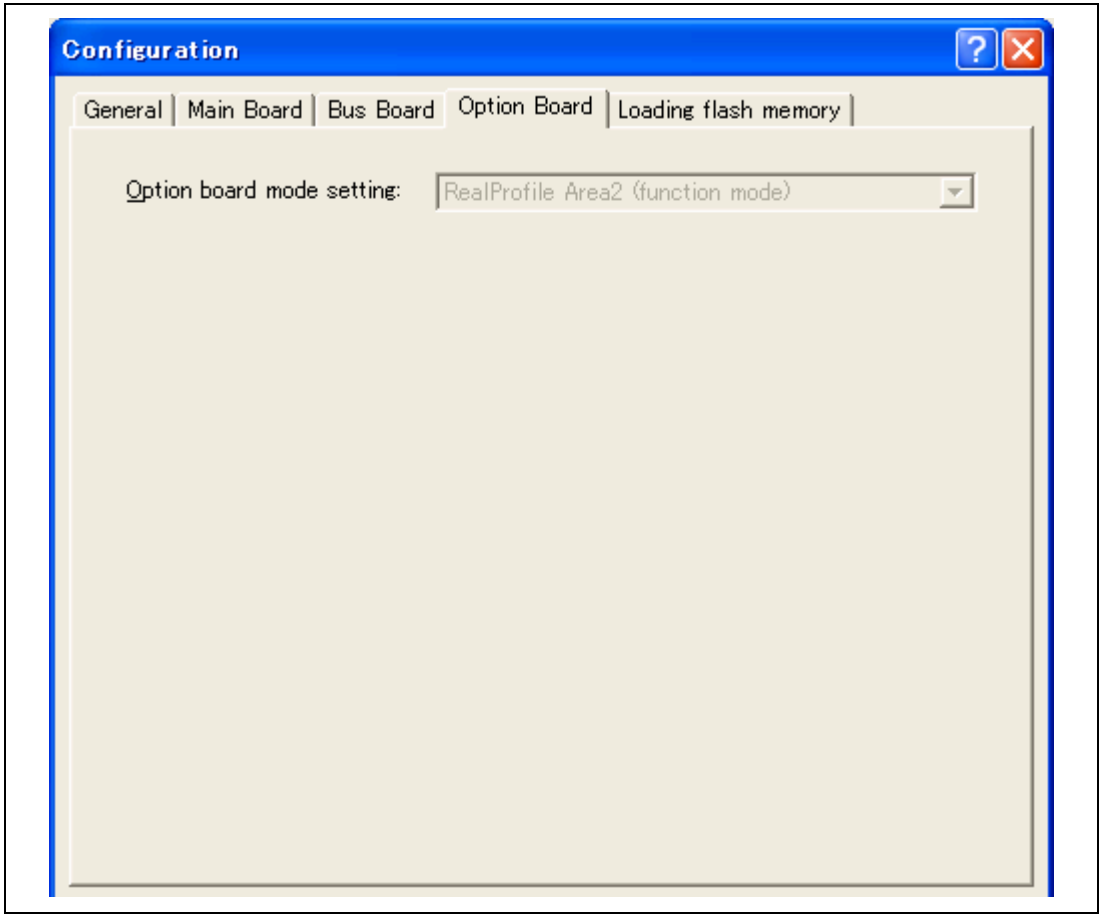


Figure 5.6 [Configuration] Dialog Box ([Option Board] Page)

Item that can be displayed in this page is listed below.

[Option board mode setting]	Displays the functions of the expansion profiling unit. [RealProfile Area 2 (function mode)]: The realtime profiling function (measurement mode: function mode) has been selected. [RealProfile Area 2 (nest mode)]: The realtime profiling function (measurement mode: nest mode) has been selected. [Coverage (8M)]: The coverage function has been selected. The 8-Mbyte area can be measured.

5.1.6 Downloading to the Flash Memory

Sets the emulator operation conditions for downloading the external flash memory.

For details, refer to section 6.22, Download Function to the Flash Memory Area.

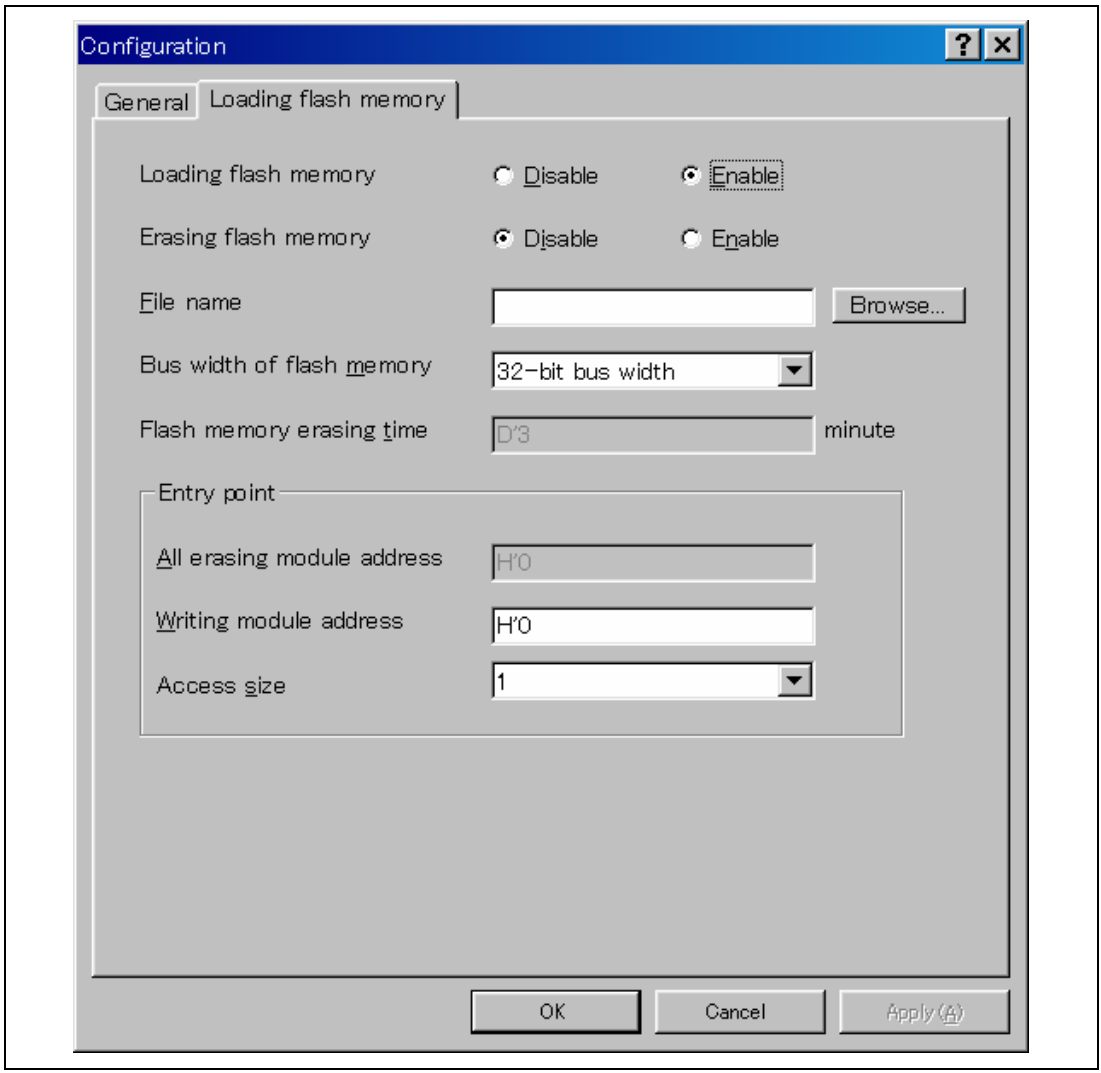


Figure 5.7 [Configuration] Dialog Box ([Loading flash memory] Page)

Items that can be displayed in this page are listed below.

[Loading flash memory]	Sets Enable for flash memory downloading. At Enable, when the flash memory is downloaded on the High-performance Embedded Workshop, the write module is always called. [Disable]: Not download to the flash memory [Enable]: Download to the flash memory
[Erasing flash memory]	Sets Enable for erasing before the flash memory is written. At Enable, the erase module is called before calling the write module. [Disable]: Not erase the flash memory [Enable]: Erase the flash memory
[File name]	Sets the write/erase module name. The file that has been set is loaded to the RAM area before loading to the flash memory.
[Bus width of flash memory]	Sets the bus width of the flash memory.
[Flash memory erasing time]	Sets the TIMEOUT value at flash memory erasing. Increase the value if erasing requires much time although the default time is three minutes. The values that can be set are as follows: D'0 (minimum) and D’65535 (maximum). Only positive integers can be input.
[Entry point]	Sets the calling destination address of the write/erase module. (It must be RAM address.) [All erasing module address]: Enters the calling destination address of the erase module. [Writing module address]: Enters the calling destination address of the write module. [Access size]: Selects the access size of the RAM area that is used for loading the write/erase module.

5.1.7 Opening the [Memory Mapping] Dialog Box

Selecting [Setup -> Emulator -> Memory Resource…] or clicking the [Emulator Memory Resource] toolbar button ( ) opens the [Memory Mapping] dialog box.

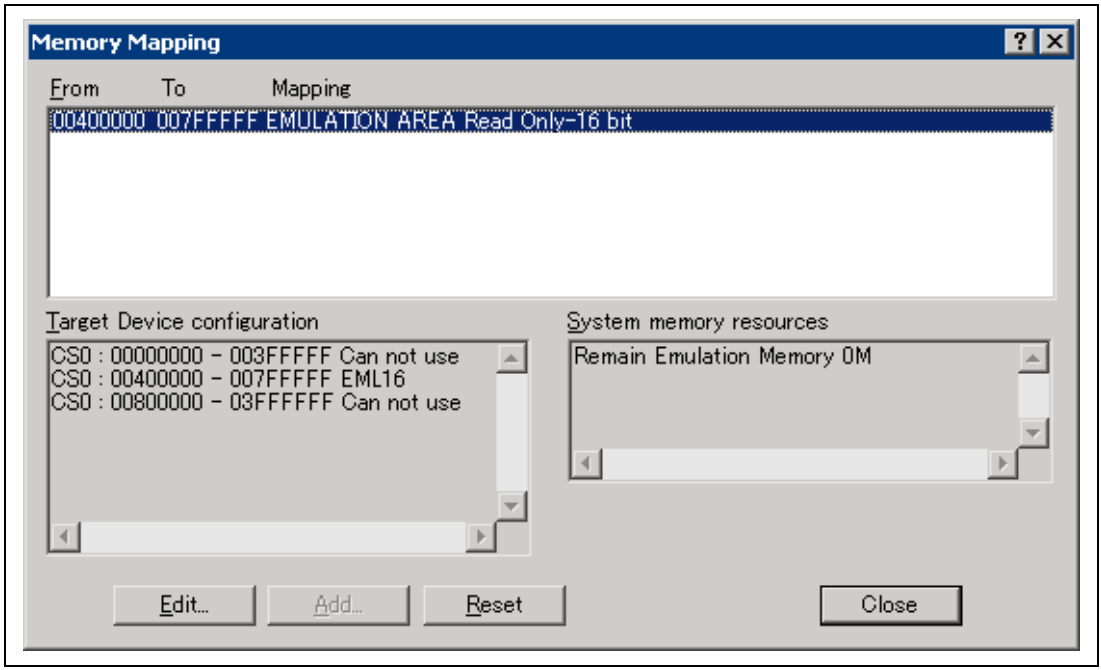


Figure 5.8 [Memory Mapping] Dialog Box

This dialog box displays the current memory map and the state of the emulation memory.

[Edit...]: Displays the [Edit Memory Mapping] dialog box, allowing the user to modify the address range and attributes of a memory map.

[Add...]: Allocates the new emulation memory.

[Reset]: Resets the map memory to its default settings.

[Close]: Closes the dialog box.

The memory configuration of the device being emulated is shown on the [Memory] sheet of the [Status] window.

Notes: 1. When the trace unit is not connected to the emulator, memory mapping cannot be set on this page.
2. The items that can be set in this dialog box depend on the emulator in use. For details, refer to the online help.
3. The items to be displayed in this dialog box depend on the emulator in use. For details, refer to the online help.

5.1.8 Changing the Memory Map Setting

Clicking the [Edit…] button on the [Memory Mapping] dialog box after selecting the information on the memory map setting you want to change opens the [Edit Memory Mapping] dialog box.

This dialog box also opens when the [Add…] button is clicked on the [Memory Mapping] dialog box.

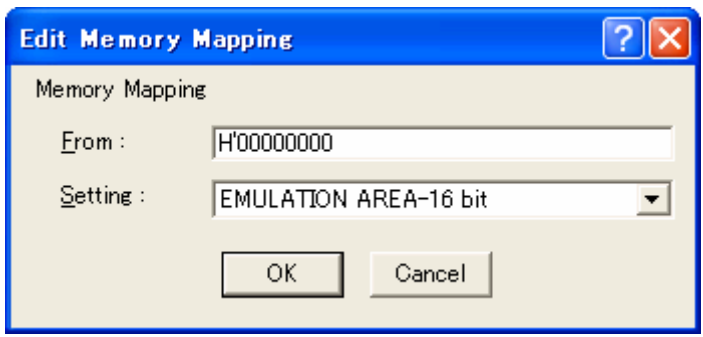


Figure 5.9 [Edit Memory Mapping] Dialog Box

Use this dialog box to change the address range and attributes of a memory map.

[From]: Enter the start address of the map range.

[Setting]: Enter the memory map setting.
The choices given are listed below. Since these items differ depending on the product, refer to the online help.

- EMULATION AREA – 16 bit: The data bus width is 16 bits.
- EMULATION AREA – 16 bit Read-Only:
  This area is write-protected and the data bus width is 16 bits.
- USER AREA: This area is specified as the user area.

Note: The unit for mapping the memory is fixed to 4 Mbytes.

5.2 Downloading a Program

This section describes how to download a program and view it as source code or assembly-language mnemonics.

Note: After a break has occurred, the High-performance Embedded Workshop displays the location of the program counter (PC). In most cases, for example if an Elf/Dwarf2-based project is moved from its original path, the source file may not be automatically found. In this case, the High-performance Embedded Workshop will open a source file browser dialog box to allow you to manually locate the file.

5.2.1 Downloading a Program

A load module to be debugged must be downloaded.

To download a program, select the load module from [Debug -> Download] or select [Download] from the popup menu opened by clicking the right-hand mouse button on the load module in [Download modules] of the [Workspace] window.

Notes: 1. Before downloading a program, it must be registered to the High-performance Embedded Workshop as a load module. For registration, refer to section 4.3, Setting at Emulator Activation.

2. When a program is downloaded to the external RAM, the bus controller must be initially set in the area for downloading. Especially, check that the initialization of SDRAM or the setting of the bus width is appropriate for the user system.

5.2.2 Viewing the Source Code

Select your source file and click the [Open] button to make the High-performance Embedded Workshop open the file in the integrated editor. It is also possible to display your source files by double-clicking on them in the [Workspace] window.

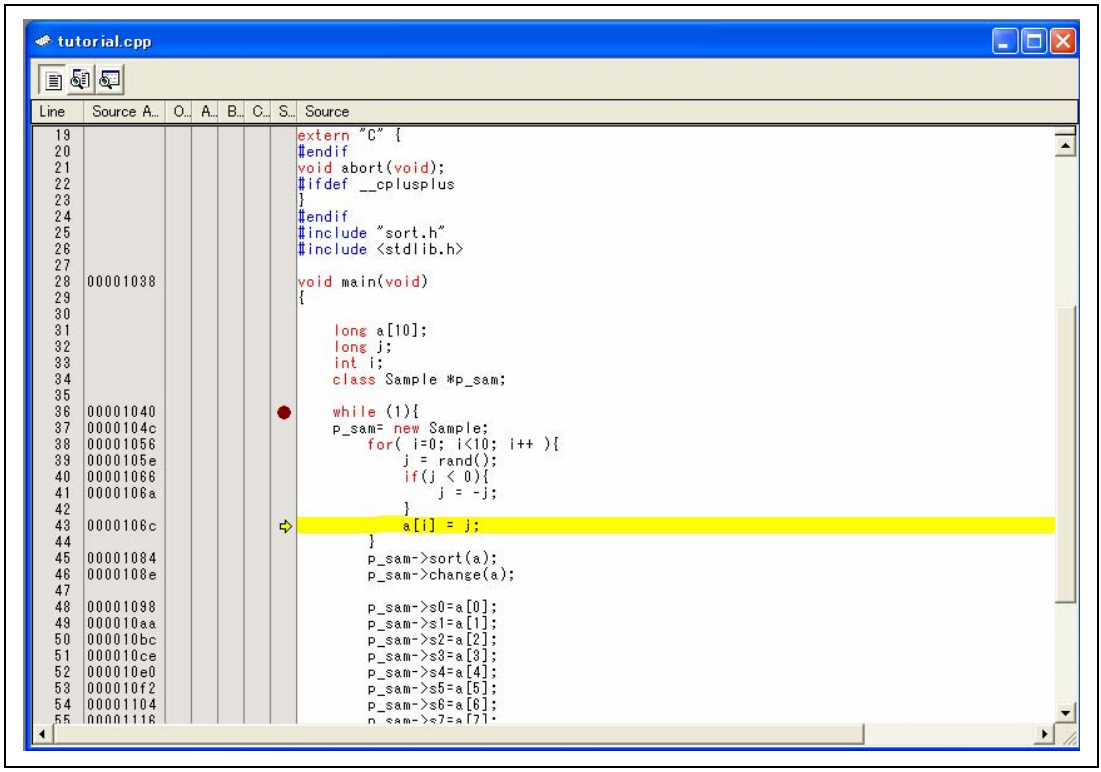


Figure 5.10 [Source] Window

In this window, the following items are shown on the left as line information.

The first column (Source address column): Address information

The second column (Onchip event column): On-chip event information

The third column (AUD event column): AUD event information

The fourth column (BUS event column): External bus event information

The fifth column (CodeCoverage column): Coverage information

The sixth column (S/W breakpoint column): PC, bookmark, and breakpoint information

Note: When the trace unit is not connected to the emulator, the BUS event column is not displayed.

Source address column

When a program is downloaded, an address for the current source file is displayed on the Source address column. These addresses are helpful when setting the PC value or breakpoints.

Onchip event column

The Onchip event column displays the following item:

●: An address condition for the on-chip event is set. The number of address conditions that can be set is the same as that of on-chip event channels at which the address condition can be set.

The bitmap symbol above is shown by double-clicking the Onchip event column. This is also set by using the popup menu.

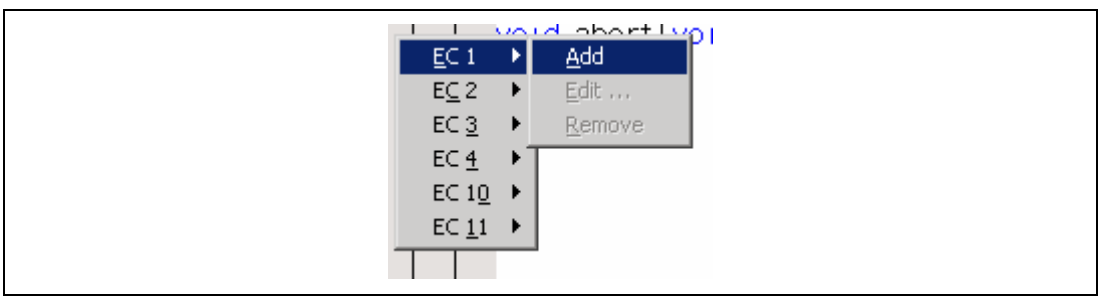


Figure 5.11 Popup Menu for the Onchip Event Column

Note: The contents of the Onchip event column are erased when conditions other than the address condition are added to each channel by using the [Edit] menu or in the [Event] window.

AUD event column

The AUD event column displays the following item:

: An AUD break is set.

: An AUD sequential break is set.

: An AUD trace acquisition is set.

: An AUD trace start is set.

: An AUD trace stop is set.

: An AUD trace sequential stop is set.

: An AUD performance start is set.

: An AUD performance stop is set.

The bitmap symbols for break above are shown by double-clicking the AUD event column. These are also set by using the popup menu.

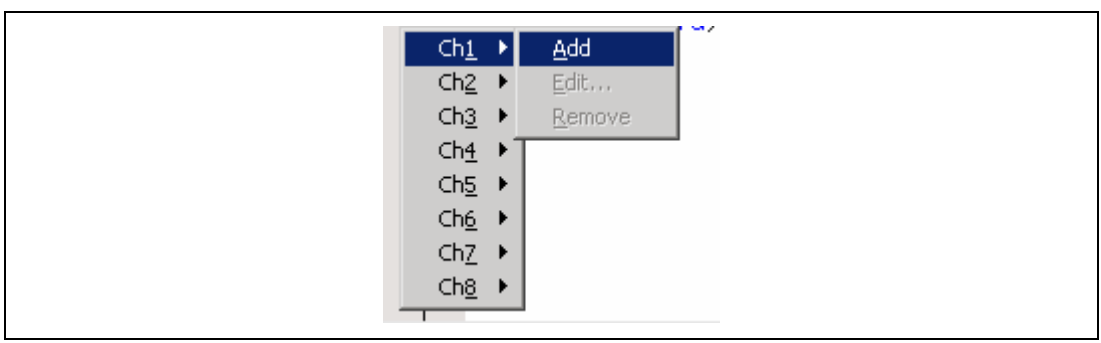


Figure 5.12 Popup Menu for the AUD Event Column

Note: The contents of the AUD event column are erased when conditions other than the address condition are added to each channel by using the [Edit] menu or in the [Event] window.

BUS event column

The BUS event column displays the following item:

: An external bus break is set.

: An external bus sequential break is set.

: An external bus trace acquisition is set.

: An external bus trace start is set.

: An external bus trace stop is set.

: An external bus trace sequential stop is set.

The bitmap symbols for break above are shown by double-clicking the BUS event column. These are also set by using the popup menu.

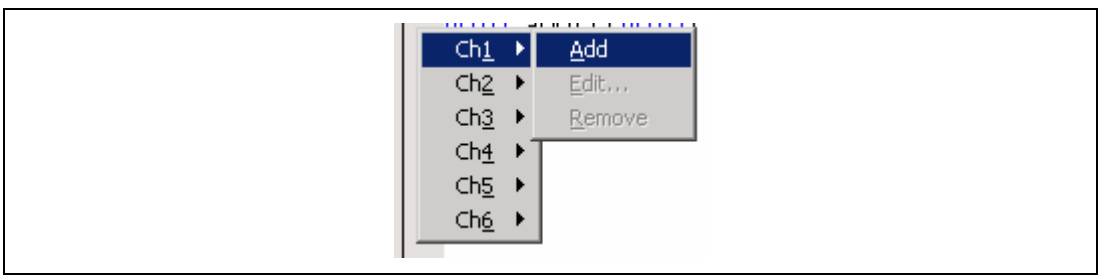


Figure 5.13 Popup Menu for the BUS event Column

Note: The contents of the BUS event column are erased when conditions other than the address condition are added to each channel by using the [Edit] menu or in the [Event] window.

S/W breakpoint column

Editor column displays the following items:

: A bookmark is set.

: A PC breakpoint is set.

: PC location

➲ To switch off a column in all source files

1. Click the right-hand mouse button on the [Source] window.
2. Click the [Define Column Format…] menu item.
3. The [Global Editor Column States] dialog box is displayed.
4. A check box indicates whether the column is enabled or not. If it is checked, the column is enabled. If the check box is gray, the column is enabled in some files and disabled in others. Deselect the check box of a column you want to switch off.
5. Click the [OK] button for the new column settings to take effect.

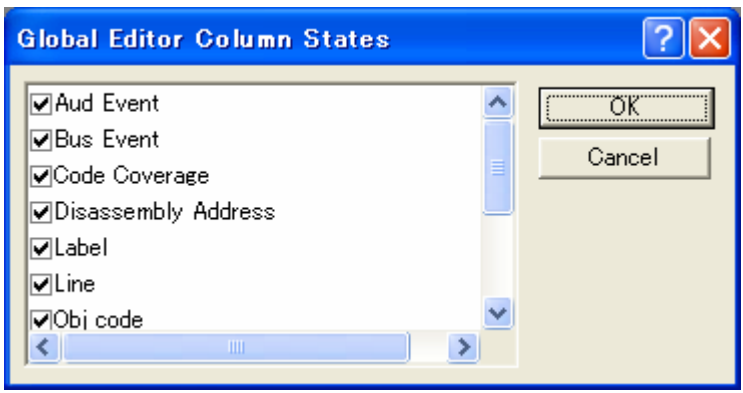


Figure 5.14 [Global Editor Column States] Dialog Box

➲ To switch off a column in one source file

1. Open the source file which contains the column you want to remove and click the right-hand mouse button on the [Editor] window.
2. Click the [Columns] menu item to display a cascaded menu item. The columns are displayed in this popup menu. If a column is enabled, it has a tick mark next to its name. Clicking the entry will toggle whether the column is displayed or not.

5.2.3 Viewing the Assembly-Language Code

Click the right-hand mouse button on the [Source] window to open the popup menu and select [Go to Disassembly] to open the [Disassembly] window at the address that corresponds to the current source file.

If you do not have a source file, but want to view code in the assembly-language level, either choose [View -> Disassembly…] or click the [Disassembly] toolbar button (). The [Disassembly] window opens at the current PC location and shows [Address] and [Code] (optional) which show the disassembled mnemonics (with labels when available).

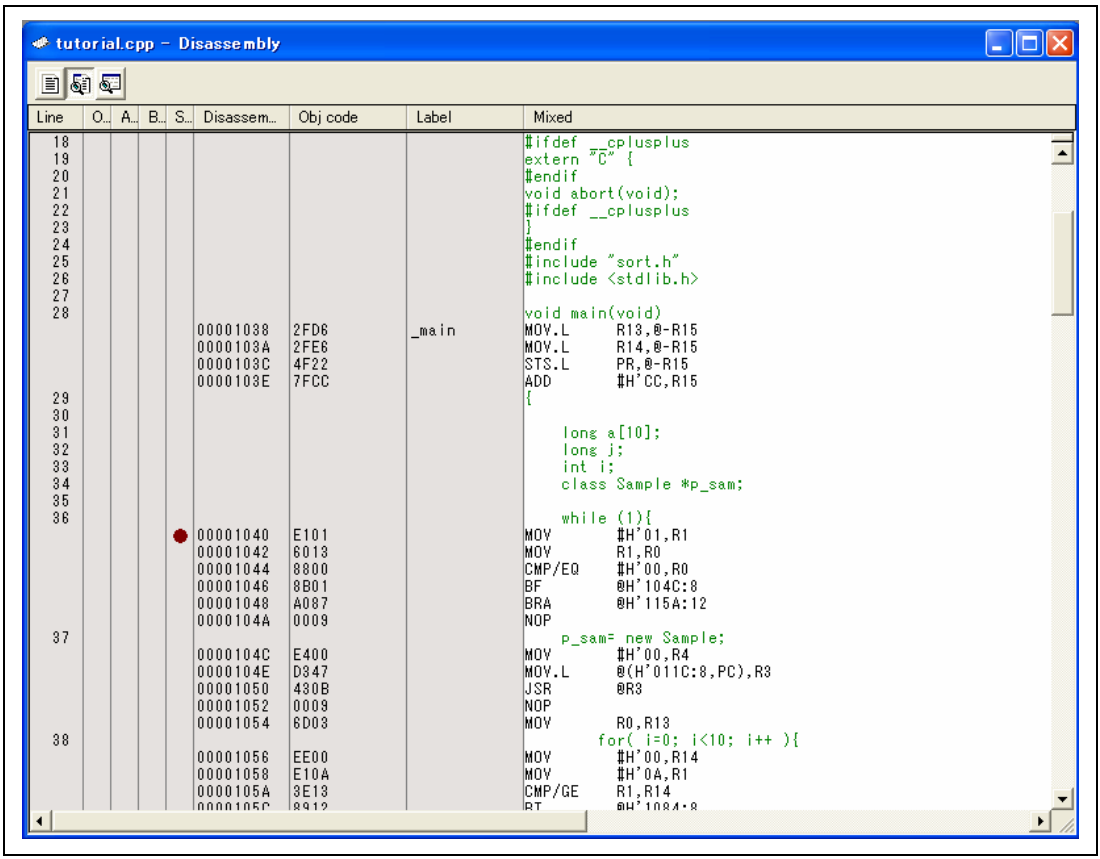


Figure 5.15 [Disassembly] Window

5.2.4 Modifying the Assembly-Language Code

You can modify the assembly-language code by double-clicking on the instruction that you want to change. The [Assembler] dialog box will be opened.

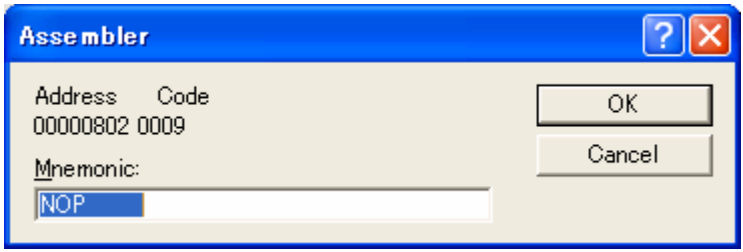


Figure 5.16 [Assembler] Dialog Box

The address, machine code, and disassembled instruction are displayed. Enter the new instruction or edit the current instruction in the [Mnemonic] field. Pressing the [Enter] key will assemble the instruction into memory and move on to the next instruction. Clicking the [OK] button will assemble the instruction into memory and close the dialog box. Clicking the [Cancel] button or pressing the [Esc] key will close the dialog box.

Note: The assembly-language display is disassembled from the actual machine code in the memory. If the memory contents are changed, the dialog box (and the [Disassembly] window) will show the new assembly-language code, but the display content of the [Source] window will not be changed. This is the same even if the source file contains an assembly codes.

5.2.5 Viewing a Specific Address

When you are viewing your program in the [Disassembly] window, you may want to look at another area of your program's code. Rather than scrolling through a lot of code in the program, you can go directly to a specific address. Select [Set Address…] from the popup menu, and the dialog box shown in figure 5.17 is displayed.

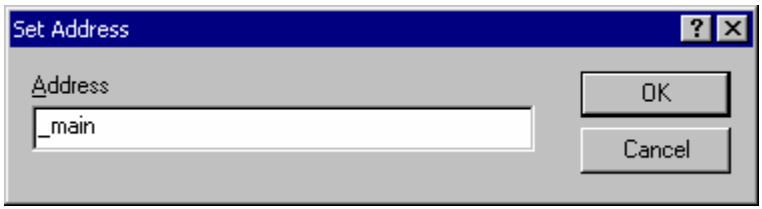


Figure 5.17 [Set Address] Dialog Box

Enter the address or label name in the edit box and either click on the [OK] button or press the [Enter] key. The [Disassembly] window will be updated to show the code at the new address. When an overloaded function or a class name is entered, the [Select Function] dialog box opens for you to select a function.

5.2.6 Viewing the Current Program Counter Address

Wherever you can enter an address or value into the High-performance Embedded Workshop, you can also enter an expression. If you enter a register name prefixed by the hash character, the contents of that register will be used as the value in the expression. Therefore, if you open the [Set Address] dialog box and enter the expression #pc, the [Source] or [Disassembly] window will display the current PC address. It also allows the offset of the current PC to be displayed by entering an expression with the PC register plus an offset, e.g., #PC+0x100.

5.3 Displaying Memory Contents in Realtime

Use the [Monitor] window to monitor the memory contents during user program execution.

When the emulator is activated, it is possible to select whether the address to be monitored is the physical address or the virtual address. For the physical address, monitoring is disabled if [Hardware break enable] is selected from the [Emulation mode] combo box in the [General] sheet of the [Configuration] dialog box.

5.3.1 Opening the [Monitor] Window

To open the [Monitor] window, select [View -> CPU -> Monitor -> Monitor Setting...] or click the [Monitor] toolbar button ( ) to display the [Monitor Setting] dialog box.

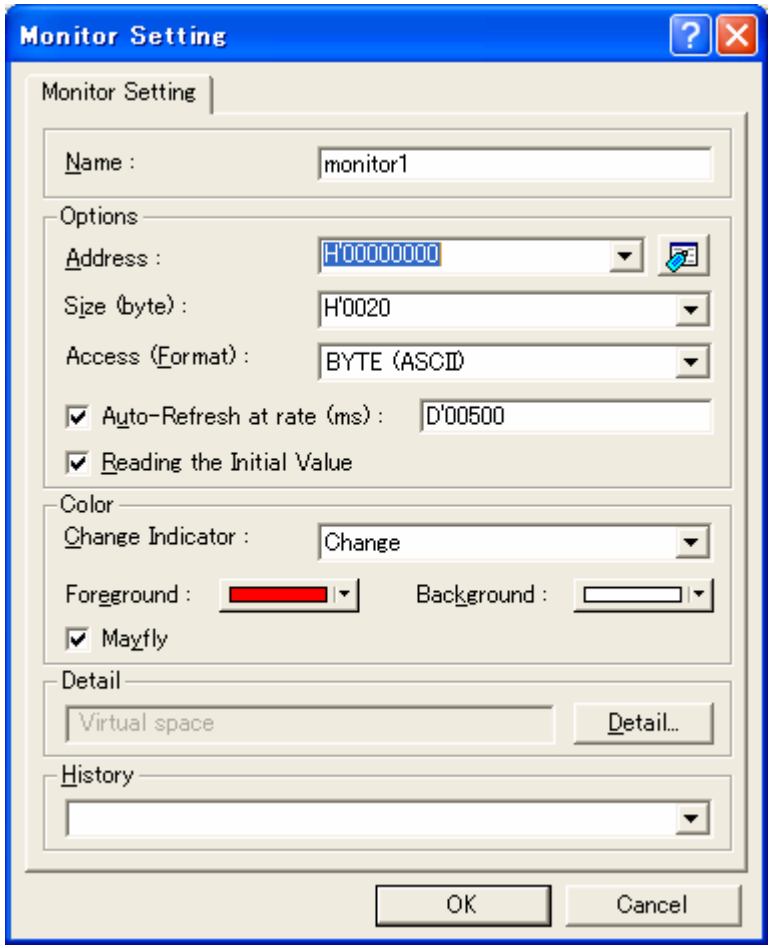


Figure 5.18 [Monitor Setting] Dialog Box

[Name]: Decides the name of the monitor window.

[Options]: Sets monitor conditions.

[Address]: Sets the start address for monitoring.

[Size]: Sets the range for monitoring.

[Access]: Sets the access size to be displayed in the monitor window.

[Auto-Refresh at rate]: Sets the interval for acquisition by monitoring.

[Reading the Initial Value]: Selects reading of the values in the monitored area when the monitor window is opened.

[Color]: Sets the method to update monitoring and the attribute of colors.

[Change Indicator]: Selects how to display the values that have changed during monitoring (available when [Reading the Initial Value] has been selected).

No change: No color change.

Change: Color is changed according to the [Foreground] and [Background] options.

Gray: Those data with values that have not been changed are displayed in gray.

Appear: A value is only displayed after changed.

[Foreground]: Sets the color used for display (available when [Change] has been selected).

[Background]: Sets the background color (available when [Change] has been selected).

[Mayfly]: A check in this box selects restoration of the color of those data which have not been updated in a specified interval to the color selected in the [Background] option. The specified interval is the interval for monitor acquisition (available when [Change], [Gray], or [Appear] has been selected).

[Detail]: Displays whether the address to be monitored is physical or virtual.

[History]: Displays the previous settings.

Note: Selection of the foreground or background color may not be available depending on the operating system in use.

After setting, clicking the [OK] button displays the [Monitor] window.

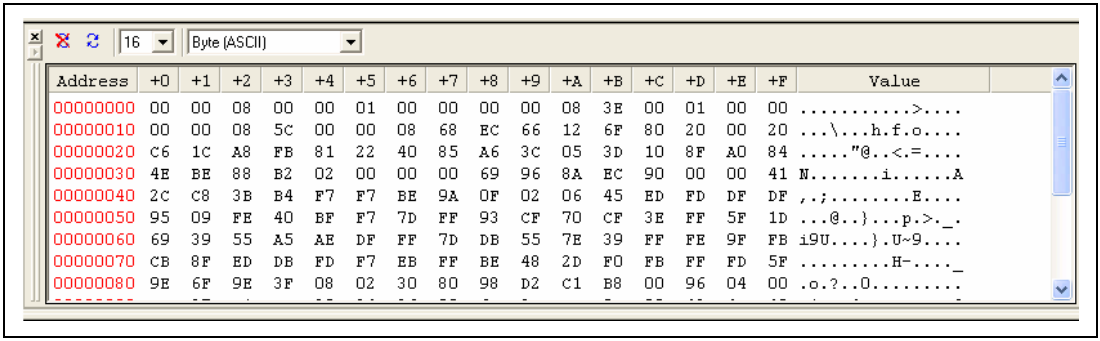


Figure 5.19 [Monitor] Window

During user program execution, the display is updated according to the setting value of the auto-update interval.

Note: Select [Refresh] from the popup menu when data is not displayed correctly after changing the address or content of memory.

5.3.2 Changing the Monitor Settings

Selecting [Monitor Settings...] from the popup menu of the [Monitor] window displays the [Monitor Setting] dialog box, which allows the settings to be changed.

Colors, the size of accesses, and the display format can be easily changed from [Color] or [Access] of the popup menu.

5.3.3 Temporarily Stopping Update of the Monitor

During user program execution, the display of the [Monitor] window is automatically updated according to the auto-update interval. Select [Lock Refresh] from the popup menu of the [Monitor] window to stop the update of display. The characters in the address section are displayed in black, and the update of display is stopped.

Selecting [Lock Refresh] again from the popup menu cancels the stopped state.

5.3.4 Deleting the Monitor Settings

Selecting [Close] from the popup menu of the [Monitor] window to be deleted closes the [Monitor] window and deletes the monitor settings.

5.3.5 Monitoring Variables

Using the [Watch] window refers to the value of any variables.

The value of the variable that has been registered in the [Watch] window can be updated during execution of the user program. Select [Auto Update] from the popup menu of the [Watch] window.

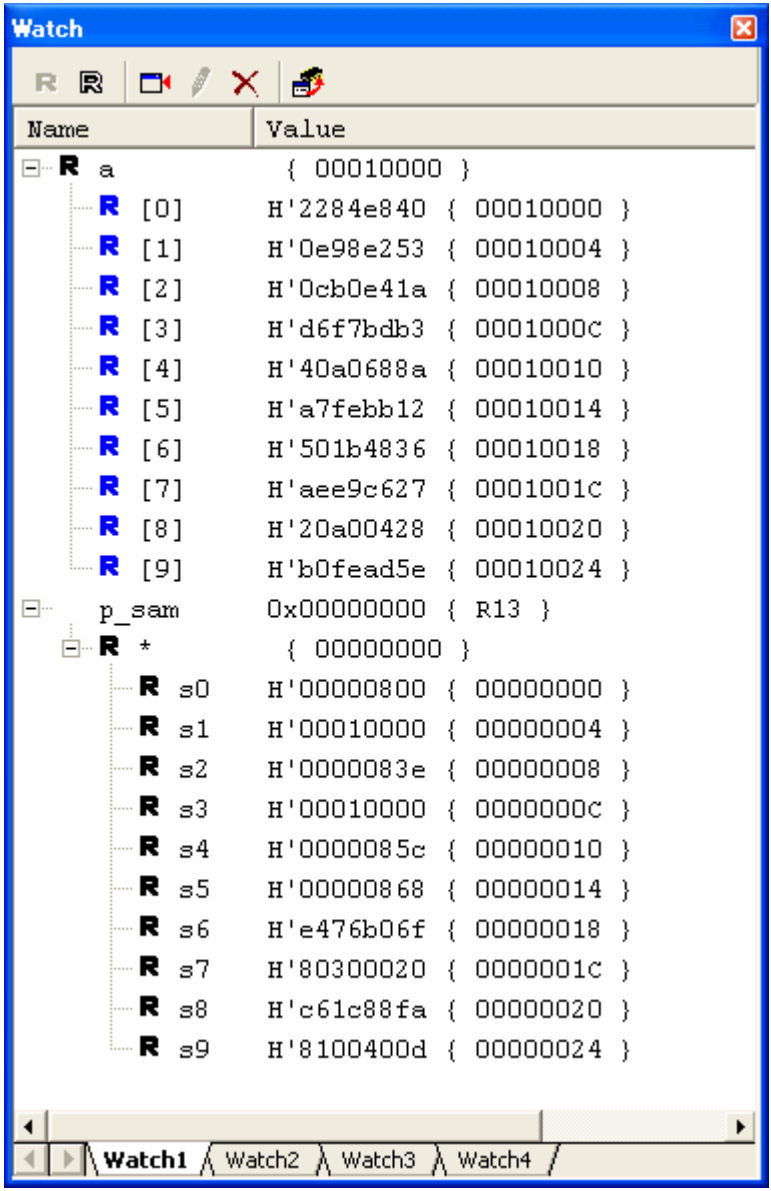


Figure 5.20 [Watch] Window

The [R] mark shows that the value of the variable can be updated during user program execution.

For reading of memory and updating of the content of the variable that has been registered in the [Watch] window during execution of the user program, there are following two methods:

1. Use the monitoring function without halting the user program

The value of the variable is updated by using the monitoring function of the emulator.

2. Temporarily stops the user program and reads the memory contents to update the value

Note: The realtime operation is disabled because the user program is stopped temporarily.

It is possible to recognize the method for updating the value during execution of the user program according to the color of the [R] mark.

Blue [R]: An updated value of the data at this location has been read by the monitoring function.

Black [R]: A value has been updated by reading the normal data.

Notes: 1. This function can be set per variable or per element or body for structures of data.
2. The information is lost when it is scrolled out of the [Watch] window and when the window is closed.
3. A variable that is allocated to a register cannot be selected for monitoring.
4. The realtime watch function is realized by using the AUD monitoring function. It is thus possible to monitor four areas in total by the AUD monitoring function and the realtime update function of the [Watch] window.

5.3.6 Hiding the [Monitor] Window

When using the Monitor function to monitor the value of a variable from the [Watch] window, hide the [Monitor] window for the effective use of the screen.

The current monitoring information is listed as the submenu when selecting [Display -> CPU -> Monitor]. The list consists of the [Monitor] window name and the address to start monitoring.

When the left of the list is checked, the [Monitor] window is being displayed.

Selecting items of the [Monitor] window you want to hide from the monitor setting list displays no [Monitor] window and removes the check mark at the left of the list.

To display the [Monitor] window again, select the hidden the [Monitor] window.

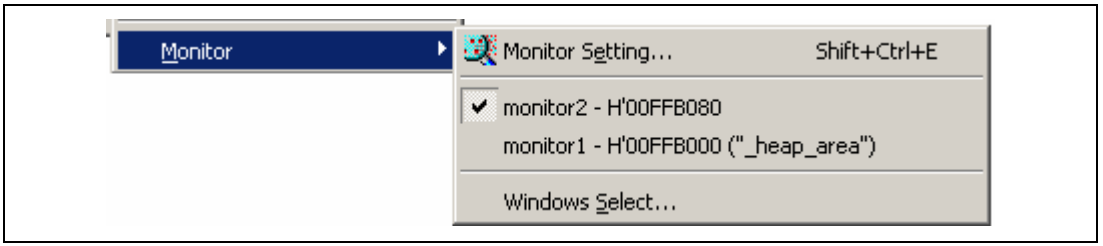


Figure 5.21 Monitor Setting List

5.3.7 Managing the [Monitor] Window

Selecting [Display -> CPU -> Monitor -> Windows Select…] displays the [Windows Select] dialog box. In this window, the current monitoring condition is checked and the new monitoring condition is added, edited, and deleted in succession.

Selecting multiple monitoring conditions enables a temporary stop of update, hiding, and deletion.

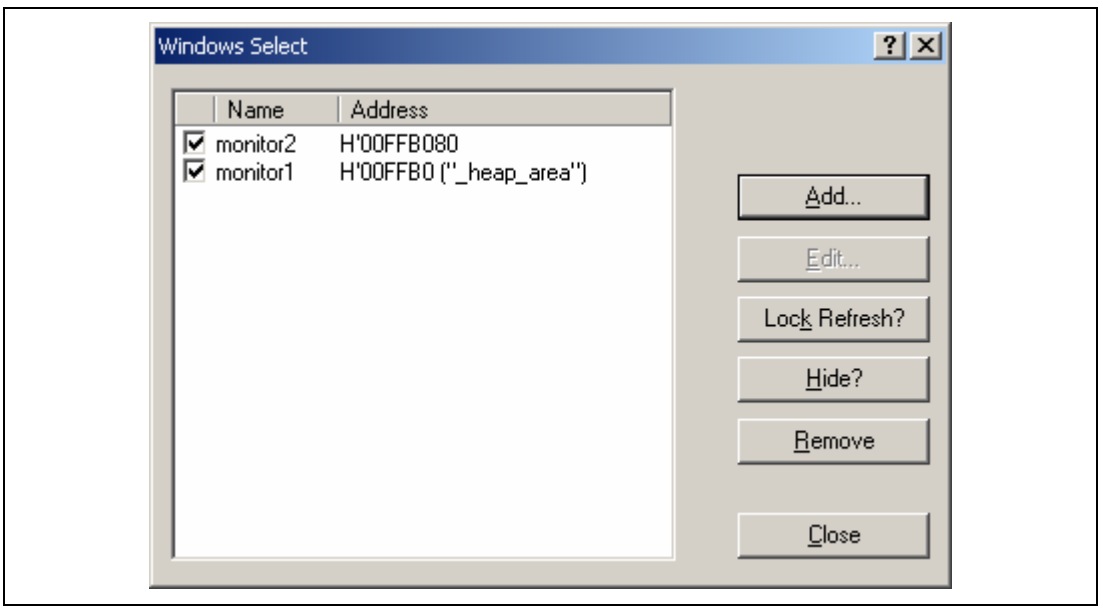


Figure 5.22 [Windows Select] Dialog Box

5.4 Viewing the Current Status

Choose [View -> CPU -> Status] or click the [View Status] toolbar button ( ) to open the [Status] window and see the current status of the debugging platform.

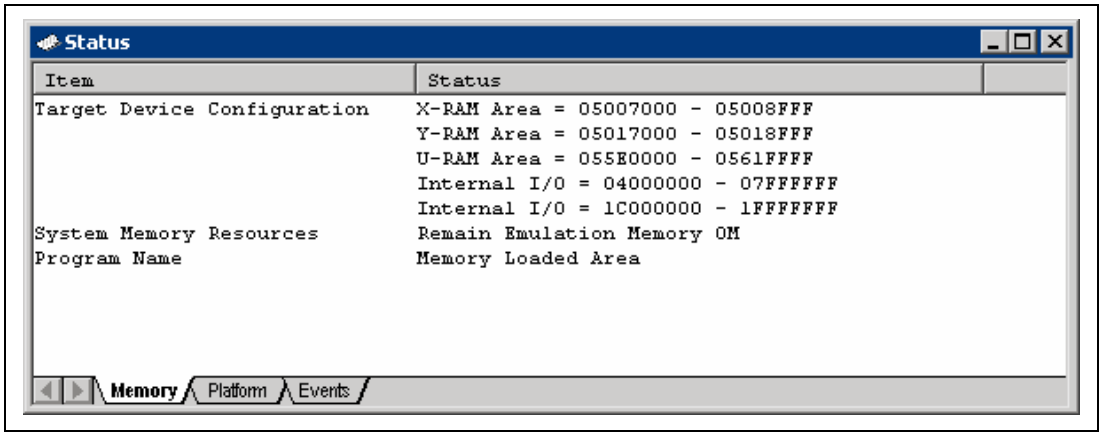


Figure 5.23 [Status] Window

The [Status] window has three sheets:

- [Memory] sheet

  Contains information about the current memory status including the memory-mapping resources and the areas used by the currently loaded object file. When the emulation memory function is used, the [System Memory Resources] item shows the number of bytes of the emulation memory that can be allocated.
- [Platform] sheet

  Contains information about the current status of the debugging platform, typically including CPU type and emulation mode, and the state of execution.
- [Events] sheet

  Contains information about the current event (breakpoint) status, including resource information.

Note: The items that can be set in this dialog box vary according to the emulator in use. For details, refer to the online help.

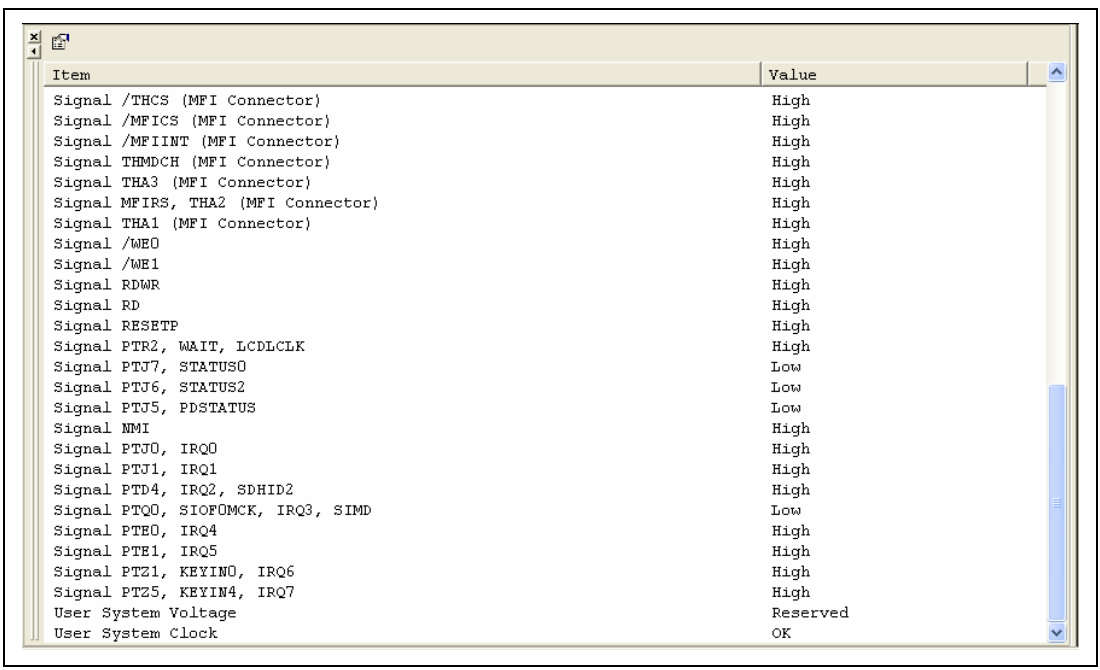
5.5 Reading and Displaying the Emulator Information Regularly

Use the [Extended Monitor] window to know the changing information on the emulator no matter the user program is running or halted.

Note: The Extended Monitor function does not affect the execution of the user program since it monitors the signal output from the user system or the MPU.

5.5.1 Opening the [Extended Monitor] Window

Selecting [View -> CPU -> Extended Monitor] or clicking the [Extended Monitor] toolbar button ( ) displays this window.



Item	Value
Signal /THCS (MFI Connector)	High
Signal /MFICS (MFI Connector)	High
Signal /MFIINT (MFI Connector)	High
Signal THMDCH (MFI Connector)	High
Signal THA3 (MFI Connector)	High
Signal MFIRS, THA2 (MFI Connector)	High
Signal THA1 (MFI Connector)	High
Signal /WE0	High
Signal /WE1	High
Signal RDWR	High
Signal RD	High
Signal RESETP	High
Signal PTR2, WAIT, LCDLCLK	High
Signal PTJ7, STATUS0	Low
Signal PTJ6, STATUS2	Low
Signal PTJ5, PDSTATUS	Low
Signal NMI	High
Signal PTJ0, IRQ0	High
Signal PTJ1, IRQ1	High
Signal PTD4, IRQ2, SDHID2	High
Signal PTQ0, SIOFOMCK, IRQ3, SIMD	Low
Signal PTE0, IRQ4	High
Signal PTE1, IRQ5	High
Signal PTZ1, KEYIN0, IRQ6	High
Signal PTZ5, KEYIN4, IRQ7	High
User System Voltage	Reserved
User System Clock	OK

Figure 5.24 [Extended Monitor] Window

Note: The items that can be set in this window vary according to the emulator in use. For details, refer to the online help.

5.5.2 Selecting Items to be Displayed

Selecting [Properties…] from the popup menu displays the [Extended Monitor Configuration] dialog box.

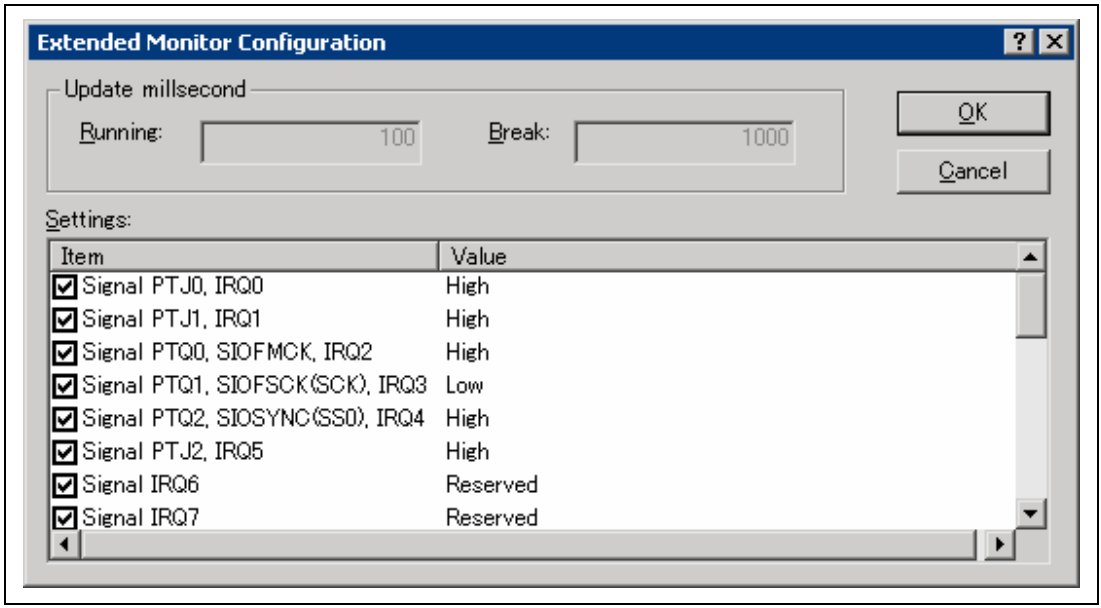


Figure 5.25 [Extended Monitor Configuration] Dialog Box

This dialog box allows the user to set the items to be displayed in the [Extended Monitor] window.

5.6 Using the Eventpoints

The emulator has the eventpoint function that performs breaking, tracing, and execution time measurement by specifying more complex conditions along with the PC breakpoints standard for the High-performance Embedded Workshop.

5.6.1 PC Breakpoints

When the instruction of the specified address is fetched, the user program is stopped. Up to 1000 points can be set.

5.6.2 Eventpoints

Eventpoints can be used for higher-level conditions such as the data condition as well as specification of the single address. Up to four eventpoints can be set in the emulator.

(1) Onchip Eventpoint

This is a function that sets eventpoints according to the information in the MPU. There are 13 event detection channels.

For an operation when an event is detected, break, internal trace acquisition/acquisition start/acquisition stop, or internal performance measurement start/end can be specified.

A combination of one or more of the Onchip Eventpoints enables specifying more complex sequential conditions.

This function can be set in the [Onchip Event] sheet of the [Event] window.

Note: The contents to be set will differ depending on the product. For details, refer to the on-line help for each product.

(2) AUD Eventpoint

This is a function that sets eventpoints according to the information output from the AUD interface. There are eight event detection channels.

For an operation when an event is detected, break, AUD trace acquisition/acquisition start/acquisition stop, or AUD performance measurement start/end can be specified.

A combination of one or more of the AUD evenpoints enables specifying more complex sequential conditions.

This function can be set in the [AUD Event] sheet of the [Event] window.

(3) BUS Eventpoint

This is a function that sets eventpoints according to the external bus of the MPU or information on the pin such as an interrupt pin. There are six event detection channels.

For an operation when an event is detected, break or external bus trace acquisition/acquisition start/acquisition stop can be specified.

A combination of one or more of the BUS evenpoints enables specifying more complex sequential conditions.

This function can be set in the [BUS Event] sheet of the [Event] window.

Notes: 1. When the trace unit is not connected to the emulator, this function is not supported.
2. When the function of the trace unit is changed, the function of event detection channels is changed. For details, refer to section 5.1.4, [Bus Board] Page.

(4) Other Eventpoint

This function can be set in the [Other Event] sheet of the [Event] window.

(a) Execution time eventpoint

Eventpoints can be defined specifying the program execution time as the condition. There is one event detection channel.

For an operation when an event is detected, break can be specified.

(b) External probe eventpoint

Eventpoints can be defined specifying four external probe signals via the probe cable as the condition. There is one event detection channel.

For an operation when an event is detected, break or AUD trace acquisition/acquisition start/acquisition stop can be specified.

(c) Other eventpoints

Event conditions can be set by some I/O analyzer functions.

5.6.3 Opening the [Event] Window

Select [View -> Code -> Eventpoints] or click the [Eventpoints] toolbar button () to open the [Event] window.

The [Event] window has the following five sheets:

[Breakpoint] sheet:	Displays the settings made for PC breakpoints. It is also possible to set, modify, and cancel PC breakpoints.
[Onchip Event] sheet:	Displays or sets the settings made for on-chip event channels.
[AUD Event] sheet:	Displays or sets the settings made for AUD event channels.
[BUS Event] sheet:	Displays or sets the settings made for external bus event channels.
[Other Event] sheet:	Displays or sets the settings made for other event channels.

5.6.4 Setting PC Breakpoints

It is possible to display, modify, and add PC breakpoints on the [Breakpoint] sheet.

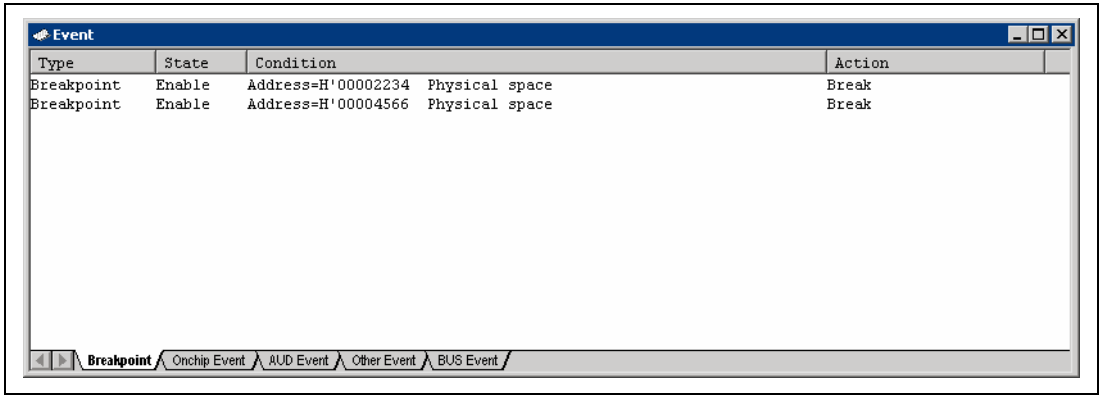


Figure 5.26 [Event] Window ([Breakpoint] Sheet)

Items that can be displayed in the sheet are listed below.

[Type] Breakpoint

[State] Whether the breakpoint is enabled or disabled

[Condition] An address that the breakpoint is set
Address = Program counter (Corresponding file name, line, and symbol name)

[Action] Operation of the emulator when a break condition is satisfied
Break: Breaks program execution

Select [Add...] or the PC breakpoint displayed in this window and then select [Edit...] from the popup menu to display the [Breakpoint] dialog box.

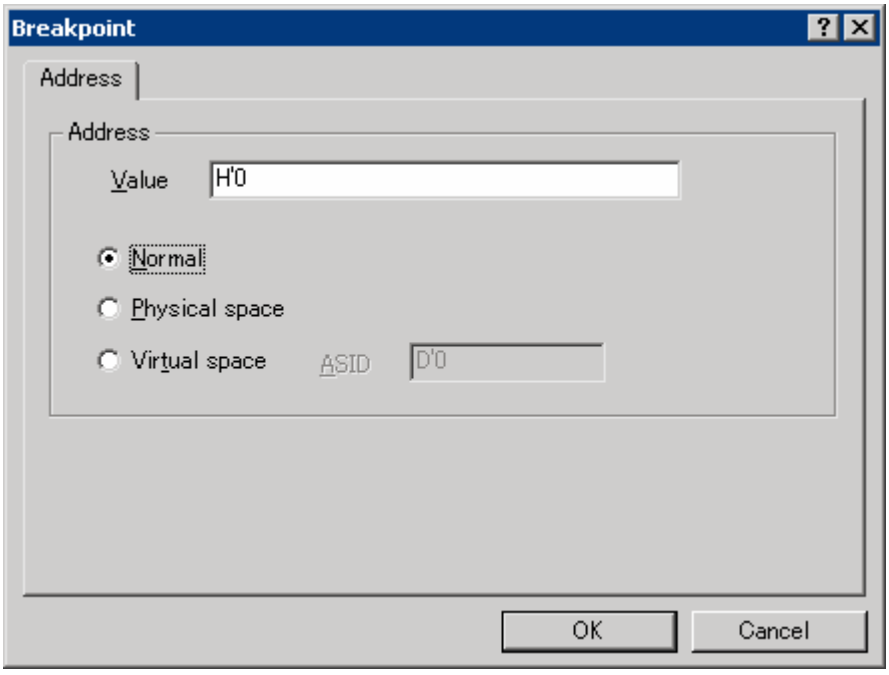


Figure 5.27 [Breakpoint] Dialog Box

This dialog box specifies address conditions of PC breakpoints.

A breakpoint address to be set is specified in the [Value] edit box. The PC register can also be specified such as #PC. Up to 1000 breakpoints can be specified.

The contents to be set differ depending on the product. For details, refer to the on-line help for each product.

When [Value] is selected, if an overloaded function or class name including a member function is specified in address, the [Select Function] dialog box opens.

Clicking the [OK] button sets the specified breakpoint conditions. Clicking the [Cancel] button closes this dialog box without setting the break conditions.

5.6.5 Setting Onchip Eventpoints

On the [Onchip Event] sheet, the settings for Onchip Eventpoints are displayed and modified.

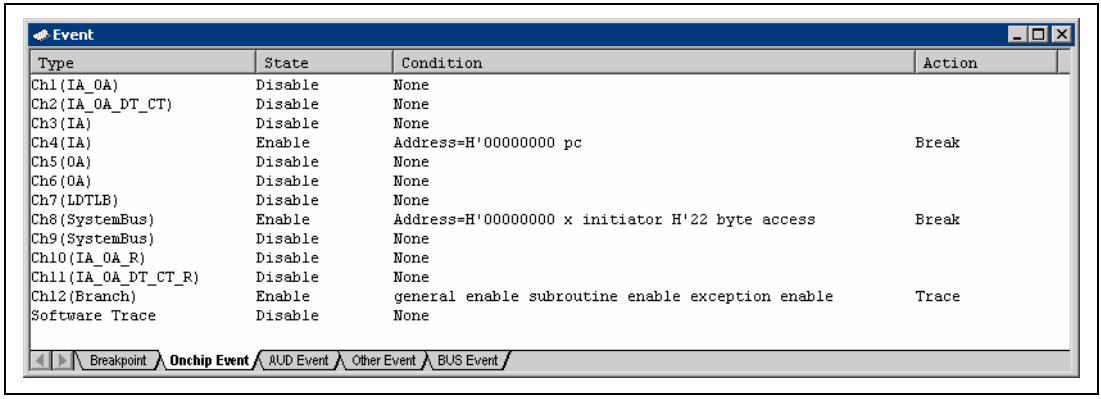


Figure 5.28 [Event] Window ([Onchip Event] Sheet)

Since the number of event detection channels and the contents to be set differ depending on the product, refer to the online help for each product.

Items that can be displayed in the sheet are listed below.

[Type] On-chip event channel number and type

[State] Whether the eventpoint is enabled or disabled

[Condition] A condition that satisfies an eventpoint. The displayed contents differ depending on the channel.

[Action] Operation of the emulator when an eventpoint condition is satisfied. The displayed contents differ depending on the channel.

When an event channel is double-clicked or selected and [Edit...] is selected from the popup menu in this window, the [Event condition x] dialog box is opened and eventpoint conditions can be modified.

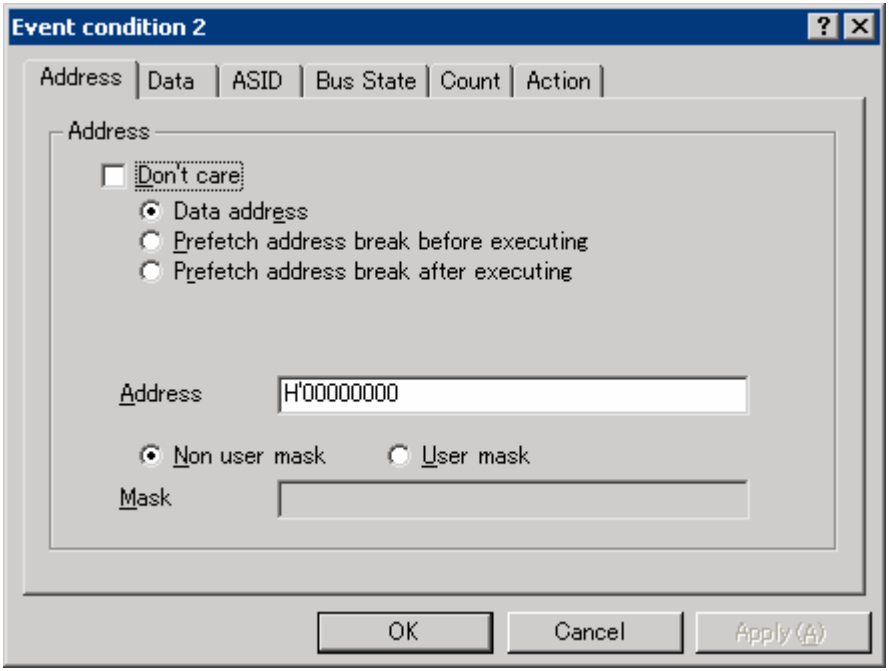


Figure 5.29 [Event condition x] Dialog Box ([Address] Page)

Note: For details on the [Event condition x] dialog box, refer to the online help for each product.

Table 5.1 lists the conditions of Event Condition.

Table 5.1 Types of Event Conditions

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

Table 5.2 lists the combination of conditions that can be set by Ch1 to Ch12 and software trace channels.

Table 5.2 Channels for Setting Event Conditions

	Function										
Channel	Address Bus Condition (Address)	Data Bus Condition (Data)	ASID Condition (ASID)	Bus State Condition (Bus Status)	Window Address Condition (Window address)	System Bus	LDTLB Instruction Break	Count Condition (Count)	Branch Condition (Branch Trace)	Software Trace	Action
Ch1 (IA_OA)	O	X	O	O	X	X	X	X	X	X	O (B and P)
Ch2 (IA_OA_ DT_CT)	O	O	O	O	X	X	X	O	X	X	O (B and P)
Ch3 (IA)	O	X	O	X	X	X	X	X	X	X	O (B and P)
Ch4 (IA)	O	X	O	X	X	X	X	X	X	X	O (B and P)
Ch5 (OA)	X	X	O	O	O	X	X	X	X	X	O (B, T, and P)
Ch6 (OA)	X	X	O	O	O	X	X	X	X	X	O (B, T, and P)
Ch7 (LDTLB)	X	X	X	X	X	X	O	X	X	X	Break fixed
Ch8 (System-Bus)	O	X	X	X	X	O	X	X	X	X	O (B, T, and P)
Ch9 (System-Bus)	O	X	X	X	X	O	X	X	X	X	O (B, T, and P)
Ch10 (IA_OA_ R)	O	X	O	O	X	X	X	X	X	X	O (B and P)
Ch11 (IA_OA_ DT_CT_ R)	O	O	O	O	X	X	X	O	X	X	O (B and P)

Table 5.2 Channels for Setting Event Conditions (cont)

	Function										
Channel	Address Bus Condition (Address)	Data Bus Condition (Data)	ASID Condition (ASID)	Bus State Condition (Bus Status)	Window Address Condition (Window address)	System Bus	LDTLB Instruction Break	Count Condition (Count)	Branch Condition (Branch Trace)	Software Trace	Action
Ch12 (Branch)	X	X	X	X	X	X	X	X	O	X	O (B, T, and P)
Software Trace	X	X	X	X	X	X	X	X	X	O	Trace fixed

Notes: 1. O: Can be set in the dialog box.
X: Cannot be set in the dialog box.

2. For the Action item,
B: Setting a break is enabled.
T: Setting a trace is enabled.
P: Setting a performance start or end condition is enabled.

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

Table 5.3 Sequential Event Conditions

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

Table 5.3 Sequential Event Conditions (cont)

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

Sequential Break Extension Setting:

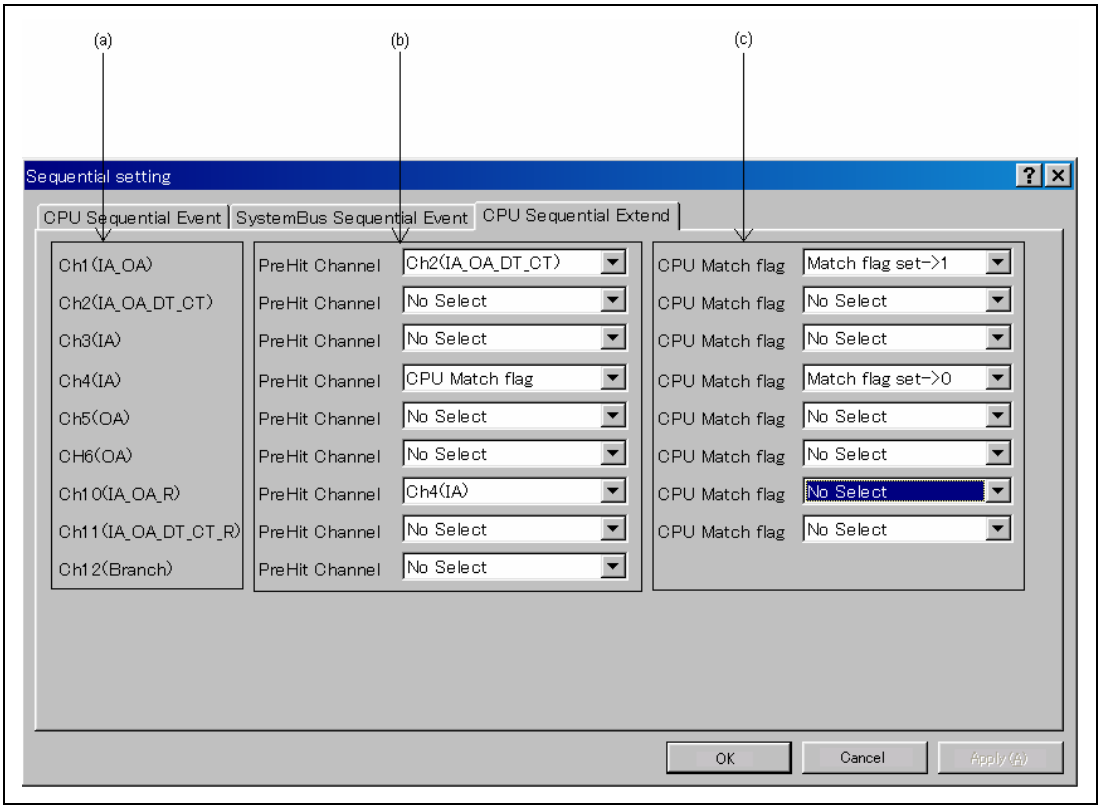


Figure 5.30 [CPU Sequential Extend] Page

(a) Indicates the channel name for setting conditions.

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

(c) After conditions have been matched, the CPU match flag is set or cleared.
When a program breaks, the CPU match flag is initialized.

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

Usage Example of Sequential Break Extension Setting: A tutorial program provided for the product is used as an example. For the tutorial program, refer to section 6, Tutorial, in the SuperH™ Family E200F Emulator User's Manual.

The conditions of Event Condition are set as follows:

1. Ch1

   Breaks address H'00001084 when the condition [Prefetch address break after executing] is satisfied.
2. Ch2

   Breaks address H'0000106c when the condition [Prefetch address break after executing] is satisfied.
3. Ch4

   Breaks address H'000010aa when the condition [Prefetch address break after executing] is satisfied.
4. Ch10

   Breaks address H'000010e0 when the condition [Prefetch address break after executing] is satisfied.
   Note: Do not set other channels.
5. Set the [CPU Sequential Extend] page as shown in figure 5.30.

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

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

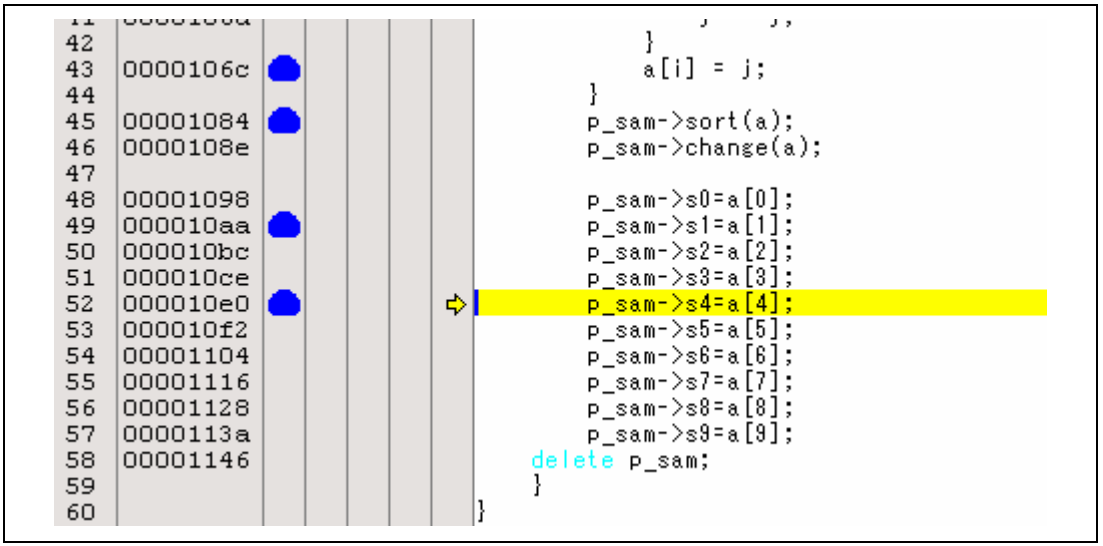


Figure 5.31 [Source] Window at Execution Halted (Sequential Break)

5.6.6 Setting AUD Eventpoints

On the [AUD Event] sheet, the settings for AUD Eventpoints are displayed and modified.

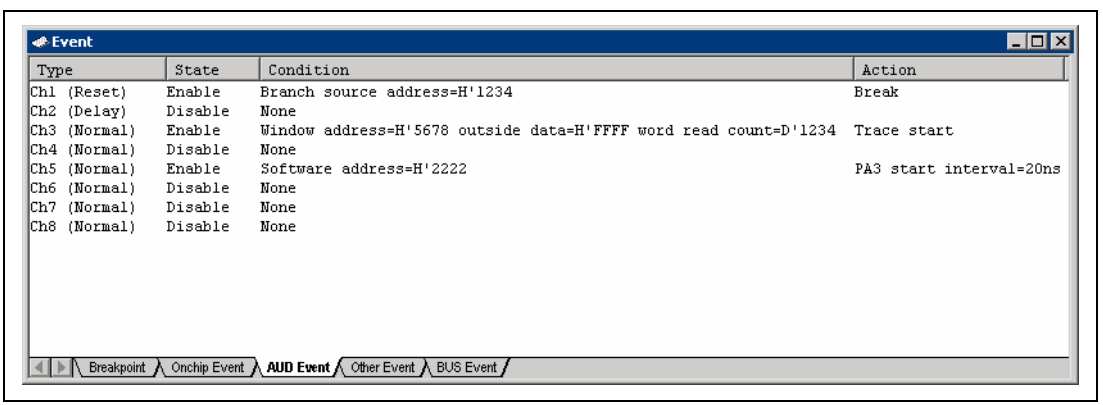


Figure 5.32 [Event] Window ([AUD Event] Sheet)

Using eight event detection channels sets eight eventpoints.

Note: Since the AUD eventpoint condition is set according to the information in the MPU output from the AUD pin, the trace acquisition condition for the Onchip Eventpoint must be set.

Set trace acquisition conditions for on-chip event channels Ch5 (OA), Ch6 (OA), Ch7 (SystemBus), Ch8 (SystemBus), and Ch12 (Branch) or software trace.

Items that can be displayed in the sheet are listed below.

[Type] AUD event channel number and type
Normal: Standard event channel
Delay: Event channel that delay conditions can be set
Reset: Event channel that can be set as the reset point of the AUD sequential event

[State] Whether the eventpoint is enabled or disabled
Enable: Valid
Disable: Invalid

[Condition] A condition that satisfies an eventpoint. The displayed contents differ depending on the channel.

[Action] Operation of the emulator when an eventpoint condition is satisfied. The displayed contents differ depending on the channel.

When an event channel is double-clicked or selected and [Edit...] is selected from the popup menu in this window, the [Chx] dialog box is displayed.

The [Chx] dialog box consists of the [General], [Branch], [Window], [Software], [SystemBus], [Count], [Delay], and [Action] pages.

The combination of the conditions set in each page is set as the detection condition of eventpoints.

(1) [General] page

Specifies AUD trace information used to set the AUD Eventpoint condition.

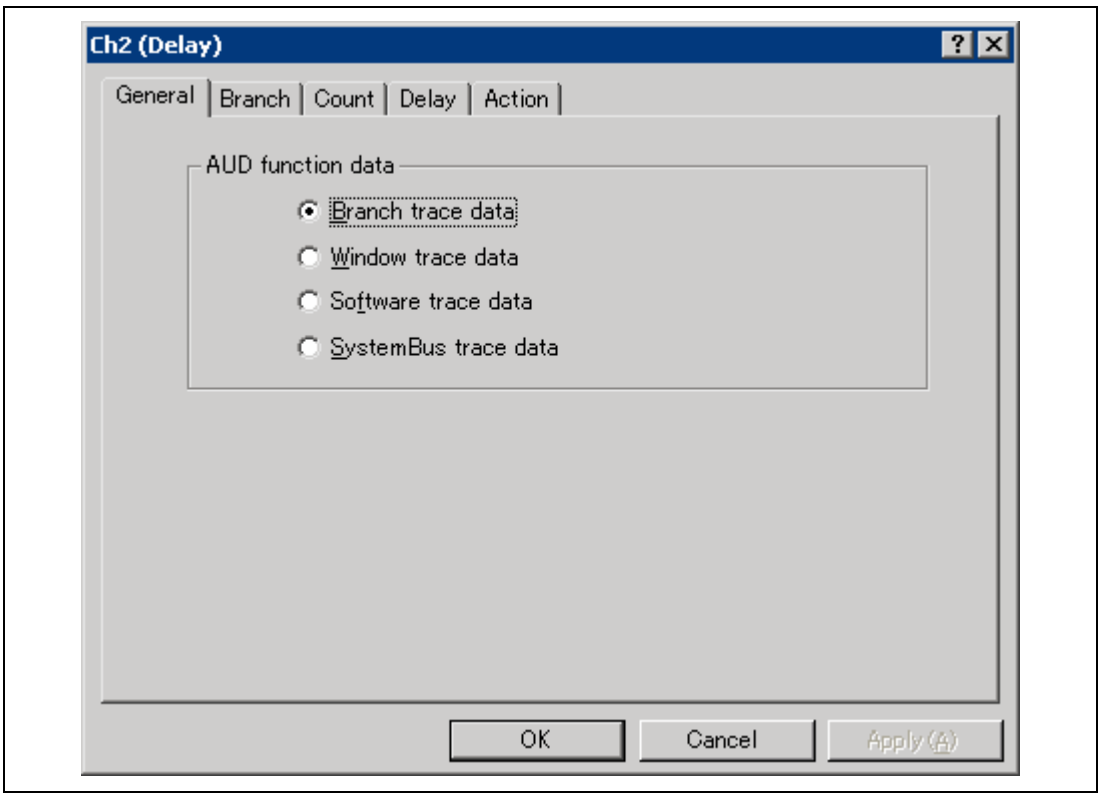


Figure 5.33 [Chx (Delay)] Dialog Box ([General] Page)

[Branch trace data]: Sets the eventpoint condition according to the branch trace information.

[Window trace data]: Sets the eventpoint condition according to the window trace information.

[Software trace data]: Sets the eventpoint condition according to the software trace information.

[SystemBus trace data]: Sets the eventpoint condition according to the system bus trace information.

(2) [Branch] page

Specifies the type and address conditions of the branch trace.

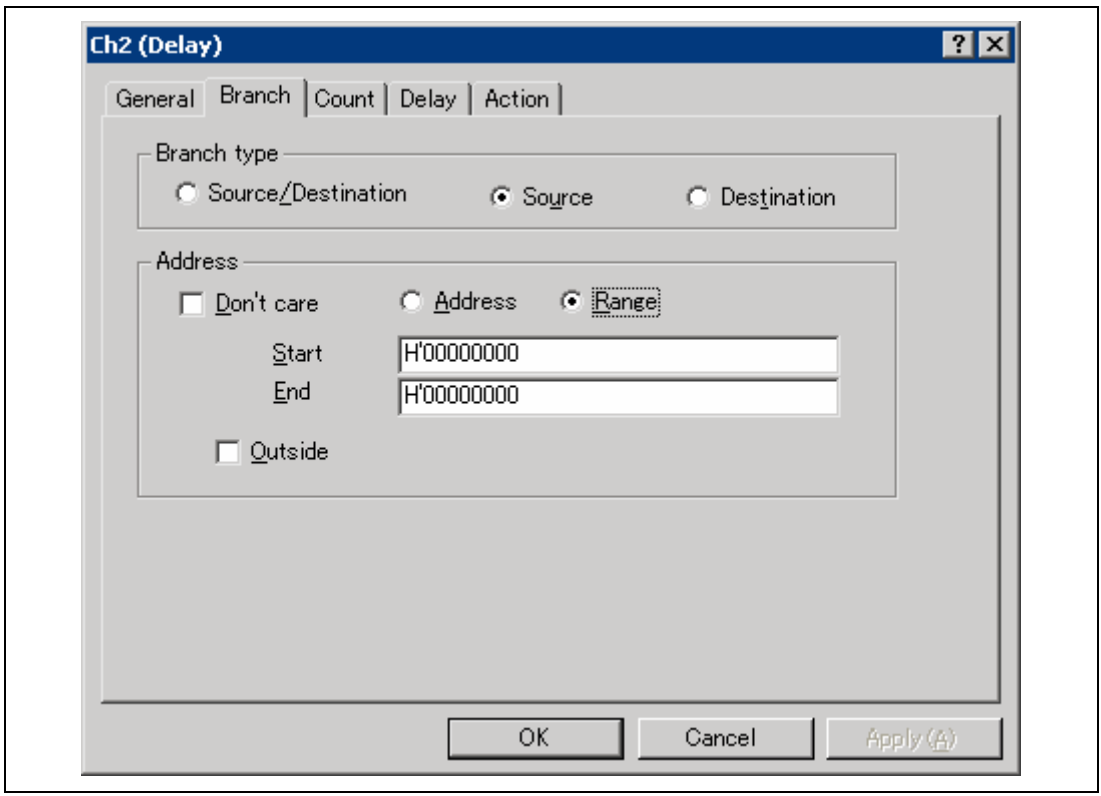


Figure 5.34 [Chx (Delay)] Dialog Box ([Branch] Page)

[Branch type]: Sets branch types.

- [Source/Destination]: Sets no branch type.
- [Source]: Sets the branch-source address condition.
- [Destination]: Sets the branch-destination address condition.

[Address]: Sets address conditions.

- [Don’t care]: Sets no address condition.
- [Address]: Sets the single address.
- [Range]: Sets the address range.

[Start]: Sets the single address or the start address of the address range.

[End]: Sets the end address of the address range.

[Outside]: Sets other value than that has been set for the single address or address range as the condition.

Notes: 1. This page is only displayed when [Branch trace data] is specified on the [General] page.
2. The address range can only be specified for AUD event channels Ch1 and Ch2.
3. The mask address can be input for [Start] when the single address is specified.
4. The mask address cannot be input for [Start] and [End] when the address range is specified.

(3) [Window] page

Specifies the address and data conditions of the window trace.

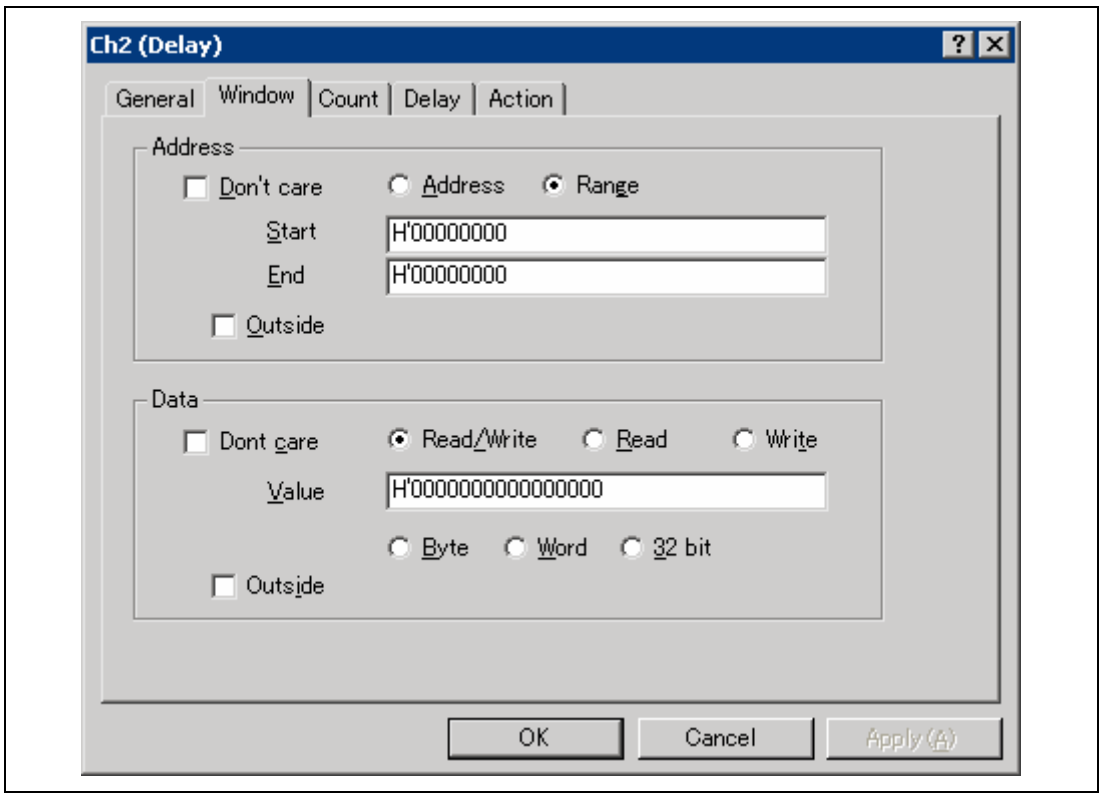


Figure 5.35 [Chx (Delay)] Dialog Box ([Window] Page)

[Address]: Sets the address condition.

- [Don’t care]: Sets no address condition.
- [Address]: Sets the single address.
- [Range]: Sets the address range.
- [Start]: Sets the single address or the start address of the address range.
- [End]: Sets the end address of the address range.
- [Outside]: Sets other value than that has been set for the single address or address range as the condition.

[Data]: Sets the data condition.

[Don’t care]: Sets no data condition.

[Read/Write]: Sets the read or write cycle as the condition.

[Read]: Sets the read cycle as the condition.

[Write]: Sets the write cycle as the condition.

[Value]: Sets the data bus value (mask data can be input).

[Byte]: Sets the byte access as the condition.

[Word]: Sets the word access as the condition.

[32 bit]: Sets the 32-bit access as the condition.

[Outside]: Sets other value than that has been set for [Value] as the condition.

Notes:
1. This page is only displayed when [Window trace data] is specified on the [General] page.
2. The address range can only be specified for AUD event channels Ch1 and Ch2.
3. The mask address can be input for [Start] when the single address is specified.
4. The mask address cannot be input for [Start] and [End] when the address range is specified.

(4) [Software] page

Specifies the address and data conditions of the software trace.

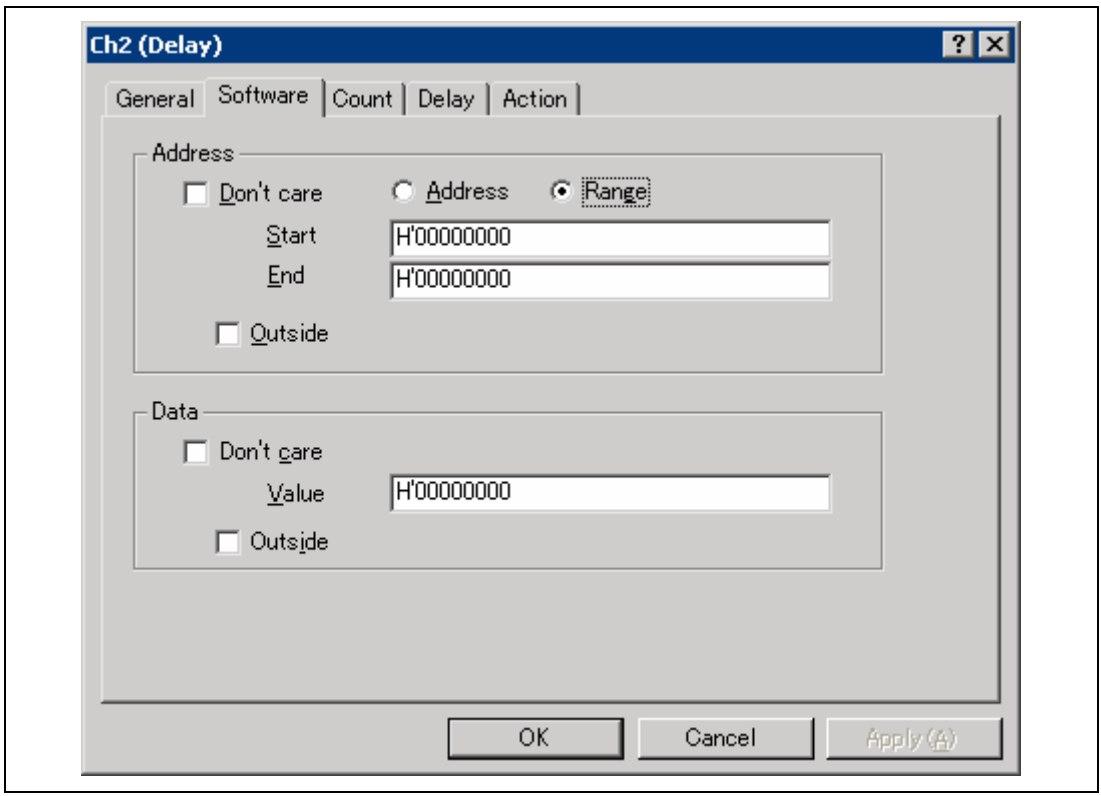


Figure 5.36 [Chx (Delay)] Dialog Box ([Software] Page)

[Address]: Sets the address condition.

[Don't care]: Sets no address condition.

[Address]: Sets the single address.

[Range]: Sets the address range.

[Start]: Sets the single address or the start address of the address range.

[End]: Sets the end address of the address range.

[Outside]: Sets other value than that has been set for the single address or address range as the condition.

[Data]: Sets the data condition.

[Don’t care]: Sets no data condition.

[Value]: Sets the data bus value (mask data can be input).

[Outside]: Sets other value than that has been set for [Value] as the condition.

Notes: 1. This page is only displayed when [Software trace data] is specified on the [General] page.
2. The address range can only be specified for AUD event channels Ch1 and Ch2.
3. The mask address can be input for [Start] when the single address is specified.
4. The mask address cannot be input for [Start] and [End] when the address range is specified.

(5) [SystemBus] page

Specifies the address and data conditions of the system bus trace.

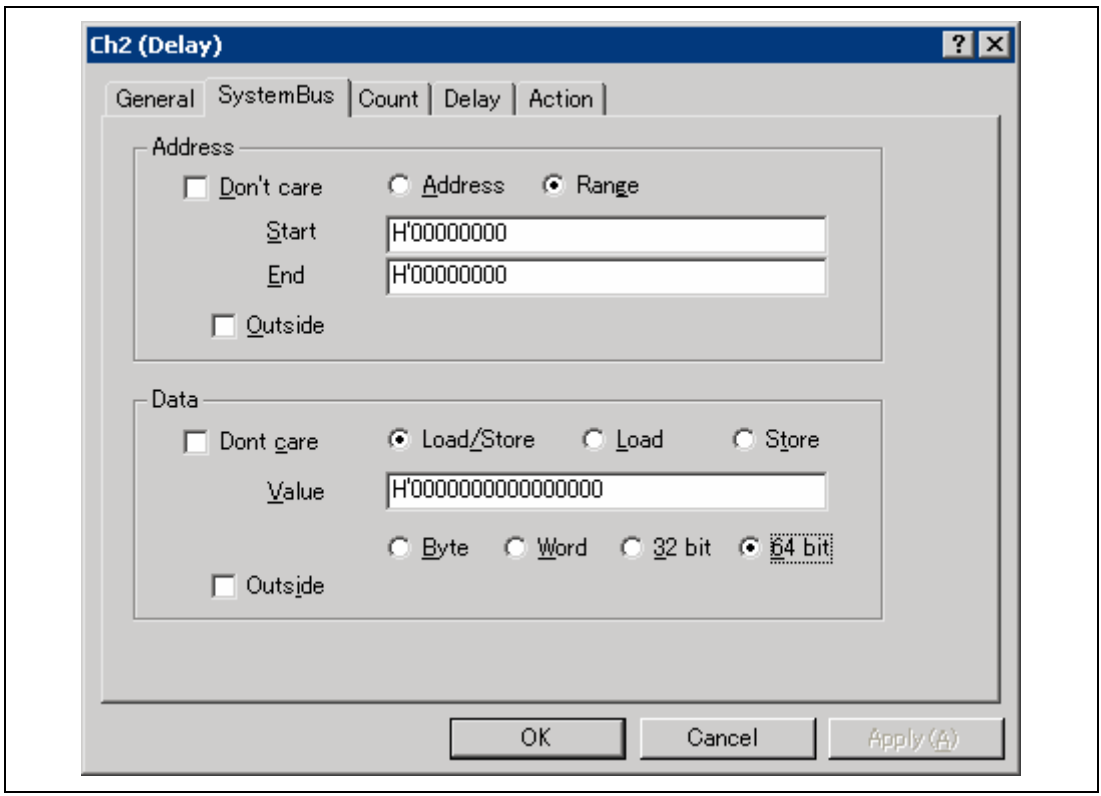


Figure 5.37 [Chx (Delay)] Dialog Box ([SystemBus] Page)

[Address]: Sets the address condition.

- [Don’t care]: Sets no address condition.
- [Address]: Sets the single address.
- [Range]: Sets the address range.
- [Start]: Sets the single address or the start address of the address range.
- [End]: Sets the end address of the address range.
- [Outside]: Sets other value than that has been set for the single address or address range as the condition.

[Data]: Sets the data condition.

[Don’t care]: Sets no data condition.

[Load/Store]: Sets the load or store cycle as the condition (not supported).

[Load]: Sets the load cycle as the condition (not supported).

[Store]: Sets the store cycle as the condition.

[Value]: Sets the data bus value (mask data can be input).

[Byte]: Sets the byte access as the condition.

[Word]: Sets the word access as the condition.

[32 bit]: Sets the 32-bit access as the condition.

[64 bit]: Sets the 64-bit access as the condition.

[Outside]: Sets other value than that has been set for [Value] as the condition.

Notes:
1. This page is only displayed when [SystemBus trace data] is specified on the [General] page.
2. The address range can only be specified for AUD event channels Ch1 and Ch2.
3. The mask address can be input for [Start] when the single address is specified.
4. The mask address cannot be input for [Start] and [End] when the address range is specified.
5. For the load cycle ([Load]), information on the data of the system bus is not output from the AUD pin. The data condition can be set only for the store cycle.

(6) [Count] page

Specifies the satisfaction count conditions.

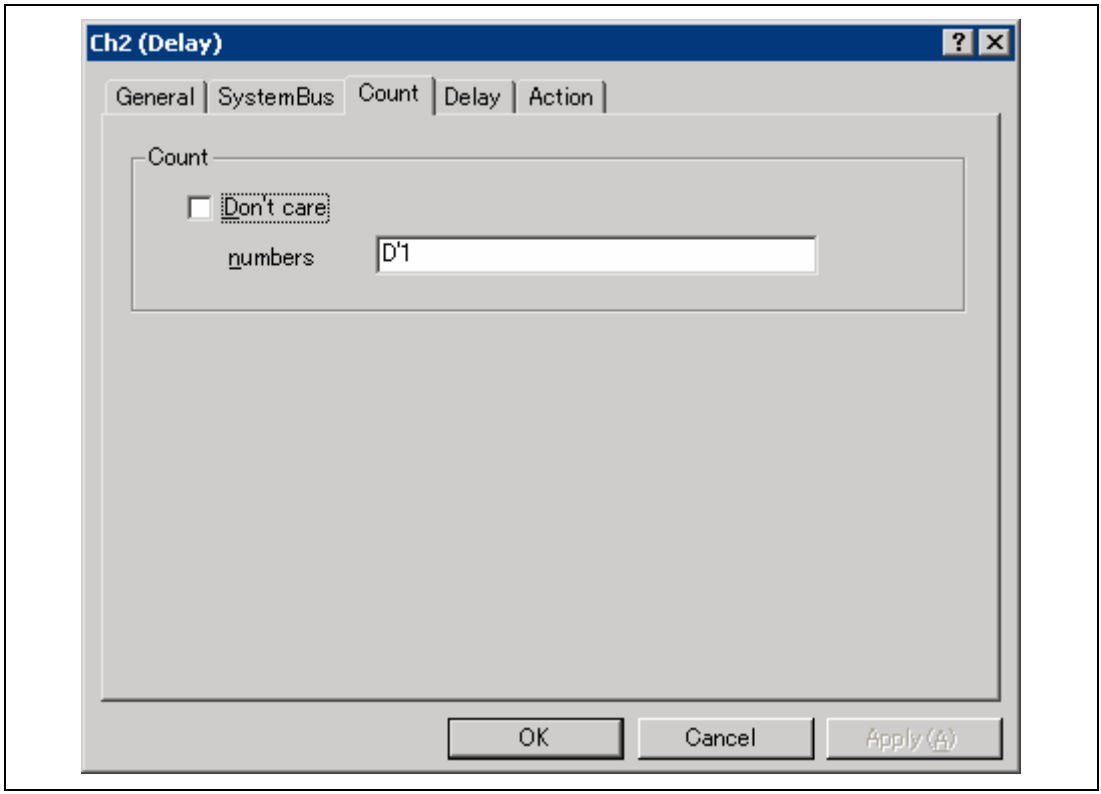


Figure 5.38 [Chx (Delay)] Dialog Box ([Count] Page)

[Don't care]: Sets no satisfaction count.

[numbers]: Sets a value as the satisfaction count condition; D'1 to D'65535 is available.

Note: If [Trace get] is selected on the [Action] page, this page will not be displayed.

(7) [Delay] page

Sets the delay cycle from the event detection to the AUD trace stop.

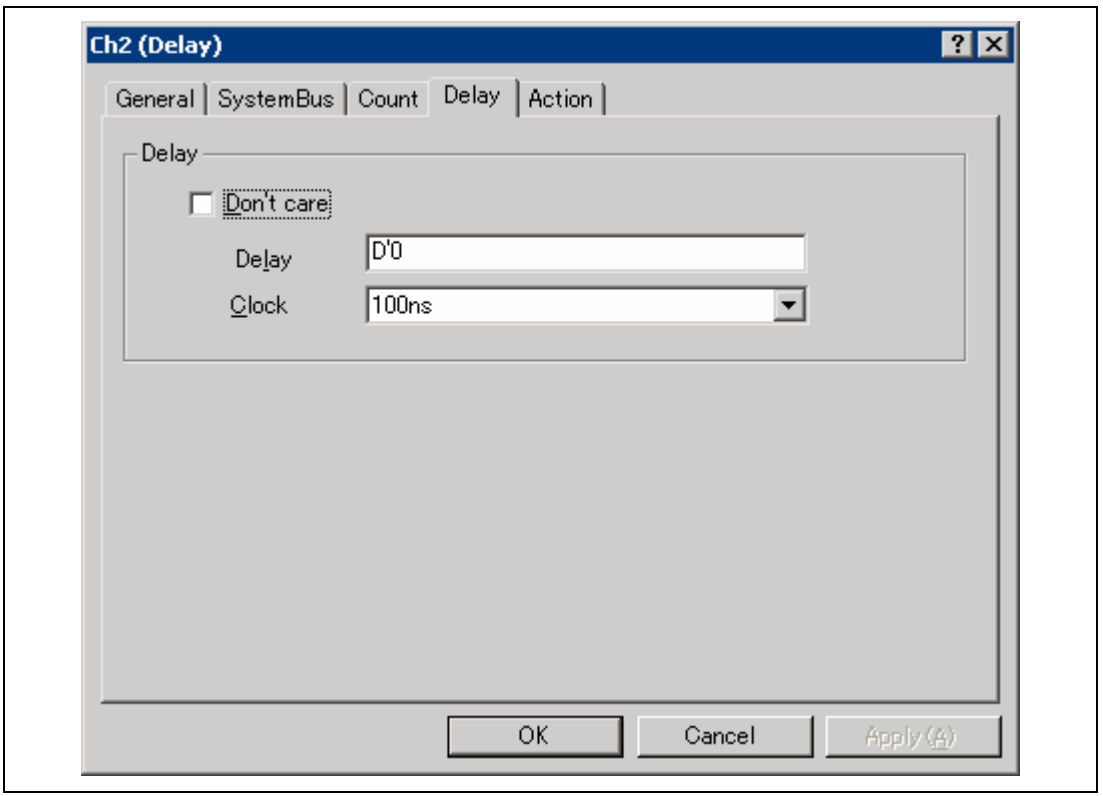


Figure 5.39 [Chx (Delay)] Dialog Box ([Delay] Page)

[Don't care]:	Sets no delay condition.	
[Delay]:	Sets the delay cycle counts; D'0 to D'262143 is available.	
[Clock]:	Sets a cycle for delay measurement.	
	[100ns]:	Specifies 100 ns as one cycle.
	[number of trace information]:	Specifies a set of AUD trace information as one cycle.

Notes: 1. This page is only displayed when AUD event channel Ch2 is selected and [Trace stop] is specified on the [Action] page.

2. When PPC information is output by the AUD trace, two sets of AUD trace information are specified as one cycle.

(8) [Action] page

Sets the operation after the condition has been satisfied.

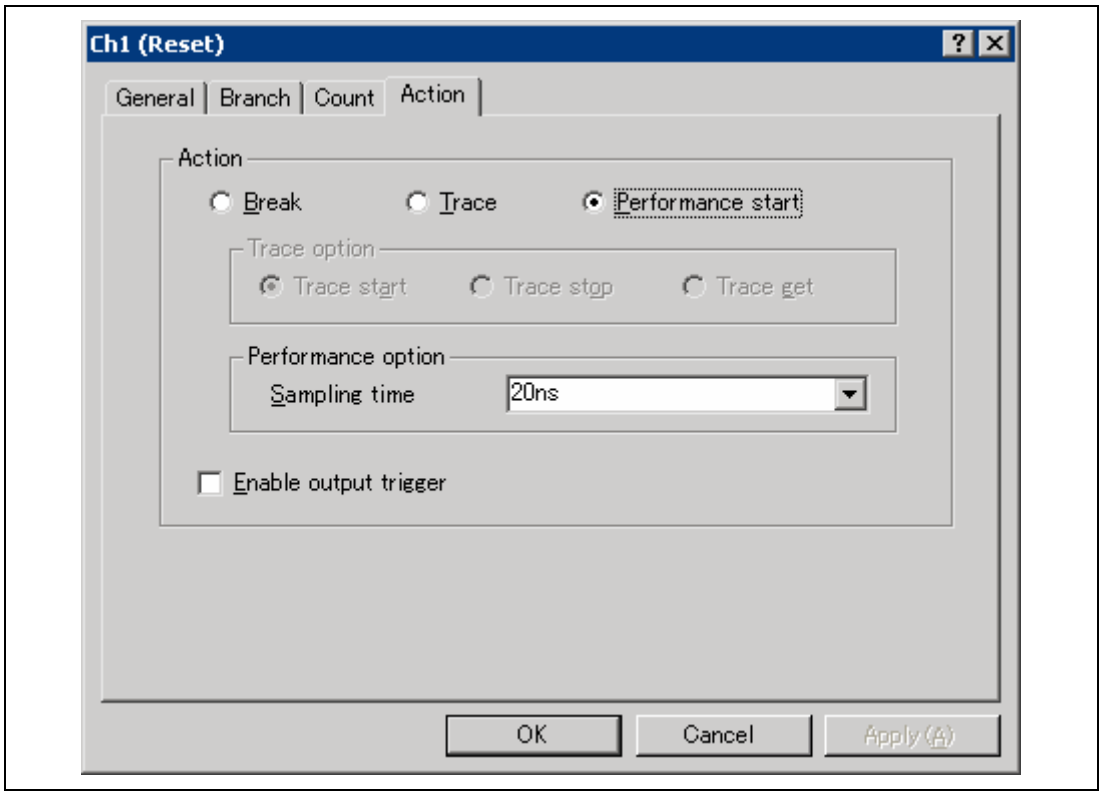


Figure 5.40 [Chx (Reset)] Dialog Box ([Action] Page)

[Break]: Breaks after conditions have been matched.

[Trace]: Enables [Trace option] and sets the AUD trace operation.
[Trace start]: Starts AUD trace after conditions have been matched.
[Trace stop]: Stops AUD trace after conditions have been matched.
[Trace get]: Acquires AUD trace after conditions have been matched.

[Performance start]: Starts or ends AUD performance measurement after conditions have been matched. When this option is selected, [Performance option] is enabled and the time interval of the performance measurement can be specified.
Sampling time: Specifies the time interval of the AUD

performance measurement as any of the following values:
20 ns, 40 ns, 100 ns, or 400 ns

[Enable output trigger]: Specifies whether or not the trigger is output after conditions have been matched.

Notes: 1. For AUD performance measurement, two AUD event channels are used to start and stop measurement. When an event channel is specified for performance measurement, the related channels (Ch1 to Ch2, Ch3 to Ch4, Ch5 to Ch6, and Ch7 to Ch8) must also be specified.

2. The [Performance option] is only displayed for AUD event channels Ch1, Ch3, Ch5, and Ch7.

(9) [Sequential AUD Event] dialog box

The sequential AUD event occurs when all the AUD Eventpoint conditions are satisfied in the specified order.

AUD event channel Ch1 can be specified as the reset point. When the reset point is passed, the satisfied eventpoint condition is disabled and checking the first eventpoint condition is started.

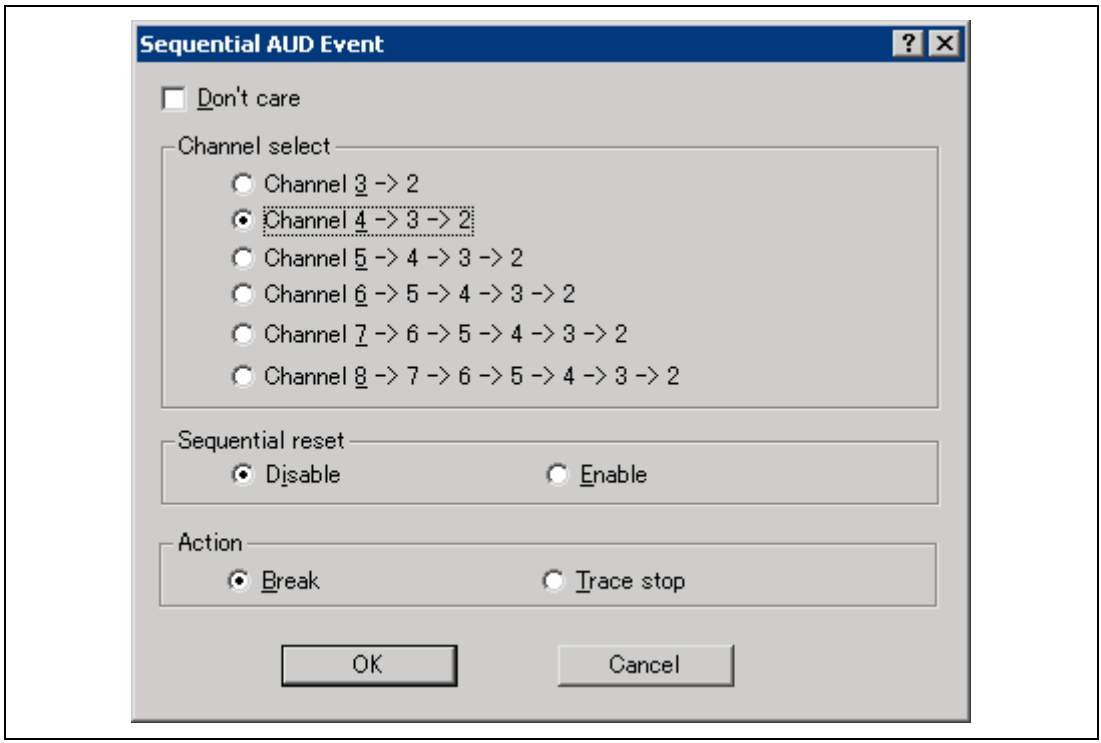


Figure 5.41 [Sequential AUD Event] Dialog Box

[Don't care]: Sets no sequential AUD event condition.

[Channel select]: Specifies the order that the sequential AUD event is satisfied.

[Channel 3 -> 2]:
After conditions have been matched in the order of AUD event channel 3 -> 2, a sequential AUD event occurs.

[Channel 4 -> 3 -> 2]:
After conditions have been matched in the order of AUD event channel 4 -> 3 -> 2, a sequential AUD event occurs.

[Channel 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of AUD event channel 5 -> 4 -> 3 -> 2, a sequential AUD event occurs.

[Channel 6 -> 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of AUD event channel 6 -> 5 -> 4 -> 3 -> 2, a sequential AUD event occurs.

[Channel 7 -> 6 -> 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of AUD event channel 7 -> 6 -> 5 -> 4 -> 3 -> 2, a sequential AUD event occurs.

[Channel 8 -> 7 -> 6 -> 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of AUD event channel 8 -> 7 -> 6 -> 5 -> 4 -> 3 -> 2, a sequential AUD event occurs.

[Sequential reset]: Selects whether or not AUD event channel Ch1 is used as the reset point.
[Disable]: Not used as the reset point.
[Enable]: Used as the reset point.

[Action]: Specifies the operation after the sequential AUD event has been detected.
[Break]: Breaks after the sequential AUD event has been detected.
[Trace stop]: Stops AUD trace after the sequential AUD event has been detected.

Note: When the sequential AUD event condition is set and the eventpoint condition is edited with the AUD event channel that has been selected for [Channel select], the [Action] page cannot be modified. To modify the settings of the [Action] page, cancel the sequential AUD event condition.

5.6.7 Setting BUS Eventpoints

On the [BUS Event] sheet, the settings for BUS Eventpoints are displayed and modified.

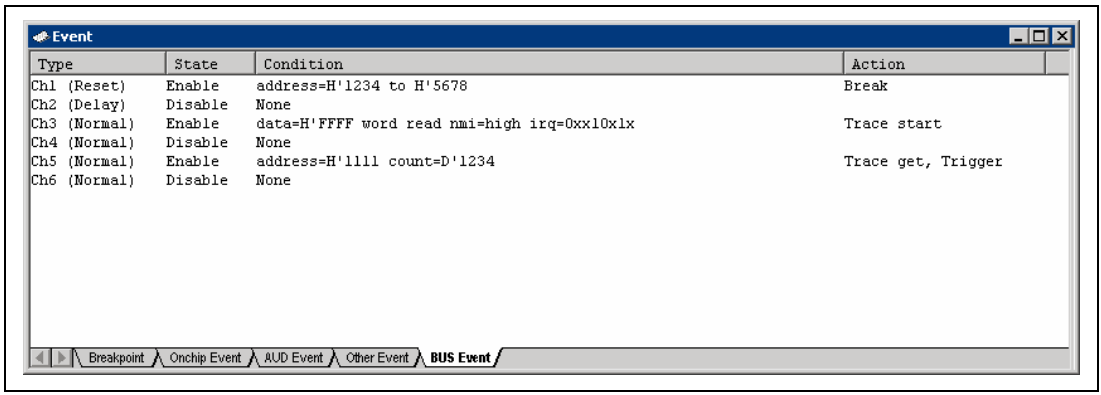


Figure 5.42 [Event] Window ([BUS Event] Sheet)

Using six event detection channels sets six eventpoints.

Notes: 1. When the trace unit is not connected to the emulator, this function is not supported.
2. When the debugging function of the trace unit is set again, the number of event detection channels is changed. For details, refer to section 5.1.4, [Bus Board] Page.

Items that can be displayed in the sheet are listed below.

[Type] External bus event channel number and type
Normal: Standard event channel
Delay: Event channel that delay conditions can be set
Reset: Event channel that can be set as the reset point of the external bus sequential event

[State] Whether the eventpoint is enabled or disabled
Enable: Valid
Disable: Invalid

[Condition] A condition that satisfies an eventpoint. The displayed contents differ depending on the channel.

[Action] Operation of the emulator when an eventpoint condition is satisfied. The displayed contents differ depending on the channel.

When an event channel is double-clicked or selected and [Edit...] is selected from the popup menu in this window, the [Chx] dialog box is displayed.

The [Chx] dialog box consists of the [Address], [Data], [Interrupt], [Count], [Delay], and [Action] pages.

The combination of the conditions set in each page is set as the detection condition of eventpoints.

(1) [Address] page

Specifies the address condition.

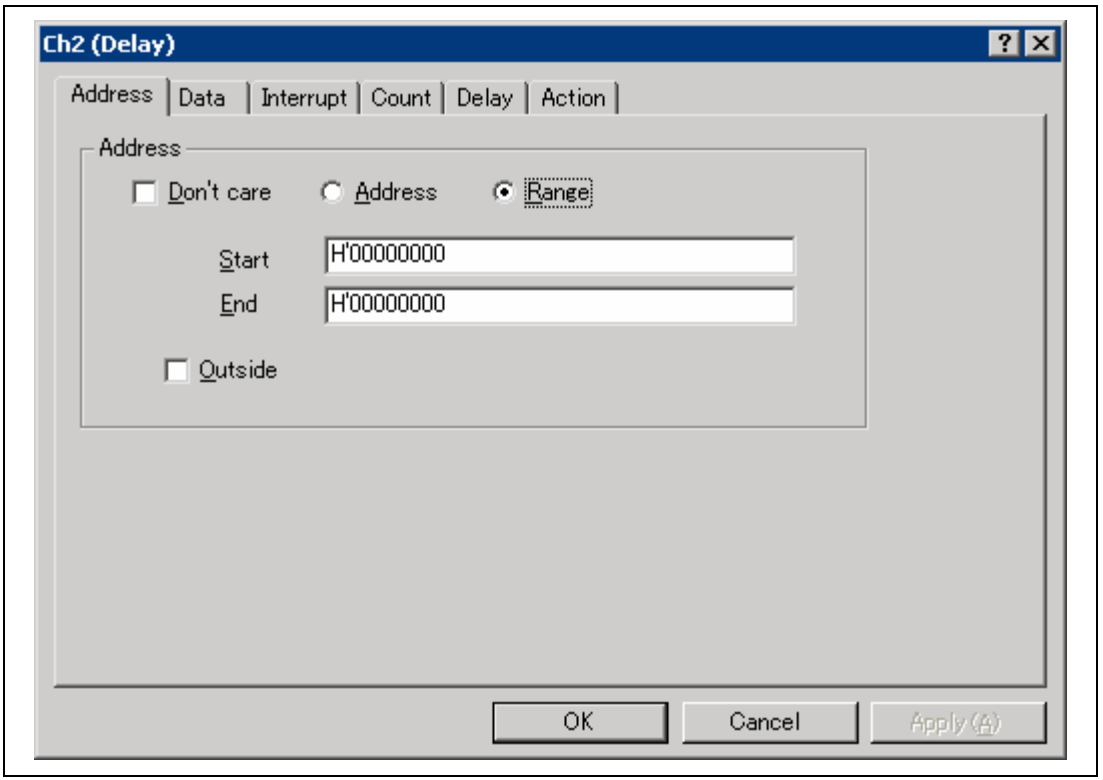


Figure 5.43 [Chx (Delay)] Dialog Box ([Address] Page)

[Address]: Sets the address condition.

[Don't care]: Sets no address condition.

[Address]: Sets the single address.

[Range]: Sets the address range.

[Start]: Sets the single address or the start address of the address range (mask address can be input).

[End]: Sets the end address of the address range (mask address can be input).

[Outside]: Sets other value than that has been set for the single address or address range as the condition.

(2) [Data] page

Specifies the data bus condition.

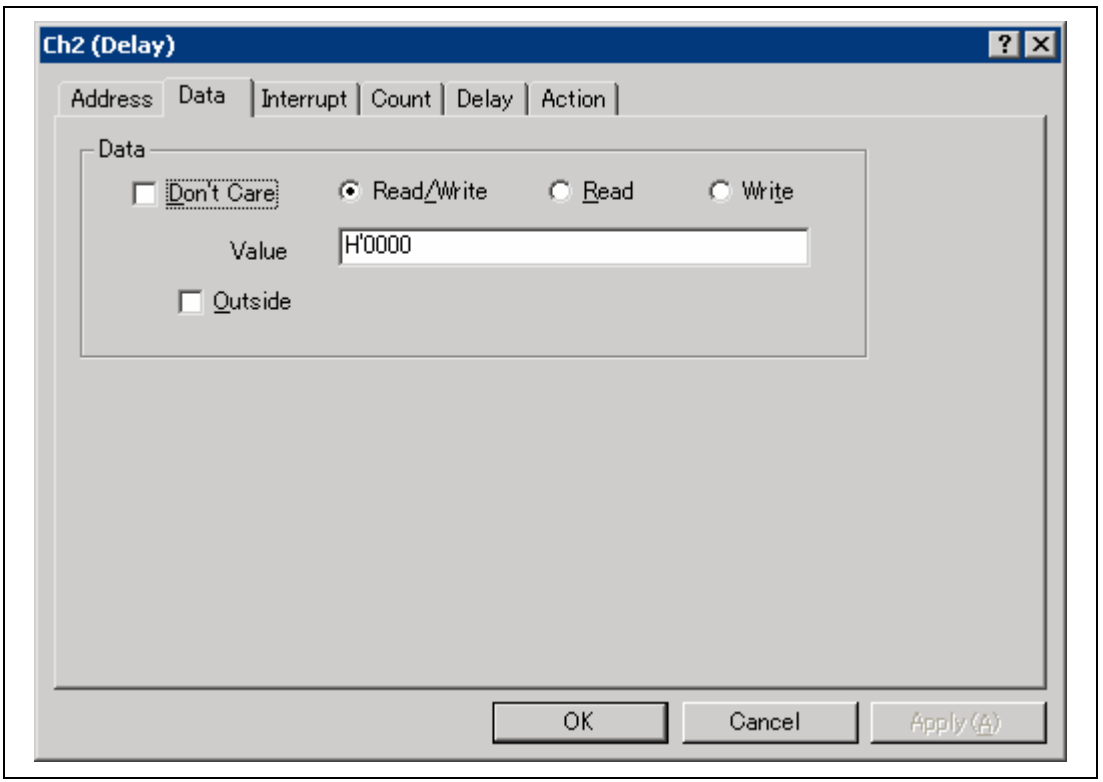


Figure 5.44 [Chx (Delay)] Dialog Box ([Data] Page)

[Data]: Sets the data condition.

[Don't care]: Sets no data condition.
[Read/Write]: Sets the read or write cycle as the condition.
[Read]: Sets the read cycle as the condition.
[Write]: Sets the write cycle as the condition.
[Value]: Sets the data bus value (mask data can be input).
[Outside]: Sets other value than that has been set for [Value] as the condition.

(3) [Interrupt] page

Specifies the signal conditions of NMI and external interrupts.

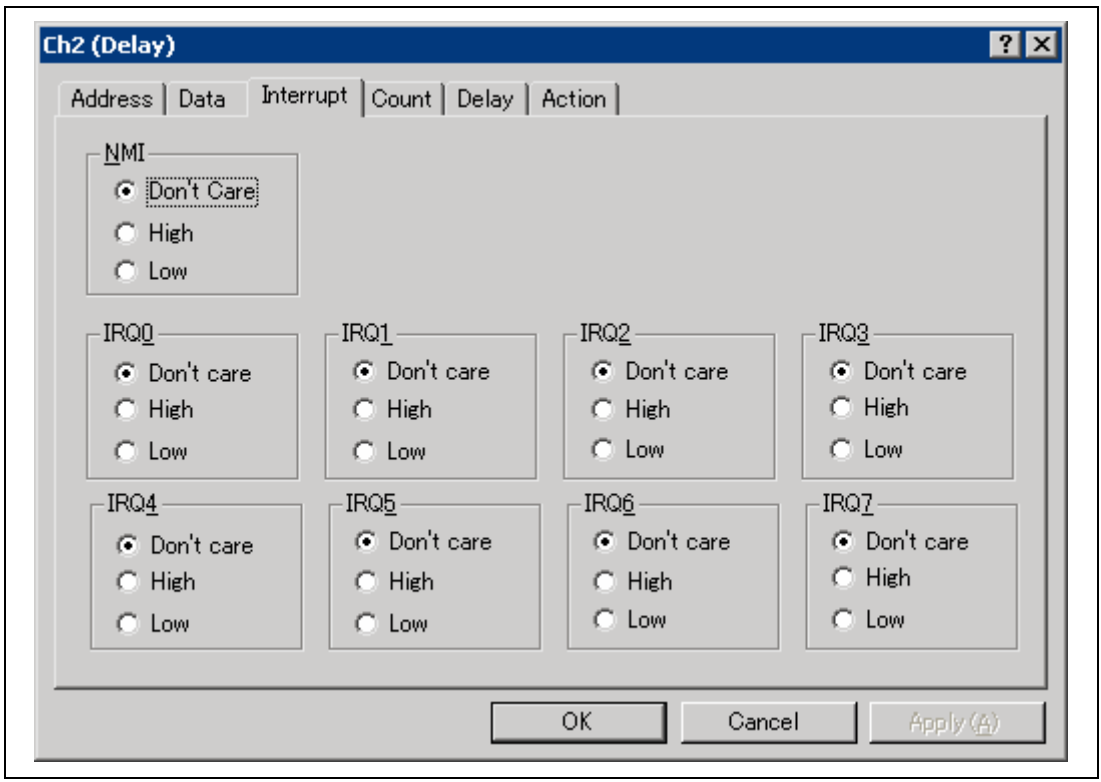


Figure 5.45 [Chx (Delay)] Dialog Box ([Interrupt] Page)

[Don’t care]: Sets no signal condition.

[High]: Satisfies a condition when the signal is high.

[Low]: Satisfies a condition when the signal is low.

Note: The external interrupt signal differs depending on the supported MPU. For details, refer to the online help.

(4) [Count] page

Specifies the satisfaction count condition.

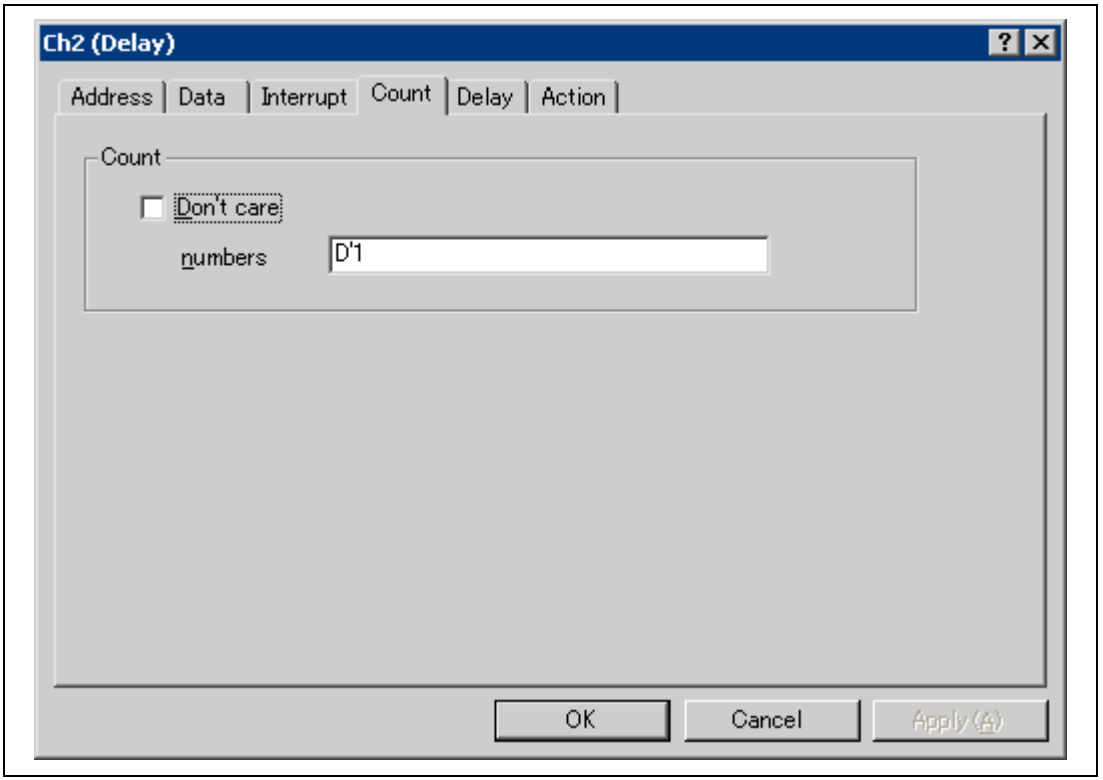


Figure 5.46 [Chx (Delay)] Dialog Box ([Count] Page)

[Don’t care]: Sets no satisfaction count.

[numbers]: Sets a value as the satisfaction count condition; D’1 to D’65535 is available.

Note: If [Trace get] is selected on the [Action] page, this page will not be displayed.

(5) [Delay] page

Sets the delay cycle from the event detection to the external bus trace stop.

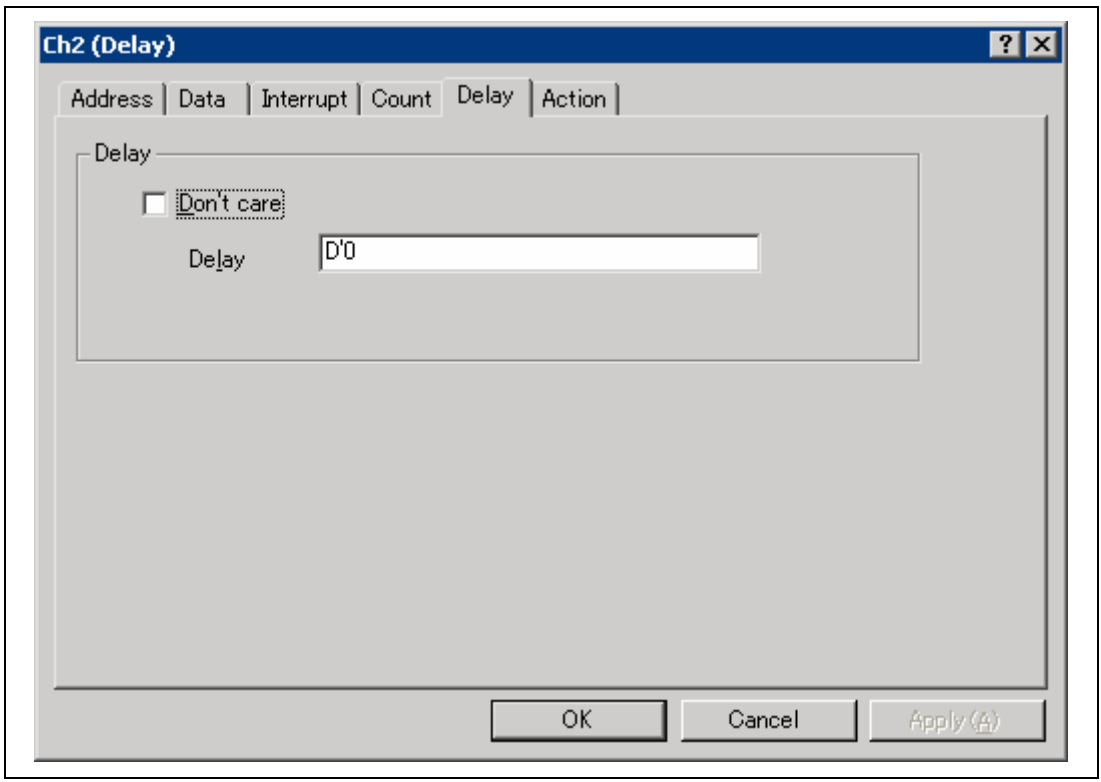


Figure 5.47 [Chx (Delay)] Dialog Box ([Delay] Page)

[Don't care]: Sets no delay condition.

[Delay]: Sets the delay cycle counts; D'0 to D'262143 is available.

Note: This page is only displayed when external bus event channel Ch2 is selected and [Trace stop] is specified on the [Action] page.

(6) [Action] page

Sets the operation after the condition has been satisfied.

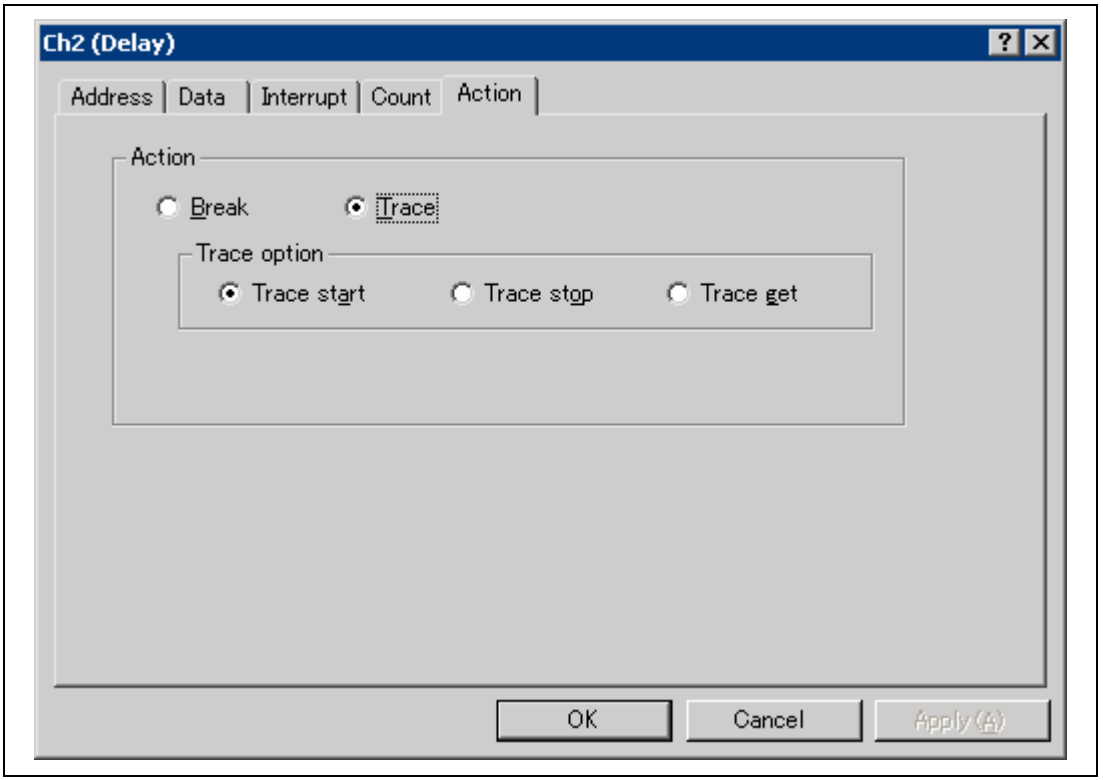


Figure 5.48 [Chx (Delay)] Dialog Box ([Action] Page)

[Break]: Breaks after conditions have been matched.

[Trace]: Enables [Trace option] and sets the external bus trace operation.
[Trace start]: Starts external bus trace after conditions have been matched.
[Trace stop]: Stops external bus trace after conditions have been matched.
[Trace get]: Acquires external bus trace after conditions have been matched.

(7) [Sequential BUS Event] dialog box

The sequential bus event occurs when all the BUS Eventpoint conditions are satisfied in the specified order.

Bus event channel Ch1 can be specified as the reset point. When the reset point is passed, the satisfied eventpoint condition is disabled and checking the first eventpoint condition is started.

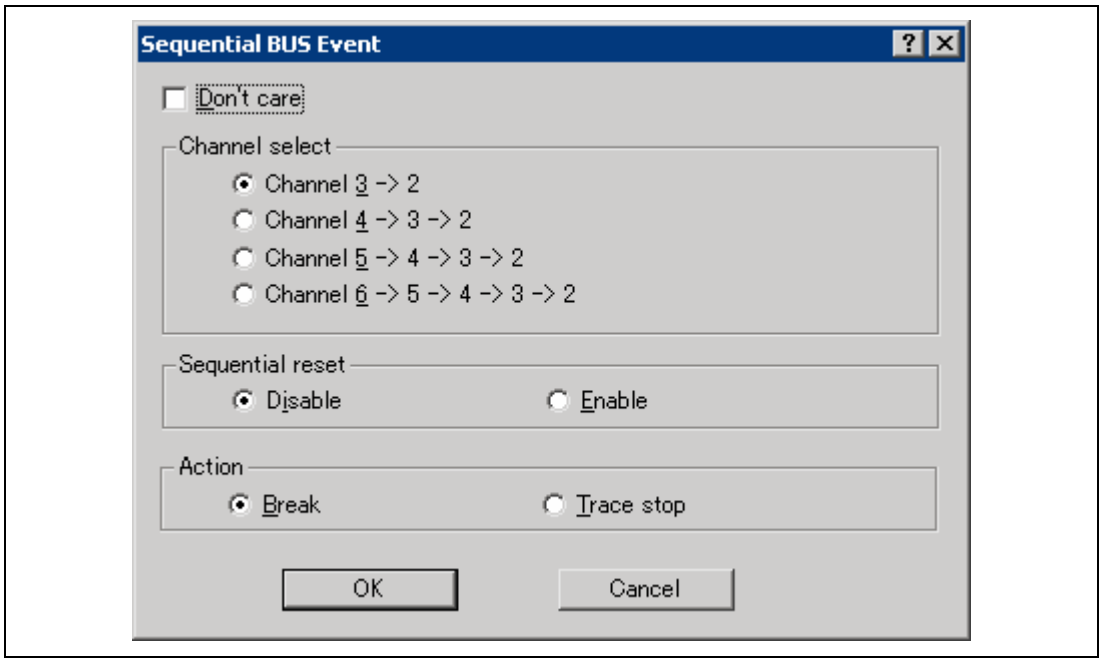


Figure 5.49 [Sequential BUS Event] Dialog Box

[Don't care]: Sets no Sequential BUS Event condition.

[Channel select]: Specifies the order that the Sequential BUS Event is satisfied.

[Channel 3 -> 2]:
After conditions have been matched in the order of bus event channel 3 -> 2, a Sequential BUS Event occurs.

[Channel 4 -> 3 -> 2]:
After conditions have been matched in the order of bus event channel 4 -> 3 -> 2, a Sequential BUS Event occurs.

[Channel 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of bus event channel 5 -> 4 -> 3 -> 2, a Sequential BUS Event occurs.

[Channel 6 -> 5 -> 4 -> 3 -> 2]:
After conditions have been matched in the order of bus event channel 6 -> 5 -> 4 -> 3 -> 2, a Sequential BUS Event occurs.

[Sequential reset]: Selects whether or not bus event channel Ch1 is used as the reset point.
[Disable]: Not used as the reset point.
[Enable]: Used as the reset point.

[Action]: Specifies the operation after the Sequential BUS Event has been detected.
[Break]: Breaks after the Sequential BUS Event has been detected.
[Trace stop]: Stops external bus trace after the Sequential BUS Event has been detected.

Note: When the Sequential BUS Event condition is set and the eventpoint condition is edited with the bus event channel that has been selected for [Channel select], the [Action] page cannot be modified. To modify the settings of the [Action] page, cancel the Sequential BUS Event condition. When a sequential event is specified, no condition will be satisfied on channels other than those selected for this event.

5.6.8 Setting Other Eventpoints

On the [Other Event] sheet, the settings for Other Eventpoints are displayed and modified.

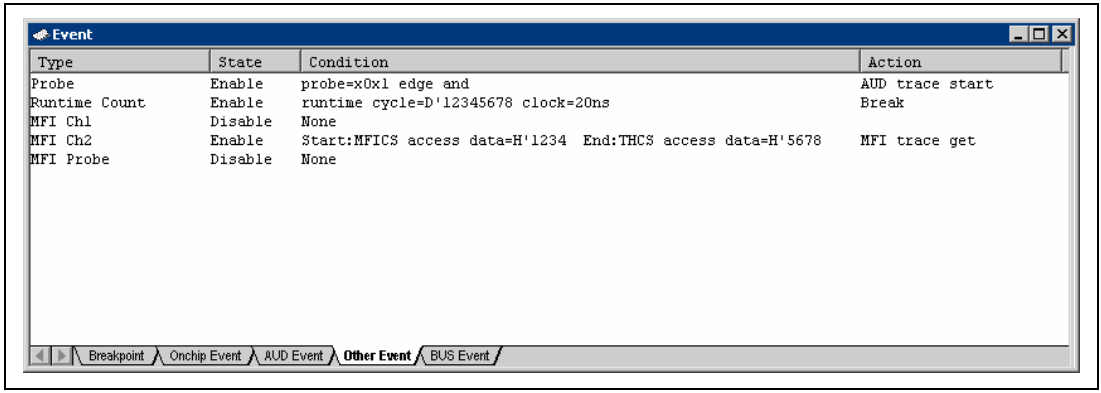


Figure 5.50 [Event] Window ([Other Event] Sheet)

Items that can be displayed in the sheet are listed below. The display content will be added depending on the status of the optional units in use.

[Type] Other event channel
Probe: External probe event channel
Runtime Count: Execution-time event channel

[State] Whether the eventpoint is enabled or disabled
Enable: Valid
Disable: Invalid

[Condition] A condition that satisfies an eventpoint. The displayed contents differ depending on the channel.

[Action] Operation of the emulator when an eventpoint condition is satisfied. The displayed contents differ depending on the channel.

When an event channel is double-clicked or selected and [Edit...] is selected from the popup menu in this window, the dialog box for setting conditions is displayed. The displayed contents of this dialog box differ depending on the event channel.

(1) [Probe] dialog box

Double-clicking the [Probe] event channel displays this dialog box. Conditions of the eventpoint can be set on the [Condition] and [Action] pages.

(a) [Condition] page:
Specifies four external probe signal conditions via the external probe cable.

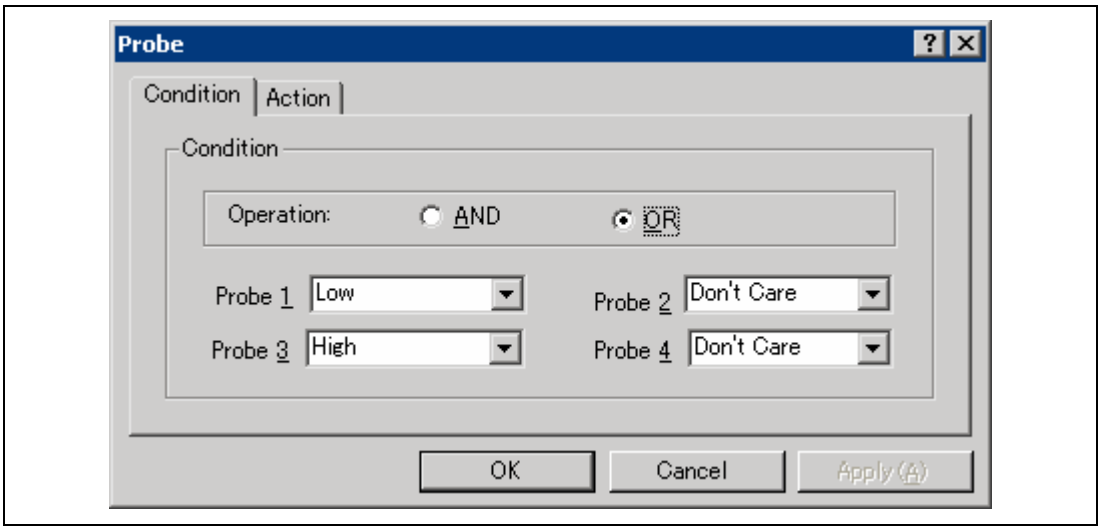


Figure 5.51 [Probe] Dialog Box ([Condition] Page)

[Operation]: Specifies how to combine the probe signal conditions.
[AND]: Determines whether or not the condition is satisfied by AND operation of the probe signal condition.
[OR]: Determines whether or not the condition is satisfied by OR operation of the probe signal condition.

[Probe 1 to 4]: Specifies the probe signal conditions.
[Don't care]: Specifies no probe signal condition.
[High]: Satisfies the condition when the probe signal is high.
[Low]: Satisfies the condition when the probe signal is low.

(b) [Action] page:
Sets the operation after the condition has been satisfied.

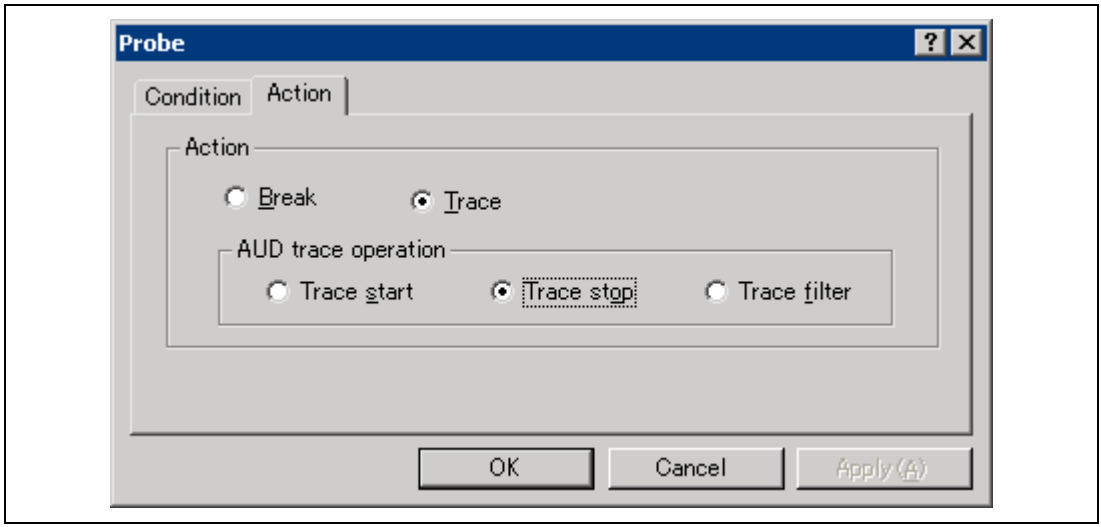


Figure 5.52 [Probe] Dialog Box ([Action] Page)

[Break]: Breaks after conditions have been matched.

[Trace]: Enables [AUD trace option] and sets the AUD trace operation.
[Trace start]: Starts AUD trace after conditions have been matched.
[Trace stop]: Stops AUD trace after conditions have been matched.
[Trace filter]: Sets the filter condition for acquiring AUD trace after conditions have been matched.

(2) [Runtime Count] dialog box

Double-clicking the [Runtime Count] event channel displays this dialog box. Conditions of the eventpoint can be set on the [Condition] and [Action] pages.

(a) [Condition] page:
Specifies the execution time of the user program.

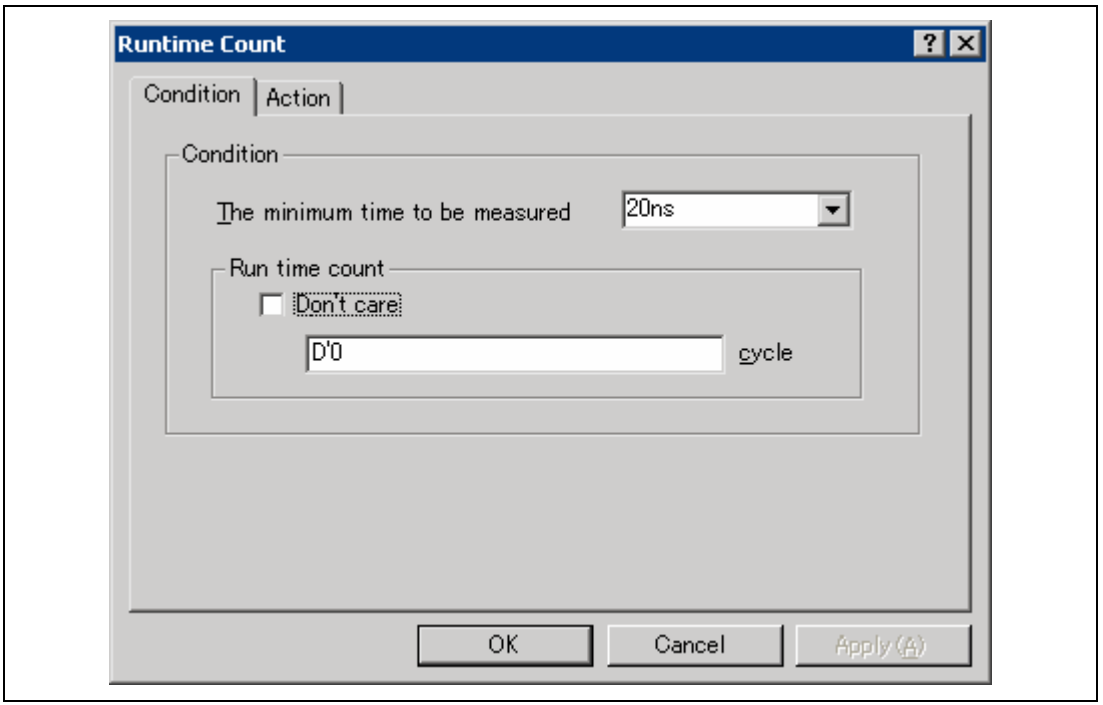


Figure 5.53 [Runtime Count] Dialog Box ([Condition] Page)

[The minimum time to be measured]: Specifies the resolution of the timer for execution time measurement as any of the following values:
20 ns, 40 ns, 100 ns, or 400 ns

[Runtime count]: Specifies the execution time conditions.
[Don’t care]: Specifies no execution time condition.
[Cycle]: Enters the measurement cycle counts; D’0 to D’1099511627775 is available.

(b) [Action] page:
Sets the operation after the condition has been satisfied.

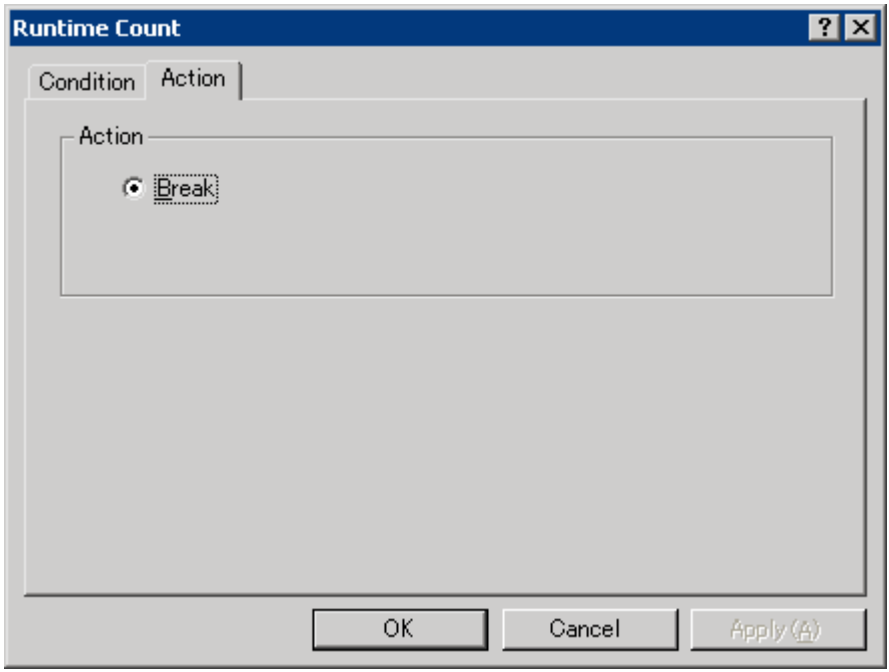


Figure 5.54 [Runtime Count] Dialog Box ([Action] Page)

[Break]: Breaks after conditions have been matched.

5.6.9 Editing Breakpoint or Eventpoint

Select a break condition to be modified, and choose [Edit...] from the popup menu to open the dialog box for the point, which allows the user to modify the breakpoints or eventpoints. The [Edit...] menu is only available when one breakpoint or eventpoint is selected.

5.6.10 Enabling Breakpoint or Eventpoint

Select a breakpoint or an eventpoint and choose [Enable] from the popup menu to enable the selected breakpoint or eventpoint.

5.6.11 Disabling Breakpoint or Eventpoint

Select a breakpoint or an eventpoint and choose [Disable] from the popup menu to disable the selected breakpoint or eventpoint. When a breakpoint or an eventpoint is disabled, a condition will not be satisfied when the specified conditions have been satisfied.

5.6.12 Deleting Breakpoint or Eventpoint

Select a breakpoint or an eventpoint and choose [Delete] from the popup menu to delete the selected breakpoint or eventpoint. To retain the break condition but not have it cause an event when its conditions are met, use the [Disable] option (see section 5.6.11, Disabling Breakpoint or Eventpoint).

Select a breakpoint or an eventpoint and click [Delete] from the popup menu. Deletes the selected breakpoint(s) or eventpoint(s). To retain the details of the breakpoint or eventpoint but not have it satisfy the condition when its conditions are met, use the [Disable] option (see section 5.6.11, Disable Breakpoint or Eventpoint).

5.6.13 Deleting All Breakpoints or Eventpoints

Choose [Delete All] from the popup menu to delete all breakpoints or eventpoints.

5.6.14 Viewing the Source Line for Breakpoints or Eventpoints

Select a breakpoint or an eventpoint and choose [Go to Source] from the popup menu to open the [Source] or [Disassembly] window at the address of the breakpoint or eventpoint. The [Go to Source] menu is only available when one breakpoint or eventpoint that has the corresponding source file is selected.

5.7 Viewing the Trace Information

In the emulator, there are functions to acquire information within and outside the MPU by a trace. The emulator acquires four types of trace information: internal trace (Internal trace), AUD trace (AUD trace), trace that is output to the user memory (Usermemory trace), and external bus trace (BUS trace). The peripheral I/O trace information can be acquired in some products.

The information on the Internal trace, AUD trace, and Usermemory trace is displayed on the [Internal/AUD/Usermemory trace] window. The information on the BUS trace is displayed on the [BUS] window.

For the description on the trace function, refer to section 1.3.3, Trace Functions.

5.7.1 Opening the [Internal/AUD/Usermemory trace] Window

To display the [Trace Window Type] dialog box, choose [View -> Code -> Trace] or click the [Trace] toolbar button ( ).

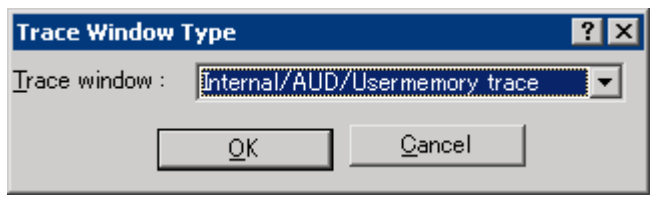


Figure 5.55 [Trace Window Type] Dialog Box

Selecting [Internal/AUD/Usermemory trace] and clicking the [OK] button displays the [Internal/AUD/Usermemory trace] window.

(1) Acquiring Internal Trace Information (Internal trace)

Selecting the [Set…] menu in the popup menu of the [Internal/AUD/Usermemory acquisition] window displays the [Internal/AUD/Usermemory acquisition] dialog box. When [Internal trace] is selected in [Trace Type] within the dialog box, the trace information is acquired by using the internal trace function.

When the emulator does not set the acquisition condition of the trace information, the trace information is acquired by the internal trace function in default.

The acquired trace information is displayed in the [Internal/AUD/Usermemory trace] window.

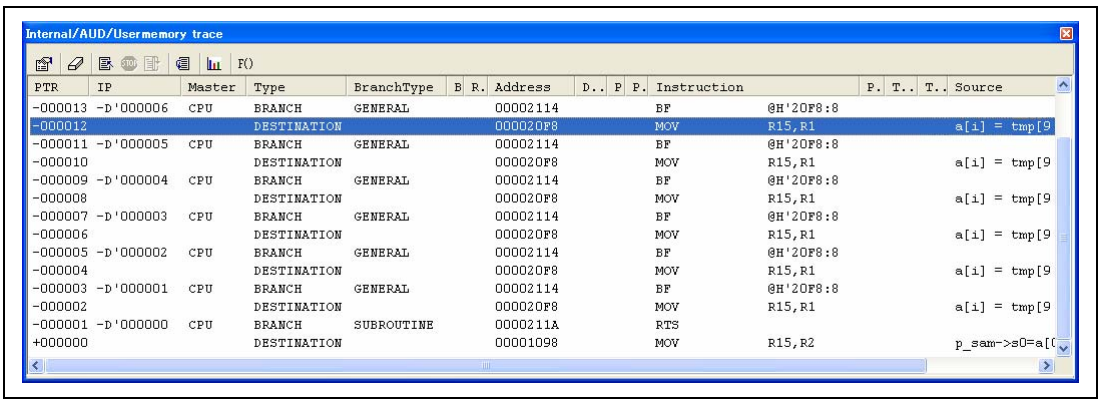


Figure 5.56 [Internal/AUD/Usermemory trace] Window (Internal Trace)

Some MPUs to be debugged display the items below. For details on the specifications of each product, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help.

[PTR] The trace buffer pointer (+0 for the last executed instruction)

[IP] The amount of acquired trace information

[Master] (Bus Master) Type of bus master accessed

[Type] Type of trace information:

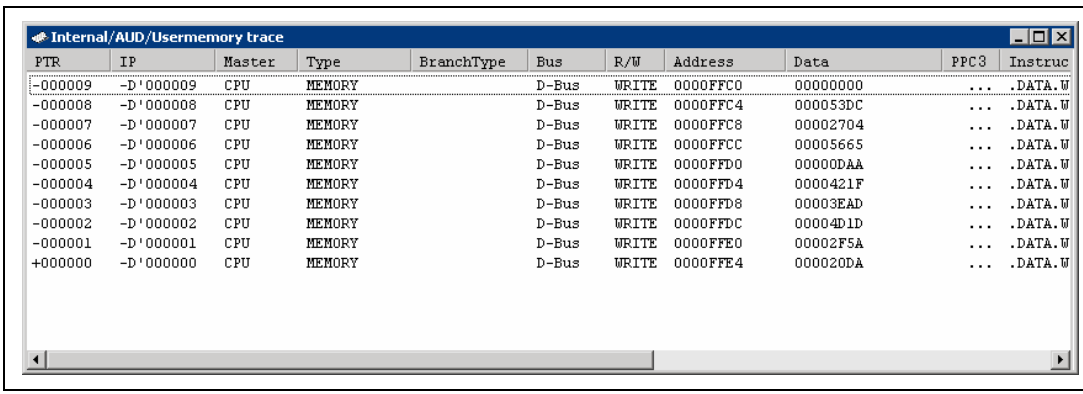
BRANCH: Branch source
DESTINATION: Branch destination
MEMORY: Memory access
S_TRACE: Executed Trace(x) function
LOST: Lost trace information (only in the realtime mode)

[Branch Type]	Type of branch (only when the branch trace is acquired): GENERAL: General branch SUBROUTINE: Subroutine branch EXCEPTION: Exception branch
[Bus]	The bus which was being accessed
[R/W]	Whether the generated data is associated with read or write access
[Address]	Address
[Data]	The data of the generated data access. When [Type] is S_TRACE, value x, a variable of function Trace(x), is displayed.
[PPC3]	Internal performance counter (Ch3) output
[PPC4]	Internal performance counter (Ch4) output
[Instruction]	Instruction mnemonic
[Source]	The C/C++ or assembly-language source program
[Label]	Label information

(2) Acquiring AUD Trace Information (AUD trace)

Selecting the [Set…] menu in the popup menu of the [Internal/AUD/Usermemory acquisition] window displays the [Internal/AUD/Usermemory acquisition] dialog box. When [AUD trace] is selected in [Trace Type] within the dialog box, the trace information is acquired by using the AUD trace function.

Note: Since the AUD trace function is implemented according to the information in the MPU output from the AUD pin, the trace acquisition condition for the Onchip Eventpoint must be set. Set trace acquisition conditions for on-chip event channels Ch (OA), Ch6 (OA), Ch7 (SystemBus), Ch8 (SystemBus), and Ch12 (Branch) or software trace.



PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	PPC3	Instruc
-000009	-D'000009	CPU	MEMORY		D-Bus	WRITE	0000FFC0	00000000	...	.DATA.W
-000008	-D'000008	CPU	MEMORY		D-Bus	WRITE	0000FFC4	000053DC	...	.DATA.W
-000007	-D'000007	CPU	MEMORY		D-Bus	WRITE	0000FFC8	00002704	...	.DATA.W
-000006	-D'000006	CPU	MEMORY		D-Bus	WRITE	0000FFCC	00005665	...	.DATA.W
-000005	-D'000005	CPU	MEMORY		D-Bus	WRITE	0000FFD0	00000DAA	...	.DATA.W
-000004	-D'000004	CPU	MEMORY		D-Bus	WRITE	0000FFD4	0000421F	...	.DATA.W
-000003	-D'000003	CPU	MEMORY		D-Bus	WRITE	0000FFD8	00003EAD	...	.DATA.W
-000002	-D'000002	CPU	MEMORY		D-Bus	WRITE	0000FFDC	00004D1D	...	.DATA.W
-000001	-D'000001	CPU	MEMORY		D-Bus	WRITE	0000FFE0	00002F5A	...	.DATA.W
+000000	-D'000000	CPU	MEMORY		D-Bus	WRITE	0000FFE4	000020DA	...	.DATA.W

Figure 5.57 [Internal/AUD/Usermemory trace] Window (AUD Trace)

This window displays the following trace information items (some of this information will not be displayed in some products):

[PTR] The AUD trace buffer pointer (+0 for the last executed instruction)

[IP] The amount of acquired trace information

[Type] Type of trace information:
BRANCH: Branch source
DESTINATION: Branch destination
MEMORY: Memory access
S_TRACE: Executed Trace(x) function
LOST: Lost trace information (only in the realtime mode)

[Bus] The bus which was being accessed

[R/W] Whether the generated data is associated with read or write access

[Address] Address

[Data] The data of the generated data access. When [Type] is S_TRACE, value x, a variable of function Trace(x), is displayed.

[PPC3] Internal performance counter (Ch3) output

[PPC4] Internal performance counter (Ch4) output

[Instruction] Instruction mnemonic

[Probe] State of the input probe

[Timestamp] Time stamp value

[Source] The C/C++ or assembly-language source program

[Label] Label information

[Timestamp-Difference] Difference value of the time stamp

Note: Since the displayed contents differ depending on the product, refer to each product's online help. Some MPUs supported may not have the AUD trace function.

(3) Acquiring Trace Information (Usermemory trace) Output to the User Memory

Selecting the [Set…] menu in the popup menu of the [Internal/AUD/Usermemory acquisition] window displays the [Internal/AUD/Usermemory acquisition] dialog box. When [Usermemory trace] is selected in [Trace Type] within the dialog box, the trace information output to the user memory is acquired.

The items to be displayed are the same as those for the internal trace information. For details, refer to section 5.7.1 (1), Acquiring Internal Trace Information.

(4) Specifying Conditions or Modes for Acquiring Trace Information

The capacity of the trace buffer is limited. When the buffer becomes full, the oldest trace information is overwritten. Setting the trace acquisition condition allows acquisition of useful trace information and effective use of the trace buffer.

The trace information acquisition conditions are implemented by eventpoints and acquisition start, acquisition end, and acquisition can be controlled. For setting eventpoints, refer to section 5.6, Using the Eventpoints.

The trace information acquisition mode is set in the [Internal/AUD/Usermemory acquisition] dialog box that is displayed by selecting [Set…] from the popup menu.

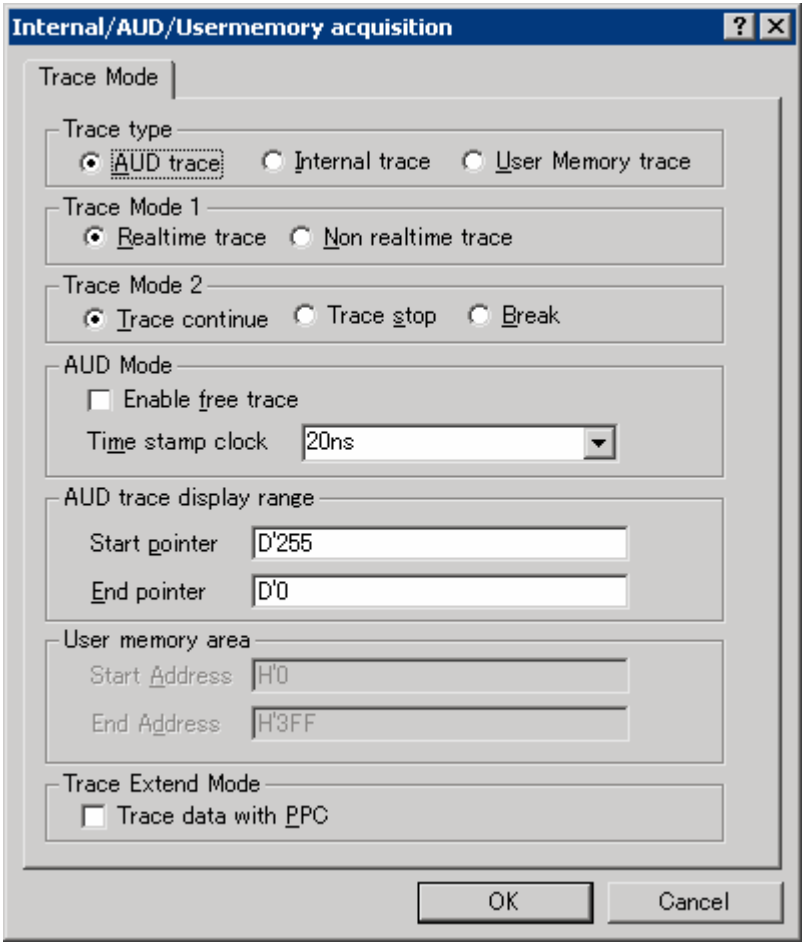


Figure 5.58 [Internal/AUD/Usermemory acquisition] Dialog Box

[Trace type]: Selects the type of trace function.

- [AUD trace] Selects the AUD trace function.
- [Internal trace] Selects the internal trace function.
- [User Memory trace] Selects the user-memory output trace function.

[Trace Mode 1]: Selects the operation mode when the trace information has been sequentially generated.

- [Realtime trace] Some trace information will not be output.

[Non realtime trace] The CPU stops operations until the trace information is output.

[Trace mode 2]: Specifies the operation when the AUD trace buffer of the emulator becomes full.

[Trace continue] The oldest trace information is overwritten to acquire the latest trace information.

[Trace stop] When the trace buffer becomes full, the trace information is no longer acquired.

[Break] A break occurs.

[AUD Mode]: Sets the mode when the AUD trace function is used.

[Enable free trace] When checked, the settings of the AUD eventpoint are ignored and the whole trace information is acquired.

[Time stamp clock] Specifies the resolution of the timer for time stamp as any of the following values:
20 ns, 40 ns, 100 ns, or 400 ns

[AUD trace display range]: Sets the trace display range when the AUD trace function is used.

[Start pointer] Enters the numerical start pointer value of the display range.

[End pointer] Enters the numerical end pointer value of the display range.

[User memory area]: Sets the trace display range when the user-memory output trace function is used.

[Start Address] Specifies the start address of the display range.

[End Address] Specifies the end address of the display range.

[Trace Extend Mode]: Sets whether or not internal performance counter Ch3 or Ch4 is included as the item for trace acquisition.

[Trace data with PPC] When checked, internal performance counter Ch3 or Ch4 is included.

Notes: 1. When [Internal trace] is selected, [Trace Mode 1], [Trace Mode 2], [AUD Mode], [AUD trace display range], and [User memory area] are disabled.
2. When only [AUD trace] is selected, [Trace Mode 2], [AUD Mode], and [AUD trace display range] are enabled.
3. When only [User Memory trace] is selected, [User memory area] is enabled.

Clicking the [OK] button stores the settings. Clicking the [Cancel] button closes this dialog box without modifying the settings.

5.7.2 Opening the [BUS/MFI trace] Window

To display the [Trace Window Type] dialog box, choose [View -> Code -> Trace] or click the [Trace] toolbar button ( ).

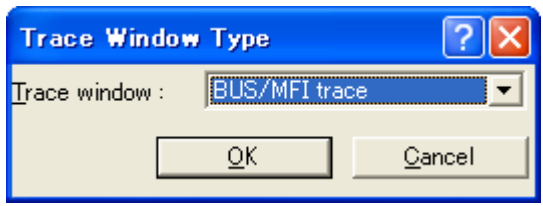


Figure 5.59 [Trace Window Type] Dialog Box

Selecting [BUS/MFI trace] and clicking the [OK] button displays the [BUS/MFI trace] window.

Notes: 1. When the trace unit is not connected to the emulator, the external bus trace function is not supported.
2. When the function of the trace unit is changed, the number of bus cycles that can be acquired by the external bus trace is changed. For details, refer to section 5.1.4, [Bus Board] Page.
3. Not only the BUS trace but also other trace functions can be displayed in this window depending on the product. For example, the display will be [BUS/MFI Trace] in the SH-Mobile MPUs. In this case, the result of the access via MFI that has been acquired by a trace is displayed. For details on the specifications of each product, refer to the online help.

(1) Acquiring External Bus Trace Information (BUS trace)

Selecting the [Set…] menu in the popup menu of the [BUS/MFI trace] window displays the [BUS acquisition] dialog box. When [BUS/MFI trace] is selected in [Trace Type] within the dialog box, the trace information is acquired by using the external bus trace function.

When the emulator does not set the acquisition condition of the trace information, the trace information is acquired by the external bus trace function in default.

The acquired trace information is displayed in the [BUS/MFI trace] window.

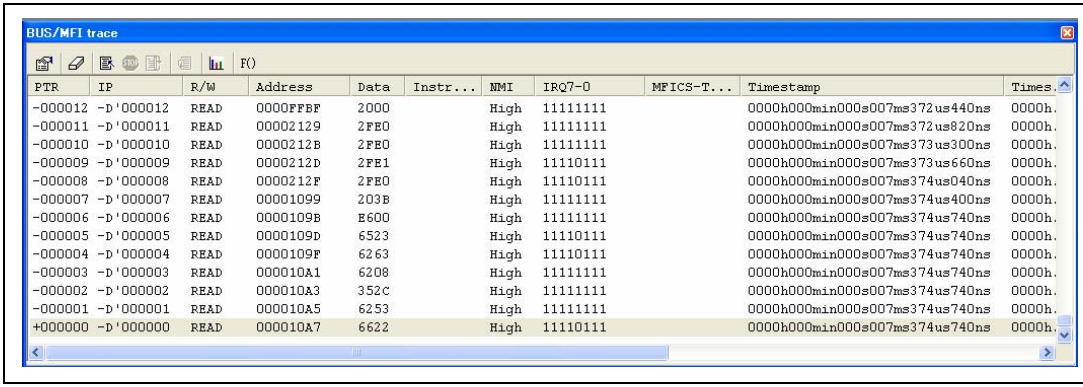


Figure 5.60 [BUS/MFI trace] Window (BUS Trace)

Items that can be displayed in this window are listed below. For details on the specifications of each product, refer to the online help.

[PTR] The bus trace buffer pointer (+0 for the last executed instruction)

[IP] The amount of acquired trace information

[R/W] Whether the generated data is associated with read or write access

[Address] Address

[Data] The data bus value

[NMI] Input state of NMI

[IRQ5-0] Input states of six IRQs

[Instruction] Instruction mnemonic

[Timestamp] Time stamp of bus cycle

[Timestamp-Difference] Difference value of the time stamp

(2) Specifying Conditions or Modes for Acquiring Trace Information

The capacity of the trace buffer is limited. When the buffer becomes full, the oldest trace information is overwritten. Setting the trace acquisition condition allows acquisition of useful trace information and effective use of the trace buffer.

The trace information acquisition conditions are implemented by eventpoints and acquisition start, acquisition end, and acquisition can be controlled. For setting eventpoints, refer to section 5.6, Using the Eventpoints.

The trace information acquisition mode is set in the [BUS acquisition] dialog box that is displayed by selecting [Set…] from the popup menu.

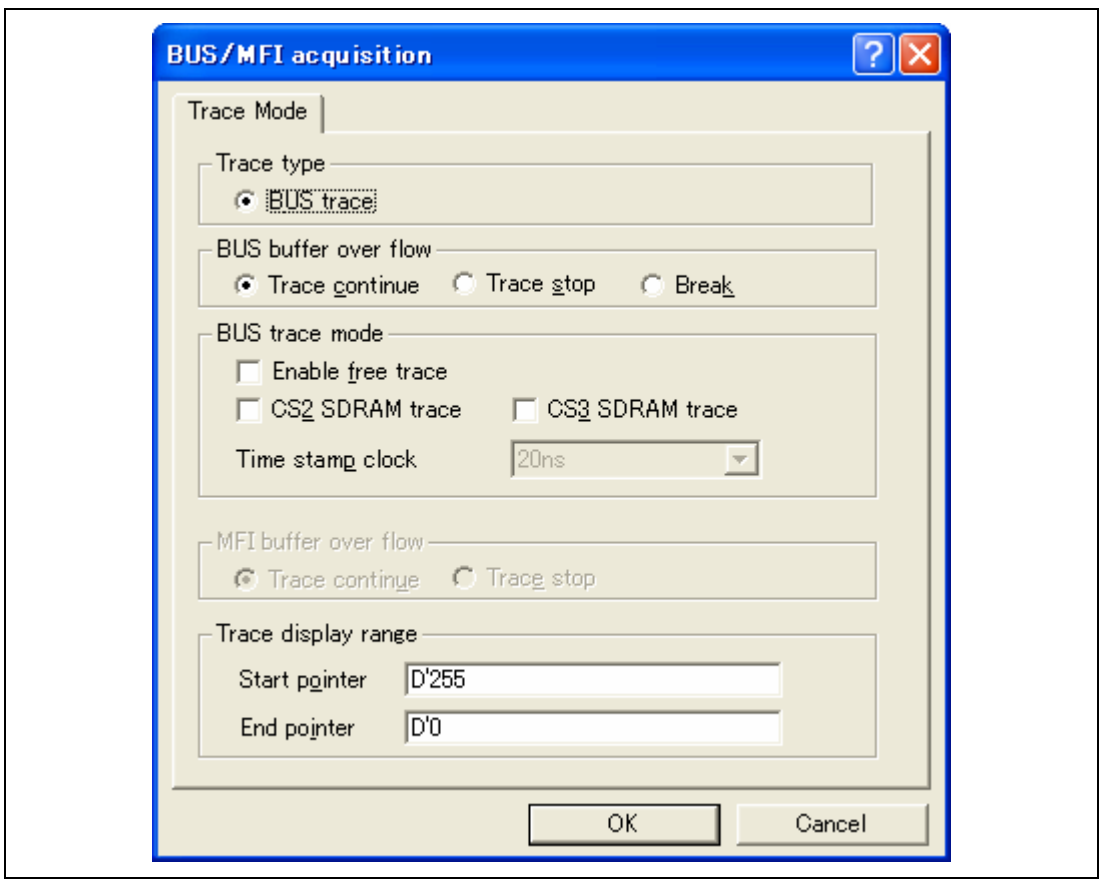


Figure 5.61 [BUS/MFI acquisition] Dialog Box

[BUS buffer over flow]: Specifies the operation when the external bus trace buffer of the emulator becomes full.

- [Trace continue]: The oldest trace information is overwritten by the latest information.
- [Trace stop]: When the trace buffer becomes full, the trace information is no longer acquired.
- [Break]: A break occurs.

[BUS trace mode]: Sets the mode when the external bus trace function is used.

- [Enable free trace]: Enables external-bus free trace.
- [CS2 SDRAM trace]: Recognizes the CS2 area as SDRAM to be analyzed by a trace.
- [CS3 SDRAM trace]: Recognizes the CS3 area as SDRAM to be analyzed by a trace.
- [Time stamp clock]: Specifies the resolution of the timer for time stamp as any of the following values:
  20 ns, 40 ns, 100 ns, or 400 ns

[Trace display range]: Sets the trace display range.

- [Start pointer]: Enters the numerical start pointer value of the display range.
- [End pointer]: Enters the numerical end pointer value of the display range.

5.7.3 Hiding the [Trace] Column

It is possible to hide any column not necessary in the [Trace] window. Selecting a column you want to hide from the popup menu displayed by clicking the right-hand mouse button on the header column hides that column. To display the hidden column, select the column from the said popup menu again. Dragging the column with the mouse changes the display order.

5.7.4 Searching for a Trace Record

Use the [Trace Find] dialog box to search for a trace record. To open this dialog box, choose [Find...] from the popup menu.

The [Trace Find] dialog box has the following options:

Table 5.4 [Trace Find] Dialog Box Pages

Page	Description
[General]	Sets the range for searching.
[Address]	Sets an address condition.
[Data]	Sets a data condition.
[Type]	Selects the type of trace information.
[Bus]	Selects the type of a bus.
[R/W]	Selects the type of access cycles.

Note: Items other than [General] and [Address] vary according to the emulator in use. For details, refer to the online help.

Clicking the [OK] button after setting conditions in those pages stores the settings and starts searching. Clicking the [Cancel] button closes this dialog box without setting conditions.

When a trace record that matches the search conditions is found, the line for the trace record will be highlighted. When no matching trace record is found, a message dialog box will appear.

Only the trace information that satisfies all the conditions set in above pages will be searched.

If a search operation is successful, selecting [Find Next] from the popup menu will move to the next found item.

(1) [General] page

Set the range for searching.

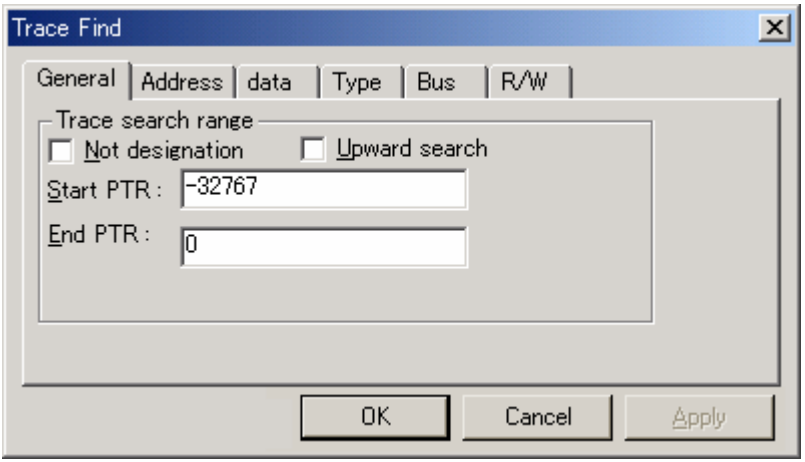


Figure 5.62 [Trace Find] Dialog Box ([General] Page)

[Trace search range]: Sets the range for searching.

[Not designation]: Searches for information that does not match the conditions set in other pages when this box is checked.

[Upward search]: Searches upwards when this box is checked.

[Start PTR]: Enters a PTR value to start a search.

[End PTR]: Enters a PTR value to end a search.

Note: Along with setting the range for searching, PTR values to start and end searching can be set in the [Start PTR] and [End PTR] options, respectively.

(2) [Address] page

Set an address condition.

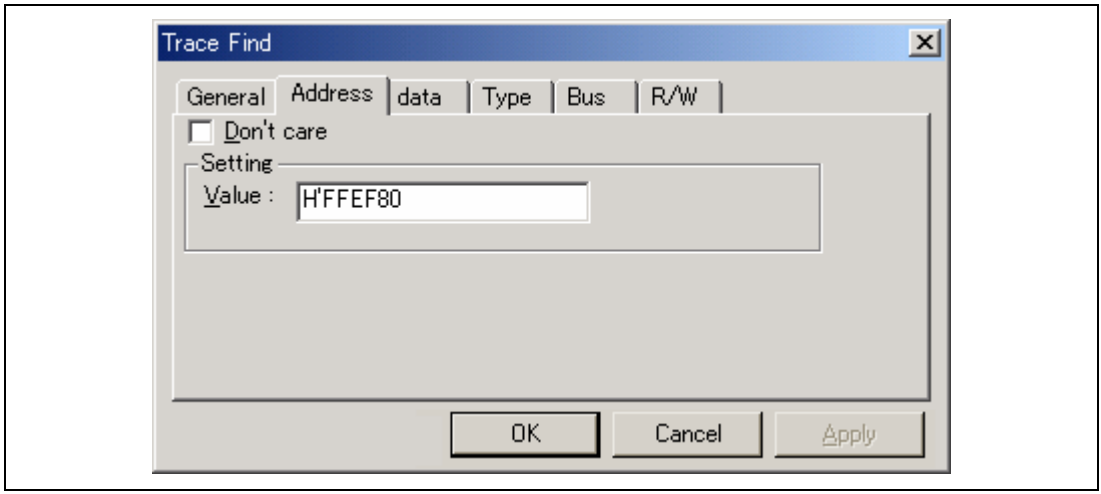


Figure 5.63 [Trace Find] Dialog Box ([Address] Page)

[Don't care]: Detects no address when this box is checked.

[Setting]: Detects the specified address.

[Value]: Enter the address value (not available when [Don’t care] has been checked).

(3) [Data] page

Set a data condition.

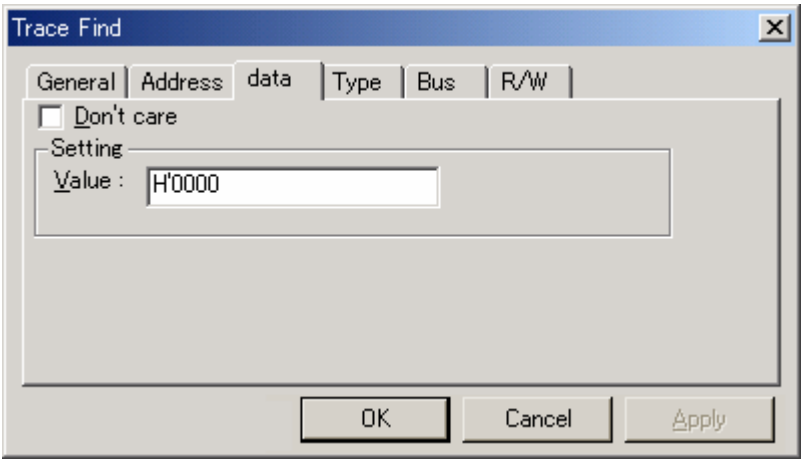


Figure 5.64 [Trace Find] Dialog Box ([data] Page)

[Don't care]: Detects no data when this box is checked.

[Setting]: Detects the specified data.

[Value]: Enter the data value (not available when [Don’t care] has been checked).

(4) [R/W] page

Select the type of access cycles.

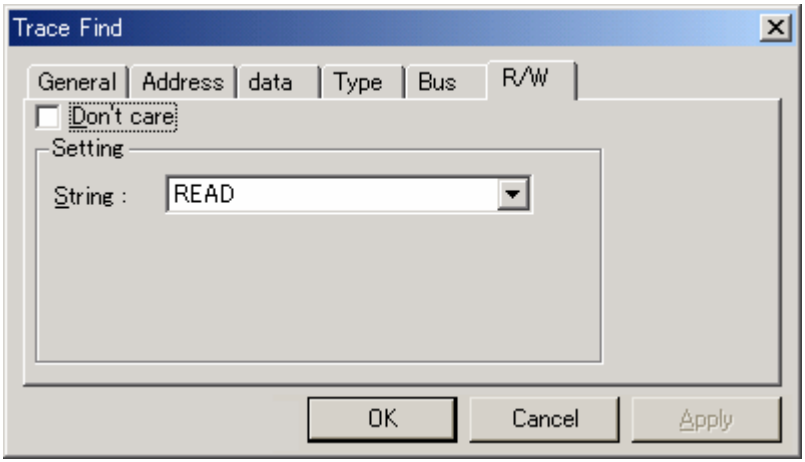


Figure 5.65 [Trace Find] Dialog Box ([R/W] Page)

[Don't care]: Detects no read/write condition when this box is checked.

[Setting]: Detects the specified read/write condition.

[String]: Select a read/write condition (not available when [Don't care] has been checked).

READ: Read cycle

WRITE: Write cycle

(5) [Type] page

Select the type being accessed. The selection is not available when a time stamp is acquired.

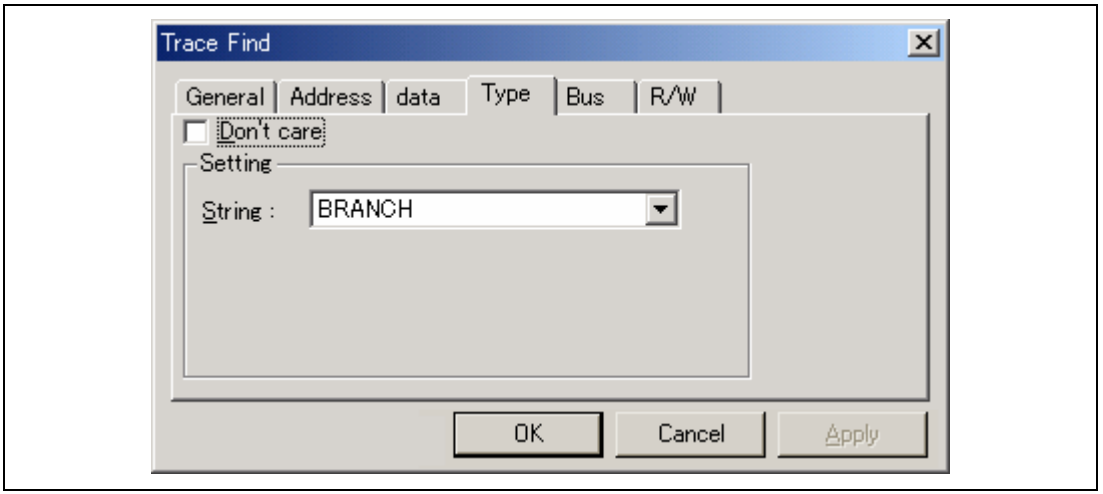


Figure 5.66 [Trace Find] Dialog Box ([Type] Page)

[Don't care]: Detects no type condition when this box is checked.

[Setting]: Detects the specified type condition.

[String]: Select a type condition (not available when [Don't care] has been checked).

(6) [Bus] page

Select the status of a bus.

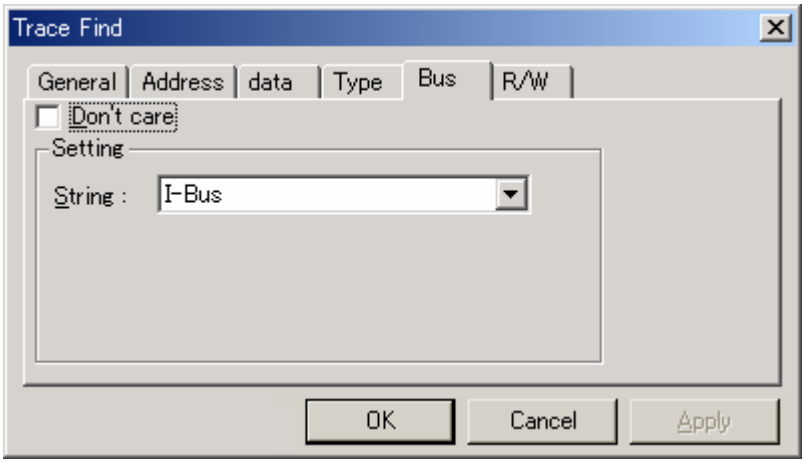


Figure 5.67 [Trace Find] Dialog Box ([Bus] Page)

[Don't care]: Detects no bus condition when this box is checked.

[Setting]: Detects the specified bus condition.

[String]: Select a bus condition (not available when [Don’t care] has been checked).

5.7.5 Clearing the Trace Information

When [Clear] is selected from the popup menu, the trace buffer that stores the trace information becomes empty.

5.7.6 Saving the Trace Information in a File

Select [Save...] from the popup menu to open the [Save As] file dialog box, which allows the user to save the information displayed in the [Trace] window as a text file. A range can be specified based on the [PTR] number (saving the complete buffer may take several minutes). Note that this file cannot be reloaded into the [Trace] window.

Note: In filtering of trace information, the range to be saved cannot be selected. All the trace information displayed in the [Trace] window after filtering will be saved. Select a filtering range on the [General] page in the [Trace Filter] dialog box if you want to save the selected range. For details on the filtering function, refer to section 5.7.10, Extracting Records from the Acquired Information.

5.7.7 Viewing the [Source] Window

The [Source] window corresponding to the selected trace record can be displayed in the following two ways:

- Select a trace record and choose [View Source] from the popup menu.
- Double-click a trace record.

The [Source] or [Disassembly] window opens and the selected line is marked with a cursor.

5.7.8 Trimming the Source

Choose [Trim Source] from the popup menu to remove the white space from the left side of the source.

When the white space is removed, a check mark is shown to the left of the [Trim Source] menu. To restore the white space, choose [Trim Source] while the check mark is shown.

5.7.9 Temporarily Stopping Trace Acquisition

To temporarily stop trace acquisition during execution of the user program, select [Halt] from the popup menu. This stops trace acquisition and updates the trace display. Use this method to check the trace information without stopping execution of the user program.

5.7.10 Extracting Records from the Acquired Information

Use the filtering function to extract the records you need from the acquired trace information. The filtering function allows the trace information acquired by hardware to be filtered by software. Unlike the settings made in the [Trace Acquisition] dialog box for acquiring trace information by conditions, changing the settings for filtering several times to filter the acquired trace information allows easy extraction of necessary information, which is useful for analysis of data. The content of the trace buffer will not be changed even when the filtering function is used. Acquiring useful information as much as possible by the [Trace Acquisition] settings improves the efficiency in analysis of data because the capacity of the trace buffer is limited.

Use the filtering function in the [Trace Filter] dialog box. To open the [Trace Filter] dialog box, select [Filter…] from the popup menu.

The [Trace Filter] dialog box has the following pages:

Table 5.5 [Trace Filter] Dialog Box Pages

Page	Description
[General]	Selects the range for filtering.
[Address]	Sets address conditions.
[Data]	Sets data conditions.
[Type]	Selects the type of trace information.
[Bus]	Selects the type of a bus.
[R/W]	Selects the type of access cycles.

Note: Items other than [General] and [Address] vary according to the emulator in use. For details, refer to the online help.

Set filtering conditions and then press the [OK] button. This starts filtering according to the conditions. Clicking the [Cancel] button closes the [Trace Filter] dialog box, which holds the settings at the time when the dialog box was opened.

In filtering, only the trace information that satisfies one or more filtering conditions set in the above pages will be displayed in the [Trace] window.

Filtering conditions can be changed several times to analyze data because the content of the trace buffer is not changed by filtering.

(1) [General] page

Set the range for filtering.

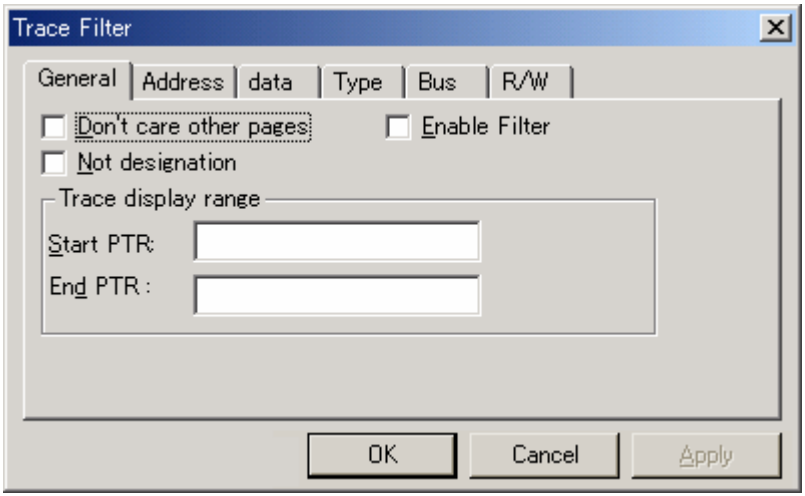


Figure 5.68 [Trace Filter] Dialog Box ([General] Page)

[Don't care other pages]: Only selects the cycle number when this box is checked. Other options become invalid.

[Enable Filter]: Enables the filter when this box is checked.

[Not designation]: Filters information that does not match the conditions set in those pages when this box is checked.

[Trace display range]: Sets the range for filtering.

[Start PTR]: Enters a PTR value to start filtering.
[End PTR]: Enters a PTR value to end filtering.

Note: Along with setting the range for filtering, PTR values to start and end filtering can be set in the [Start PTR] and [End PTR] options, respectively.

(2) [Address] page

Set address conditions.

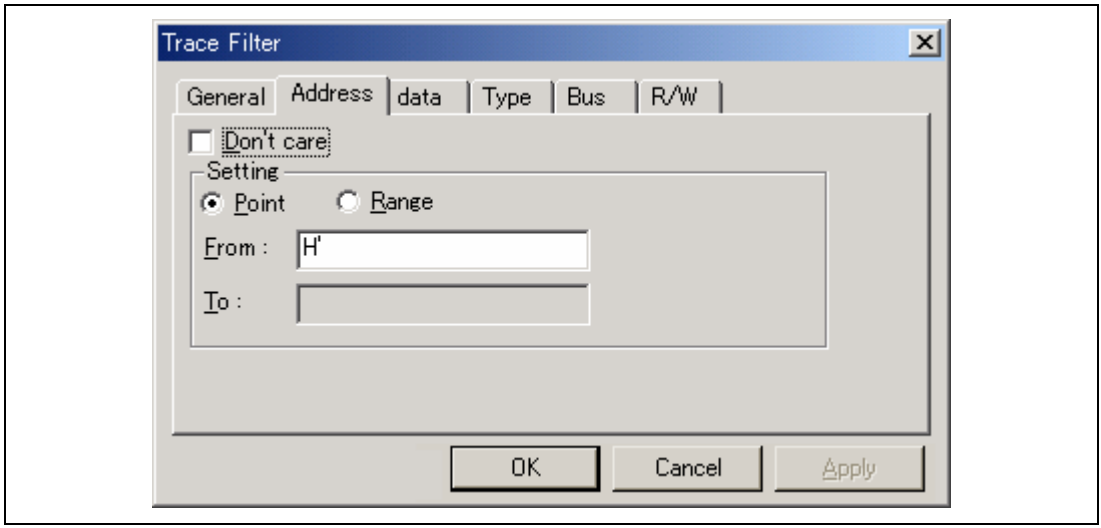


Figure 5.69 [Trace Filter] Dialog Box ([Address] Page)

[Don't care]: Detects no address when this box is checked.

[Setting]: Detects the specified address.

[Point]: Specifies a single address (not available when [Don't care] has been checked).

[Range]: Specifies an address range (not available when [Don't care] has been checked).

[From]: Enter a single address or the start of the address range (not available when [Don't care] has been checked).

[To]: Enter a single address or the end of the address range (only available when [Range] has been selected).

Note: Along with setting the address range, the start and end of the address range can be set in the [From] and [To] options, respectively.

(3) [Data] page

Set a data condition.

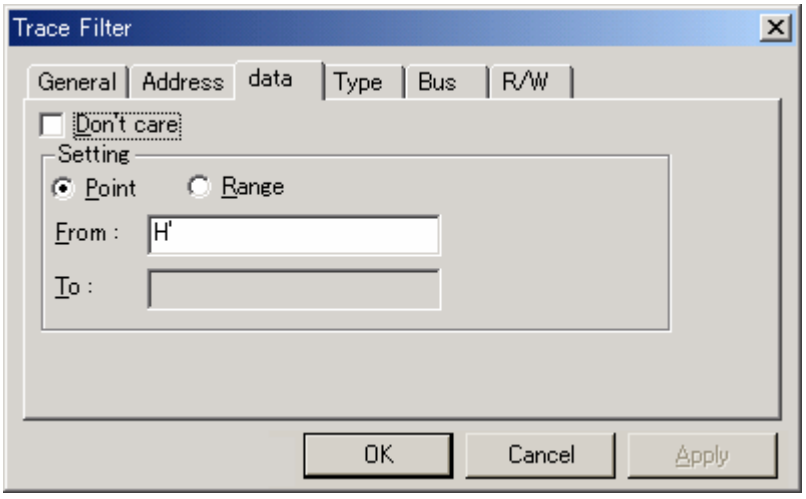


Figure 5.70 [Trace Filter] Dialog Box ([data] Page)

[Don't care]: Detects no data when this box is checked.

[Setting]: Detects the specified data.

[Point]: Specifies single data (not available when [Don't care] has been checked).

[Range]: Specifies a data range (not available when [Don't care] has been checked).

[From]: Enter single data or the minimum value of the data range (not available when [Don't care] has been checked).

[To]: Enter the maximum value of the data range (only available when [Range] has been selected).

Note: Along with setting the data range, the minimum and maximum values can be set in the [From] and [To] options, respectively.

(4) [R/W] page

Select the type of access cycles.

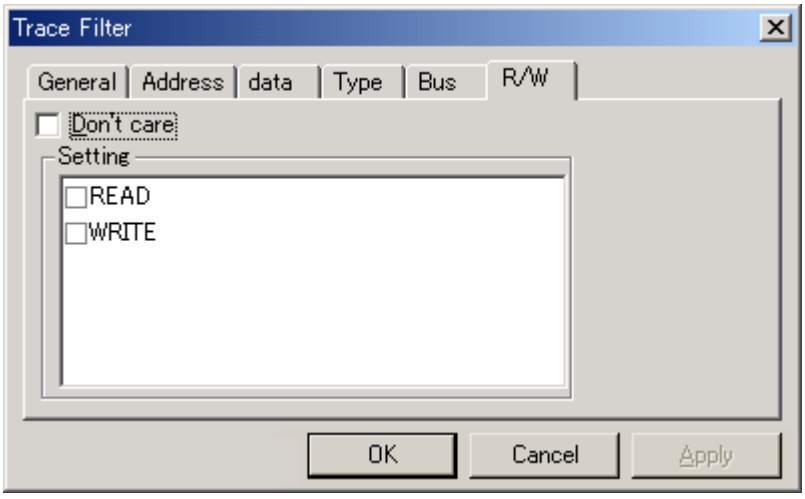


Figure 5.71 [Trace Filter] Dialog Box ([R/W] Page)

[Don't care]: Detects no read/write condition when this box is checked.

[Setting]: Detects the specified read/write condition.

READ: Detects read cycles when this box is checked (not available when [Don’t care] has been checked).

WRITE: Detects write cycles when this box is checked (not available when [Don’t care] has been checked).

(5) [Type] page

Select the type being accessed. The selection is not available when a time stamp is acquired.

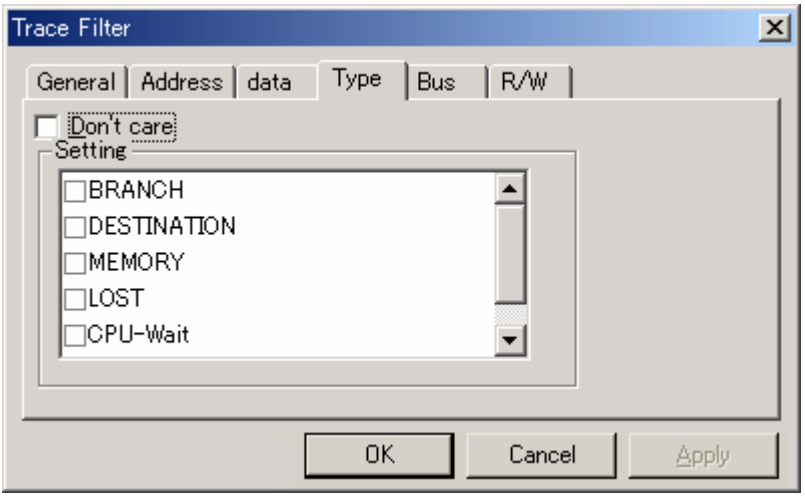


Figure 5.72 [Trace Filter] Dialog Box ([Type] Page)

[Don't care]: Detects no type condition when this box is checked.

[Setting]: Detects the specified type condition (not available when [Don’t care] has been checked).

(6) [Bus] page

Select the status of a bus. The selection is not available when a time stamp is acquired.

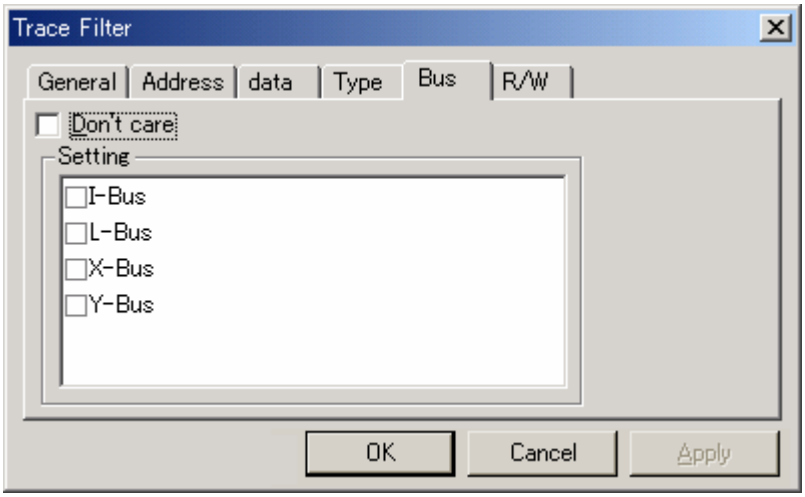


Figure 5.73 [Trace Filter] Dialog Box ([Bus] Page)

[Don't care]: Detects no bus condition when this box is checked.

[Setting]: Detects the specified bus condition (not available when [Don’t care] has been checked).

5.7.11 Analyzing Statistical Information

Choose [Statistic…] from the popup menu to open the [Statistic] dialog box and analyze statistical information under the specified conditions.

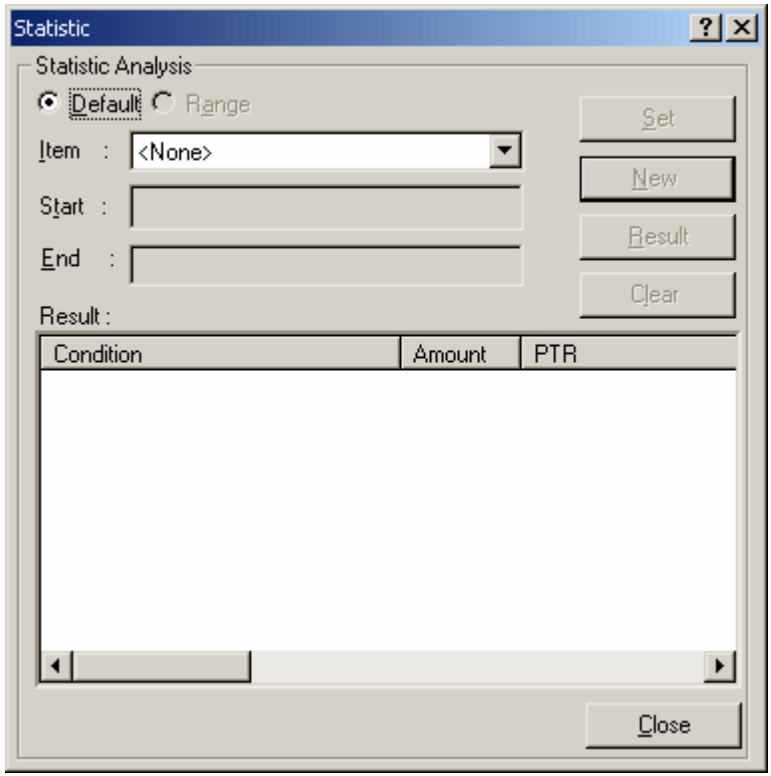


Figure 5.74 [Statistic] Dialog Box

[Statistic Analysis]: Setting required for analysis of statistical information.

[Default]: Sets a single input value or character string.

[Range]: Sets the input value or character string as a range.

[Item]: Sets the item for analysis.

[Start]: Sets the input value or character string. To set a range, the start value must be specified here.

[End]: Specify the end value if a range has been set (only available when [Range] has been selected).

[Set]: Adds a new condition to the current one.

[New]: Creates a new condition.

[Result] button: Obtains the result of statistical information analysis.

[Clear]: Initializes the settings.

[Result] list box: Clears all conditions and results of statistical information analysis.

[Close]: Closes this dialog box. All the results displayed in the [Result] list will be cleared.

This dialog box allows the user to analyze statistical information concerning the trace information. Set the target of analysis in [Item] and the input value or character string by [Start] and [End]. Click the [Result] button after setting a condition by pressing the [New] or [Add] button to analyze the statistical information and display its result in the [Result] list.

Note: In this emulator, only [PTR] can be set as a range. Each of other items must be specified as a character string. In analysis of statistical information, character strings are compared with those displayed in the [Trace] window. Only those that completely match are counted. Note, however, that this test is not case sensitive. The number of blanks will not be cared either.

5.7.12 Extracting Function Calls from the Acquired Trace Information

To extract function calls from the acquired trace information, select [Function Call…] from the popup menu. The [Function Call Display] dialog box will be displayed.

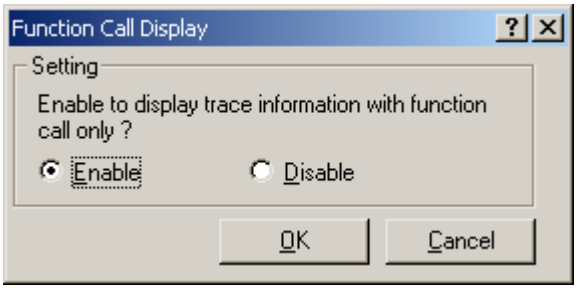


Figure 5.75 [Function Call Display] Dialog Box

[Setting]: Selects whether or not to extract function calls.

[Enable]: Extracts function calls.

[Disable]: Does not extract function calls.

When [Enable] is selected, only the cycles that include function calls are extracted for display from the acquired trace information. The content of the trace buffer is not changed by extraction of function calls. Using this function for the trace information that includes function calls allows the user to know the order of function calls.

5.8 Analyzing Performance

Use the performance analysis function to measure execution performance.

The emulator has three types of performance analysis functions: on-chip performance analysis (Onchip Performance Analysis), AUD performance analysis (AUD Performance Analysis), and external bus performance analysis (BUS Performance Analysis). For details on the performance analysis functions, refer to section 1.3.7, Performance Measurement Function.

5.8.1 Opening the [Onchip Performance Analysis] Window

To open the [Onchip Performance Analysis] window, choose [View -> Performance -> Performance Analysis] or click the [PA] toolbar button ( ) to open the [Select Performance Analysis Type] dialog box.

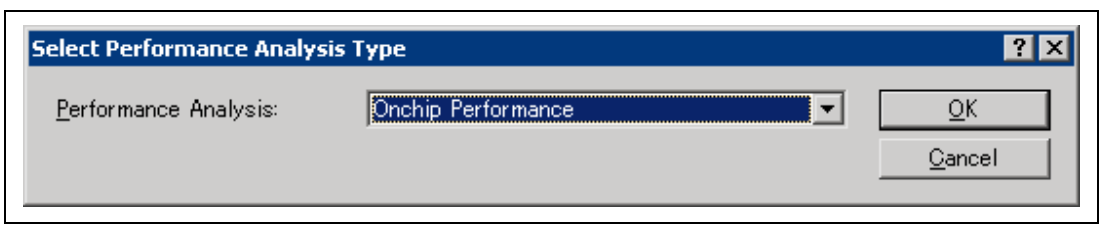


Figure 5.76 [Select Performance Analysis Type] Dialog Box

Select [Onchip Performance] and click the [OK] button to open the [Performance Analysis] window.

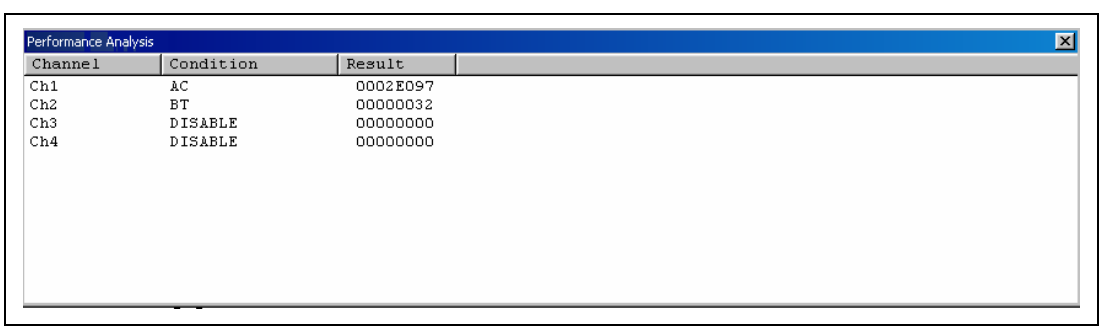


Figure 5.77 [Performance Analysis] Window (Onchip Performance)

The Onchip Performance Analysis function does not affect the realtime operation because it uses the performance measurement function in the MPU.

Note: The measurement conditions and the number of channels for the on-chip performance analysis function differ depending on the product. The time measurement continues even if a break occurs before execution reaches the measurement stop address (after passing the measurement start address) of the AUD performance function.

When a measurement channel is double-clicked or selected and [Set...] is selected from the popup menu in this window, the [Performance Analysis] dialog box is opened and measurement conditions can be modified.

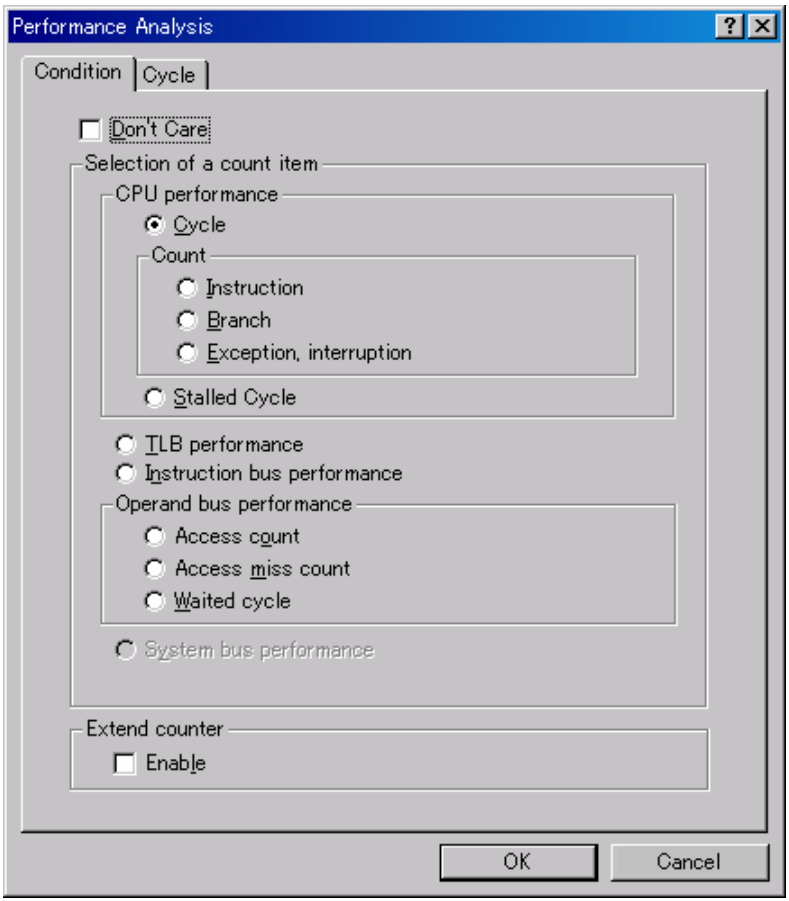


Figure 5.78 [Performance Analysis] Dialog Box

For details on the [Performance Analysis] dialog box, refer to the online help for each product.

5.8.2 Opening the [AUD Performance Analysis] Window

To open the [AUD Performance Analysis] window, choose [View -> Performance -> Performance Analysis] or click the [PA] toolbar button ( ) to open the [Select Performance Analysis Type] dialog box.

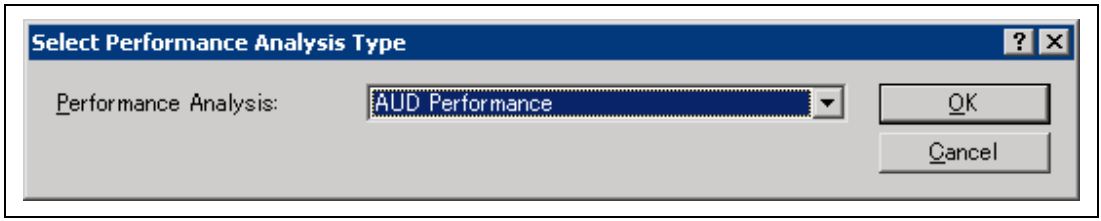


Figure 5.79 [Select Performance Analysis Type] Dialog Box

Select [AUD Performance] and click the [OK] button to open the [Performance Analysis] window.

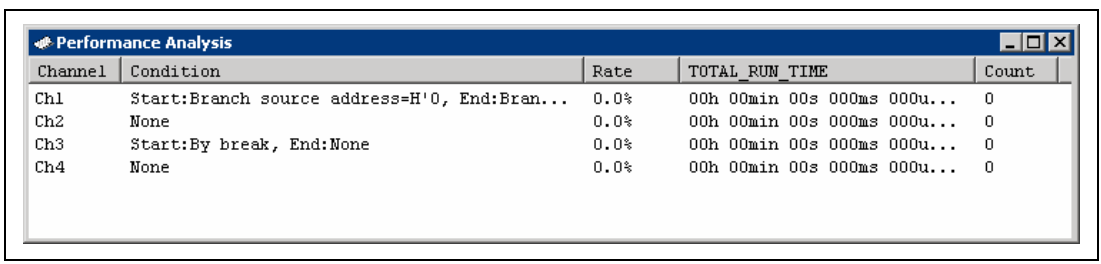


Figure 5.80 [Performance Analysis] Window (AUD Performance)

The AUD Performance Analysis function does not affect the realtime operation because it measures execution performance in the specified range by using the circuit for performance measurement on the emulator main unit case.

The start and end conditions of the measurement channel of the AUD Performance Analysis use one AUD event channel, respectively. Four measurement channels use eight AUD event channels.

Note: Since the AUD performance function is implemented according to the information in the MPU output from the AUD pin, the trace acquisition condition for the Onchip Performance analysis must be set. Set trace acquisition conditions for on-chip event channels Ch5 (OA), Ch6 (OA), Ch7 (SystemBus), Ch8 (SystemBus), and Ch12 (Branch) or software trace. The time measurement does not continue when it is executed again if a break occurs before execution reaches the measurement stop address (after passing the measurement start address) of the AUD performance function.

When a measurement channel is double-clicked or selected and [Set...] is selected from the popup menu in this window, the [Performance Analysis AUD Channel x] dialog box is opened and measurement conditions can be modified.

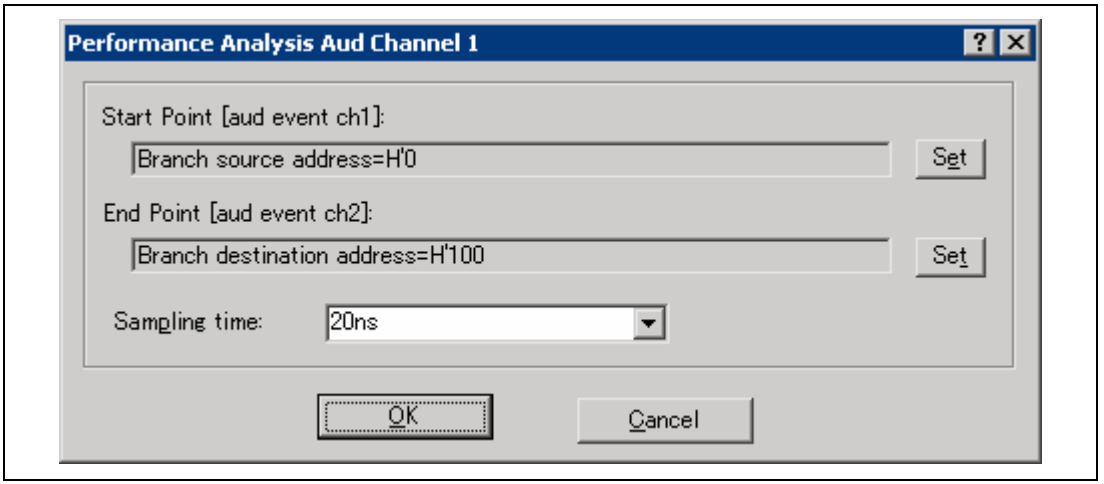


Figure 5.81 [Performance Analysis AUD Channel x] Dialog Box

[Start Point [aud event chx]]:	Displays the start pointer condition of the measurement channel.
[End Point [aud event chx]]:	Displays the end pointer condition of the measurement channel.
[Set]:	Displays the dialog box to set the AUD event channel for the start or end pointer.
[Sampling time]:	Specifies the resolution of the measurement timer as any of the following values: 20 ns, 40 ns, 100 ns, or 400 ns

Clicking the [Set] button opens the dialog box to set the corresponding AUD event channel to modify the measurement start or end condition.

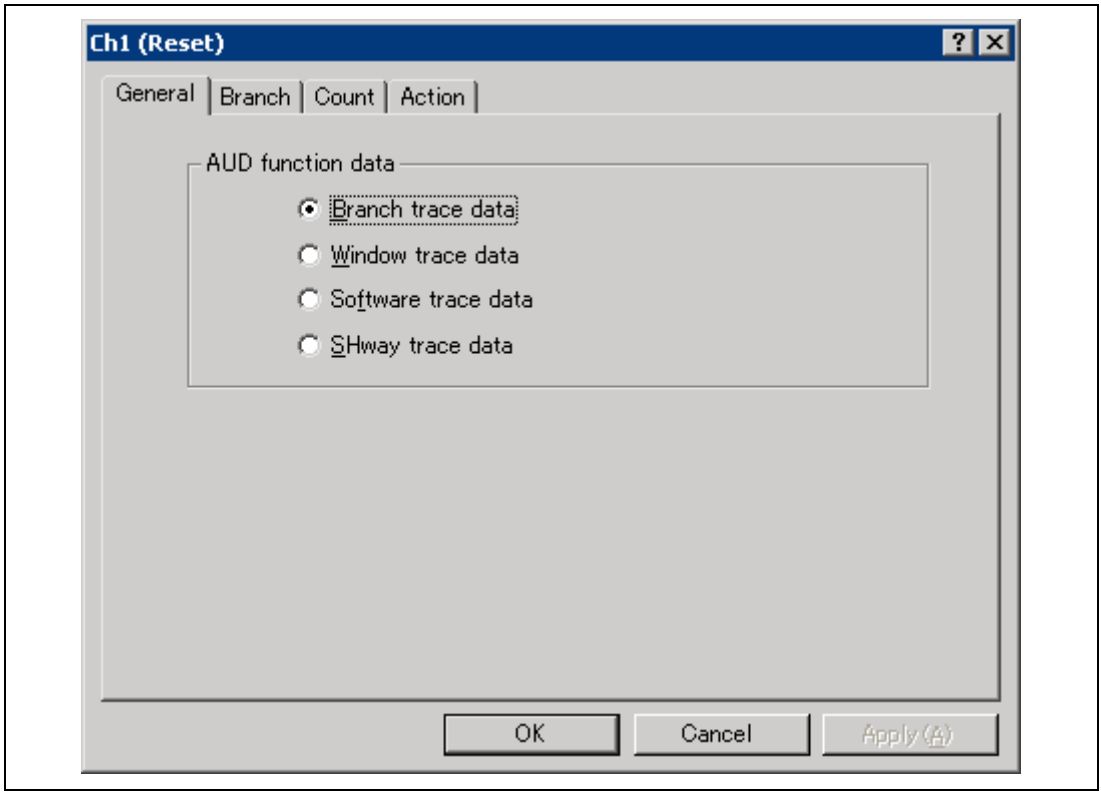


Figure 5.82 Editing the AUD Performance Analysis Measurement Condition (Dialog Box for Setting the AUD Event)

Notes: 1. When the AUD performance analysis function is used, set [Performance start/stop] on the [Action] page of each channel.

2. When the measurement condition is modified, the contents are reflected on the [AUD Event] sheet in the [Event] window.

5.8.3 Opening the [BUS Performance Analysis] Window

To open the [BUS Performance Analysis] window, choose [View -> Performance -> Performance Analysis] or click the [PA] toolbar button ( ) to open the [Select Performance Analysis Type] dialog box.

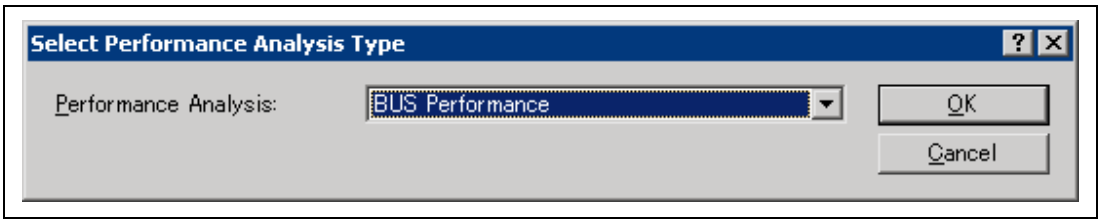
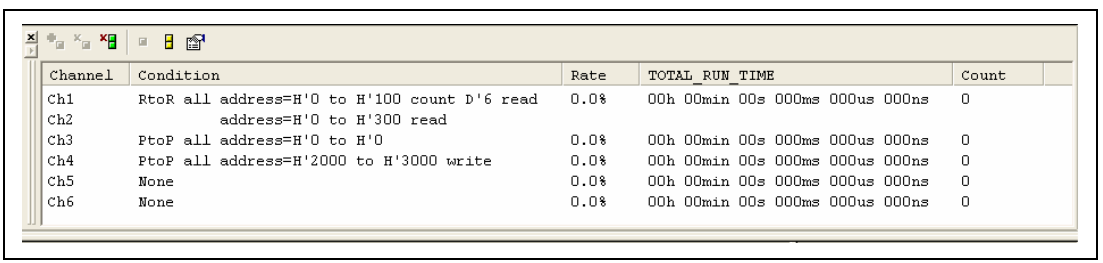


Figure 5.83 [Select Performance Analysis Type] Dialog Box

Select [BUS Performance] and click the [OK] button to open the [Performance Analysis] window.



Channel	Condition	Rate	TOTAL_RUN_TIME	Count
Ch1	RtoR all address=H'0 to H'100 count D'6 read	0.0%	00h 00min 00s 000ms 000us 000ns	0
Ch2	address=H'0 to H'300 read			
Ch3	PtoP all address=H'0 to H'0	0.0%	00h 00min 00s 000ms 000us 000ns	0
Ch4	PtoP all address=H'2000 to H'3000 write	0.0%	00h 00min 00s 000ms 000us 000ns	0
Ch5	None	0.0%	00h 00min 00s 000ms 000us 000ns	0
Ch6	None	0.0%	00h 00min 00s 000ms 000us 000ns	0

Figure 5.84 [Performance Analysis] Window (BUS Performance)

The BUS performance analysis function does not affect the realtime operation because it measures execution performance in the specified range by using the circuit for performance measurement on the trace unit of the emulator.

Select either of the following three modes according to the purpose of measurement.

Available Measurement Modes

Mode	Description	Purpose
Time Of Specified Range Measurement	Measures the execution time and execution count in the specified range.	Measurement of time taken for processing of functions except for that required for child functions called from the functions.
Start Point To End Point Measurement	Measures the execution time and execution count between the specified addresses.	Measurement of time taken for processing of functions.
Start Range To End Range Measurement	Measures the execution time from a specified range to another specified range.	Measurement of execution time spent from calling of any of sequential subroutines to calling of any of another sequential subroutines in a program that includes subroutines in sequence, such as an assembly program.

For the measurement channel, up to eight points can be set.

Note, however, since the start and end conditions use one measurement channel when Start Range To End Range Measurement is selected, two channels are used for one measurement item.

Notes: 1. When the trace unit is not connected to the emulator, the external bus function is not supported.
2. When the function of the trace unit is changed, the number of event detection channels is changed. For details, refer to section 5.1.4, [Bus Board] Page.

When a measurement channel is double-clicked or selected and [Set...] is selected from the popup menu in this window, the [Performance Analysis BUS Channel x] dialog box is opened and measurement conditions can be modified.

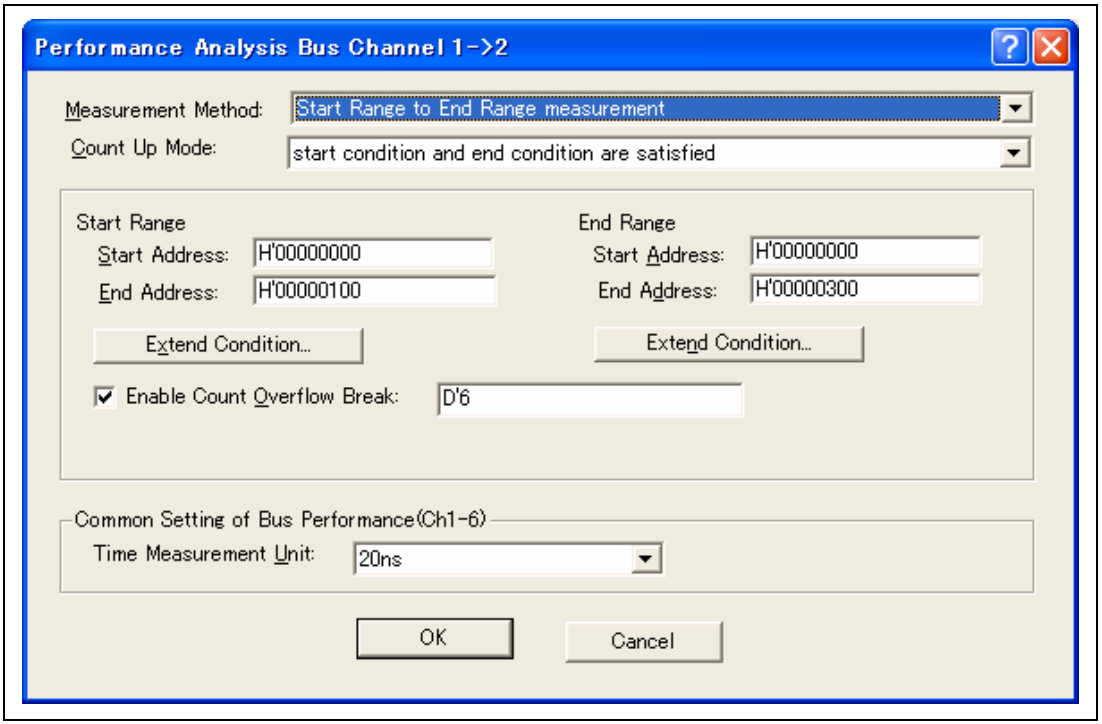


Figure 5.85 [Performance Analysis BUS Channel x] Dialog Box

[Measurement Method]: Specifies the measurement mode.
[Specified Range measurement]:
Measurement within the specified range.
[Start Point to End Point measurement]:
Measurement between specified addresses.
[Start Range to End Range measurement]:
Measurement between specified ranges.

[Count Up Mode]: Specifies the counting-up method of execution counts.
[start condition and end condition are satisfied]:
Counted up when the end condition is satisfied after the start condition has been satisfied.
[end condition is satisfied]:
Counted up whenever the end condition is satisfied.

[Start Range]: Sets the measurement condition or the measurement start condition for Start Range to End Range measurement.
[Start Address]: Enters the start address.

	[End Address]: Enters the end address. [Extend Condition]: Sets the extension condition.
[End Range]:	Sets the measurement end condition for Start Range to End Range measurement. [Start Address]: Enters the start address. [End Address]: Enters the end address. [Extend Condition]: Sets the extension condition.
[Enable Count Overflow Break]:	A break occurs if the execution count exceeds the specified count.
[Time measurement Unit]:	Specifies the resolution of the measurement timer as any of the following values: 20 ns, 50 ns, or 400 ns

Notes: 1. The [End Range] item is only displayed when [Start Range to End Range measurement] is selected.

2. The [Enable Count Overflow Break] is only displayed when measurement channel 1 is selected.

Clicking the [Extend Condition...] button opens the dialog box to set the extension condition. The bus state condition can be set. When the range is selected as the measurement condition, the read cycle is selected.

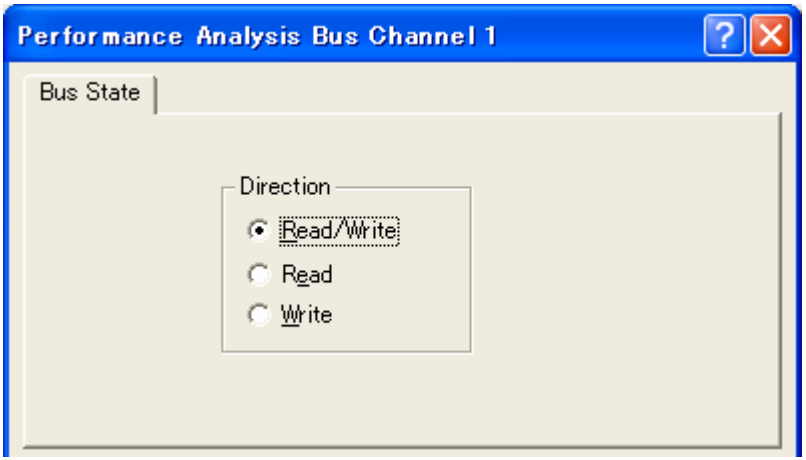


Figure 5.86 Dialog Box for Setting the Bus Performance Analysis Extension Condition

5.8.4 Hiding the Column

It is possible to hide any column not necessary in the [Performance Analysis] window. Selecting a column you want to hide from the popup menu displayed by clicking the right-hand mouse button on the header column hides that column. To display the hidden column, select the column from the said popup menu again.

5.8.5 Starting Performance Data Acquisition

Executing the user program clears the result of previous measurement and automatically starts measuring execution performance according to the conditions that have been set. Stopping the user program displays the result of measurement in the [Performance Analysis] window.

5.8.6 Deleting a Measurement Condition

Select [Reset] from the popup menu with a measurement condition selected to delete the condition.

5.8.7 Deleting All Measurement Conditions

Select [Reset All] from the popup menu to delete all the conditions that have been set.

5.9 Viewing the Profile Information

The profile function enables function-by-function measurement of the performance of the application program in execution. This makes it possible to identify parts of an application program that degrade its performance and the reasons for such degradation.

The High-performance Embedded Workshop displays the results of measurement in three windows, according to the method and purpose of viewing the profile data.

5.9.1 Stack Information Files

The profile function allows the High-performance Embedded Workshop to read the stack information files (extension: ".SNI") which are output by the optimizing linker (ver. 7.0 or later). Each of these files contains information related to the calling of static functions in the corresponding source file. Reading the stack information file makes it possible for the High-performance Embedded Workshop to display this information to do with the calling of functions without executing the user application (i.e. before measuring the profile data). However, this feature is not available when [Setting->Only Executed Functions] is checked in the pop-up menu of the [Profile] window.

When the High-performance Embedded Workshop does not read any stack information files, the data about the functions executed during measurement will be displayed by the profile function.

To make the linker create a stack information file, choose [Build -> SuperH Risc engine Standard Toolchain...], and select [Other] from the [Category] list box and check the [Stack information output] box in the [Link/Library] sheet of the [Standard Toolchain] dialog box.

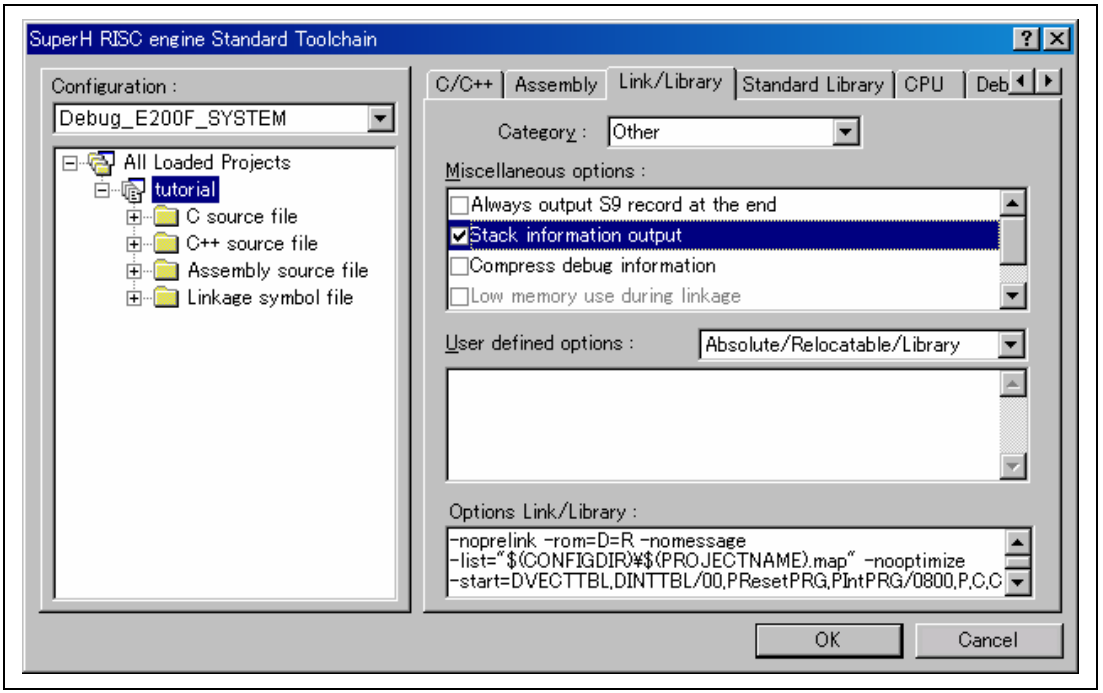


Figure 5.87 [Standard Toolchain] Dialog Box (1)

5.9.2 Profile Information Files

To create a profile information file, choose the [Output Profile Information Files…] menu option from the pop-up menu of the [Profile] window and specify the file name, after measuring a profile data of the application program.

This file contains information on the number of times functions are called and global variables are accessed. The optimizing linker (ver. 7.0 or later) is capable of reading the profile information file and optimizing the allocation of functions and variables in correspondence with the status of the actual operation of the program.

To input the profiler information file to the linker, choose [Optimize] from the [Category] list box and check the [Include profile] box in the [Link/Library] sheet of the [Standard Toolchain] dialog box, and specify the name of the profile information file.

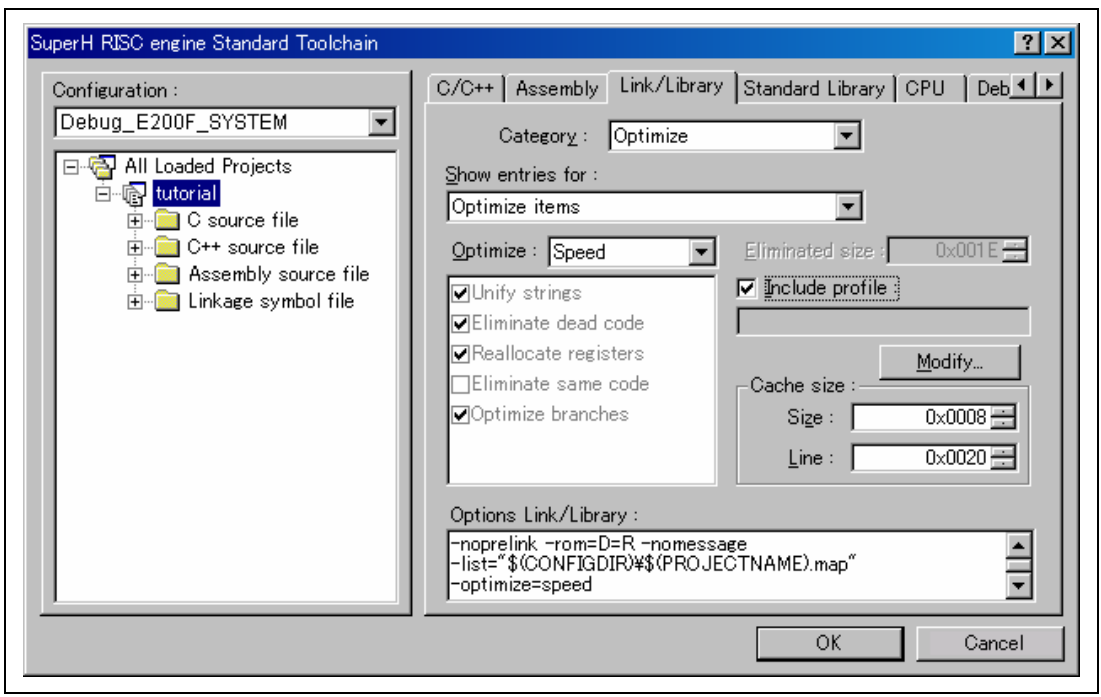


Figure 5.88 [Standard Toolchain] Dialog Box (2)

To enable the settings in the [Include profile] box, specify the [Optimize] list box as some setting other than [None].

5.9.3 Loading Stack Information Files

You can select whether or not to read the stack information file in a message box for confirmation that is displayed when a load module is loaded. Clicking the [OK] button of the message box loads the stack information file. The message box for confirmation will be displayed when:

- There are stack information files (extension: "*.SNI").
- The [Load Stack Information Files (SNI files)] check box is checked in the [Confirmation] page of the [Options] dialog box (figure 5.89) that can be opened by choosing [Tools ->Options…] from the main menu.

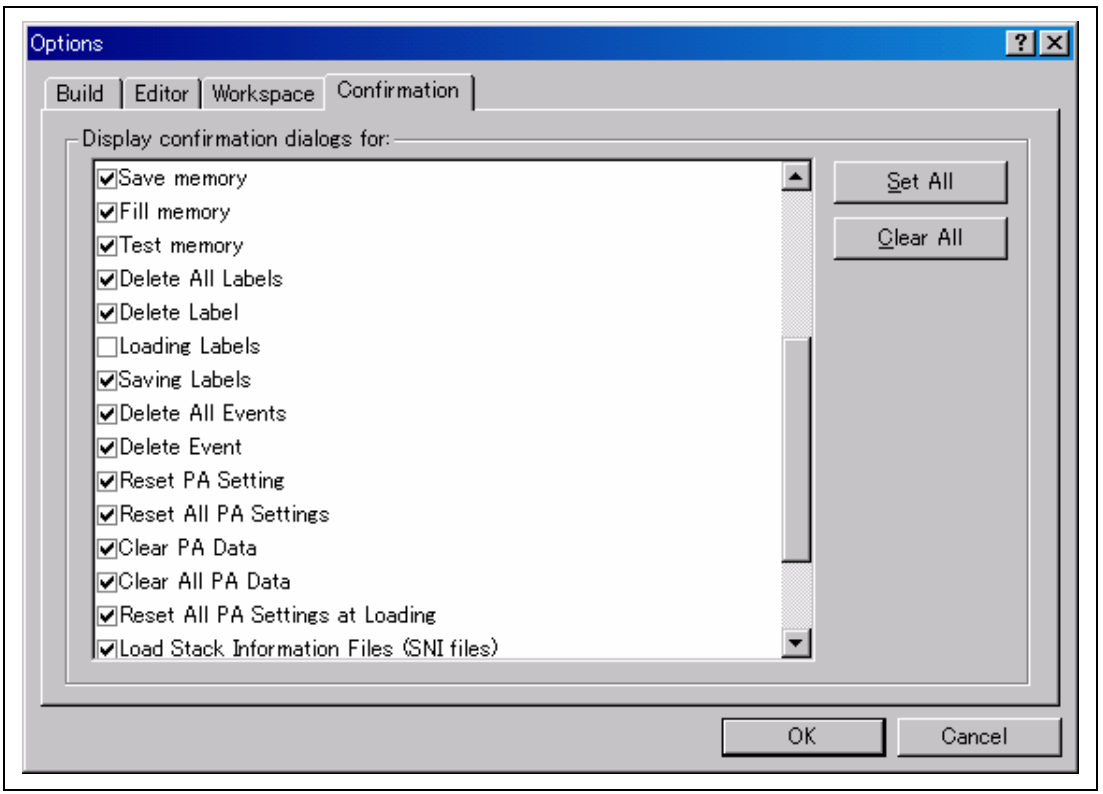


Figure 5.89 [Options] Dialog Box

5.9.4 Enabling the Profile

Choose [View->Performance->Profile] to open the [Profile] window.

Choose [Enable Profiler] from the pop-up menu of the [Profile] window. The item on the menu will be checked.

5.9.5 Specifying Measuring Mode

You can specify whether to trace functions calls while profile data is acquired. When function calls are traced, the relations of function calls during user program execution are displayed as a tree diagram. When not traced, the relations of function calls cannot be displayed, but the time for acquiring profile data can be reduced.

To stop tracing function calls, choose [Disable Tree (Not traces function call)] from the pop-up menu in the [Profile] window (a check mark is shown to the left of the menu item).

When acquiring profile data of the program in which functions are called in a special way, such as task switching in the operating system, stop tracing function calls.

5.9.6 Executing the Program and Checking the Results

After the user program has been executed and execution has been halted, the results of measurement are displayed in the [Profile] window.

The [Profile] window has two sheets; a [List] sheet and a [Tree] sheet.

5.9.7 [List] Sheet

This sheet lists functions and global variables and displays the profile data for each function and variable.

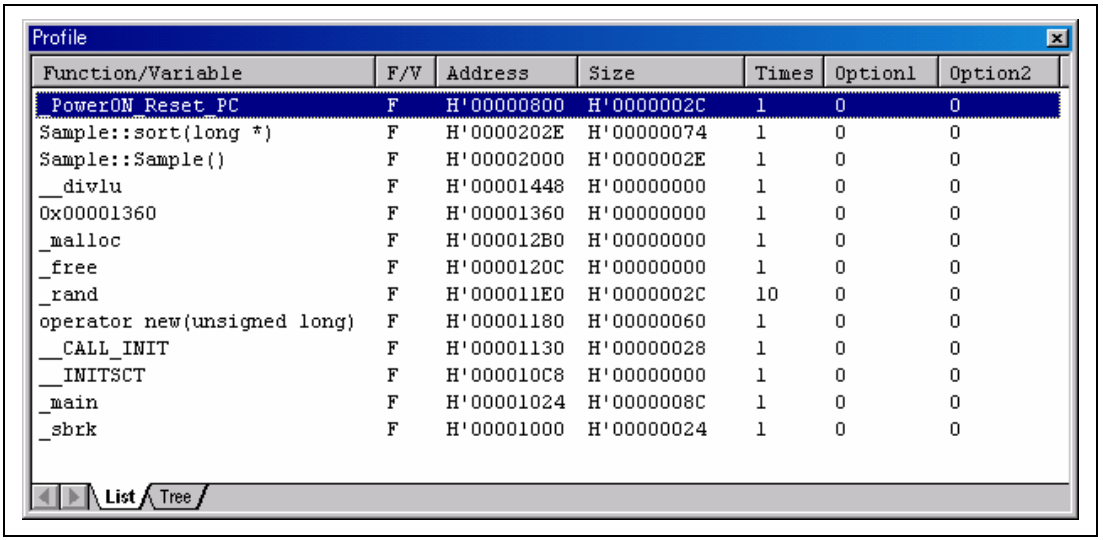


Figure 5.90 [List] Sheet

Clicking the column header sorts the items in an alphabetical or ascending order. Double-clicking the [Function/Variable] or [Address] column displays the source program corresponding to the address in the line.

Right-clicking on the mouse within the window displays a pop-up menu. For details on this pop-up menu, refer to section 5.9.8, [Tree] Sheet.

5.9.8 [Tree] Sheet

This sheet displays the relation of function calls along with the profile data that are values when the function is called. This sheet is available when [Disable Tree (Not traces function call)] is not selected from the pop-up menu in the [Profile] window.

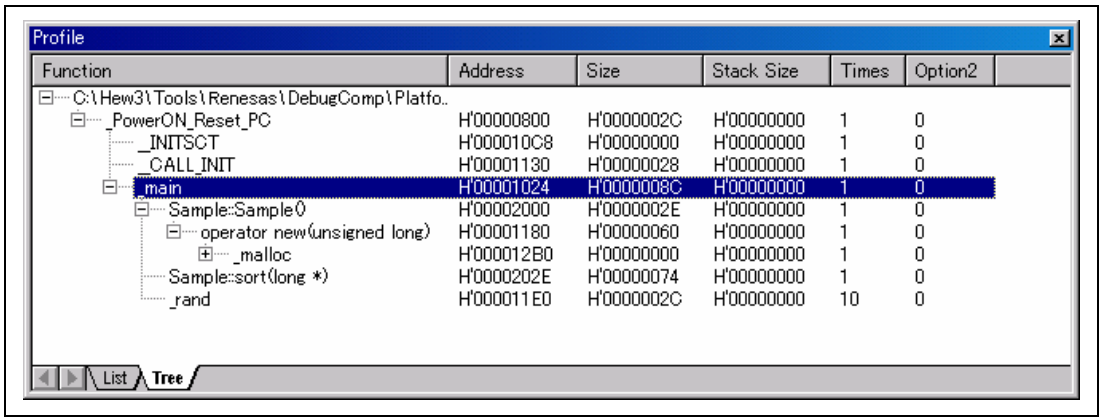


Figure 5.91 [Tree] Sheet

Double-clicking a function in the [Function] column expands or reduces the tree structure display. The expansion or reduction is also provided by the "+" or "-" key. Double-clicking the [Address] column displays the source program corresponding to the specific address.

Right-clicking on the mouse within the window displays a pop-up menu. Supported menu options are described in the following:

- View Source

  Displays the source program or disassembled memory contents for the address in the selected line.
- View Profile-Chart

  Displays the [Profile-Chart] window focused on the function in the specified line.
- Enable Profiler

  Toggles acquisition profile data. When profile data acquisition is active, a check mark is shown to the left of the menu text.
- Not trace the function call

  Stops tracing function calls while profile data is acquired. This menu is used when acquiring profile data of the program in which functions are called in a special way, such as task switching in the operating system.

To display the relation of function calls in the [Tree] sheet of the [Profile] window, acquire profile data without selecting this menu. In addition, do not select this menu when optimizing the program by the optimizing linkage editor using the acquired profile information file.

- Find…

  Displays the [Find Text] dialog box to find a character string in the [Function] column. Search is started by inputting a character string to be found in the edit box and clicking [Find Next] or pressing the Enter key.

- Find Data…

  Displays the [Find Data] dialog box.

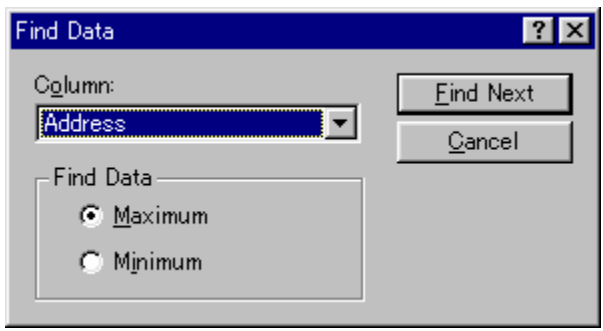


Figure 5.92 [Find Data] Dialog Box

By selecting the column to be searched in the [Column] combo box and the search type in the [Find Data] group and entering [Find Next] button or Enter key, search is started. If the [Find Next] button or the Enter key is input repeatedly, the second larger data (the second smaller data when the Minimum is specified) is searched for.

- Clear Data

  Clears the number of times functions are called and profile data. Data in the [List] sheet of the [Profile] window and the data in the [Profile-Chart] window are also cleared.

- Output Profile Information Files…

  Displays the [Save Profile Information Files] dialog box. Profiling results are saved in a profile information file (.pro extension). The optimizing linkage editor optimizes user programs according to the profile information in this file. For details of the optimization using the profile information, refer to the manual of the optimizing linkage editor.

Note: If profile information has been acquired by choosing the [Not trace the function call] menu, the program cannot be optimized by the optimizing linkage editor.

- Output Text File…

  Displays the [Save Text of Profile Data] dialog box. Displayed contents are saved in a text file.
- Setting

  This menu has the following submenus (the menus available only in the [List] sheet are also included).

  — Show Functions/Variables

  Displays both functions and global variables in the [Function/Variable] column.

  — Show Functions

  Displays only functions in the [Function/Variable] column.

  — Show Variables

  Displays only global variables in the [Function/Variable] column.

  — Only Executed Functions

  Only displays the executed functions. If a stack information file (.sni extension) output from the optimizing linkage editor does not exist in the directory where the load module is located, only the executed functions are displayed even if this check box is not checked.

  — Include Data of Child Functions

  Sets whether or not to display information for a child function called in the function as profile data.
- Properties...

  Sets the items to be measured.

5.9.9 [Profile-Chart] Window

The [Profile-Chart] window displays the relation of calls for a specific function. This window displays the specified function in the middle, with the callers of the function on the left and the callees of the function on the right. The numbers of times the function calls the called functions or is called by the calling functions are also displayed in this window.

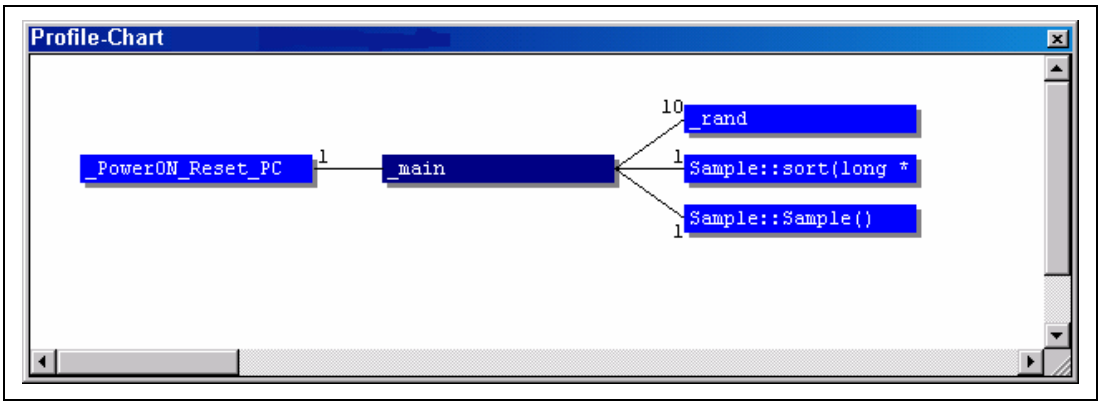


Figure 5.93 [Profile-Chart] Window

5.9.10 Types and Purposes of Displayed Data

The profiling function is able to acquire the following information:

Address	You can see the locations in memory to which the functions are allocated. Sorting the list of functions and global variables in order of their addresses allows the user to view the way the items are allocated in the memory space.
Size	Sorting in order of size makes it easy to find small functions that are frequently called. Setting such functions as inline may reduce the overhead of function calls. If you execute larger functions, more of the cache memory will need to be updated. This information allows you to check if those functions that may cause cache misses are frequently called.
Stack Size	When there is deep nesting of function calls, pursue the route of the function calls and obtain the total stack size for all of the functions on that route to estimate the amount of stack being used.
Times	Sorting by the number of calls or accesses makes it easy to identify the frequently called functions and frequently accessed global variables.

Profile Data — Measurement of a variety of CPU-specific data is also available as well as items that can be measured with the performance measurement function. For details, refer to the online help.

5.9.11 Creating Profile Information Files

To create a profile information file, choose the [Output Profile Information Files…] menu option from the pop-up menu. The [Save Profile Information Files] dialog box is displayed. Pressing the [Save] button after selecting a file name will write the profile information to the selected file. Pressing the [Save All] button will write the profile information to all of the profile information files.

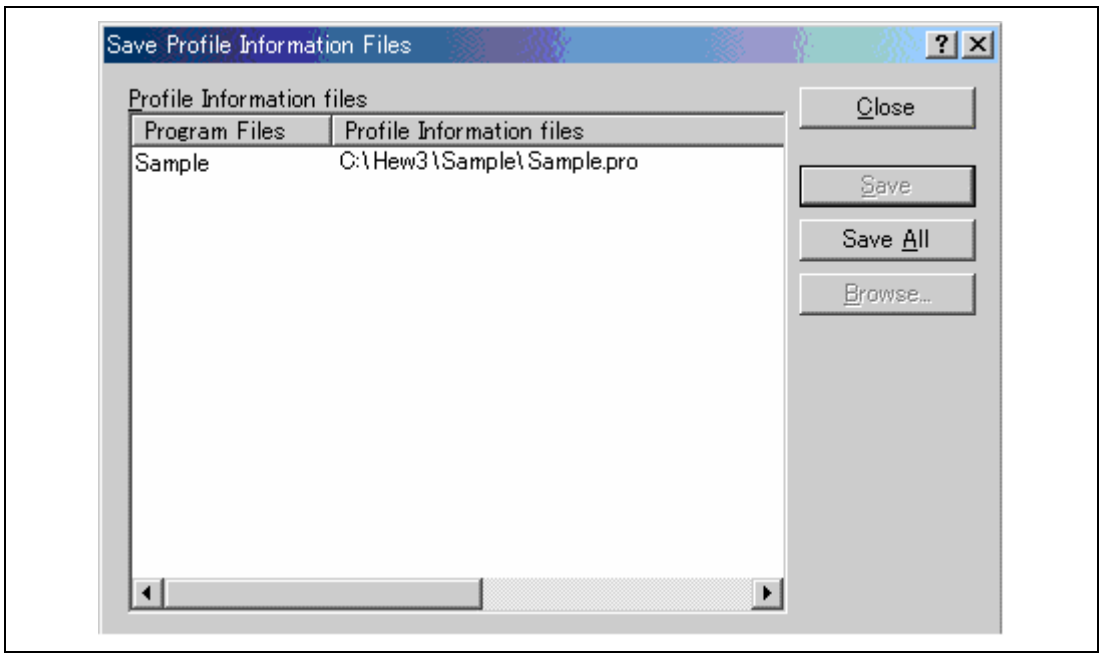


Figure 5.94 [Save Profile Information Files] Dialog Box

5.9.12 Notes

1. Tolerances

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

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

   The measured value of this function should be the reference.

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

   (a) On-chip performance analysis function

   The on-chip performance analysis function cannot be used when the profiling function is enabled.

   (b) Step function

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

   (c) Internal trace function

   When the profiling function is enabled, it is invalid to select the internal trace mode as all items of the internal trace mode are internally selected. Do not use the internal trace when the profiling function is enabled.

   (d) Halt function

   When the profiling function is enabled, do not use the halt function of the internal, AUD, external bus, and MFI traces.

   (e) Memory access during user program execution

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

   (f) When the profiling function is used, a break occurs if a branch instruction is generated. Accordingly, the realtime emulation will not be performed. In addition, since the emulator firmware is controlled on generation of a break, the executed result of the branch instruction may be displayed on the [Internal/AUD/Usermemory trace] window when the execution is returned to the user program from the emulator firmware. In this case, **EML** is displayed.

   (g) When the profiling function is used, do not use the function of Break Condition 3.

   (h) When the profiling function is used, do not set [Hardware break enable] for [Emulation mode] in the [Configuration] dialog box.

   (i) When the profiling function is used, the PC and SR values are displayed as undefined on the status bar.

3. Others
   (a) When the profiling function is used, an internal break occurs in the execution of the user program. Therefore, the measurement results of AUD performance analysis, bus performance analysis, and realtime profile will contain errors.
   (b) When the profiling function is used, the contents that have been set in the on-chip performance measurement function or data that has been measured will be deleted.
   (c) Since the profiling function is implemented with the internal break, it takes a long time to start and end the user program execution. The stopping time of the user program depends on the performance of the host machine in use or the JTAG clock; that time under the following environment is shown below:
   Environment:
   Host machine: 850 MHz (Pentium® III)
   Memory: 256 Mbytes
   SH7323: 10 MHz (TCK clock)
   OS: Windows® 2000
   Execution program: 10,000 nested calls
      (i) When the profiling function is not used: 1 second or lower
      (ii) When the profiling function is used in the setting without including a child function: 63 seconds
      (iii) When the profiling function is used in the setting including a child function: 72 seconds
   (d) When the profiling function is used, the trace settings are internally changed and the [Internal trace] mode is set. Note that a part of emulator functions cannot be used if the mode is not changed to [AUD trace].

5.10 Viewing Realtime Profile Information

The realtime profiling function is used to measure the execution performance in the specified range of the application program in a function unit. This function is effective to investigate the position and the cause of the lowered performance in the application program.

The realtime profiling function does not affect the realtime operation because it measures the performance by using the profiler measurement circuit on the main unit case and expansion profiling unit of the emulator.

The following shows the realtime profiling measurement modes:

- Function mode
  This function does not include the subroutine execution time when accumulation of the function execution time is displayed.
- Nest mode
  This function includes the subroutine execution time when accumulation of the function execution time is displayed.

Determine which mode is to be used in the [Function select] dialog box that is displayed when the emulator is activated.

Note: There are following restrictions in the realtime profiling function.

(1) Restrictions on all the realtime profiling functions

(i) Areas to be measured

In the emulator, 512 Kbytes are considered as a unit to acquire the profiling information on all the functions in the areas of a maximum of 24 blocks.

The hardware of the emulator has a maximum of three types of memory for measuring eight blocks to implement the realtime profiling function.

Note that the adjacent address areas can be set in each block, however, it is impossible to set a function of which address ranges are extended on the eight-block boundaries. If such a function is set, a warning message will be displayed and correct measurement will not be performed.

(ii) Inline expansion

When the functions are inline-expanded in accordance with optimization of the compiler, they are not displayed in the [Realtime Profile] window.

(iii) Recursive function

The execution time of the recursive function can be correctly measured, however, the execution count will be once.

(iv) AUD trace

The realtime profiling function uses the data that is output in AUD trace. Therefore, when the function is used in the realtime trace mode, the trace data may be lost and correct measurement will not be performed. In such a case, it is recommended that the non-realtime trace mode be used.

(v) Overlay program

No measurement will be performed if a part of the program is rewritten due to an overlay.

(2) Restrictions on using the function mode

(i) Tail call

When a tail call is used for a function as shown below, the return value of the callee function becomes the return address of the caller function. The execution time or execution counts of the callee function cannot be measured correctly.

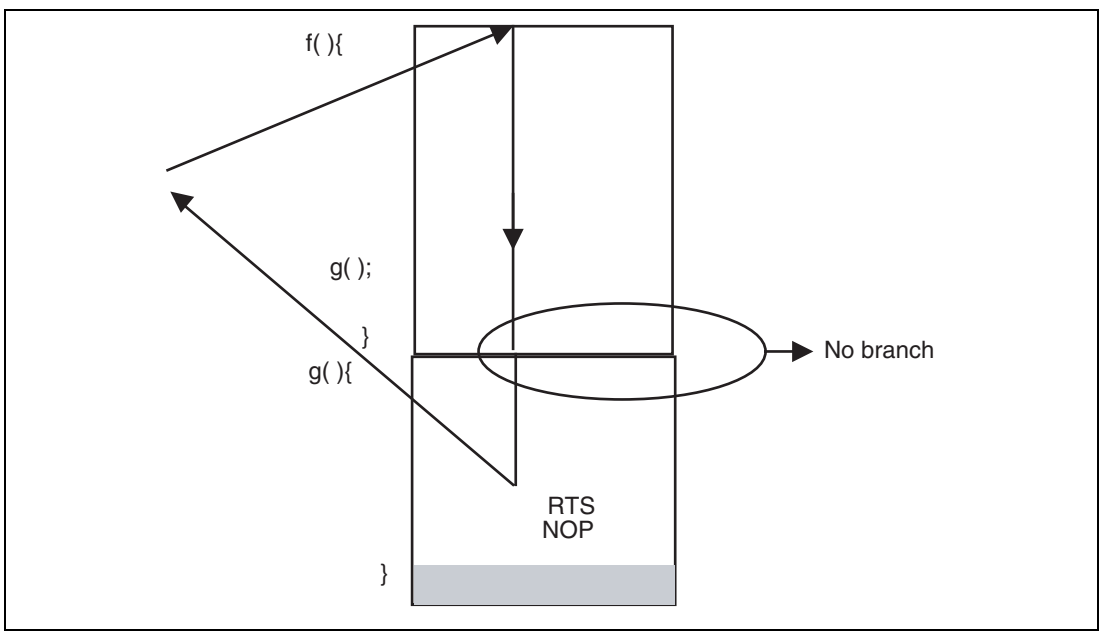


Figure 5.95 Tail Call (Function Mode)

(ii) Relationships between the Go-start address, break address, and measurable ranges

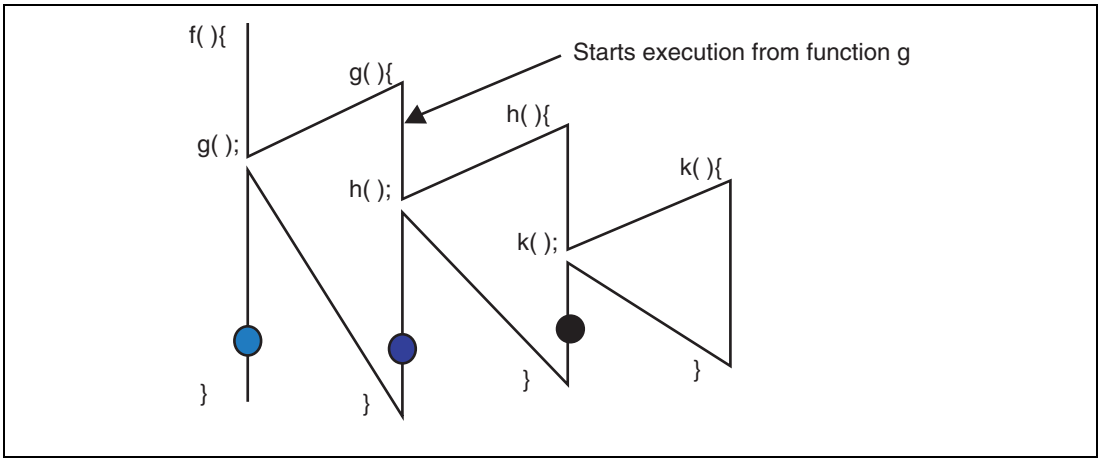


Figure 5.96 Measurable Ranges (Function Mode)

Measurable ranges when a break occurs at the position of black circle:

- Execution time and counts of functions h and k

Measurable ranges when a break occurs at the position of red circle:

- Execution time and counts of functions h and k

Measurable ranges when a break occurs at the position of blue circle:

- Execution time and counts of functions h and k
- Execution time of function g; counts cannot be measured.

It is recommended that a break occur within the function where execution is started. When execution returns to the upper function, the execution counts of the function cannot be measured.

(3) Restrictions on using the nest mode

(i) Tail call

— When a tail call is used for a function as shown below, the return value of the callee function becomes the return address of the caller function. The execution time of the callee function cannot be measured correctly but the execution counts can be masured.

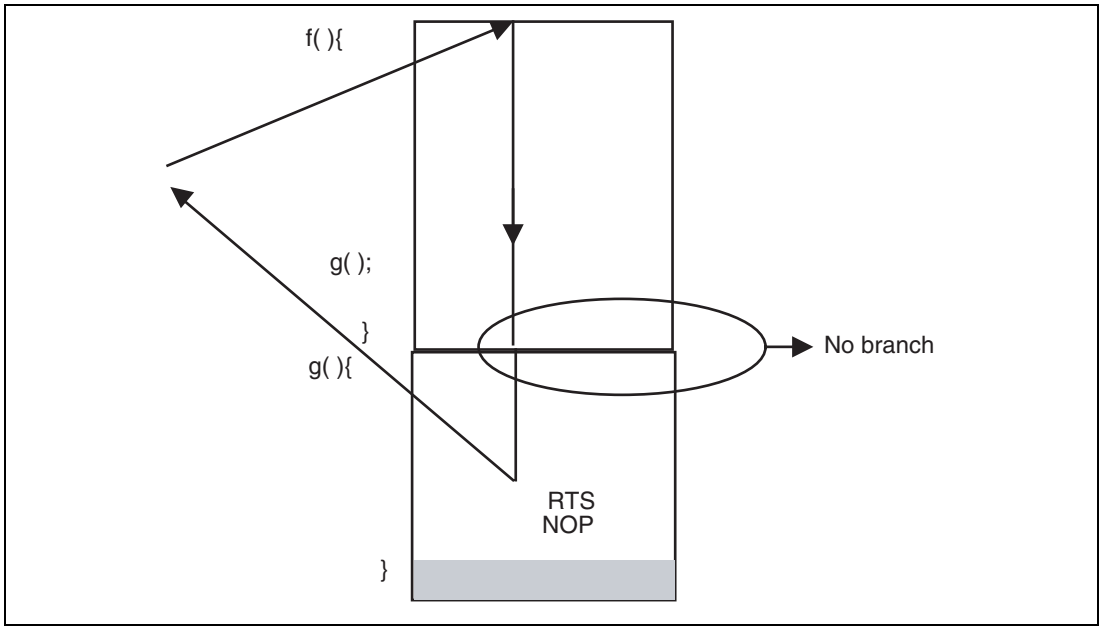


Figure 5.97 Tail Call (Nest Mode)

— There is a restriction when the tail call is used to call another function from the function that has been called by the tail call. If the tail call occurs continuously, three-step measurement will be correctly performed.

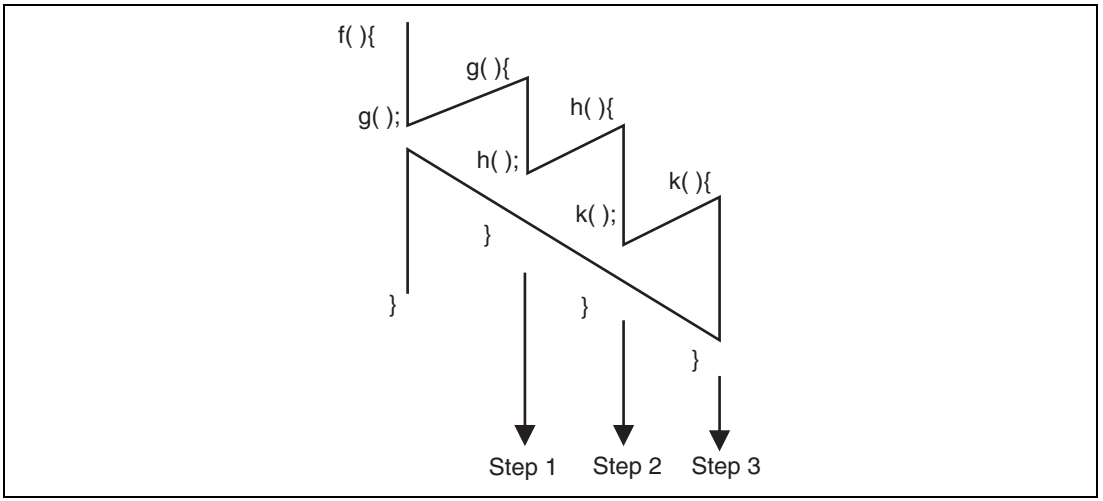


Figure 5.98 Restriction on Call

(ii) Nesting functions

If 32-step or more calls are generated from the top function within the range to be measured, correct measurement will not be performed. A warning message will be displayed.

(iii) Calling from a function outside the measurement range

Correct measurement will not be performed if a function outside the measurement range calls a function to be measured and the callee function cannot return correctly to the caller function.

Even if the callee function returns correctly to the caller function, correct measurement will not be performed if other functions are called within three instructions from the return address.

The user program execution starts from a function outside the measurement range and then moves to a function within the measurement range. After that, the execution branches to another function outside the measurement range.

(iv) Relationships between the Go-start address, break address, and measurable ranges

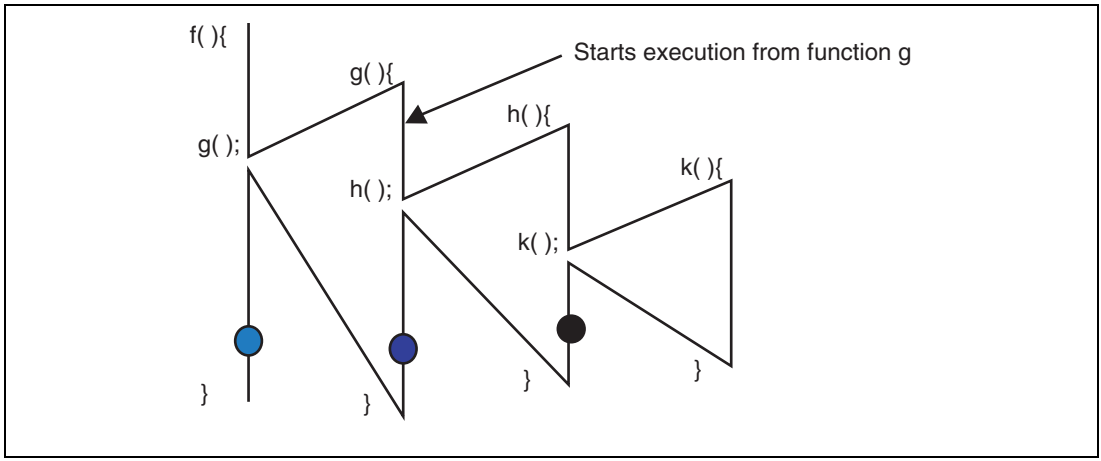


Figure 5.99 Measurable Ranges (Nest Mode)

Measurable ranges when a break occurs at the position of black circle:

- Execution time and counts of function h

Measurable ranges when a break occurs at the position of red circle:

- Execution time and counts of functions h and k

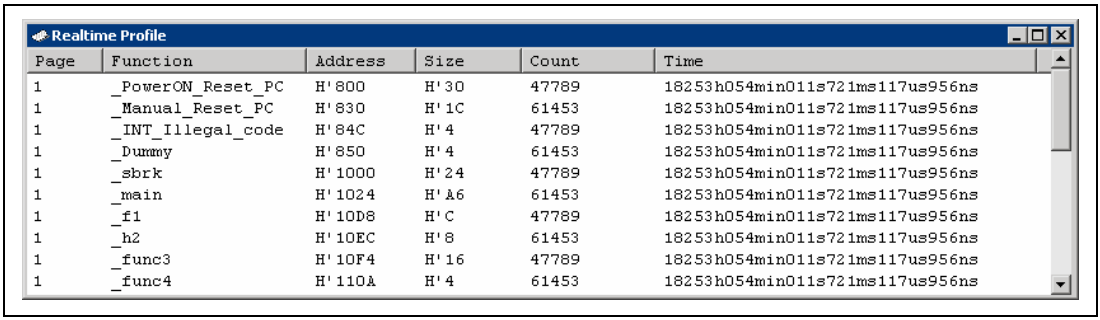
Measurable ranges when a break occurs at the position of blue circle:

- Execution time and counts of functions h and k

It is recommended that a break occur within the function where execution is started. When execution returns to the upper function, the execution counts of the function cannot be measured.

5.10.1 Opening the [Realtime Profile] Window

Select [View -> Performance -> Realtime Profile] to open the [Realtime Profile] window. Clicking the column header changes the ascending or descending order of items in displayed contents.



Page	Function	Address	Size	Count	Time
1	_PowerON_Reset_PC	H'800	H'30	47789	18253h054min011s721ms117us956ns
1	_Manual_Reset_PC	H'830	H'1C	61453	18253h054min011s721ms117us956ns
1	_INT_Illegal_code	H'84C	H'4	47789	18253h054min011s721ms117us956ns
1	_Dummy	H'850	H'4	61453	18253h054min011s721ms117us956ns
1	_sbrk	H'1000	H'24	47789	18253h054min011s721ms117us956ns
1	_main	H'1024	H'A6	61453	18253h054min011s721ms117us956ns
1	_f1	H'10D8	H'C	47789	18253h054min011s721ms117us956ns
1	_h2	H'10EC	H'8	61453	18253h054min011s721ms117us956ns
1	_func3	H'10F4	H'16	47789	18253h054min011s721ms117us956ns
1	_func4	H'110A	H'4	61453	18253h054min011s721ms117us956ns

Figure 5.100 [Realtime Profile] Window

The following information can be acquired:

Address — You can see the locations in memory to which the functions are allocated.

Size — Sorting in order of size makes it easy to find small functions that are frequently called. Setting such functions as inline may reduce the overhead of function calls. If you execute larger functions, more of the cache memory will need to be updated. This information allows you to check if those functions that may cause cache misses are frequently called.

Count — The number of times that functions have been called is displayed.

Times — The total execution time is displayed.

5.10.2 Specifying the Measurement Range

Clicking the right-hand mouse button on the window displays the popup menu. When [Add Range] is selected from the popup menu, the [EDIT] dialog box opens to specify the measurement range of realtime profile.

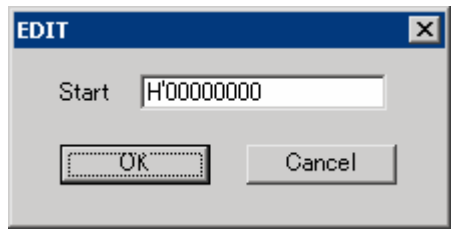


Figure 5.101 [EDIT] Dialog Box

When the expansion profiling unit is connected to the emulator, it is possible to measure the profile information within the 6-Mbyte ranges in total. When the expansion profiling unit is not connected to the emulator, it is possible to measure the profile information within the 2-Mbyte ranges in total.

In the emulator, 512 Kbytes are considered as a unit to acquire the profiling information on all the functions in the eight-block areas. Addresses to be specified for each block need not be adjacent. When the expansion profiling unit is connected to the emulator, 16 blocks of 512 Kbytes are added and the 24-block areas can be measured in total.

The measured data in each block is displayed on each page of the [Realtime Profile] window. To switch the page to be displayed, select that page and click the [OK] button in the [Select Page] dialog box that is opened by selecting [Select Page] from the popup menu of the [Realtime Profile] window.

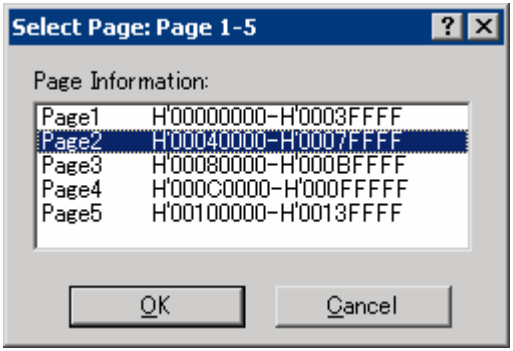


Figure 5.102 [Select Page] Dialog Box

5.10.3 Starting Measurement

When the user program is executed, measurement is started.

When the user program is halted, the measurement result is displayed in the [Realtime Profile] window.

5.10.4 Clearing Measurement Result

Selecting [Clear Data] from the popup menu clears the measurement result of the [Column] and [Time] columns.

5.10.5 Deleting Measurement Range

Selecting [Delete] from the popup menu deletes all the specified measurement range and clears the measurement result.

5.10.6 Setting the Minimum Unit of the Measurement Time

In the emulator, it is possible to change the minimum unit of the measurement time as any of 20 ns, 40 ns, 100 ns, or 400 ns.

At 20 ns, the maximum time that can be measured is about three hours.

At 40 ns, the maximum time that can be measured is about six hours.

At 100 ns, the maximum time that can be measured is about 15 hours.

At 400 ns, the maximum time that can be measured is about 61 hours.

To change the minimum unit, select [Set] from the popup menu of the [Realtime Profile] window and open the [Properties] dialog box.

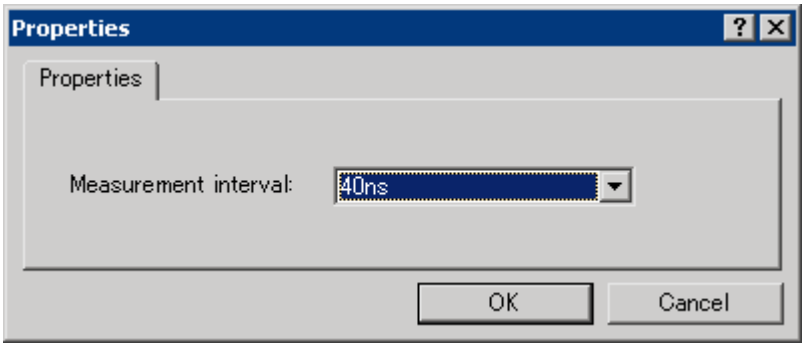


Figure 5.103 [Properties] Dialog Box

Note: Be sure to set the following when using the realtime profiling function:

- [Acquisition] dialog box that is opened by selecting [Set] from the popup menu of the [Trace] window:
  [Trace type]: Select [AUD Trace].
  [Trace Extend Mode]: Uncheck the [Trace data with PPC] check box.
- Ch12 of the [Eventpoint] window:
  [Branch event] sheet: check all the branch conditions.
  [Action] sheet: Check [Acquire trace].

These settings above are default when the emulator is activated.

5.11 Acquiring Code Coverage

The emulator acquires code coverage information (C0 coverage) from the address range specified by the user and displays the results.

Notes: 1. The information on the executed delay slot cannot be acquired.

2. Be sure to set the following when using the code coverage function:
   - [Acquisition] dialog box that is opened by selecting [Set] from the popup menu of the [Trace] window:
     [Trace type]: Select [AUD Trace].
     [Trace Extend Mode]: Uncheck the [Trace data with PPC] check box.
   - Ch12 of the [Eventpoint] window:
     [Branch event] sheet: check all the branch conditions.
     [Action] sheet: Check [Acquire trace].

   These settings above are default when the emulator is activated.
3. In this function, the data that is output by AUD trace is used. Accordingly, when the emulator is used in realtime trace mode, the trace data may be lost and the code coverage information will not be collected normally. In such a case, it is recommended to use the emulator in non-realtime trace mode.
   By default, the emulator is activated in realtime trace mode.

5.11.1 Opening the [Code Coverage] Window

Select [View -> Code -> Code Coverage…] or click the [Code Coverage] toolbar button (). The following dialog box opens when the coverage function is used for the first time after initiation of the emulator.

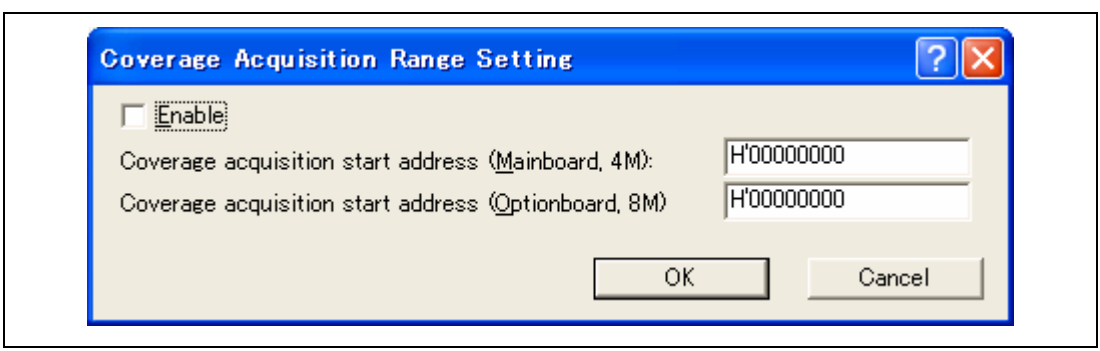


Figure 5.104 [Coverage Acquisition Range Setting] Dialog Box

[Enable]: Checking this box allows acquisition of code coverage information.

[Coverage acquisition start address (Mainboard, 4M)]:
The emulator specifies the measurement start address in the range of 4 Mbytes.

[Coverage acquisition start address (Optionboard, 8M)]:
The emulator specifies the measurement start address in the range of 8 Mbytes. This function is only available when the expansion profiling unit is connected to the emulator.

[OK]: Closes this dialog box by setting the specified condition.

[Cancel]: Closes this dialog box without setting the specified condition.

This dialog box will only be displayed once after initiation of the emulator. To open this dialog box, select [Hardware Settings…] from the popup menu of the [Code Coverage] window. When the [Coverage Acquisition Range Setting] dialog box is closed, the [Open Coverage] dialog box appears.

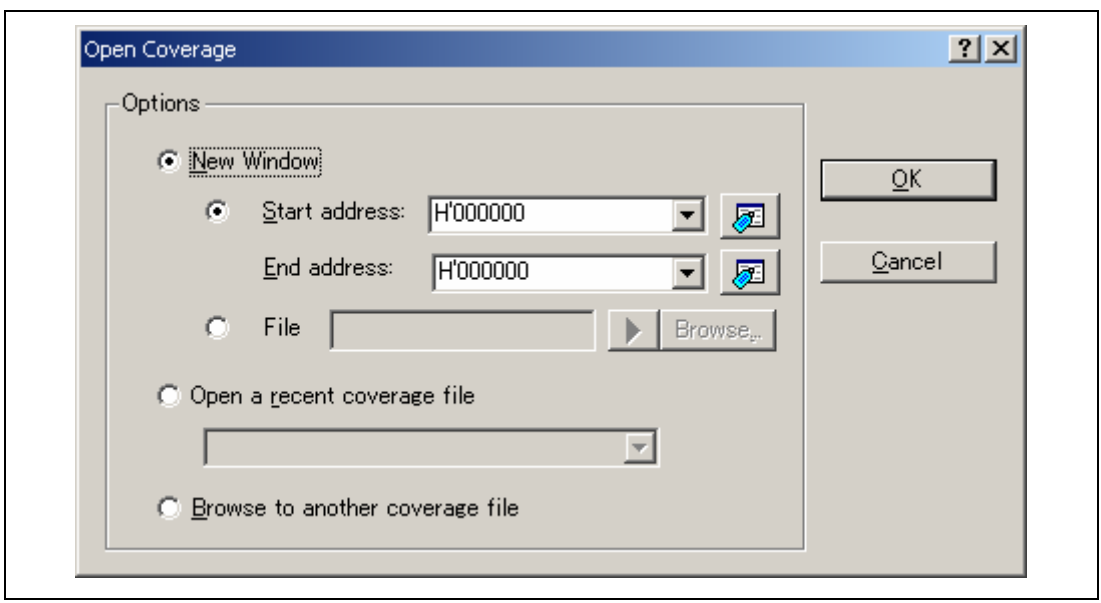


Figure 5.105 [Open Coverage] Dialog Box

In this dialog box, specify a range to be shown in the [Code Coverage] window.

[New Window]: Specifies a new coverage range.

[Start address]: Enter the start address of coverage information to be displayed (in hexadecimal when the prefix is omitted).

[End address]: Enter the end address of coverage information to be displayed (in hexadecimal when the prefix is omitted).

[File]: Specifies a source file that has ".C" or ".CPP" as its file type name from the current project. This allows a function included in the specified file to be set as the coverage range. ".C" will be automatically specified if the file type name is omitted. Files with a type name other than ".C" or ".CPP" cannot be specified here. A placeholder or the [Browse…] button can be used.

[Open a recent coverage file]: Lists a maximum of four files that were saved most recently. Select one from the list.

[Browse to another coverage file]: Allows specification of a file that is not included in the list.

Clicking the [OK] button in this dialog box opens the [Code Coverage] window. The display format differs depending on whether the address range or source file is specified.

Note: In one [Coverage window], up to 2-Mbyte address ranges can be opened. To check the whole measurement range, use multiple windows to show the addresses separately.

[Code Coverage] Window (Specifying an Address)

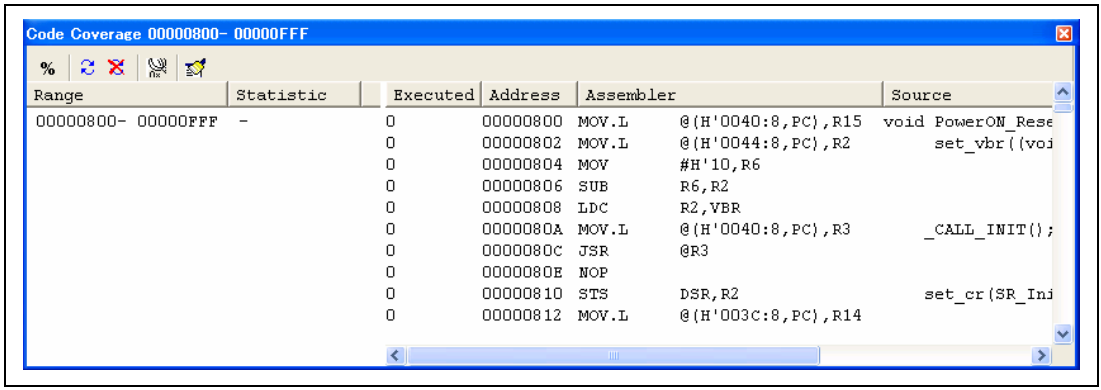


Figure 5.106 [Code Coverage] Window (Specifying an Address)

The [Code Coverage] window is divided into two by the splitter.

- **Left-hand window**

Displays the coverage range and statistical information of the coverage. The items are displayed as follows:

[Range]: Address range

[Statistic]: C0 coverage value (in percentage)

The percentage will be displayed in the [Statistic] column by selecting [Percentage] from the popup menu on the left-hand window.

Figure 5.107 Percentage Shown in the [Code Coverage] Window

- **Right-hand window**

Displays coverage information at the C/C++ or assembly-language level. The items are displayed as follows:

[Executed]: 1 (the instruction was executed) or 0 (the instruction was not executed)

[Address]: Instruction address

[Assembler]: Disassembled program

[Source]: C/C++ or assembly source program

[Code Coverage] Window (Specifying a Source File)

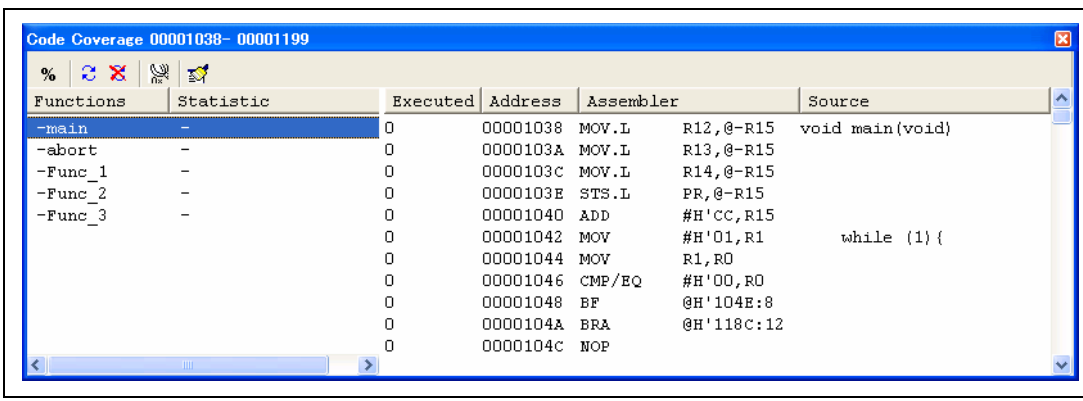


Figure 5.108 [Code Coverage] Window (Specifying a Source File)

This window is divided into two by the splitter.

- **Left-hand window**

Displays the coverage range and statistical information of the coverage. The items are displayed as follows:

[Functions]: Function to be selected for coverage

[Statistic]: C0 coverage value (in percentage)

The percentage will be displayed in the [Statistic] column by selecting [Percentage] from the popup menu in the left-hand window. Clicking the column tab allows the function names or C0 coverage values to be listed in descending or ascending order.

- **Right-hand window**

Displays coverage information of the function selected by double-clicking on the left-hand window at the C/C++ or assembly-language level. The items are displayed as follows:

[Executed]: 1 (the instruction was executed) or 0 (the instruction was not executed)

[Address]: Instruction address

[Assembler]: Disassembled program

[Source]: C/C++ or assembly source program

5.11.2 Displaying a Source File

Selecting [Display a Source File] from the popup menu opens the [Editor] window, which displays the source file for the address at the cursor location on the [Code Coverage] window.

5.11.3 Changing the Address to be Displayed

Selecting [Go To Address…] from the popup menu opens the [Go To Address] dialog box, which allows the user to change the address to be displayed in the [Code Coverage] window.

Figure 5.109 [Go To Address] Dialog Box

5.11.4 Changing the Coverage Range to be Displayed

- **When the coverage range for display has been specified by address**

Selecting [Set Range…] from the popup menu opens the [Coverage Display Range] dialog box, which allows the user to change the condition for acquisition of information on executed instructions.

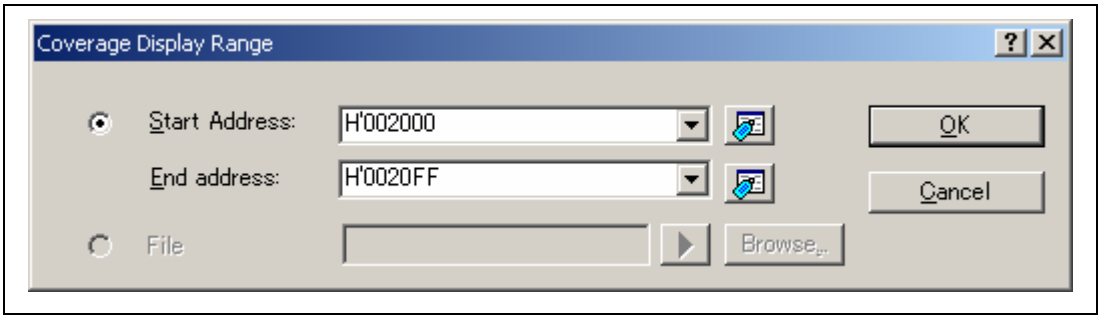


Figure 5.110 [Coverage Display Range] Dialog Box

The following items can be specified.

[Start Address]: Start address (in hexadecimal when the prefix is omitted)

[End Address]: End address (in hexadecimal when the prefix is omitted)

Clicking the [OK] button changes the coverage range to be displayed.

- **When the coverage range for display has been specified by a source file**

Selecting [Set Range…] from the popup menu opens the [Coverage Display Range] dialog box, which allows the user to change the condition for acquisition of information on executed instructions.

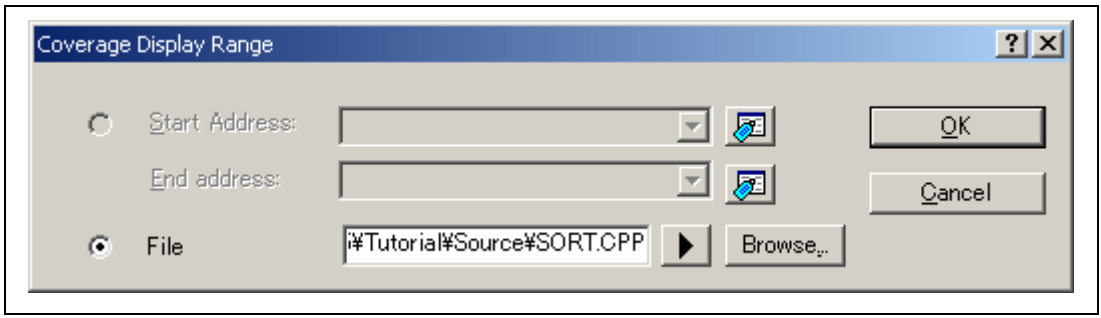


Figure 5.111 [Coverage Display Range] Dialog Box (Specifying a Source File)

The following item can be specified.

[File]: Specifies a source file that has ".C" or ".CPP" as its file type name from the current project. This allows a function included in the specified file to be set as the coverage range. ".C" will be provided if the file type name is omitted. Files with a type name other than ".C" or ".CPP" cannot be specified here. A placeholder or the [Browse…] button can be used.

Clicking the [OK] button changes the coverage range to be displayed.

5.11.5 Clearing the Coverage Information

- **Clearing the coverage information of the specified range**

Selecting [Clear Coverage Range…] from the popup menu opens the [Clear Coverage Range] dialog box.

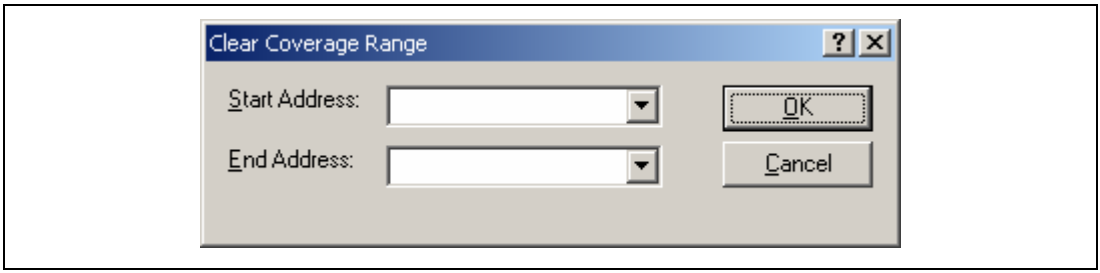


Figure 5.112 [Clear Coverage Range] Dialog Box

Enter the start and end addresses of the range to be cleared. Clicking the [OK] button clears the coverage information of the selected range.

- **Clearing all the coverage information**

Selecting [Clear the Entire Coverage] from the popup menu clears all the coverage information.

5.11.6 Saving the Coverage Information to a File

Selecting [Save Data…] from the popup menu opens the [Save Coverage Data] dialog box.

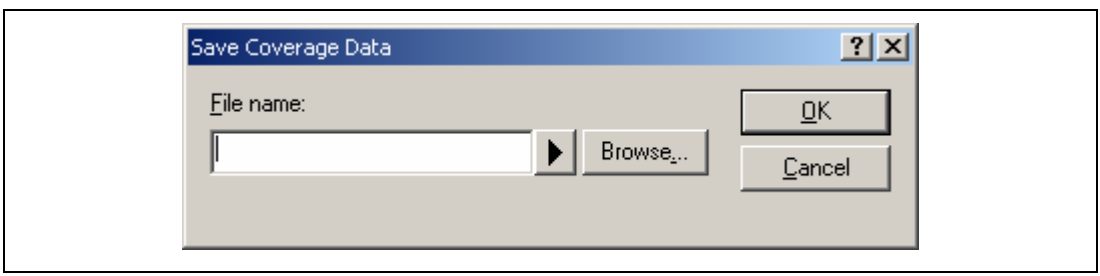


Figure 5.113 [Save Coverage Data] Dialog Box

Specify the location of the coverage information file to be saved and its name. A placeholder or the [Browse…] button can be used. If the file extension is omitted, “.COV” will automatically be added as the file extension. An error message will appear if a file extension other than “.COV” or “.TXT” is entered.

5.11.7 Loading Coverage Information from a File

Selecting [Load Data…] from the popup menu opens the [Load Data] dialog box.

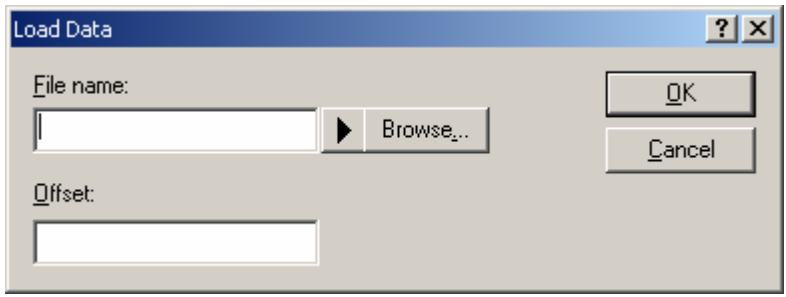


Figure 5.114 [Load Data] Dialog Box

Specify the location of the coverage information file to be loaded. A placeholder or the [Browse…] button can be used. The only file extension available is ".COV". An error message will appear if any other file extension is entered.

Note: If the coverage range has been specified by a source file, the saved ".COV" file cannot be loaded.

5.11.8 Updating Coverage Information

Selecting [Refresh] from the popup menu updates the content of the [Code Coverage] window.

5.11.9 Preventing Update of Coverage Information

Selecting [Lock Refresh] from the popup menu prevents update of the [Code Coverage] window while the user program execution is stopped. This also prevents access to the emulator for acquisition of coverage information.

5.11.10 [Confirmation Request] Dialog Box

- **Clearing code coverage information or closing the [Code Coverage] window**

A confirmation dialog box appears before clearing code coverage information or closing the [Code Coverage] window.

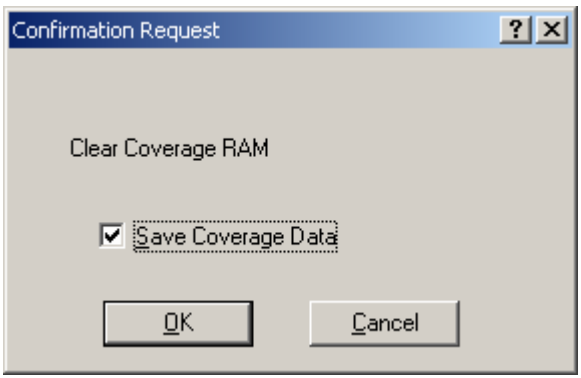


Figure 5.115 [Confirmation Request] Dialog Box

When [Save Coverage Data] is checked, the coverage data can be saved into a file before clearing. Clicking the [OK] button clears the coverage information.

- **When [File -> Save Session] has been selected**

When one or more [Code Coverage] windows are open, the number of [Save Code Coverage] dialog boxes to be opened is the same as that of the [Code Coverage] window(s). The data of the windows can be saved separately or together.

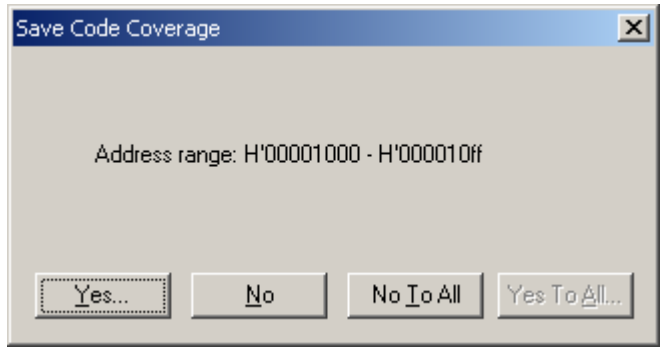


Figure 5.116 [Save Code Coverage] Dialog Box

Clicking [No To All] closes this dialog box without saving any coverage information. Clicking [Yes To All] saves data of all the [Code Coverage] windows into one file.

5.11.11 Displaying Code Coverage Information in the [Editor] Window

Highlighting the Code Coverage column that corresponds to a source line where an instruction has been executed allows the coverage information to also be displayed in the [Editor] window. If the user changes any setting regarding the coverage information in the [Code Coverage] window, the content of the corresponding Code Coverage column will also be updated.

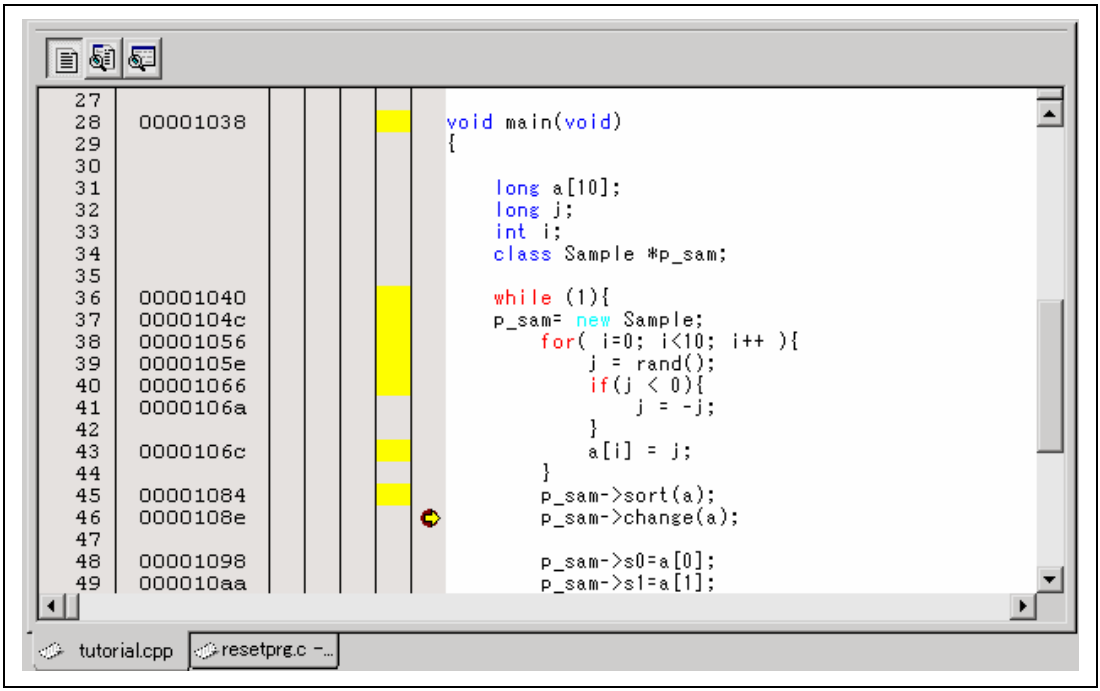


Figure 5.117 Code Coverage Column

5.12 Synchronizing Multiple Debugging Platforms

Multiple debugging platforms can be operated at the same time in the High-performance Embedded Workshop. Initiating a High-performance Embedded Workshop from another High-performance Embedded Workshop synchronizes multiple debugging platforms. The High-performance Embedded Workshop that initiates another High-performance Embedded Workshop is called the master, and the initiated High-performance Embedded Workshop is called the slave. Choose [Tools -> Launch Slave HEW...] or click the [Launch Slave HEW] toolbar button () to initiate a slave High-performance Embedded Workshop.

The slave High-performance Embedded Workshop has the same functionality as the master High-performance Embedded Workshop.

The slave High-performance Embedded Workshop is notified of the following actions done in the master High-performance Embedded Workshop to ensure synchronization of the slave High-performance Embedded Workshop and the master High-performance Embedded Workshop.

- Reset go
- Go
- Stop debugging

Note: The master High-performance Embedded Workshop can initiate multiple slave High-performance Embedded Workshop applications, but slave High-performance Embedded Workshop applications cannot be nested (no slave High-performance Embedded Workshop can initiate another slave High-performance Embedded Workshop).

When selecting [Renesas High-performance Embedded Workshop] -> [High-performance Embedded Workshop] from [Programs] in the [Start] menu to initiate another High-performance Embedded Workshop, two emulators can be used separately for debugging.

5.12.1 Distinguishing Two Emulators

Connect two emulators to the USB connector. Then, initiate the High-performance Embedded Workshop using the tutorial workspace. The following message is displayed.

Figure 5.118 Message for Driver Selection

Click the [OK] button. The following dialog box will appear.

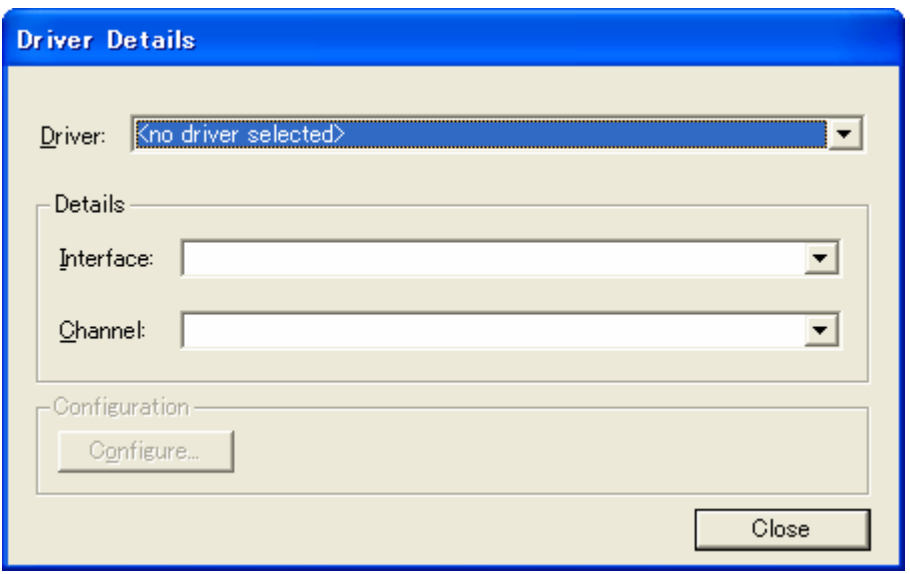


Figure 5.119 [Driver Details] Dialog Box (1)

Select [Renesas E-Series USB Driver] from the [Driver] drop-down list box and open the [Channel] drop-down list box. Channel information for two emulators is displayed in the [Channel] drop-down list box as shown in figure 5.120.

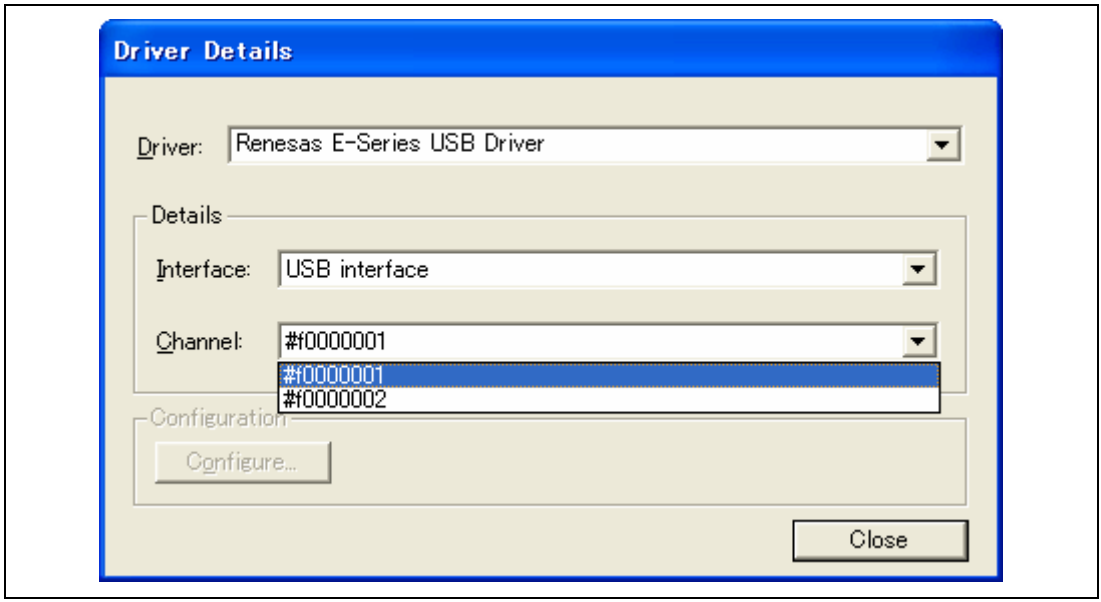


Figure 5.120 [Driver Details] Dialog Box (2)

Information displayed in figure 5.120 is that of the serial number of the emulator.

When initiating the High-performance Embedded Workshop, in the [Channel] drop-down list box, select the information on the serial number of the emulator that is connected to the master CPU. Initiate the High-performance Embedded Workshop using the normal procedures.

When initiating the slave High-performance Embedded Workshop, in the [Channel] drop-down list box, select the information on the serial number of the emulator that is connected to the slave CPU.

Section 6 Tutorial

6.1 Introduction

This section describes the main functions of the emulator by using a tutorial program.

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

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

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

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

2. This section describes general usage examples for the emulator. For the specifications of particular products, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help.
3. The operation address of `Tutorial.abs` attached to each product differs depending on the product. Replace the address used in this section with upper 16 bits of the actually loaded address.
   Example: Although the PC address is H'0000006c in the manual, enter H'0C00xxxx when the loaded address of `Tutorial.abs` is H'0C00006c (upper bit H'0000 is changed to H'0C00).

6.2 Running the High-performance Embedded Workshop

To run the High-performance Embedded Workshop, refer to section 4.1, System Check.

6.3 Setting up the Emulator

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

- AUD clock
  A clock used in acquiring AUD traces.
  If its frequency is set too low, complete data may not be acquired during realtime tracing.
  Set the frequency not to exceed the upper limit for the MPU's AUD clock.
  The AUD clock is only needed for using emulators that have an AUD trace function.
- JTAG (H-UDI) clock (TCK)
  A communication clock used except for acquiring AUD trace.
  If its frequency is set too low, the speed of downloading will be lowered.
  Set the frequency not to exceed the upper limit for the MPU's guaranteed TCK range.

For details of the limitations on both clocks, refer to section 2.2.2, Notes on Using the JTAG (H-UDI) Clock (TCK) and AUD Clock (AUDCK), in the additional document, Supplementary Information on Using the SHxxxx. The following is a description of the procedure used to set the clocks.

6.4 Setting the [Configuration] Dialog Box

- Select [Emulator] then [Systems…] from the [Options] menu to set a communication clock. The [Configuration] dialog box is displayed.

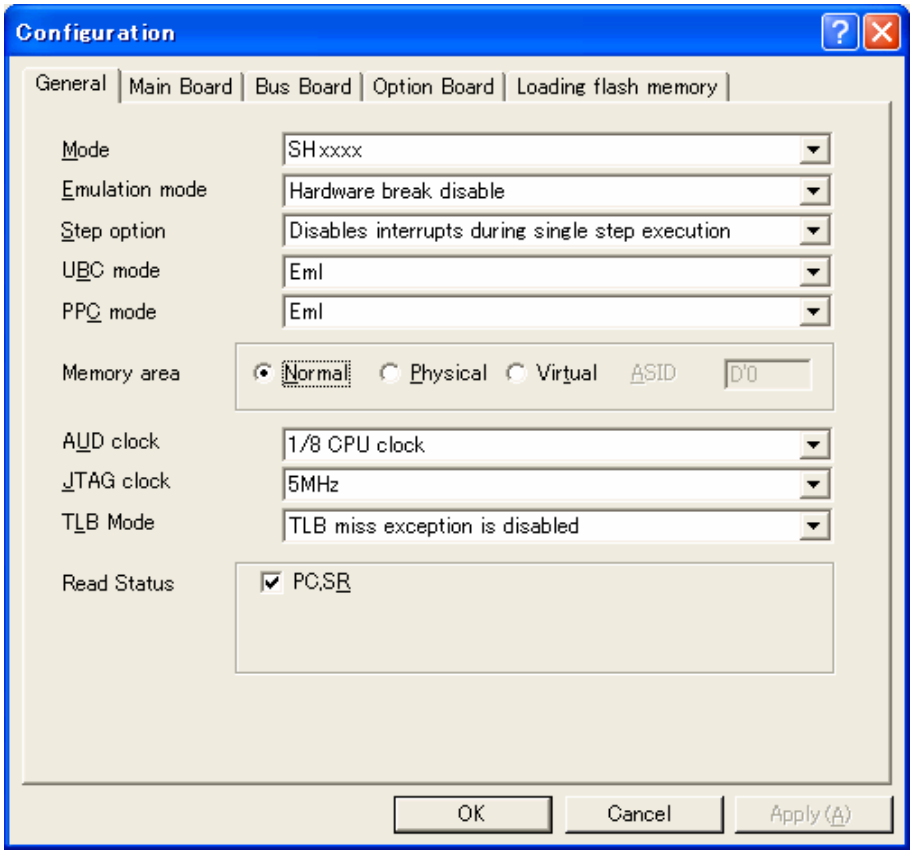


Figure 6.1 [Configuration] Dialog Box

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

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

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

6.5 Checking the Operation of the Target Memory for Downloading

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

When the destination memory is SDRAM or DRAM, a register in the bus controller of the MPU must be set before downloading. Set the bus controller correctly in the [IO] window according to the memory type to be used.

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

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

- Select [Memory…] from the [CPU] submenu of the [View] menu and enter ***H'00000000*** in the [Display Address] edit box.

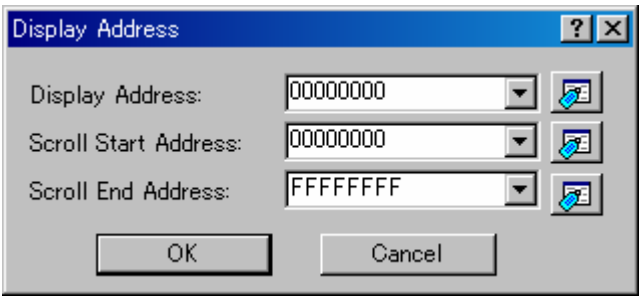


Figure 6.2 [Display Address] Dialog Box

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

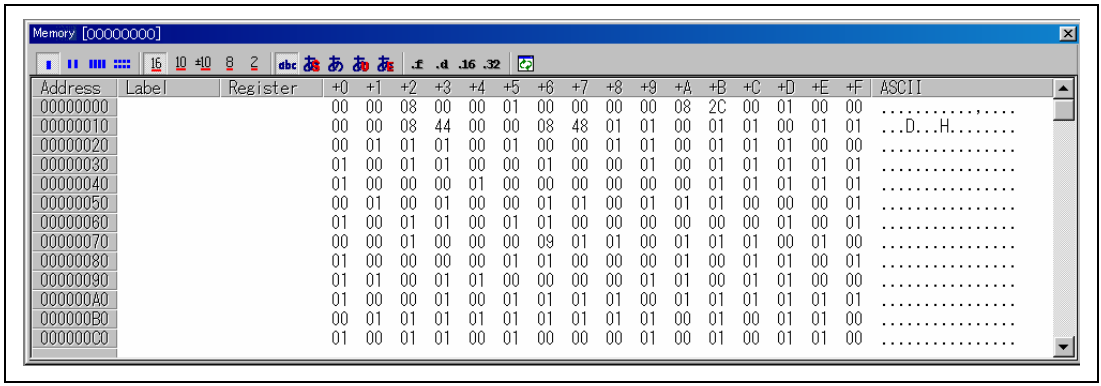


Figure 6.3 [Memory] Window

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

For details on the usage of the [Memory] window, such as modifying the displayed data size, refer to the High-performance Embedded Workshop User's Manual.

6.6 Downloading the Tutorial Program

6.6.1 Downloading the Tutorial Program

Download the object program to be debugged.

- Select [Download module] from [Tutorial.abs] under [Download modules].

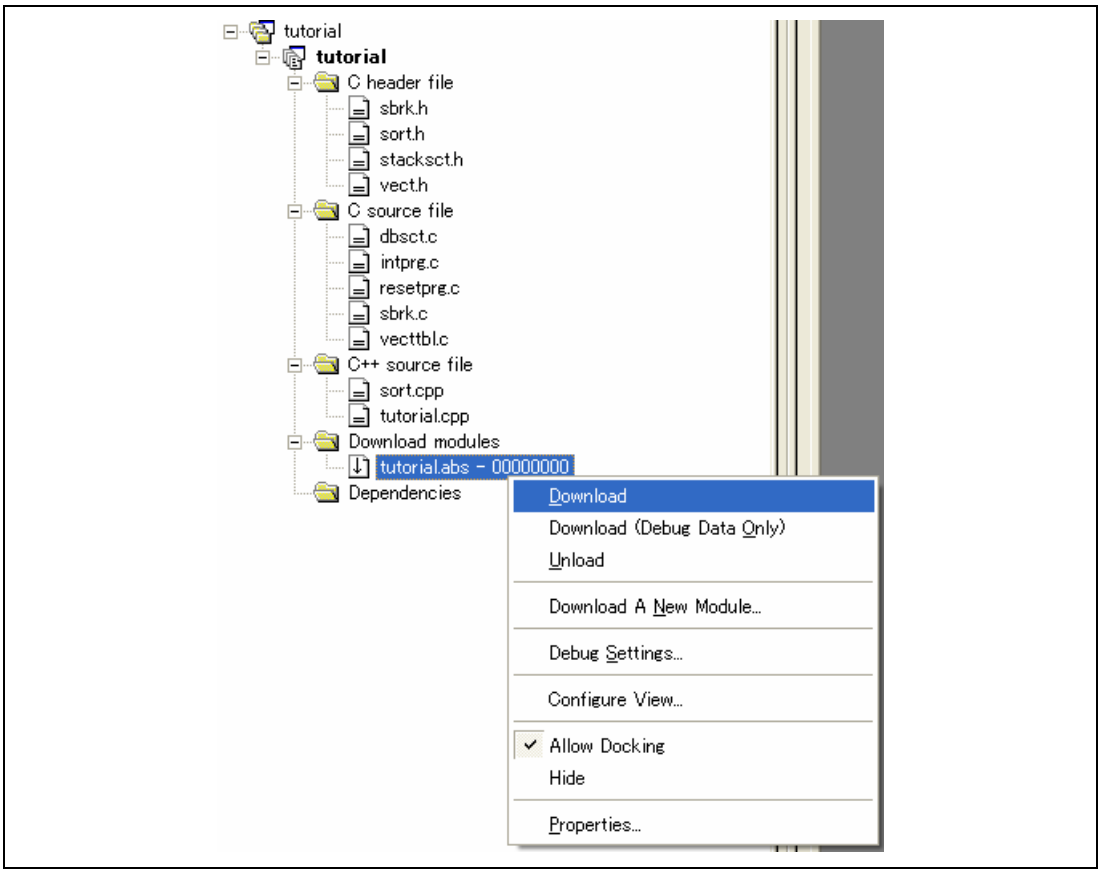


Figure 6.4 Downloading the Tutorial Program

6.6.2 Displaying the Source Program

The High-performance Embedded Workshop allows the user to debug a user program at the source level.

- Double-click [tutorial.cpp] under [C++ source file].

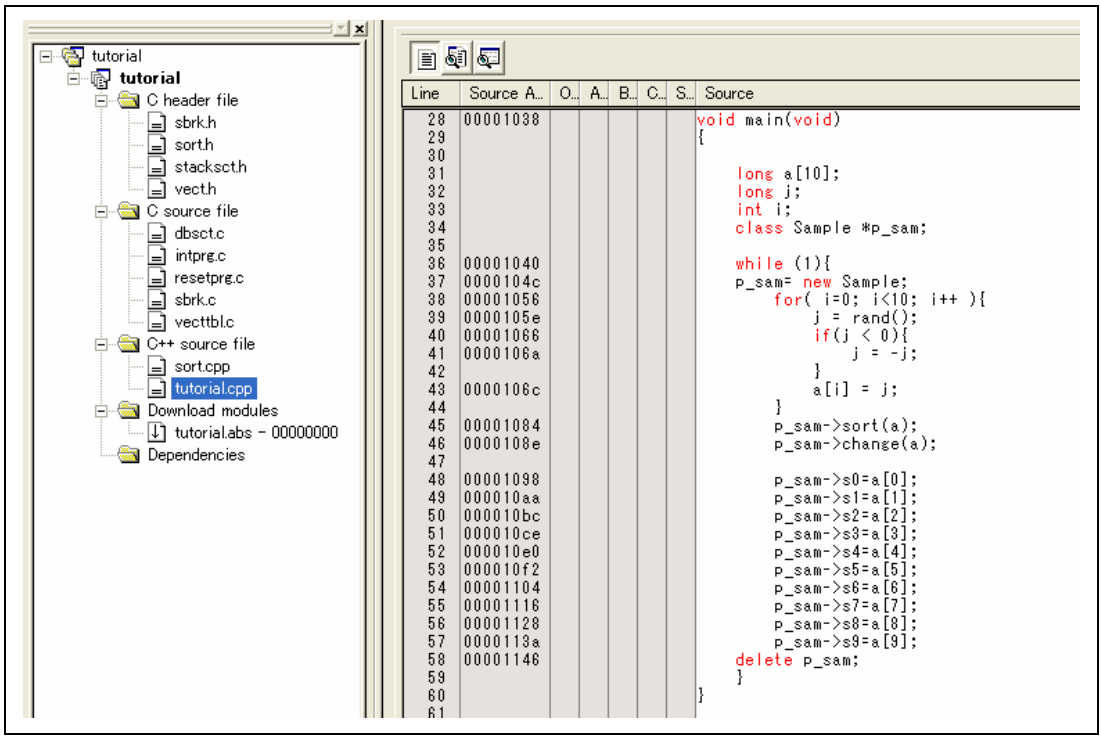


Figure 6.5 [Source] Window (Displaying the Source Program)

- Select a font and size that are legible from the [Format Views…] option in the [Setup] menu if necessary.

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

6.7 Setting a PC Breakpoint

A PC breakpoint is a simple debugging function.

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

- Select by double-clicking the [S/W Breakpoints] column on the line containing the `sort` function call.

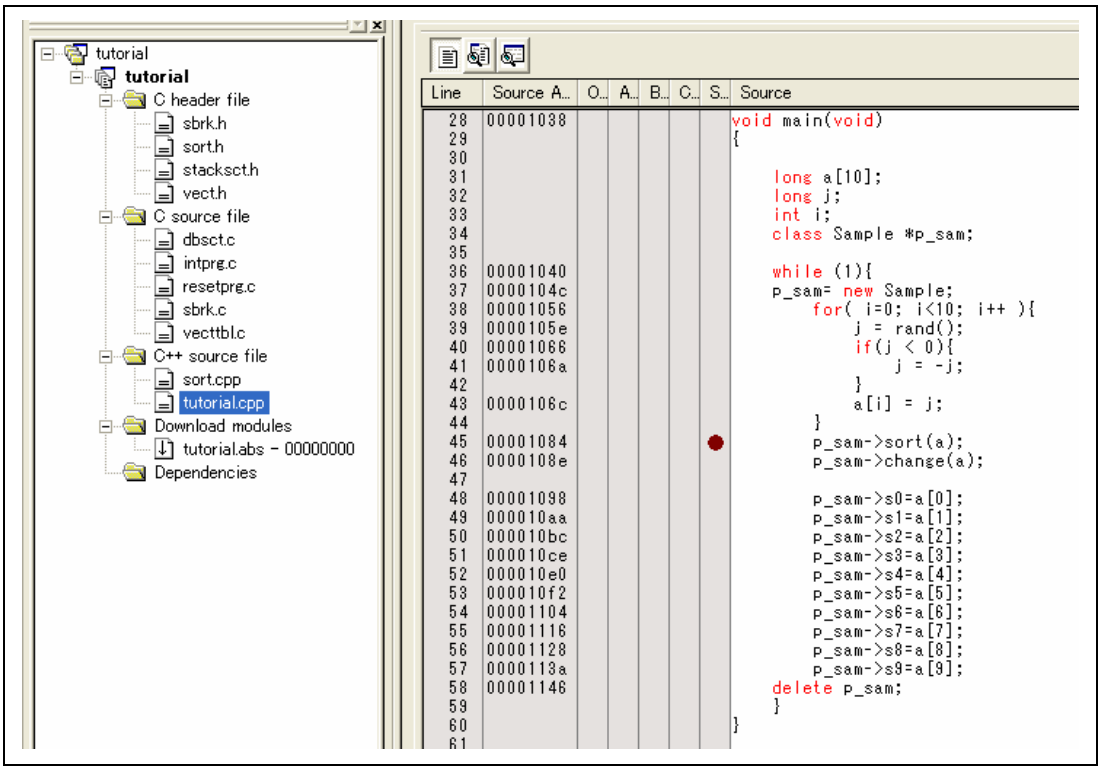


Figure 6.6 [Source] Window (Setting a PC Breakpoint)

The symbol • will appear on the line containing the `sort` function. This shows that a PC breakpoint has been set.

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

6.8 Setting Registers

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

- Select [Registers] from the [CPU] submenu of the [View] menu. The [Register] window is displayed.

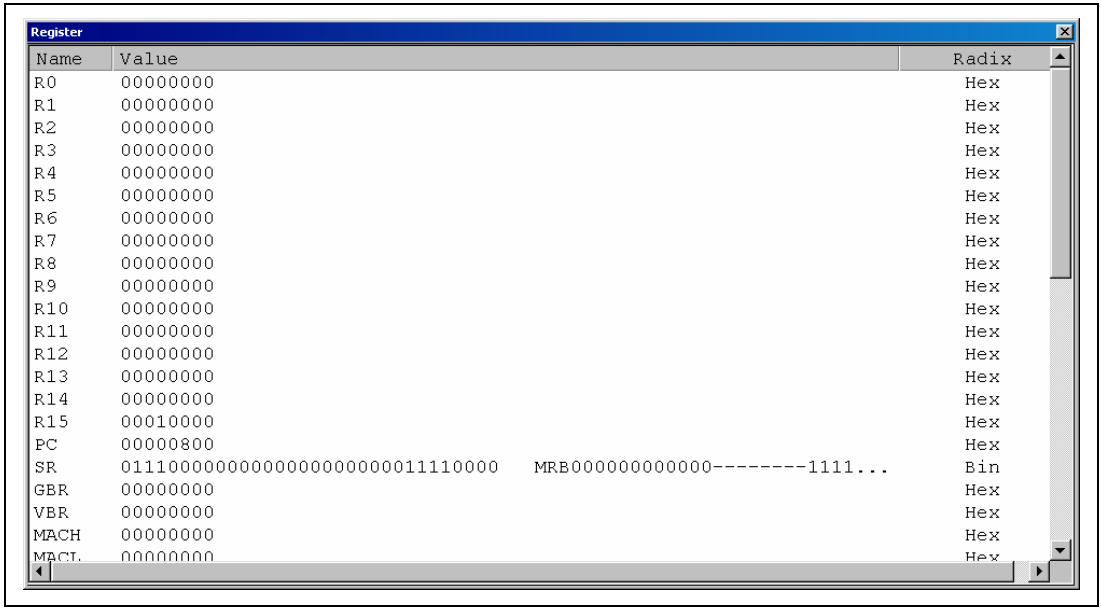


Figure 6.7 [Register] Window

- To change the value of the program counter (PC), double-click the value area in the [Register] window with the mouse. The following dialog box is then displayed, and the value can be changed. Set the program counter to H'00000800 in this tutorial program, and click the [OK] button.

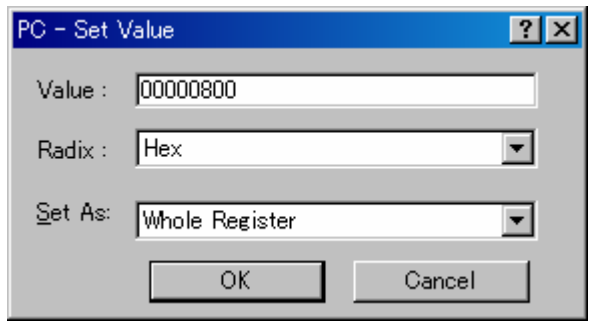


Figure 6.8 [Register] Dialog Box (PC)

- Change the value of the stack pointer (SP) in the same way. Set H'00010000 for the value of the stack pointer in this tutorial program.

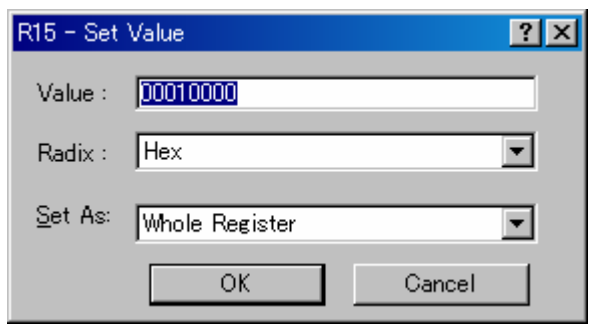


Figure 6.9 [Register] Dialog Box (R15)

6.9 Executing the Program

Execute the program as described in the following:

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

Figure 6.10 [Go] Button

When the program execution is started, `'**RUNNING'` is displayed on the status bar, and then the executed PC address is displayed. The program will be executed up to the breakpoint that has been set, and an arrow will be displayed in the [Source] column to show the position that the program has halted, with the message `[BREAKPOINT]` in the status bar.

Notes: 1. When the source file is displayed after a break, a path of the source file may be inquired. The location of the source file is as follows:
<Drive where the OS has been installed>:
\Workspace\Tutorial\E200F\xxxx\Tutorial\source

2. If program execution is failed, select [Reset CPU] from the [Debug] menu, reset the device, and restart the procedure from figure 6.8.

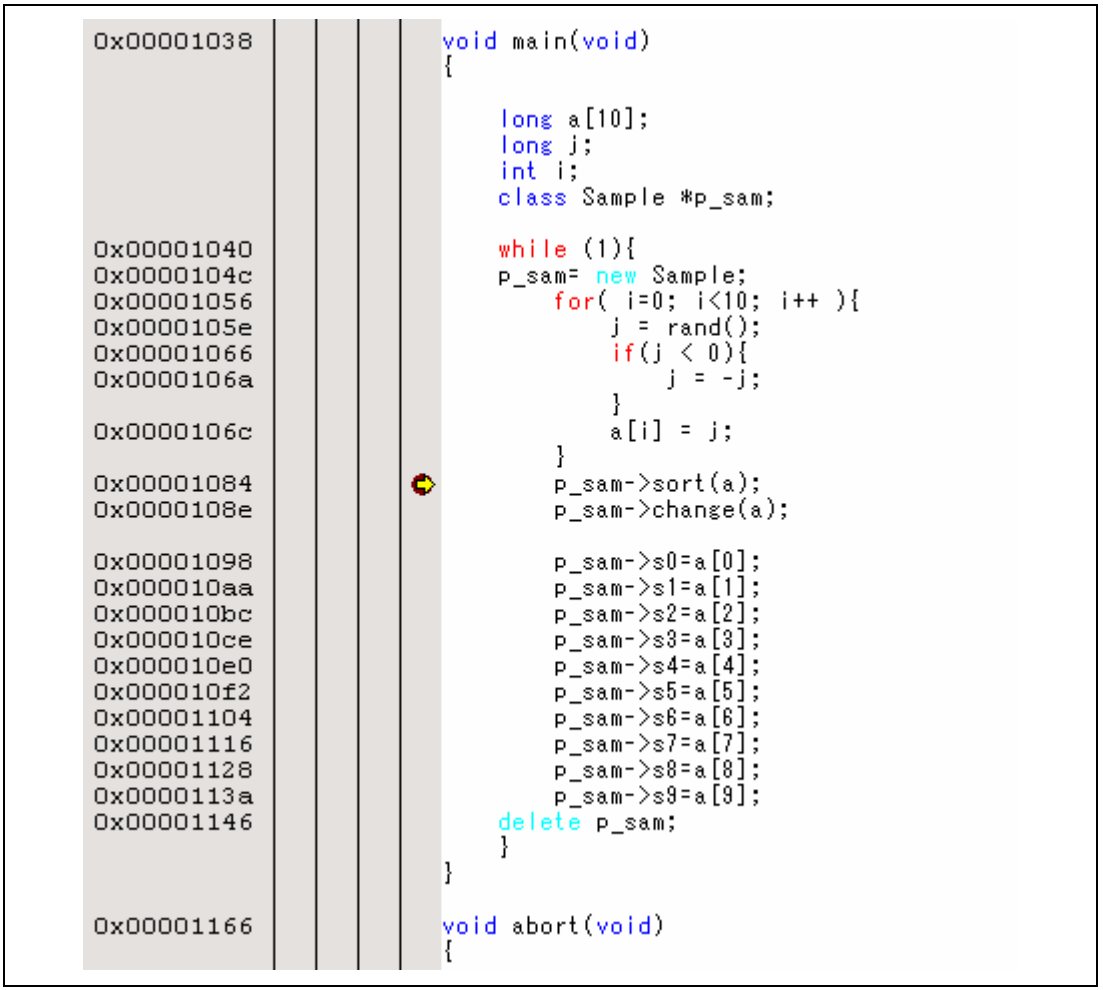


Figure 6.11 [Source] Window (Break State)

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

- Select [Status] from the [CPU] submenu of the [View] menu. After the [Status] window is displayed, open the [Platform] sheet, and check the `Status` of `Cause of last break`.

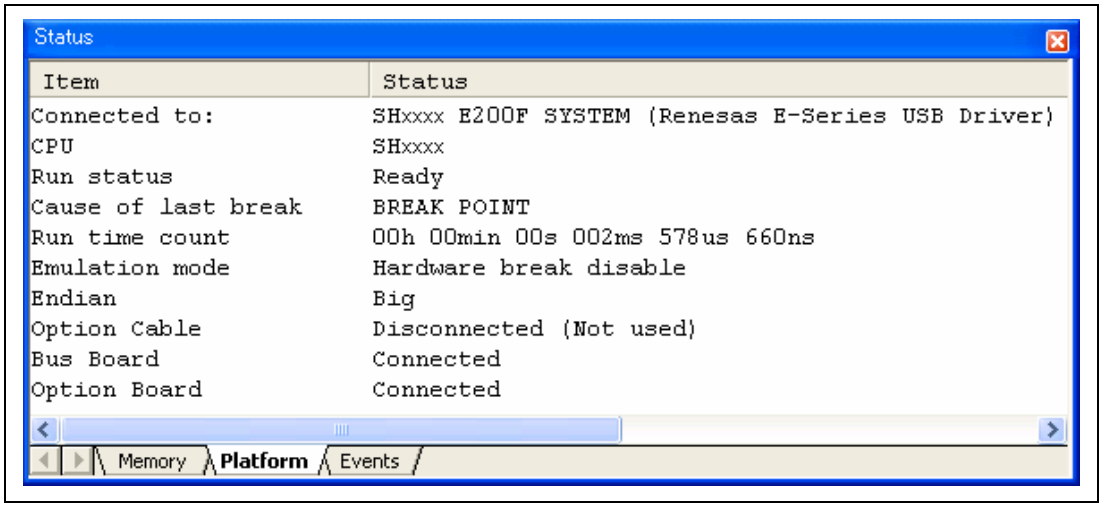


Figure 6.12 [Status] Window

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

6.10 Reviewing Breakpoints

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

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed. Select the [Breakpoint] sheet.

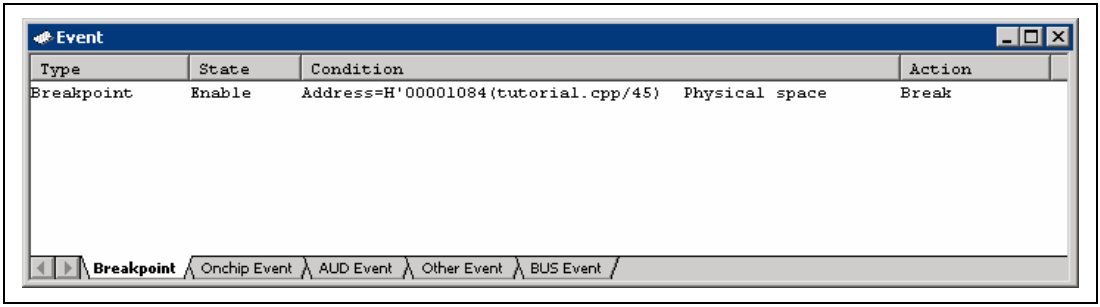


Figure 6.13 [Event] Window

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

6.11 Referring to Symbols

The [Label] window can be used to display the information on symbols in modules.

Select [Label] from the [Symbol] submenu of the [View] menu. The [Label] window is displayed so that the user can refer to the addresses of symbols in modules.

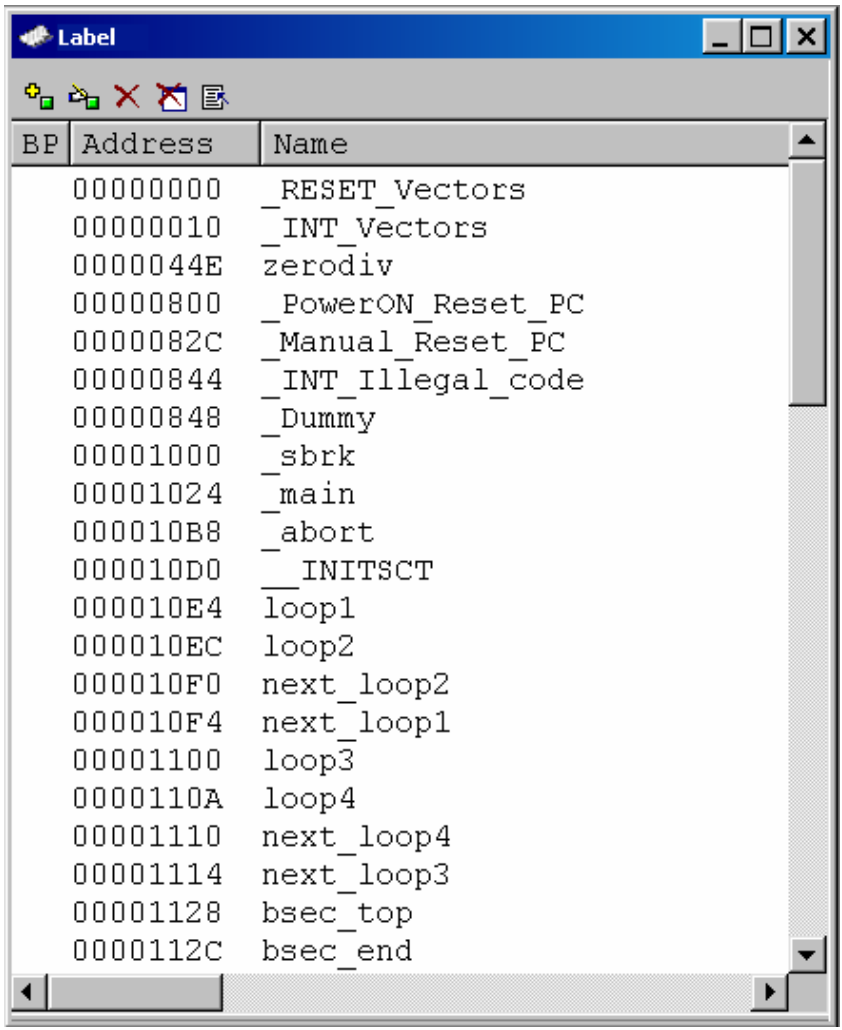


Figure 6.14 [Label] Window

6.12 Viewing Memory

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

- Select [Memory …] from the [CPU] submenu of the [View] menu, enter ***_main*** in the [Display Address] edit box.

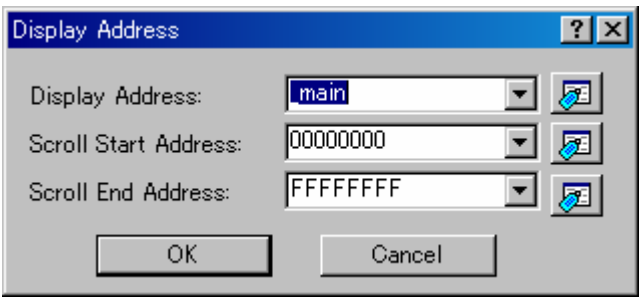


Figure 6.15 [Format] Dialog Box

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

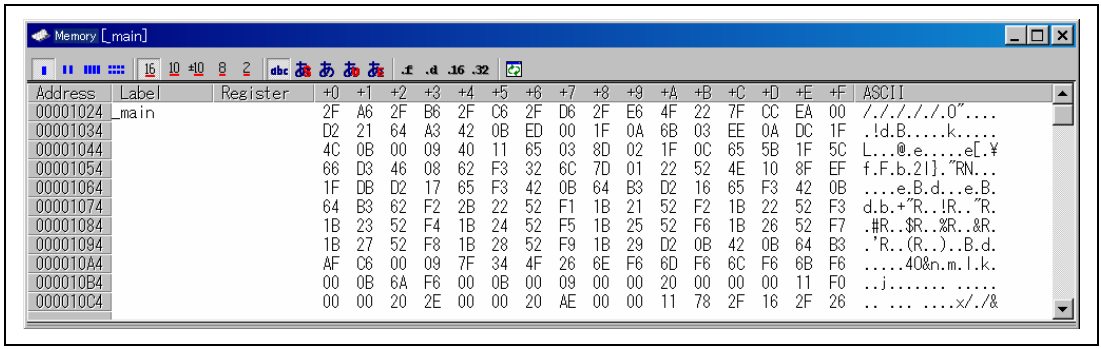


Figure 6.16 [Memory] Window

6.13 Watching Variables

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

- Click the left of displayed array `a` in the [Source] window to position the cursor.
- Select [Instant Watch...] with the right-hand mouse button.

The following dialog box will be displayed.

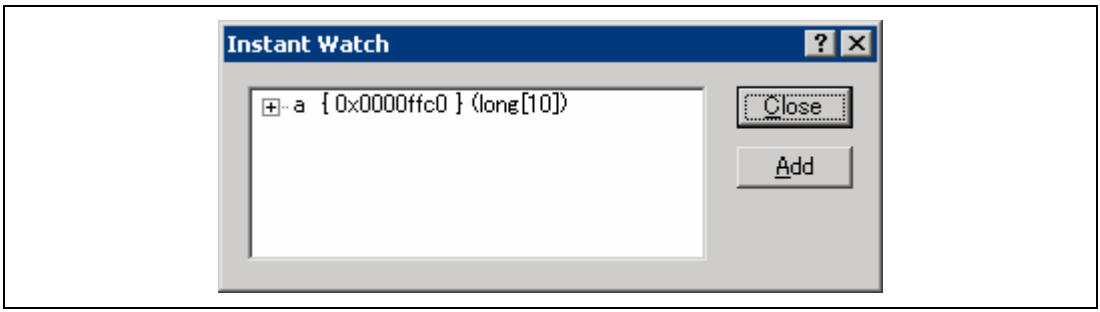


Figure 6.17 [Instant Watch] Dialog Box

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

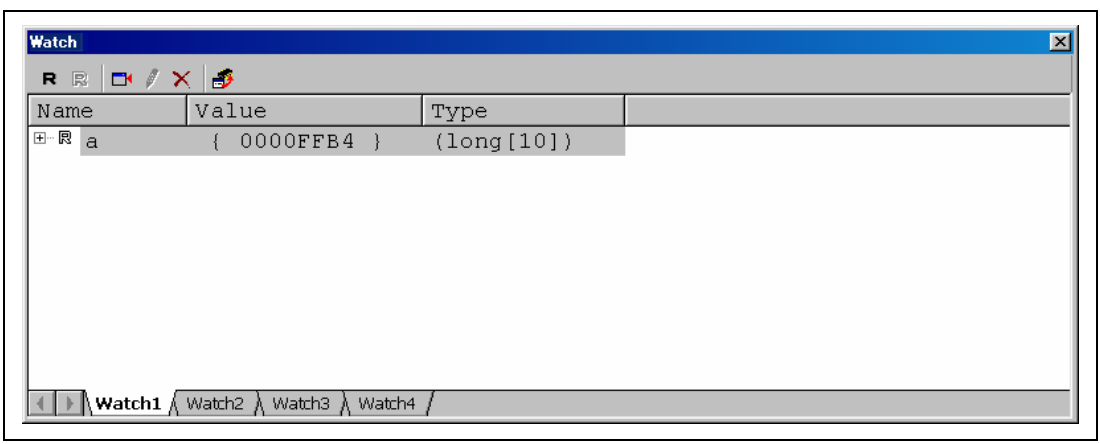


Figure 6.18 [Watch] Window (Displaying the Array)

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

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

The following dialog box will be displayed. Enter variable `i`.

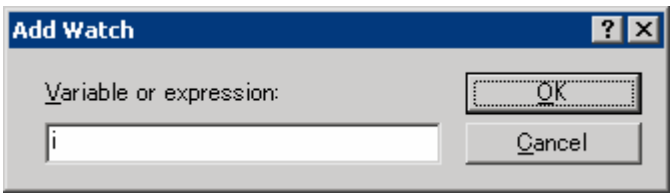


Figure 6.19 [Add Watch] Dialog Box

- Click the [OK] button.

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

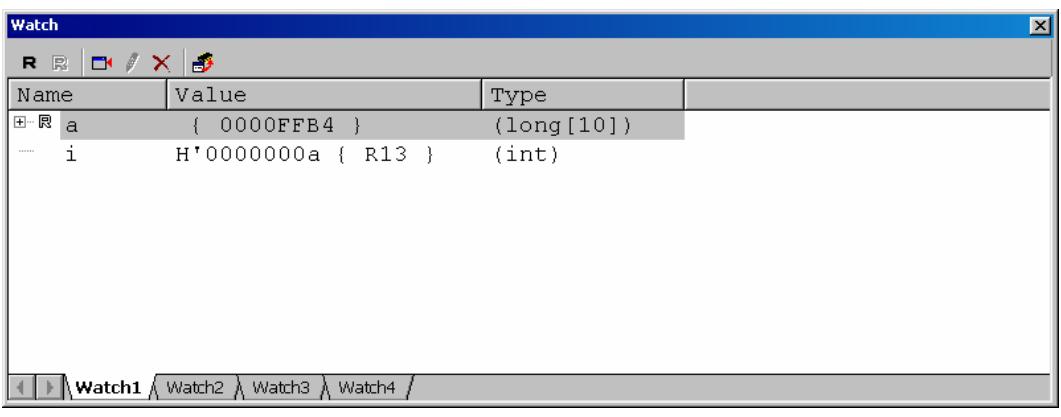


Figure 6.20 [Watch] Window (Displaying the Variable)

The user can click mark ‘+’ at the left side of array a in the [Watch] window to watch all the elements.

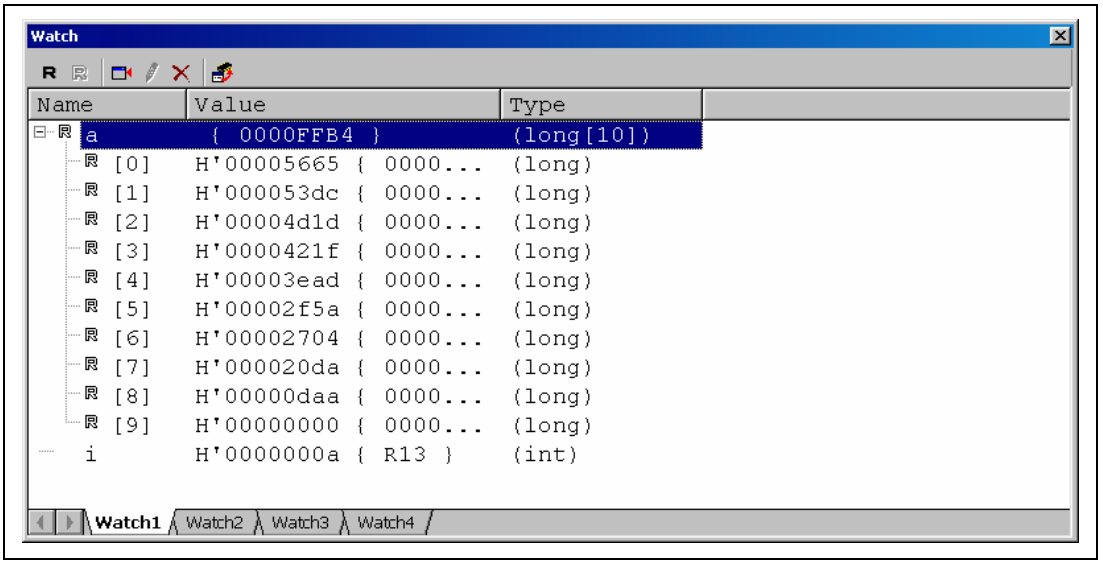


Figure 6.21 [Watch] Window (Displaying Array Elements)

6.14 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 four local variables: `a`, `j`, `i`, and `p_sam`.

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

The [Locals] window shows the local variables in the function currently pointed to by the program counter, along with their values. Note, however, that the [Locals] window is initially empty because local variables are yet to be declared.

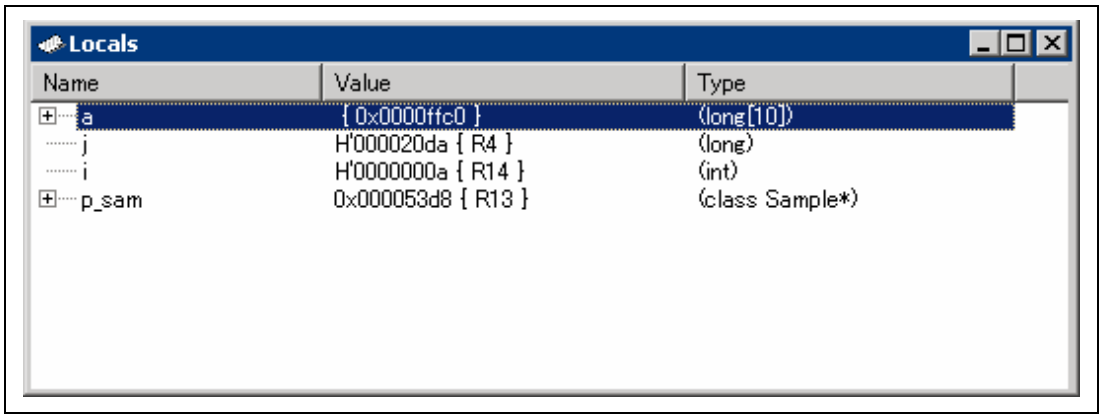


Figure 6.22 [Locals] Window

- Click mark '+' at the left side of array `a` in the [Locals] window to display the elements.
- 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.

6.15 Stepping Through a Program

The High-performance Embedded Workshop provides a range of step menu commands that allow efficient program debugging.

Table 6.1 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.

6.15.1 Executing [Step In] Command

The [Step In] command 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 [Debug] menu, or click the [Step In] button on the toolbar.

Figure 6.23 [Step In] Button

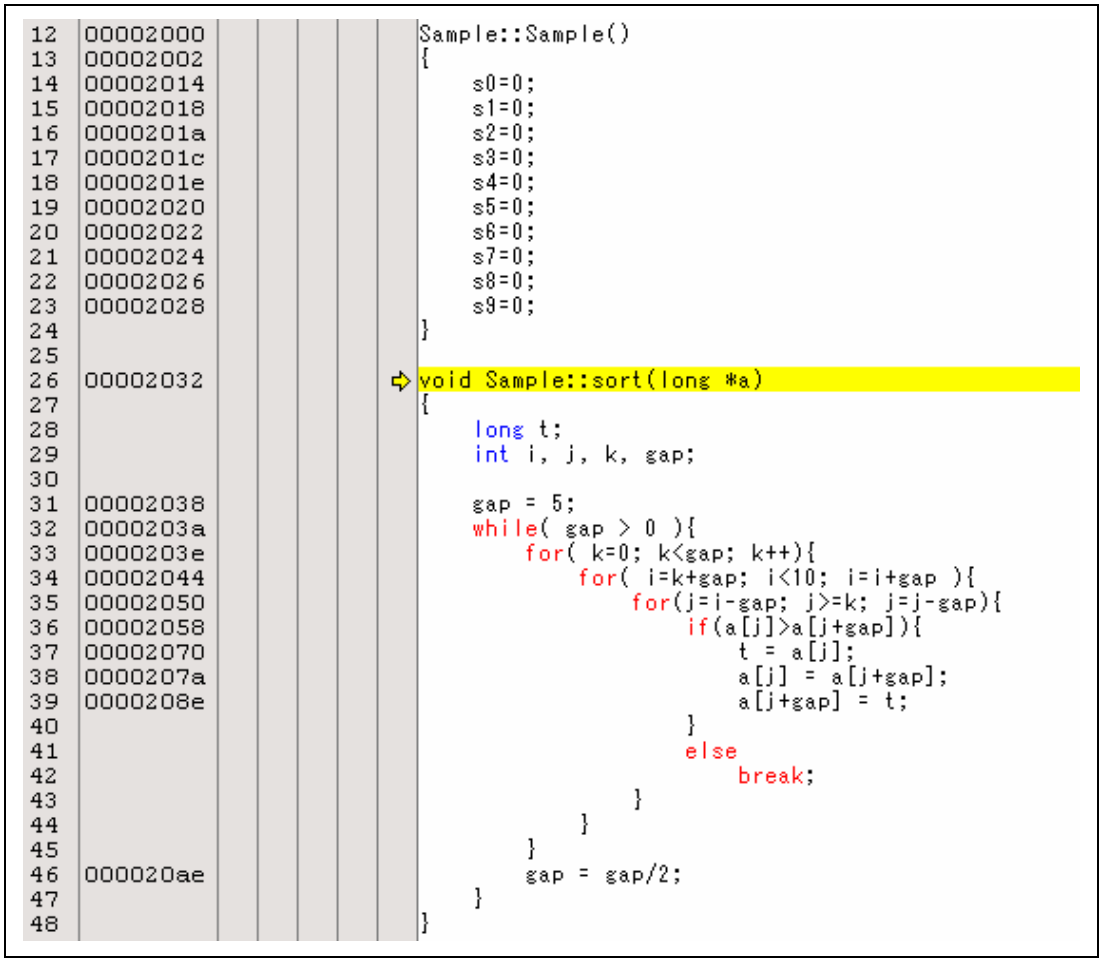


Figure 6.24 [Source] Window (Step In)

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

6.15.2 Executing [Step Out] Command

The [Step Out] command 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 [Debug] menu, or click the [Step Out] button on the toolbar.

Note: It takes time to execute this function. When the calling source is clarified, use [Go To Cursor].

Figure 6.25 [Step Out] Button

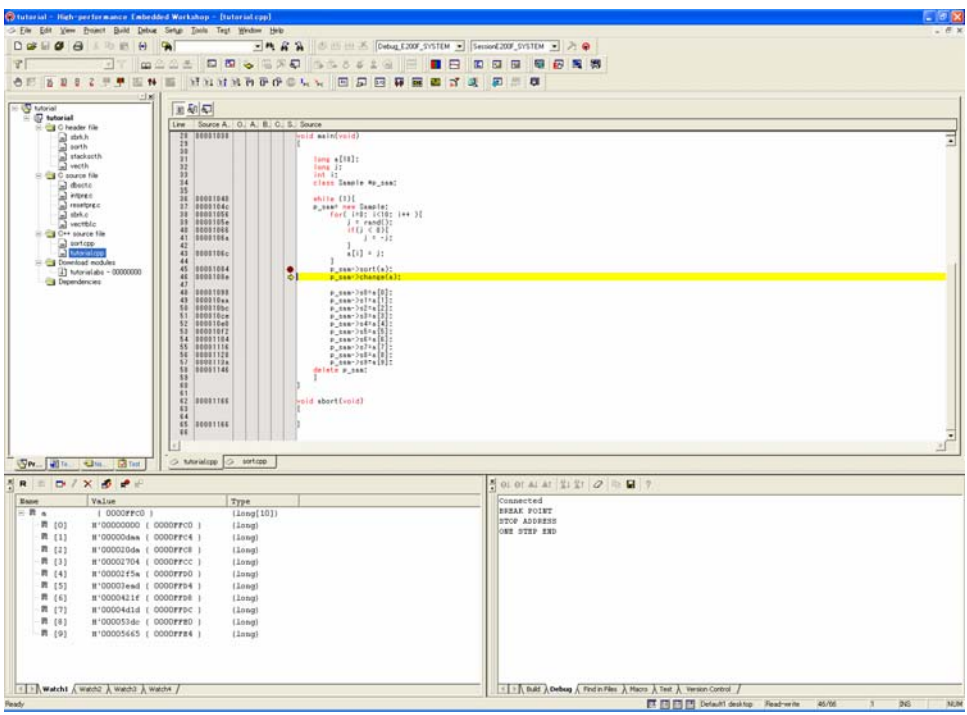


Figure 6.26 [High-performance Embedded Workshop] Window (Step Out)

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

6.15.3 Executing [Step Over] Command

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

- Move to the `change` function following the procedures described in section 6.15.1, Executing [Step In] Command.
- To step through all statements in the `change` function at a single step, select [Step Over] from the [Debug] menu, or click the [Step Over] button on the toolbar.

Figure 6.27 [Step Over] Button

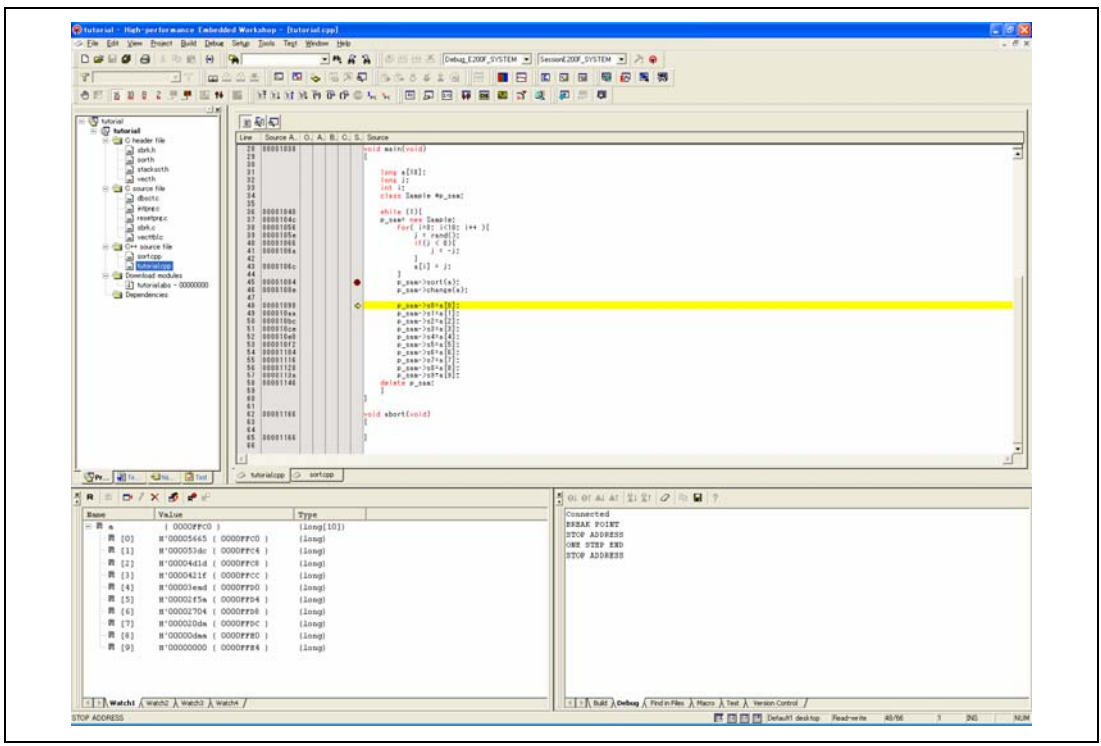


Figure 6.28 [High-performance Embedded Workshop] Window (Step Over)

6.16 Forced Breaking of Program Executions

The High-performance Embedded Workshop can force a break in the execution of a program.

- Cancel all breaks.
- To execute the remaining sections of the `main` function, select [Go] from the [Debug] menu or the [Go] button on the toolbar.

Figure 6.29 [Go] Button

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

Figure 6.30 [Stop] Button

6.17 Break Function

The emulator has PC break functions and break functions by eventpoints. With the High-performance Embedded Workshop, a PC breakpoint can be set using the [Breakpoint] sheet of the [Event] window. The eventpoint condition setting can be set by the event type using the [Onchip Event], [AUD Event], [Other Event], and [BUS Event] sheets.

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

6.17.1 PC Break Function

The emulator can set up to 1000 PC breakpoints. Other methods for setting a PC breakpoint than in section 6.7, Setting a PC Breakpoint, are described below.

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed.
- Select the [Breakpoint] sheet.

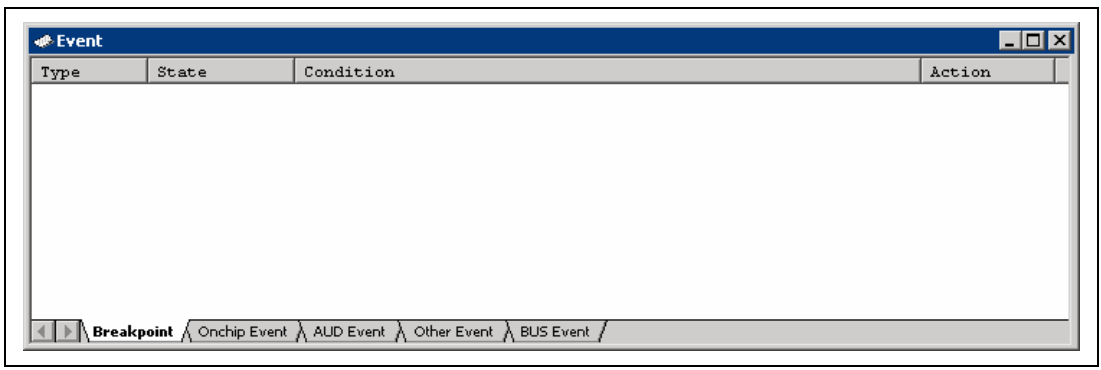


Figure 6.31 [Event] Window (Before PC Breakpoint Setting)

- Click the [Event] window with the right-hand mouse button and select [Add…] from the popup menu.
- Enter ***H'00001098*** in the [Address] edit box.

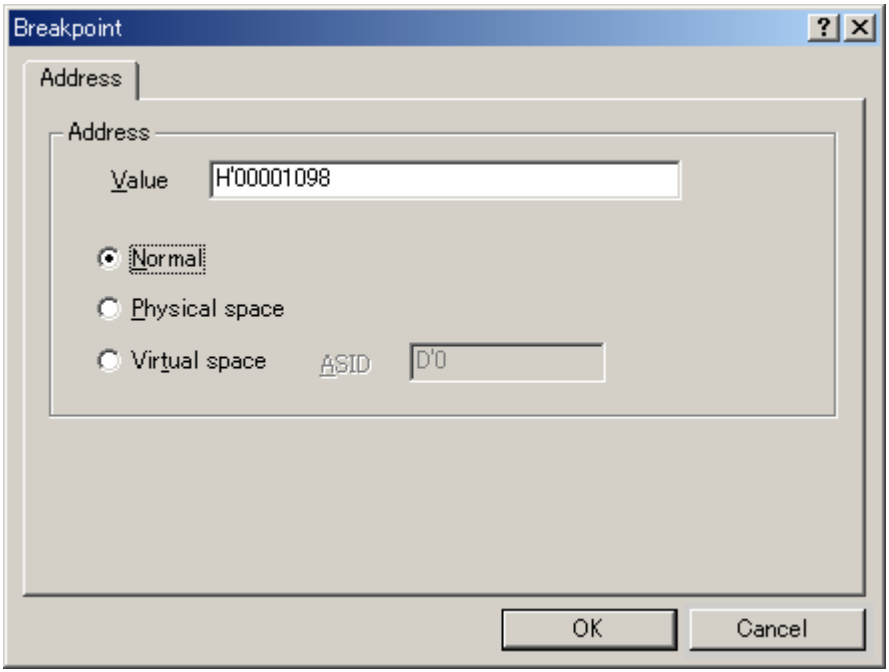


Figure 6.32 [Breakpoint] Dialog Box

Note: This dialog box differs according to the product. For the items of each product, refer to the online help.

- Click the [OK] button.

The PC breakpoint that has been set is displayed in the [Event] window.

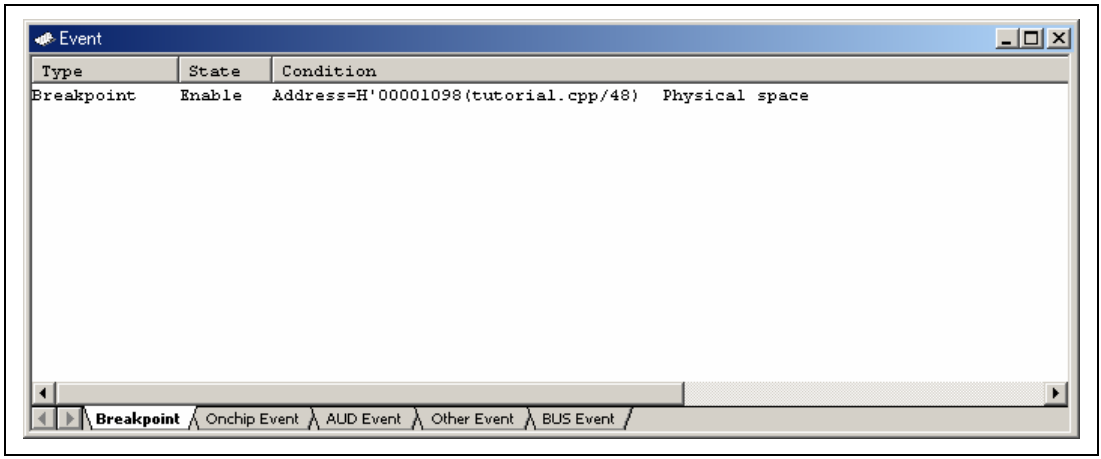


Figure 6.33 [Event] Window (PC Breakpoint Setting)

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

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

- Set the program counter and stack pointer values (PC = H'00000800 and R15 = H'00010000) that were set in section 6.8, Setting Registers, in the [Register] window. Click the [Go] button.
- If program execution is failed, reset the device and execute again the procedures above.

The program runs, and stops at the set PC breakpoint.

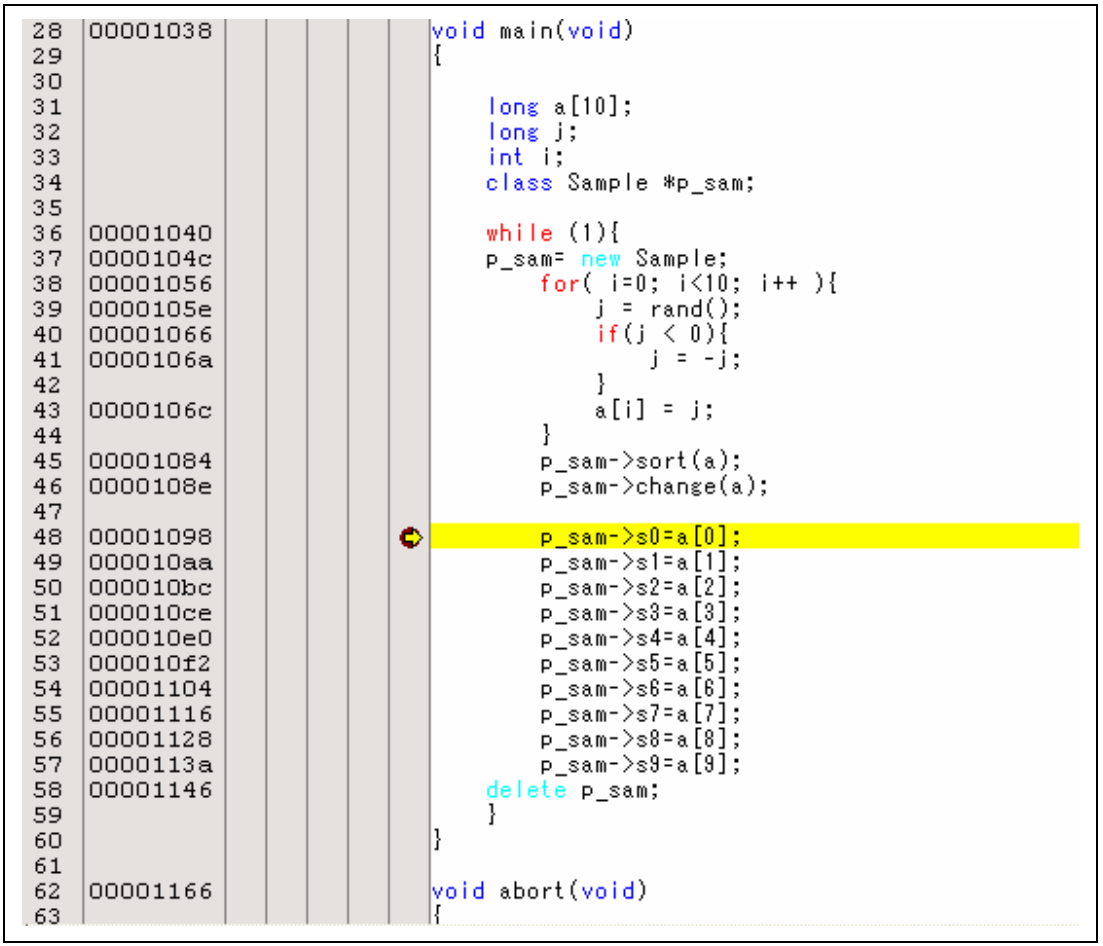


Figure 6.34 [Source] Window at Execution Stop (PC Break)

The [Status] window displays the following contents.

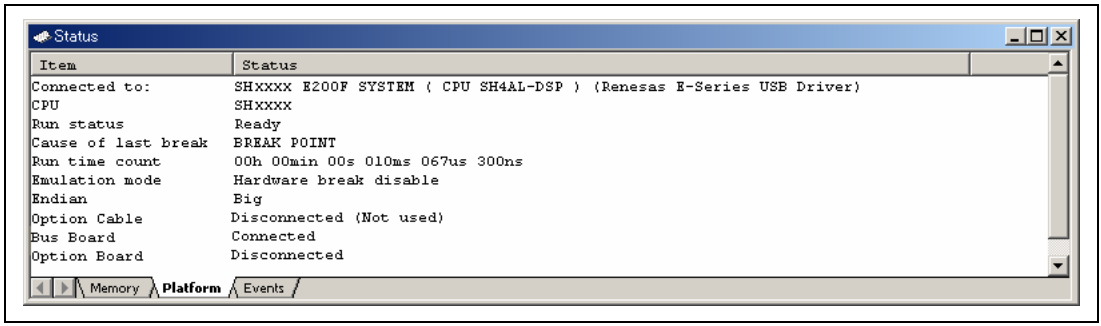


Figure 6.35 Displayed Contents of the [Status] Window (PC Break)

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

6.18 Break Function by an Eventpoint

With this emulator, an Onchip Eventpoint, AUD Eventpoint, Other Eventpoint, and BUS Eventpoint can be set. An example of how to set a break using an Onchip Eventpoint is shown below. For other eventpoint setting methods, refer to section 5.6, Using the Eventpoints.

6.18.1 Setting the Break by an Onchip Eventpoint

A method is given below in which the address bus condition is set under event channel 1 (Ch1 (IA_OA)) of the Onchip Eventpoint.

- Select [Eventpoints] from the [Code] submenu of the [View] menu. The [Event] window is displayed.
- The PC breakpoint that has been previously set is deleted. Click the [Event] window with the right-hand mouse button and select [Delete All] from the popup menu to cancel all PC breakpoints that have been set.
- To set an Onchip Eventpoint, click on [Onchip Eventpoint].

Up to 13 conditions can be set independently for the Onchip Eventpoints. In this example, set event channel 1.

Note: The number of event channels of the Onchip Eventpoint differs according to the product. For the number that can be specified for each product, refer to the online help.

- Select a line of Ch1(IA_OA) in the [Event] window. When highlighted, double-click this line.

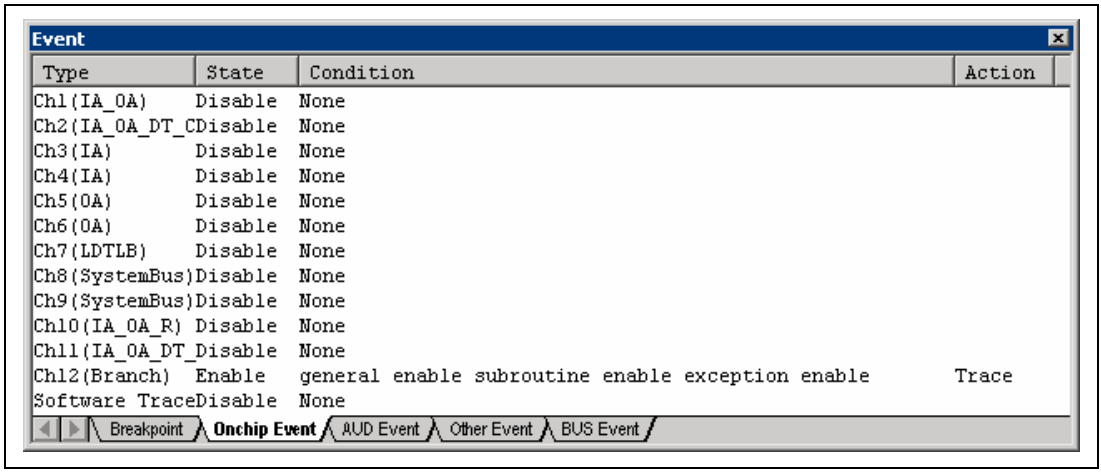


Figure 6.36 [Event] Window ([Onchip Event] Sheet)

- The [Event condition 1] dialog box is displayed.
- Clear the [Don't care] check box in the [Address] page.
- Select the [Prefetch address break before executing] radio button and enter ***H'00001084*** as the value in the [Address] edit box.

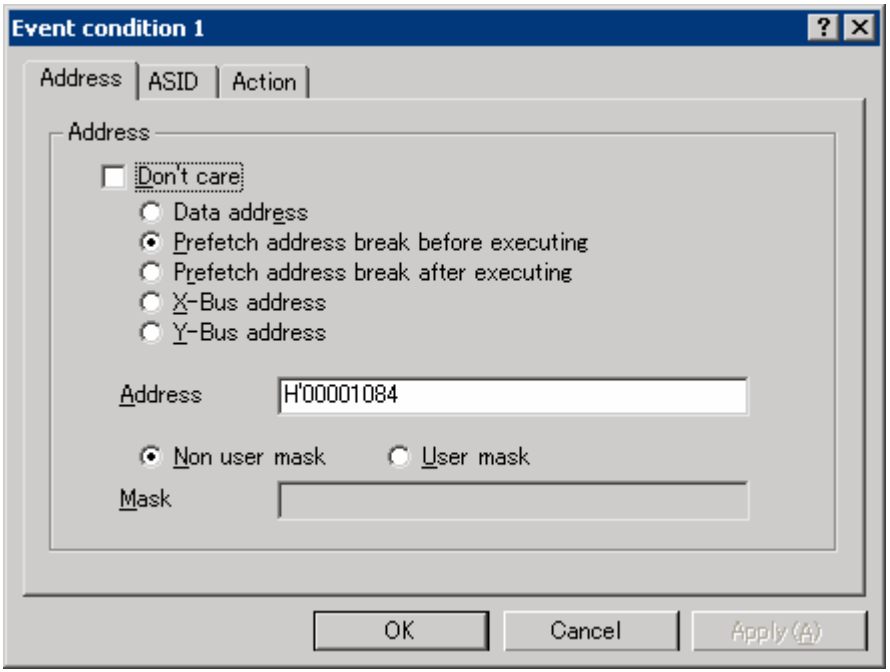


Figure 6.37 [Address] Page ([Event condition 1] Dialog Box)

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

- Click the [OK] button.
- The first point display in the State line changes from `Disable` to `Enable`.
- The first point display in the Condition line changes from `None` to `Address = H'00001084 pc`.
- Set the program counter and stack pointer values (PC = H'00000800 and R15 = H'00010000) that were set in section 6.8, Setting Registers, in the [Register] window. Click the [Go] button.
- If program execution is failed, reset the device and execute again the procedures above.

The program runs and then stops at the condition specified under event channel 1.

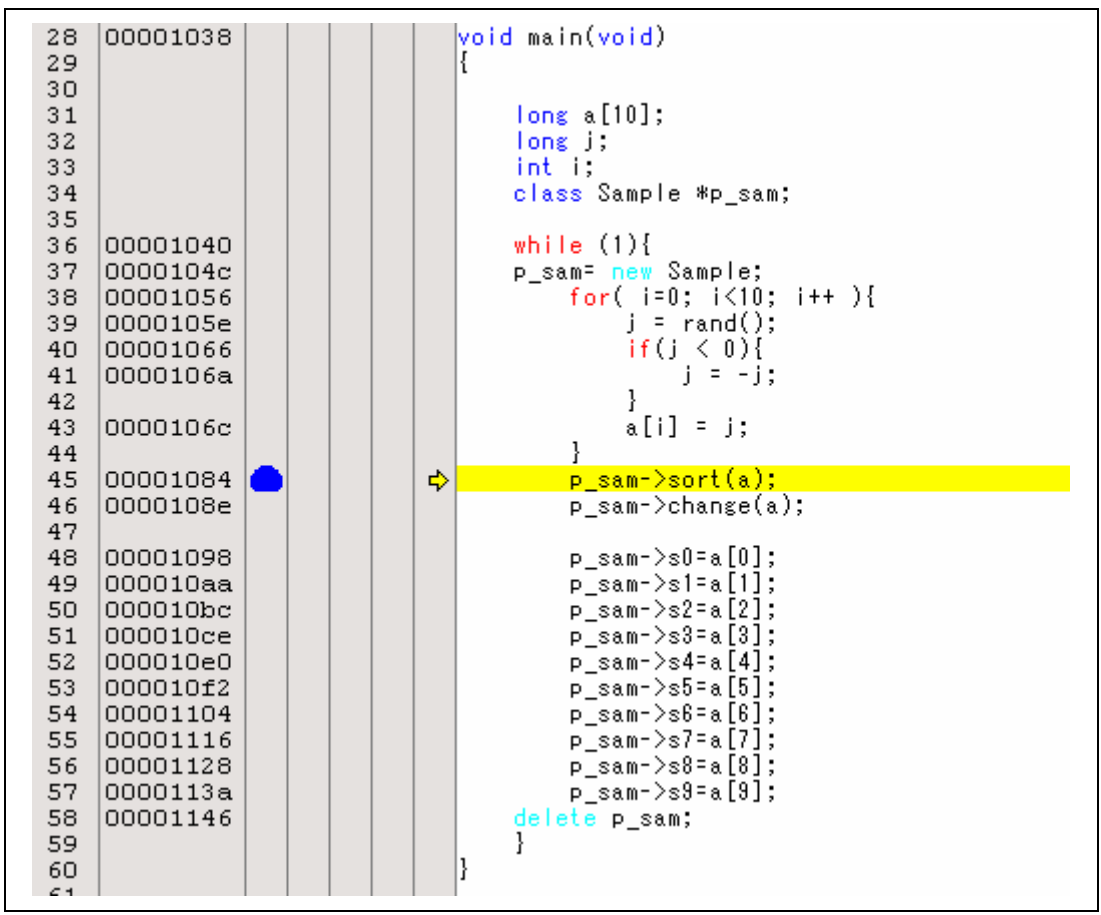


Figure 6.38 [Source] Window at Execution Stop (Onchip Eventpoint Channel 1)

The [Status] window displays the following contents.

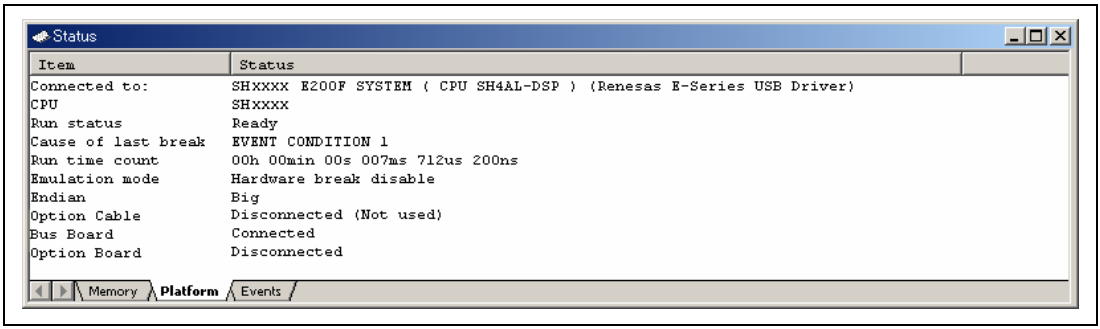


Figure 6.39 Displayed Contents of the [Status] Window (Onchip Event Ch1(IA_OA))

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

6.18.2 Setting the Sequential Onchip Eventpoint

The sequential break is enabled by the combination of Onchip Eventpoints.

Set the satisfaction conditions of Onchip Eventpoint as follows:

Ch1(IA_OA): A break condition is satisfied immediately before when an instruction of address H'00001084 is executed.

Ch2(IA_OA_DT_CT): A break condition is satisfied immediately before when an instruction of address H'0000106C is executed.

Follow the setting method described in the previous section.

To set these eventpoints as sequential:

- Select [Sequential Setting] from the popup menu by clicking the [Event] window with the right-hand mouse button. The [Sequential setting] dialog box will open.

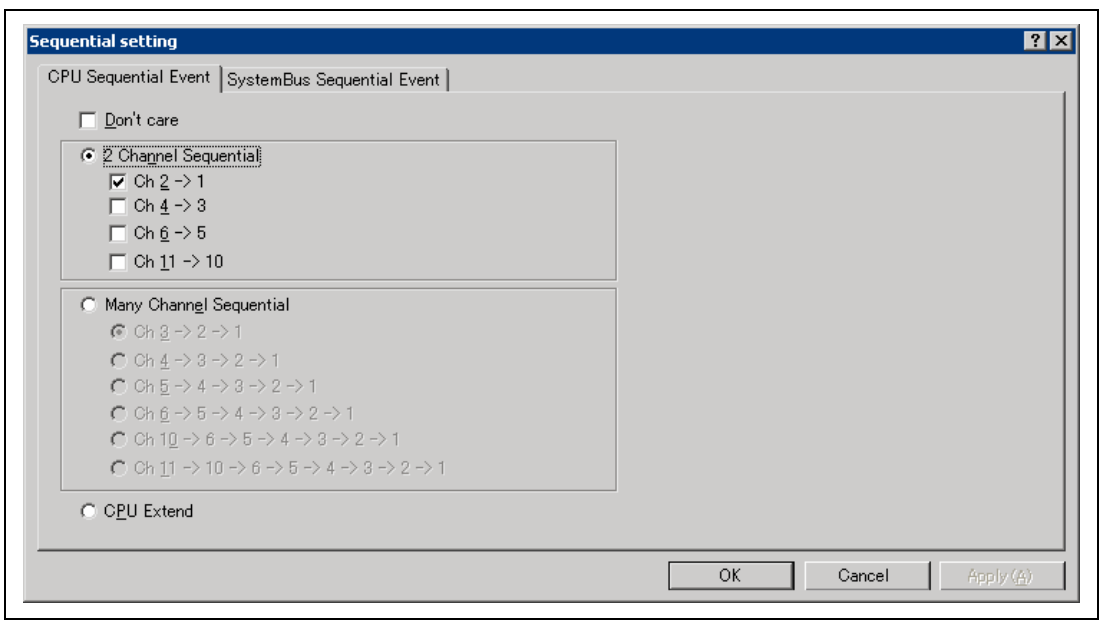


Figure 6.40 [Sequential setting] Dialog Box

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

- Select the [2 Channel Sequential] radio button and [Channel 2 -> 1], and click the [OK] button.

When the setting is completed, the [Event] window will be as shown in figure 6.41.

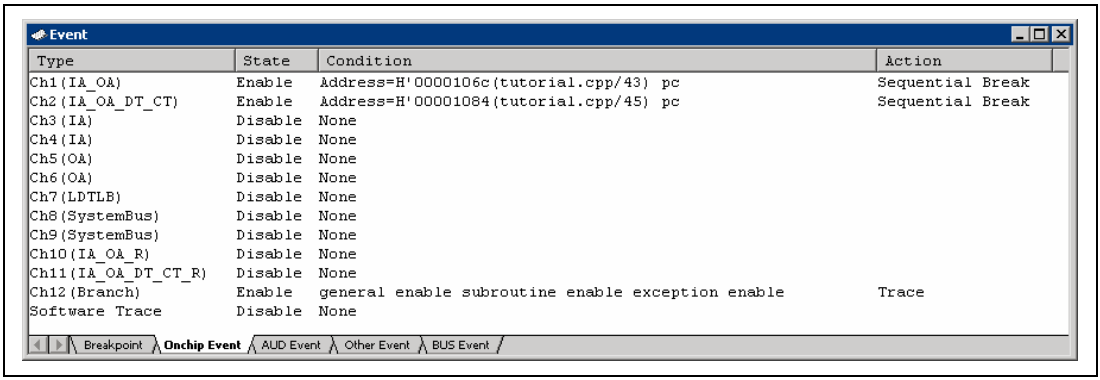


Figure 6.41 [Onchip Event] Sheet

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

- Set the program counter and stack pointer values (PC = H'00000800 and R15 = H'00010000) that were set in section 6.8, Setting Registers, in the [Register] window. Click the [Go] button.
- If program execution is failed, reset the device and execute again the procedures above.

The program runs and then stops at the condition specified under event channel 1.

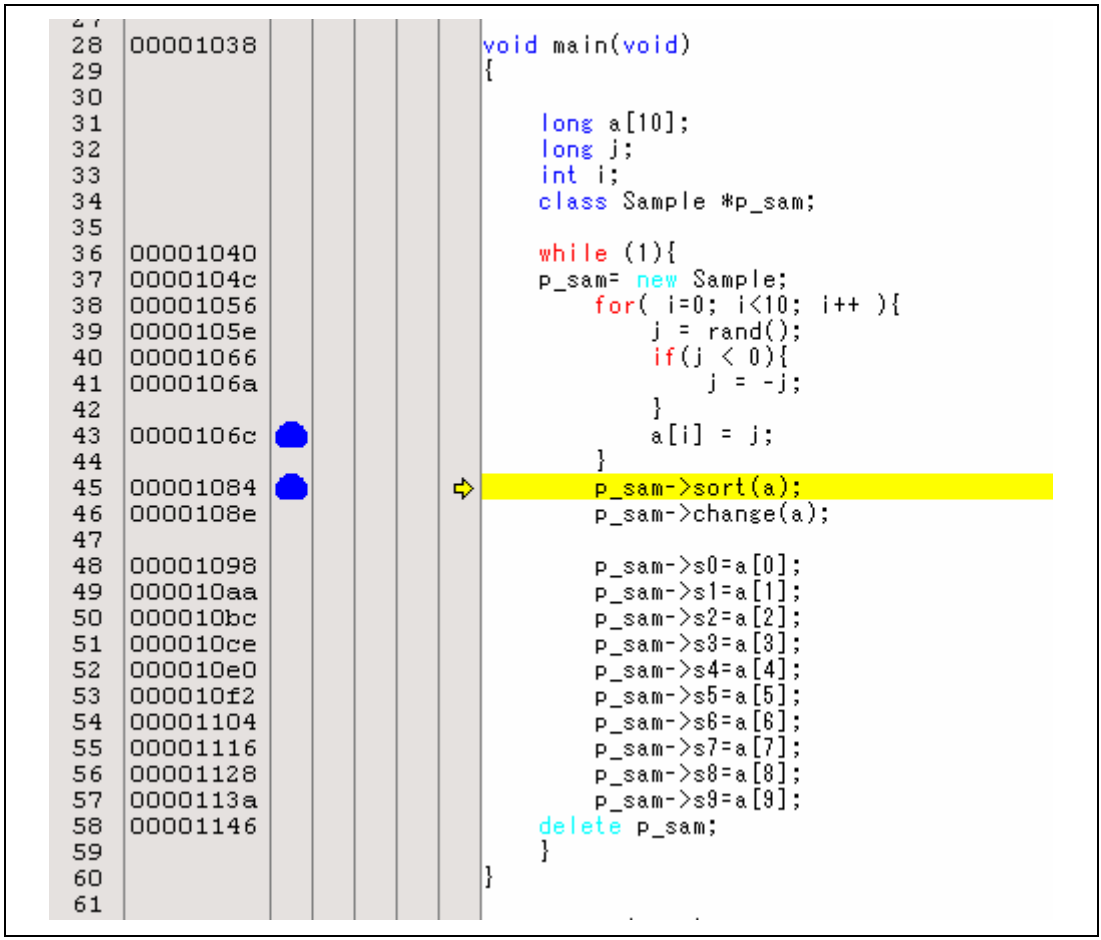


Figure 6.42 [Source] Window at Execution Stop (Sequential Break)

The [Status] window displays the following contents.

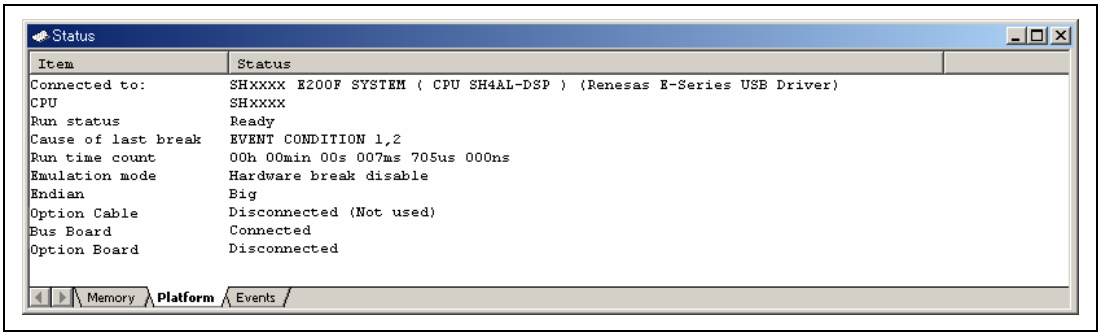


Figure 6.43 Displayed Contents of the [Status] Window (Sequential Break)

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

- The sequential break conditions that have been previously set are deleted. Click the [Event] window with the right-hand mouse button and select [Delete All] from the popup menu to cancel all eventpoint conditions that have been set.
- Select [Sequential Setting] from the popup menu by clicking the [Event] window with the right-hand mouse button. The [Sequential setting] dialog box will open (figure 6.40).
- Select the [Don't care] radio button and click the [OK] button.

6.19 Trace Functions

The emulator has the five trace functions listed below.

- Internal Trace Function

  Since this function uses the trace buffer built into the MPU, a realtime trace can be acquired.

The following information can be acquired:

- Types of trace information: Branch information, memory access information from the CPU, and PC or Rn value during the Trace Rn instruction execution
- Trace acquisition address values
- Data values
- Mnemonics
- Operands
- Source lines

Notes: 1. The number of branch instructions that can be acquired by a trace differs according to the product. For the number that can be specified for each product, refer to the online help.
2. The internal trace function is not supported for all products. For details on the specifications of each product, refer to the online help.
3. The internal trace function is not extended for all products. For details on the specifications of each product, refer to the online help.

- AUD Trace Function

  This is the large-capacity trace function that is enabled when the AUD pin is connected to the emulator. When a set of the branch source and branch destination instructions is one branch, the maximum amount of information acquired by a trace is 262,144.

The following information can be acquired:

- Types of trace information: Branch information, memory access information from the CPU, and PC or Rn value during the Trace Rn instruction execution
- Trace acquisition address values
- Data values
- External probe pin states
- Time stamp values
- Mnemonics

- Operands
- Source lines

Notes: 1. The AUD trace function is not supported for all products. For details on the specifications of each product, refer to the online help.
2. The types of trace information that can be acquired by an AUD trace function differ according to the product. For details on the specifications of each product or the number of acquired branches, refer to the online help.
3. When multiple loops are performed to reduce the number of AUD trace displays, only the IP counts up according to the product.

- Memory Output Trace Function

  This function is used to write the trace result to the specified memory range. The data is read from the memory range written in the [Trace] window and the result is then displayed. This is large-capacity trace function that is useful when the AUD pin of the device is not connected to the emulator.

Note: Some products do not support the memory output trace function. For details on the specifications of each product, refer to the online help.

- External Bus Trace Function

  This is a large-capacity trace function that is valid when the external bus pin of the MPU is connected to the emulator.
  The external bus trace function can acquire the information of a maximum of 262,144 cycles per bus cycle.

The following information can be acquired:

- External bus address values
- External bus data values
- Interrupt signal states
- Time stamp values
- Mnemonics
- Operands

Note: The types of trace information that can be acquired differ according to the product. For details on the specifications, refer to the online help.

- Trace Function as the I/O Analyzer Function
  The emulator can acquire a trace as part of the I/O analyzer function.

6.19.1 Displaying the [Internal/AUD/Usermemory trace] Window

Select [Trace] from the [Code] submenu of the [View] menu. The [Trace Window Type] dialog box is displayed.

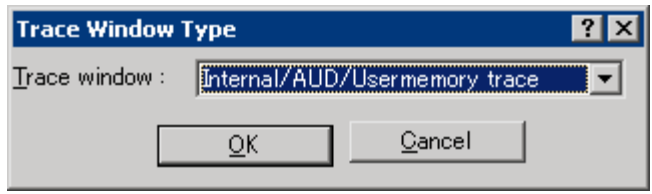


Figure 6.44 [Trace Window Type] Dialog Box

Select [Internal/AUD/Usermemory trace] and click the [OK] button. The [Internal/AUD/Usermemory trace] window will appear.

(1) Internal trace function

The methods to acquire the internal trace are described below.

(a) Setting the trace acquisition mode

Click the [Internal/AUD/Usermemory trace] window with the right-hand mouse button and select [Acquisition…] from the popup menu to display the [Internal/AUD/Usermemory acquisition] dialog box.

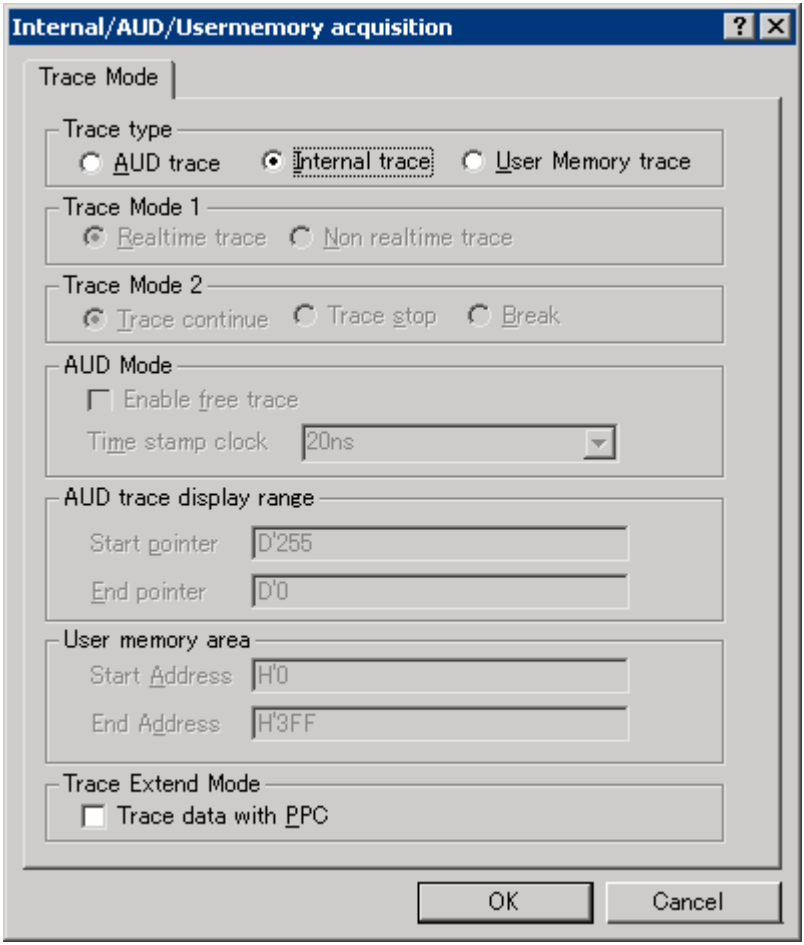


Figure 6.45 [Internal/AUD/Usermemory acquisition] Dialog Box

Select [Internal trace] for [Trace Type] and click the [OK] button.

(b) Setting the trace acquisition condition

The following is an example of acquiring branch source and branch destination information.

For acquiring other trace information, refer to section 5.6, Using the Eventpoints.

Select and double-click [Ch12 (Branch)] on the [Onchip Event] sheet of the [Event] window. The [Event condition 12] dialog box will appear.

Select the type of branch information. After selecting [Acquire trace] on the [Action] page, click the [OK] button.

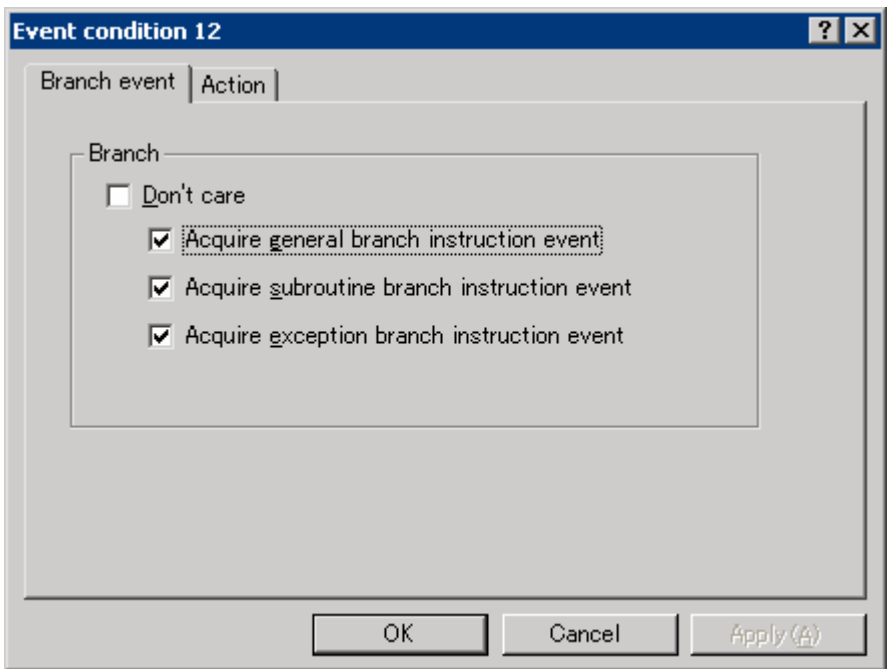
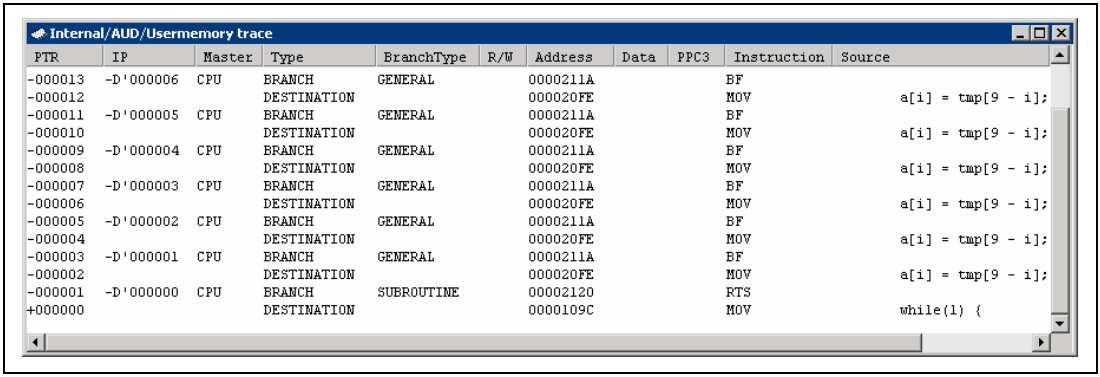


Figure 6.46 [Event condition 12] Dialog Box

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

(c) Displaying the trace result

Run the program as shown in the example of section 6.17.1, PC Break Function. The internal trace results are displayed in the [Internal/AUD/Usermemory trace] window after the program execution is completed.



PTR	IP	Master	Type	BranchType	R/W	Address	Data	PPC3	Instruction	Source
-000013	-D'000006	CPU	BRANCH	GENERAL		0000211A			BF	
-000012			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000011	-D'000005	CPU	BRANCH	GENERAL		0000211A			BF	
-000010			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000009	-D'000004	CPU	BRANCH	GENERAL		0000211A			BF	
-000008			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000007	-D'000003	CPU	BRANCH	GENERAL		0000211A			BF	
-000006			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000005	-D'000002	CPU	BRANCH	GENERAL		0000211A			BF	
-000004			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000003	-D'000001	CPU	BRANCH	GENERAL		0000211A			BF	
-000002			DESTINATION			000020FE			MOV	a[i] = tmp[9 - i];
-000001	-D'000000	CPU	BRANCH	SUBROUTINE		00002120			RTS	
+000000			DESTINATION			0000109C			MOV	while(1) {

Figure 6.47 [Internal/AUD/Usermemory trace] Window

- If necessary, adjust the column widths by dragging borders in the header bar (immediately below the title bar).

Note: The type and the amount of information that can be acquired by a trace differ according to the product. For details on the specifications of each product, refer to the online help.

(2) AUD trace function

This function is available when the AUD pin of the MPU is connected to the emulator.

An Onchip Eventpoint must be set in order to output MPU information from the AUD pin.

The following is an example of acquiring memory access information between addresses H'FFC0 to H'FFE4 (array a).

The following is the procedure for setting the AUD trace function.

(a) Setting the Onchip Eventpoint

Set the Onchip Eventpoint so that the AUD trace acquisition condition can be set to the execution information on the MPU that is output from the AUD pin, referring to section 5.6, Using the Eventpoints.

- Set the window trace from H'0 to H'10000 to Onchip Event channel Ch5 (OA).
- Select and double-click [Ch5(OA)] in the [Onchip Event] sheet of the [Event] window.
- The [Event condition 5] dialog box will appear.
- In the [Window address] page, specify the range of the memory access information to be output from the AUD pin.

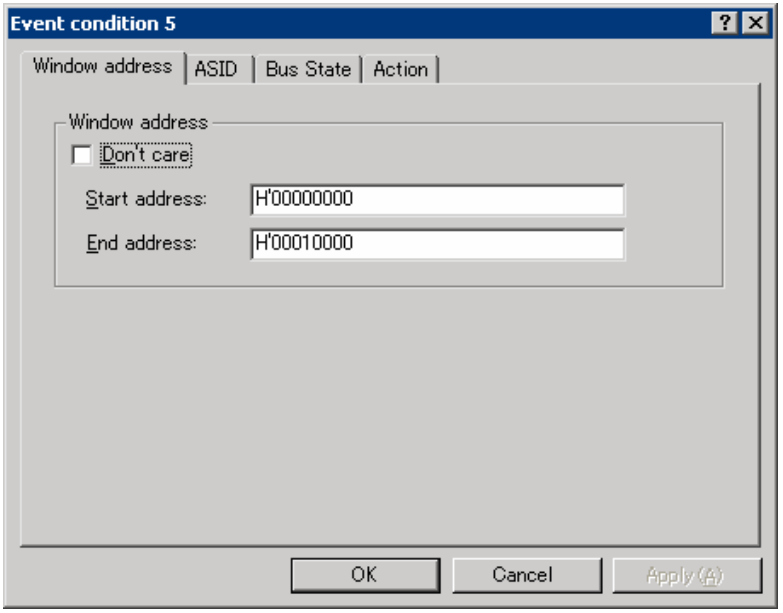


Figure 6.48 Setting the Onchip Eventpoint (Ch5(OA) Channel)

- Input H’0 into the [Start address] and H’10000 into the [End address].
- In the [Action] page, select [acquire trace] and double-click the [OK] button.

(b) Setting the trace acquisition mode

Display the [Internal/AUD/Usermemory trace] window. Click this window with the right-hand mouse button and select [Acquisition…] from the popup menu to display the [Internal/AUD/Usermemory acquisition] dialog box.

The AUD trace acquisition condition is set.

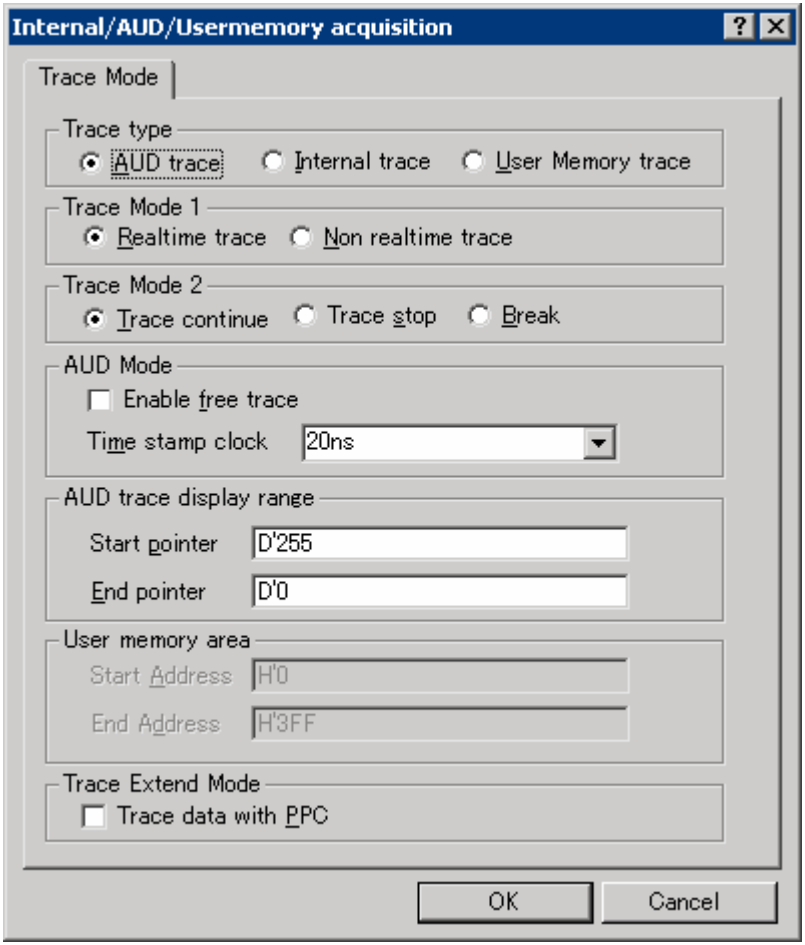


Figure 6.49 [Internal/AUD/Usermemory acquisition] Dialog Box

[Trace type]: Selects the type of trace function.

- [AUD trace] Selects the AUD trace function.
- [Internal trace] Selects the internal trace function.
- [User Memory trace] Selects the user-memory output trace function.

[Trace Mode 1]: Selects the operation mode when the trace information has been sequentially generated.

- [Realtime trace] Some trace information will not be output.
- [Non realtime trace] The CPU stops operations until the trace information is output.

[Trace mode 2]: Specifies the operation when the AUD trace buffer of the emulator becomes full.

- [Trace continue] The oldest trace information is overwritten to acquire the latest trace information.
- [Trace stop] When the trace buffer becomes full, the trace information is no longer acquired.
- [Break] A break occurs.

[AUD Mode]: Sets the mode when the AUD trace function is used.

- [Enable free trace] When checked, a free trace is enabled.
- [Time stamp clock] Specifies the resolution of the timer for time stamp as any of the following values:
  20 ns, 40 ns, 100 ns, or 400 ns

[AUD trace display range]: Sets the trace display range when the AUD trace function is used.

- [Start pointer] Enters the numerical start pointer value of the display range.
- [End pointer] Enters the numerical end pointer value of the display range.

[Trace Extend Mode]: Sets whether or not internal performance counter Ch3 or Ch4 is included as the item for trace acquisition.

- [Trace data with PPC] When checked, internal performance counter Ch3 or Ch4 is included.

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

(c) Setting the trace acquisition condition

The following is an example of acquiring memory access information.

For acquiring other trace information, refer to section 5.6, Using the Eventpoints.

Select and double-click [Ch2 (Delay)] on the [AUD Event] sheet of the [Event] window. The [Ch2 (Delay)] dialog box will appear.

Select the type of trace information to be acquired on the [General] page.

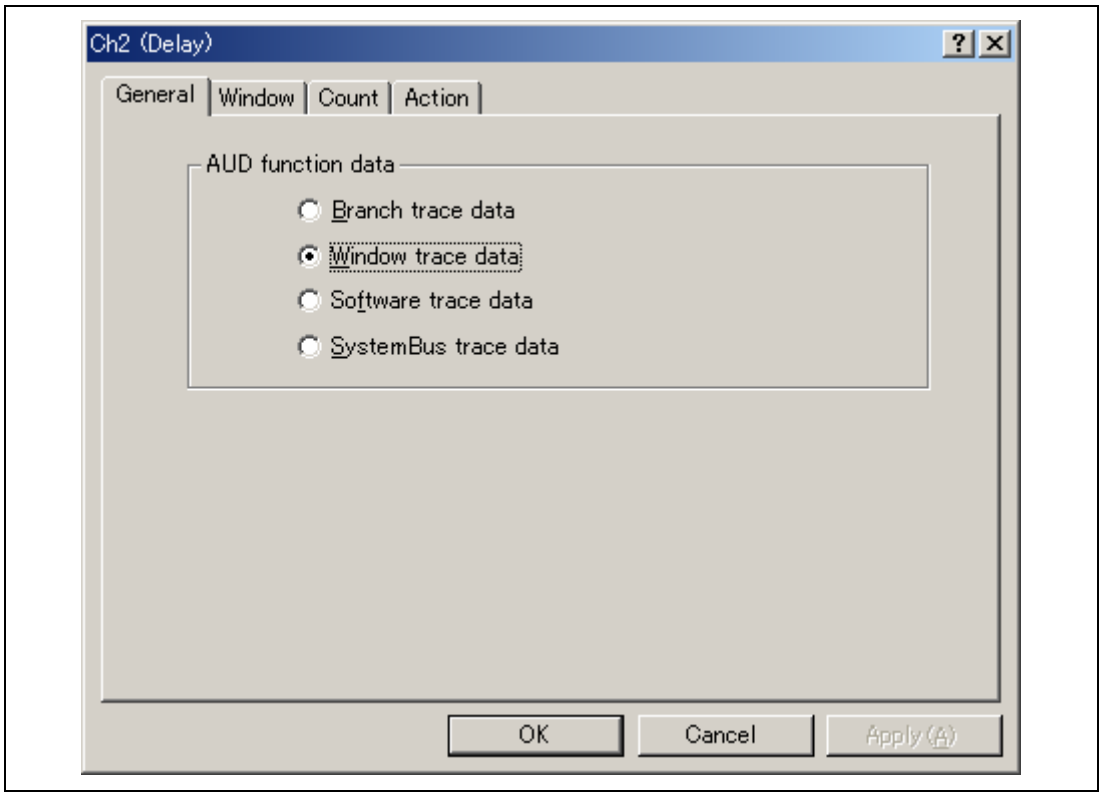


Figure 6.50 [Ch2 (Delay)] Dialog Box ([General] Page)

- Select [Window trace data] and click the [Window] page.
- The memory access range condition can be set in the [Window] page.

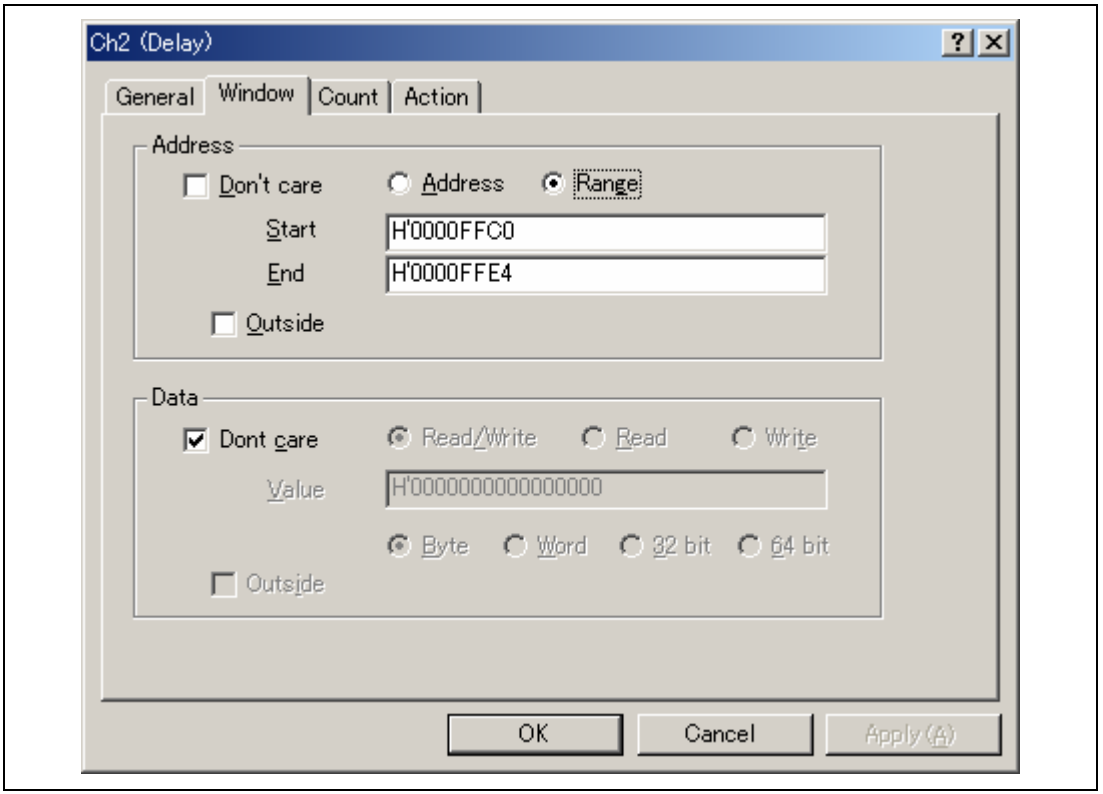
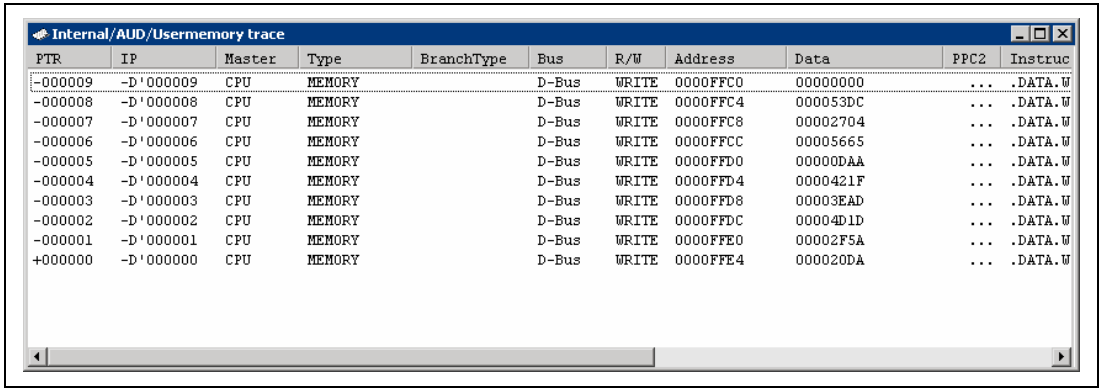


Figure 6.51 [Ch2 (Delay)] Dialog Box ([Windows] Page)

- Select [Range] to acquire trace in the address range.
- Input H'FFC0 and H'FFE4 into [Start] and [End], respectively.
- Memory access information between addresses H'FFC0 to H'FFE4 (array a) is acquired.
- Select [Trace get] in the [Action] page and click the [OK] button.

(d) Displaying the trace result

Run the program as shown in the example of section 6.17.1, PC Break Function. The trace results are displayed in the [Internal/AUD/Usermemory trace] window after the program execution is completed.



PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	PPC2	Instruc
-000009	-D'000009	CPU	MEMORY		D-Bus	WRITE	0000FFC0	00000000	...	.DATA.W
-000008	-D'000008	CPU	MEMORY		D-Bus	WRITE	0000FFC4	000053DC	...	.DATA.W
-000007	-D'000007	CPU	MEMORY		D-Bus	WRITE	0000FFC8	00002704	...	.DATA.W
-000006	-D'000006	CPU	MEMORY		D-Bus	WRITE	0000FFCC	00005665	...	.DATA.W
-000005	-D'000005	CPU	MEMORY		D-Bus	WRITE	0000FFD0	00000DAA	...	.DATA.W
-000004	-D'000004	CPU	MEMORY		D-Bus	WRITE	0000FFD4	0000421F	...	.DATA.W
-000003	-D'000003	CPU	MEMORY		D-Bus	WRITE	0000FFD8	00003EAD	...	.DATA.W
-000002	-D'000002	CPU	MEMORY		D-Bus	WRITE	0000FFDC	00004D1D	...	.DATA.W
-000001	-D'000001	CPU	MEMORY		D-Bus	WRITE	0000FFE0	00002F5A	...	.DATA.W
+000000	-D'000000	CPU	MEMORY		D-Bus	WRITE	0000FFE4	000020DA	...	.DATA.W

Figure 6.52 [Internal/AUD/Usermemory trace] Window (Example)

- The value of array a can be confirmed by opening the [Watch] window. For details, refer to section 6.13, WatchingVariables.

(3) Memory Output Trace Function

This function is used to write the trace result to the specified memory range.

The following shows the procedure for acquiring the memory output trace function.

(a) Setting the trace acquisition mode

Display the [Internal/AUD/Usermemory trace] window. Click this window with the right-hand mouse button and select [Acquisition…] from the popup menu to display the [Internal/AUD/Usermemory acquisition] dialog box.

The memory output trace acquisition mode is set.

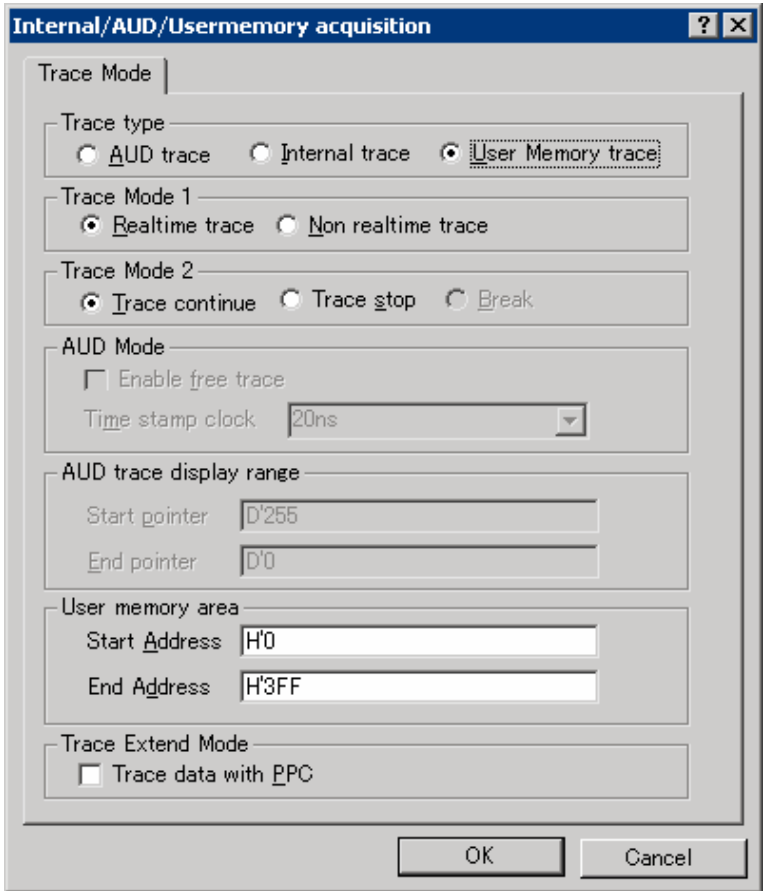


Figure 6.53 [Internal/AUD/Usermemory acquisition] Dialog Box

[Trace type]: Selects the type of trace function.

[AUD trace] Selects the AUD trace function.

[Internal trace] Selects the internal trace function.

[User Memory trace] Selects the user-memory output trace function.

[Trace Mode 1]: Selects the operation mode when the trace information has been sequentially generated.

[Realtime trace] Some trace information will not be output.

[Non realtime trace] The CPU stops operations until the trace information is output.

[Trace mode 2]: Specifies the operation when the trace buffer becomes full.

[Trace continue] The oldest trace information is overwritten to acquire the latest trace information.

[Trace stop] When the trace buffer becomes full, the trace information is no longer acquired.

[User memory area]: Sets the trace display range when the user-memory output trace function is used.

[Start Address] Specifies the start address of the display range.

[End Address] Specifies the end address of the display range.

[Trace Extend Mode]: Sets whether or not internal performance counter Ch3 or Ch4 is included as the item for trace acquisition.

[Trace data with PPC] When checked, internal performance counter Ch3 or Ch4 is included.

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

- Set the start and end addresses, which are in the memory range that outputs the trace result, in the [Start] and [End] edit boxes of [User memory area], respectively.
- Enter the addresses depending on your environment. Do not specify the range that the program has been downloaded to, as the memory contents are overwritten by the trace output result.

(b) Setting the trace acquisition condition

The setting for the acquisition condition of the memory output trace function is the same as that of the internal trace function. Refer to section 6.19.1 (1), Internal trace function.

(c) Displaying the trace result

Run the program as shown in the example of section 6.17.1, PC Break Function. The trace results are displayed in the [Internal/AUD/Usermemory trace] window after the program execution is completed.

The following is an example of the trace display.

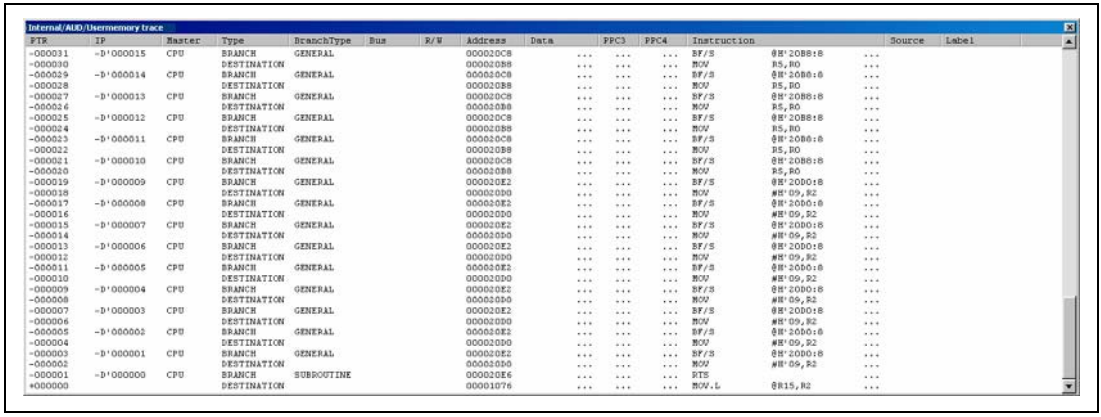


Figure 6.54 [Internal/AUD/Usermemory trace] Window (Example)

6.19.2 Displaying the [BUS/MFI trace] Window

Select [Trace] from the [Code] submenu of the [View] menu. The [Trace Window Type] dialog box is displayed.

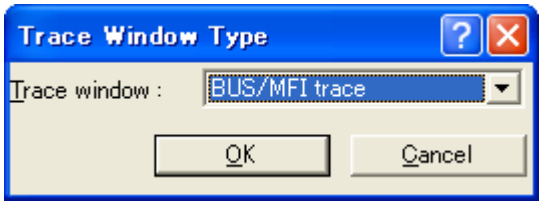


Figure 6.55 [Trace Window Type] Dialog Box

Select [BUS/MFI trace] and click the [OK] button. The [BUS/MFI trace] window will appear.

External bus trace function

This function is enabled when the external bus pins of the MPU are connected to the emulator. The methods to acquire the external bus trace are described below.

(a) Setting the bus trace

Set the multiplexed state of the bus pins or the connected memory, referring to section 2.7.2, External Bus Trace/Break Function and External Bus Performance Function.

(b) Setting the trace acquisition mode

Click the [BUS/MFI trace] window with the right-hand mouse button and select [Acquisition…] from the popup menu to display the [BUS acquisition] dialog box. Set the acquisition mode for the external bus trace as shown in figure 6.56.

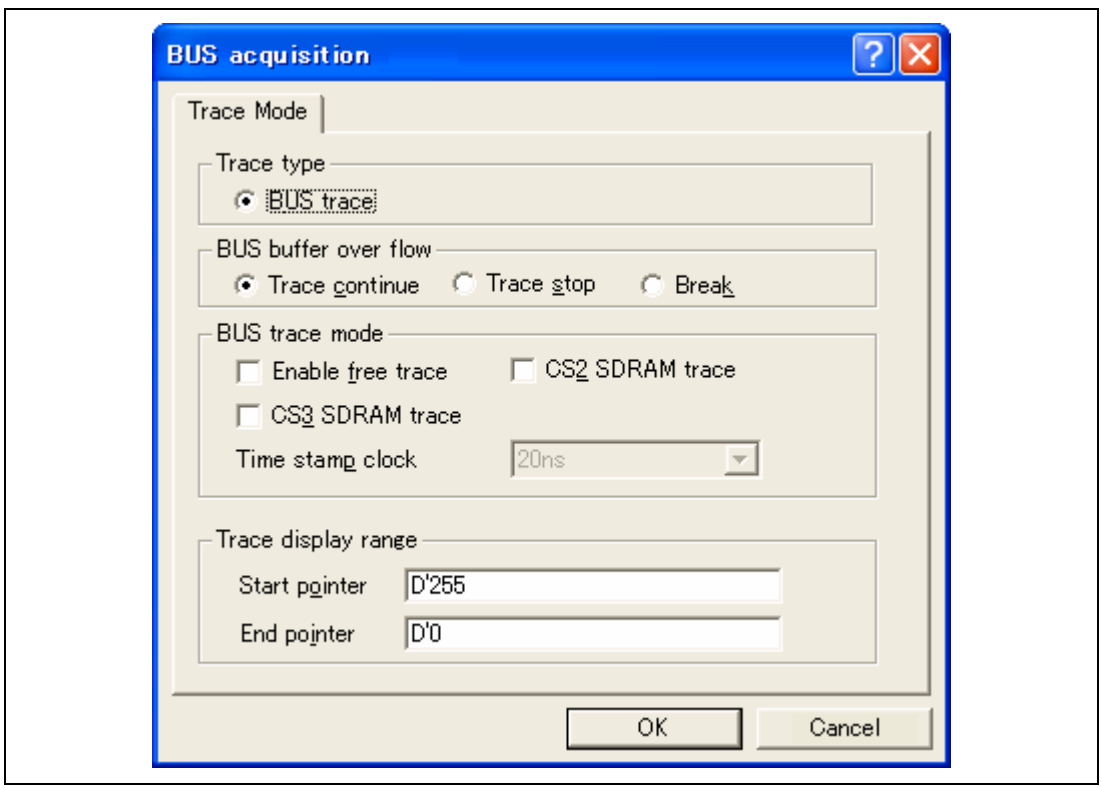


Figure 6.56 [BUS acquisition] Dialog Box

(c) Setting the trace acquisition condition

The following is an example of setting the address condition.

For setting other trace conditions, refer to section 5.6, Using the Eventpoints.

Select and double-click [Ch3 (Normal)] on the [BUS Event] sheet of the [Event] window. The [Ch3 (Normal)] dialog box will appear.

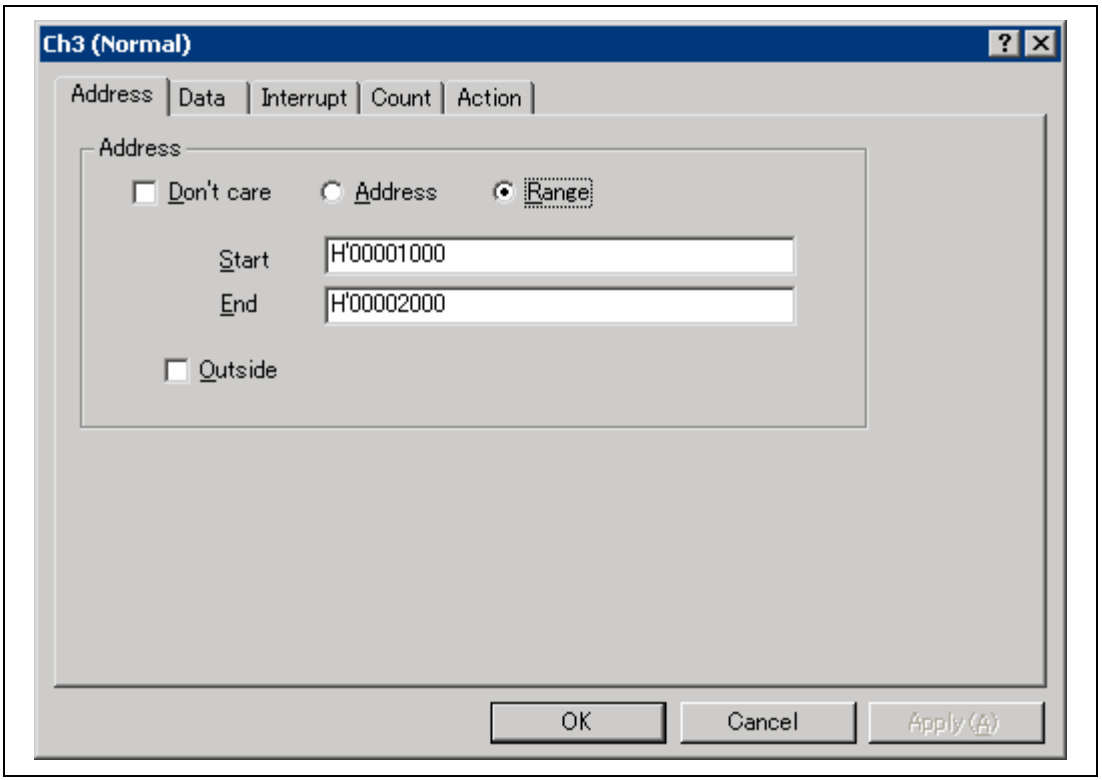
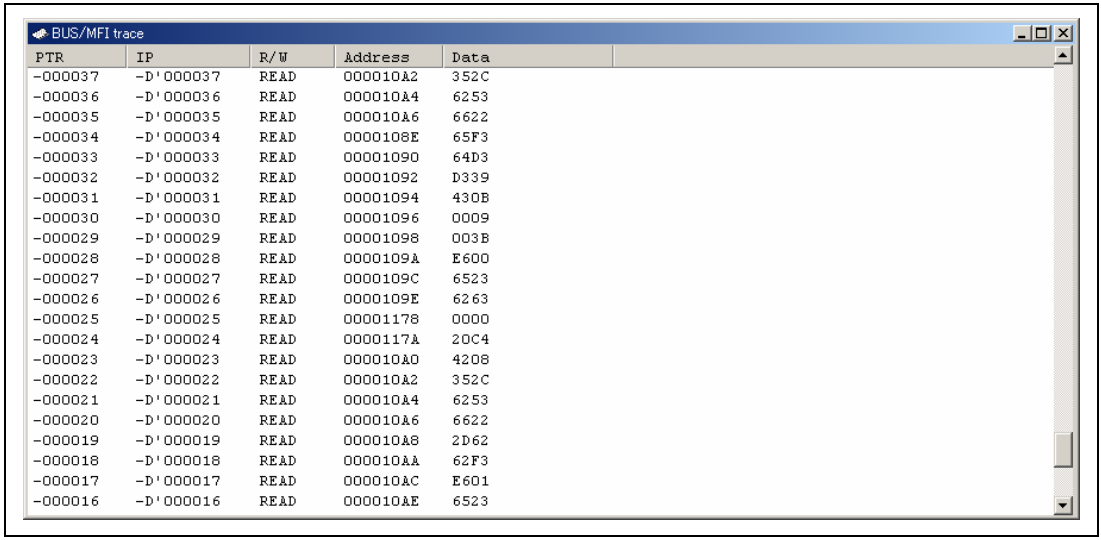


Figure 6.57 [Ch3 (Normal)] Dialog Box ([Address] Page)

- Select [Range].
- Input H’1000 and H’2000 into [Start] and [End], respectively.
- Select [Trace get] in the [Action] page and click the [OK] button.
- The trace information when external memory H’1000 to H’2000 has been accessed is acquired.

(d) Displaying the trace result

Run the program as shown in the example of section 6.17.1, PC Break Function. The internal trace results are displayed in the [BUS/MFI trace] window after the program execution is completed.



PTR	IP	R/W	Address	Data
-000037	-D'000037	READ	000010A2	352C
-000036	-D'000036	READ	000010A4	6253
-000035	-D'000035	READ	000010A6	6622
-000034	-D'000034	READ	0000108E	65F3
-000033	-D'000033	READ	00001090	64D3
-000032	-D'000032	READ	00001092	D339
-000031	-D'000031	READ	00001094	430B
-000030	-D'000030	READ	00001096	0009
-000029	-D'000029	READ	00001098	003B
-000028	-D'000028	READ	0000109A	E600
-000027	-D'000027	READ	0000109C	6523
-000026	-D'000026	READ	0000109E	6263
-000025	-D'000025	READ	00001178	0000
-000024	-D'000024	READ	0000117A	20C4
-000023	-D'000023	READ	000010A0	4208
-000022	-D'000022	READ	000010A2	352C
-000021	-D'000021	READ	000010A4	6253
-000020	-D'000020	READ	000010A6	6622
-000019	-D'000019	READ	000010A8	2D62
-000018	-D'000018	READ	000010AA	62F3
-000017	-D'000017	READ	000010AC	E601
-000016	-D'000016	READ	000010AE	6523

Figure 6.58 [BUS/MFI trace] Window

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

Note: The type and the amount of information that can be acquired by a trace differ according to the product. For details on the specifications of each product, refer to the online help.

6.20 MMU Support

This function can be used when the supported MPU has an MMU.

- TLB window

In the emulator, the contents of the TLB table can be easily displayed and edited by selecting [CPU -> TLB] from the [View] menu. For details, refer to the online help.

- VP_MAP translation function

The MPU, 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 MPU. 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 6.59.

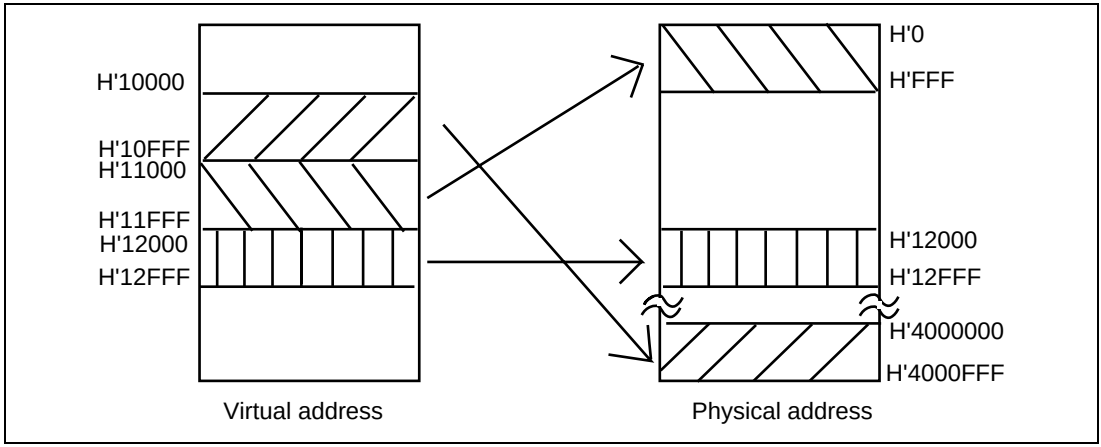


Figure 6.59 Address Translation according to VP_MAP Tables

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 `Physical` radio button is selected:

  The address is not translated.

- 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.

Table 6.2 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 [Memory area] group box in the [Configuration] dialog box.

6.21 Stack Trace Function

The emulator uses the information on the stack to display the names of functions in the sequence of calls that led to the function to which the program counter is currently pointing.

Note: This function can be used only when the load module that has the Dwarf2-type debugging information is loaded. Such load modules are supported in SHC/C++ compiler (including OEM and bundle products) V6.0 or later.

- Double-click the [Source] column in the `sort` function and set a PC breakpoint.

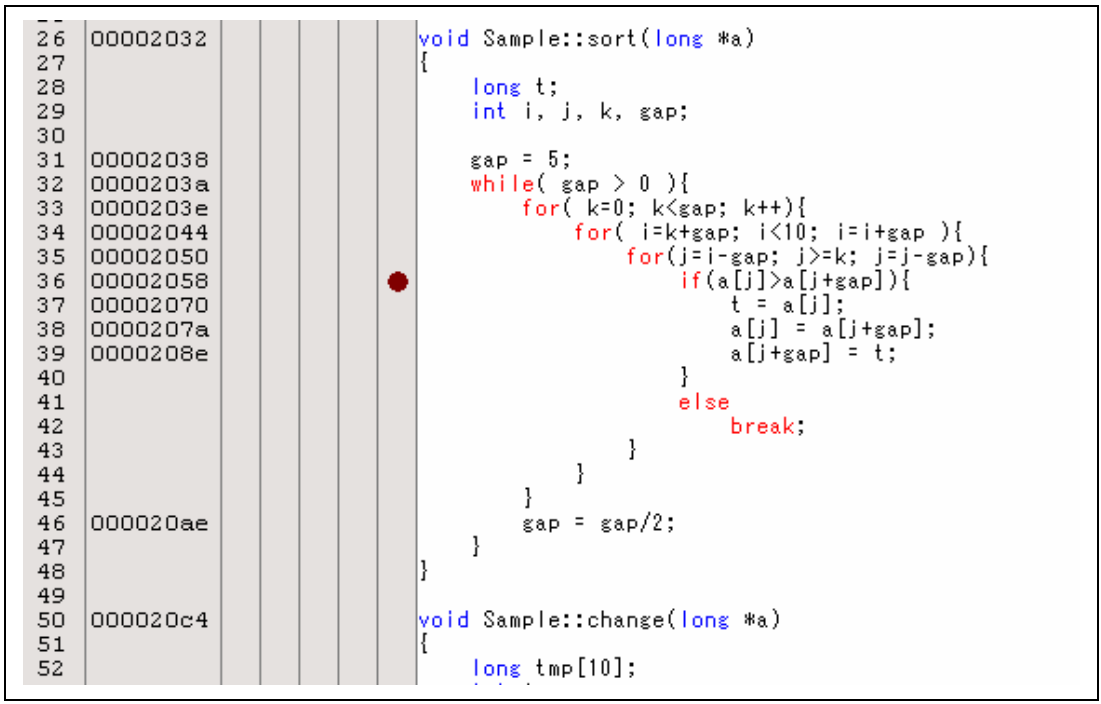


Figure 6.60 [Source] Window (PC Breakpoint Setting)

- Set the same program counter and stack pointer values (PC = H'00000800 and R15 = H'00010000) as were set in section 6.8, Setting Registers (again, use the [Register] window). Click the [Go] button.
- If program execution is failed, reset the device and execute again the procedures above.
- After the break in program execution, select [Stack Trace] from the [Code] submenu of the [View] menu to open the [Stack Trace] window.

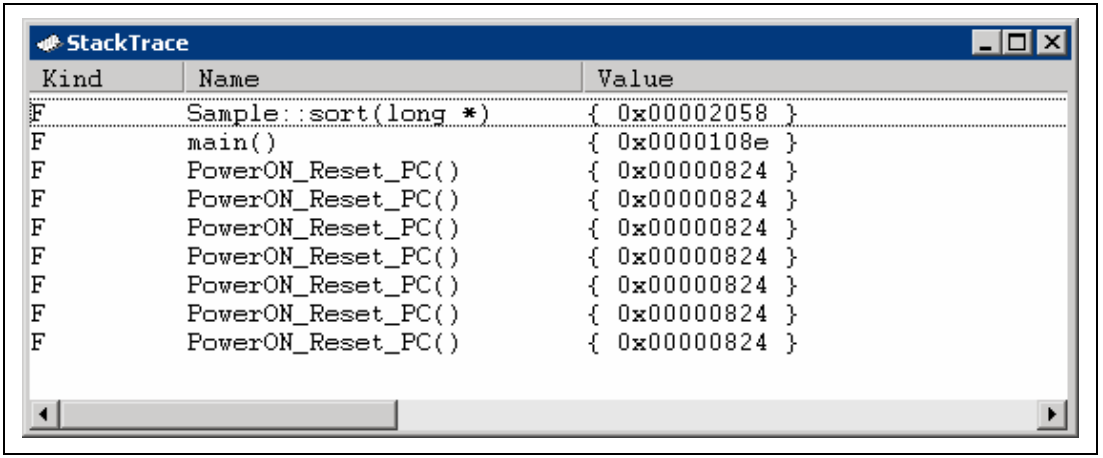


Figure 6.61 [Stack Trace] Window

Figure 6.61 shows that the position of the program counter is currently at the selected line of the `sort()` function, and that the `sort()` function is called from the `main()` function.

To remove the PC breakpoint, double-click the [Source] column in the `sort` function again.

Note: For details on this function, refer to the online help.

6.22 Download Function to the Flash Memory Area

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

Note: The write and erase modules must be prepared by the user.

- Interface with write and erase modules and emulator firmware

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

  — Describe all the write and erase modules with the assembly language.

  — Save and return all the general register values and control register values before and after calling the write or erase module.

  — Return the write or erase module to the calling source after processing.

  — The write and erase module must be a Motorola-type file.

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

Table 6.3 Module Interface

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

Note: The (L) means the longword size.

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

- Flash memory download method

  For downloading to the flash memory, set the items on the [Loading flash memory] page in the [Configuration] dialog box, which is opened from [System…], then [Emulator] from the [Options] menu.

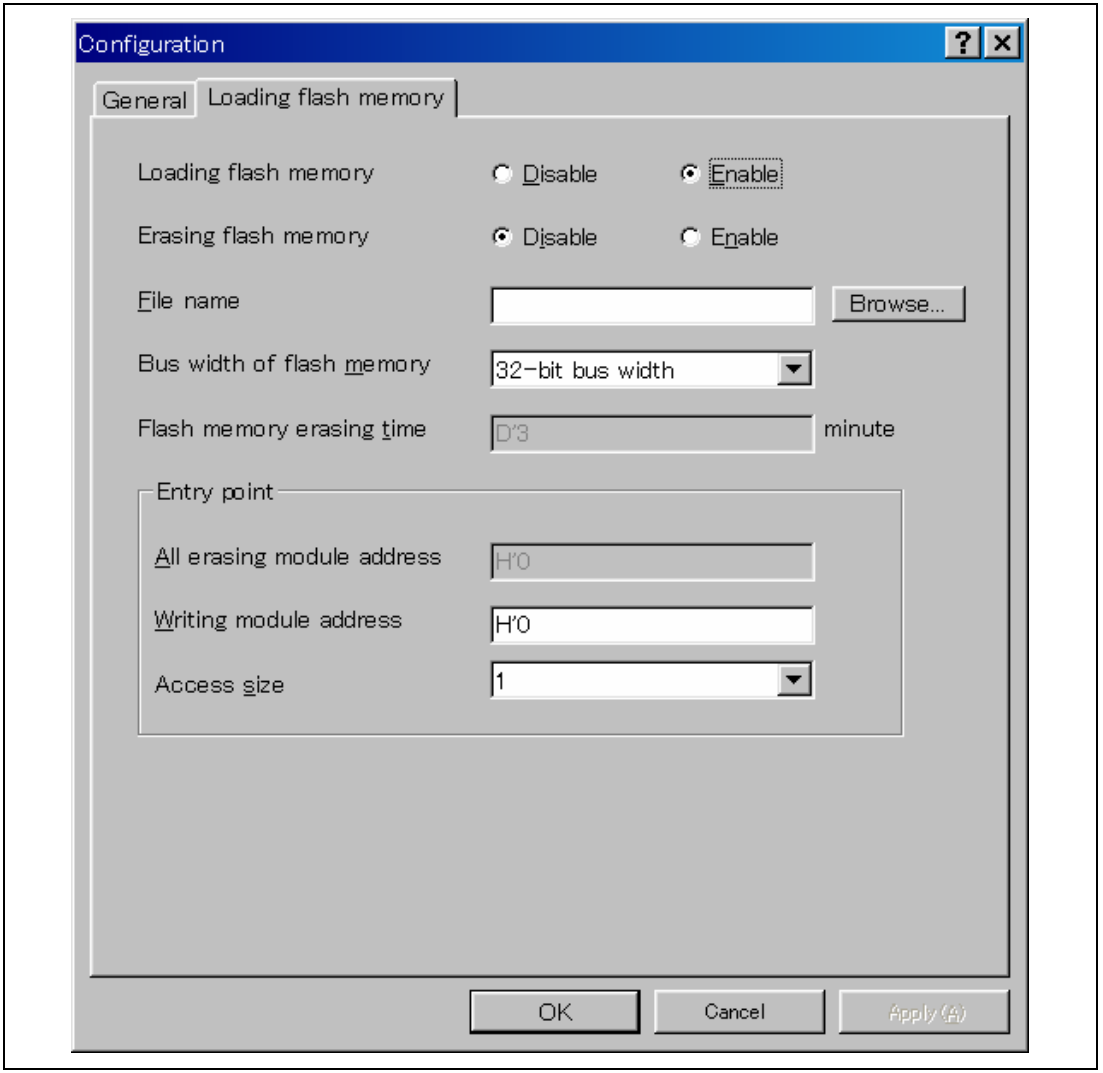


Figure 6.62 [Loading flash memory] Page

Table 6.4 shows the options for the [Loading flash memory] page.

Table 6.4 [Loading flash memory] Page Options

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

Note: Although the values that can be set are D'1 to D'65535, the TIMEOUT period may be extended according to the set value. Therefore, it is recommended to input the minimum value by considering the erasing time of the flash memory in use.

- Notes on using the flash memory download function

  The following are notes on downloading to the flash memory.

  — When the flash memory download is enabled, downloading to areas other than the flash memory area is disabled.

  — Downloading is only enabled to the flash memory area. Perform memory write or PC break only to the RAM area.

  — When the flash memory erase is enabled, the [Stop] button cannot stop erasing.

— The area for the write and erase modules must be set in an MMU-disabled space.

- An example of downloading to the flash memory

  The following is an example of downloading to the flash memory manufactured by Intel Corporation (type number: G28F640J5-150) that has been connected as shown in figure 6.63. A sample is provided in the \Fmtool folder in the installation destination folder. Create a program that suits the user specifications by referring to this sample.

Table 6.5 Board Specifications

Item	Contents
SDRAM address	H'0C000000 to H'0FFFFFFF
Flash memory address	H'00000000 to H'01FFFFFF
Bus width of flash memory	32 bits
Operating environment Endian	Big endian

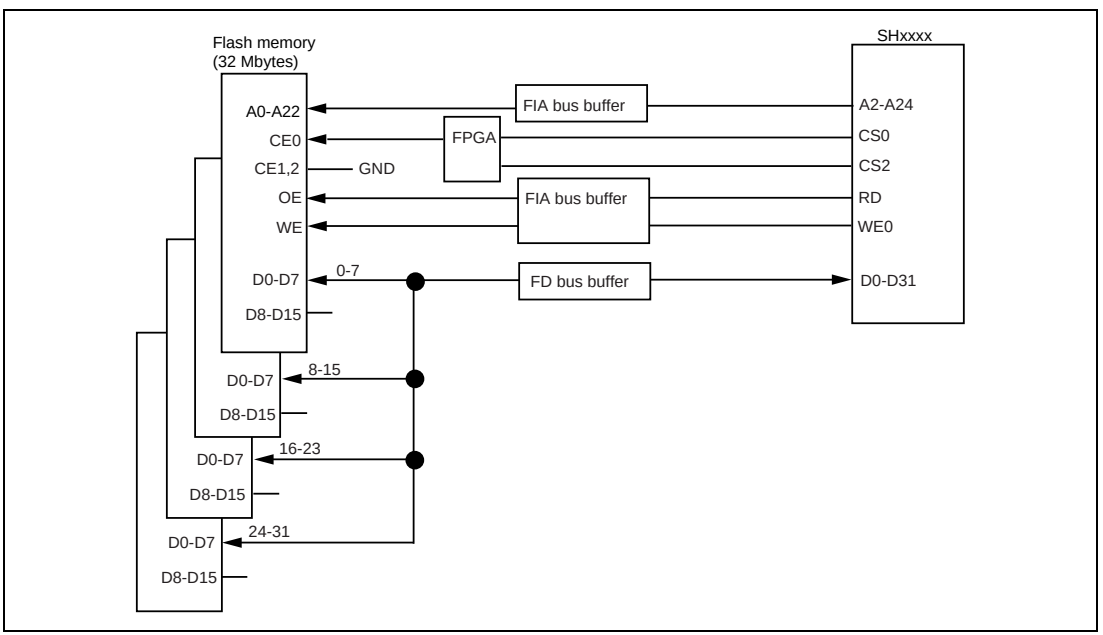


Figure 6.63 Flash Memory Wiring

Table 6.6 Sample Program Specifications

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

- Since the SDRAM is used, the bus controller must be set.
- Set the options on the [Loading flash memory] page in the [Configuration] dialog box as follows:

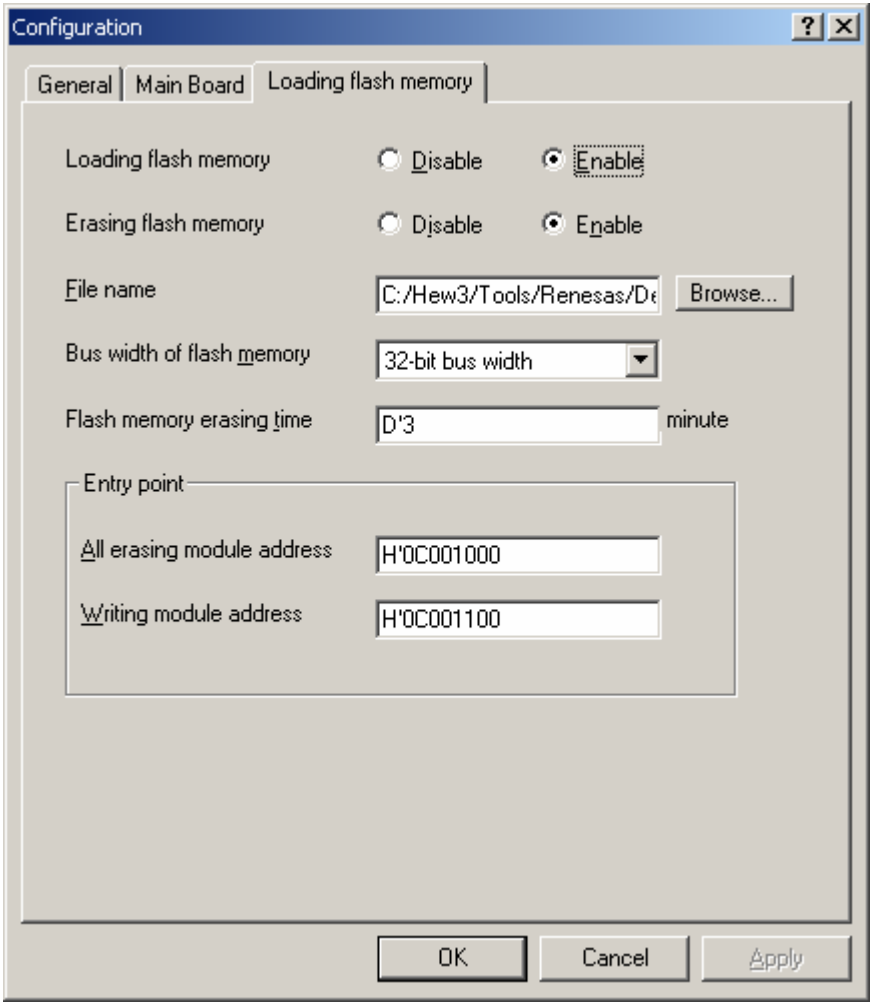


Figure 6.64 [Loading flash memory] Page

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

2. When [Erasing flash memory] is selected, it takes about one minute to erase the flash memory (in this example).

- Select the object for downloading to the flash memory area.

6.23 What Next?

This tutorial has described the major features of the emulator and the use of the High-performance Embedded Workshop.

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.

Section 7 Troubleshooting

1. ***I have a text file open in the editor but syntactic color-coding is not being displayed.***

   Ensure that you have named the file (i.e. saved it) and that the "Syntax coloring" check box is set on the "Editor" tab of the "Options" dialog box, which is launched via **[Setup -> Options...]**. The High-performance Embedded Workshop looks up the filename extension to determine the group to which the file belongs and decides whether or not coloring should be applied to the file. To view the currently defined filename extensions and file groups, select **[Project -> File Extensions...]** to launch the "File Extensions" dialog box. To view the coloring information, select **[Setup -> Format]** to display the "Color" tab of the "Format" dialog box (for further details, see the "Syntax Coloring" section in Chapter 4, "Using the Editor," of the volume on the High-performance Embedded Workshop).

2. ***I want to change the settings of a tool but the [Tools ->Administration…] menu option is not selectable.***

   [Tools ->Administration...] is not selectable while a workspace is open. To open the "Tool Administration" dialog box, close the current workspace.

3. ***I opened a workspace from my PC, and one of my colleagues opened the same workspace simultaneously from another PC. I changed the settings of the workspace and saved it. My colleague saved the workspace after me. I opened the workspace again and found that the settings of the workspace differed from those I had made.***

   The last settings to be saved are effective. While a workspace is open in the High-performance Embedded Workshop, updating of the workspace is within the memory. The settings are not saved in a file unless the user intentionally saves the workspace.

In addition to above, refer to FAQs on the emulator and High-performance Embedded Workshop on the Renesas web site (www.renesas.com).

Section 8 Maintenance and Guarantee

This section describes maintenance, guarantee, repair provisions, and how to request for repair of the emulator.

8.1 User Registration

When you purchase our product, be sure to register as a user. For user registration, refer to the section of ‘User Registration’ (p. iii) of this user's manual.

8.2 Maintenance

1. If dust or dirt collects on any equipment of this product, wipe the board dry with a soft cloth. Do not use thinner or other solvents because these chemicals can cause the equipment's surface coating to separate.
2. When you do not use this product for a long period, for safety purposes, disconnect the power cable from the power supply.

8.3 Guarantee

If your product becomes faulty within one year after its purchase while being used under good conditions by observing ‘IMPORTANT INFORMATION’ described in this user's manual, we will repair or replace your faulty product free of charge. Note, however, that if your product's fault is raised by any one of the following causes, we will repair it or replace it with new one with extra-charge:

- Misuse, abuse, or use under extraordinary conditions
- Unauthorized repair, remodeling, maintenance, and so on
- Inadequate user's system or misuse of it
- Fires, earthquakes, and other unexpected disasters

In the above cases, contact your local distributor. If your product is being leased, consult the leasing company or the owner.

8.4 Repair Provisions

8.4.1 Repair with Extra-Charge

The products elapsed more than one year after purchase can be repaired with extra-charge.

8.4.2 Replacement with Extra-Charge

If your product's fault falls in any of the following categories, the fault will be corrected by replacing the entire product instead of repair, or you will be advised to purchase new one, depending on the severity of the fault.

- Faulty or broken mechanical parts
- Flaw, separation, or rust in coated or plated parts
- Flaw or cracks in plastic parts
- Faults or breakage caused by improper use or unauthorized repair or modification
- Heavily damaged electric circuits due to overvoltage, overcurrent or shorting of power supply
- Cracks in the printed circuit board or burnt-down patterns
- Wide range of faults that makes replacement less expensive than repair
- Unlocatable or unidentified faults

8.4.3 Expiration of the Repair Period

When a period of one year elapses after the model was dropped from production, repairing products of the model may become impossible.

8.4.4 Transportation Fees at Sending Your Product for Repair

Send your product to us for repair at your expense.

8.5 How to Make a Request for Repair

If your product is found faulty, follow the procedure below to send your product for repair.

Fill in the Repair Request Sheet included with this product, then send it along with this product for repair to your local distributor. Make sure that information in the Repair Request Sheet is written in as much detail as possible to facilitate repair.

⚠ CAUTION

Note on Transporting the Product:

When sending your product for repair, use the packing box and cushion material supplied with this product when delivered to you and specify handling caution for it to be handled as precision equipment. If packing of your product is not complete, it may be damaged during transportation. When you pack your product in a bag, make sure to use conductive polyvinyl supplied with this product (usually a blue bag). When you use other bags, they may cause a trouble on your product because of static electricity.

Appendix A Menus

Table A.1 shows GUI menus.

Table A.1 GUI Menus

Menu	Option			Shortcut	Toolbar Button	Remarks
View	Disassembly			Ctrl + D		Opens the [Disassembly] window.
	Command Line			Ctrl + L		Opens the [Command Line] window.
	Workspace			Alt + K		Opens the [Workspace] window.
	Output			Alt + U		Opens the [Output] window.
	Difference					Opens the [Difference] window.
	CPU	Registers		Ctrl + R		Opens the [Register] window.
		Memory…		Ctrl + M		Opens the [Memory] window.
		IO		Ctrl + I		Opens the [IO] window.
		Status		Ctrl + U		Opens the [Status] window.
		Cache		Shift + Ctrl + C		Opens the [Cache] window.
		TLB		Shift + Ctrl + X		Opens the [TLB] window.
		Monitor	Monitor Setting	Shift + Ctrl + E		Opens the [Monitor Setting] dialog box.
			Window Select			Opens the [Window Select] dialog box.
		Extended Monitor				Opens the [Extended Monitor] window.

Table A.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
View (cont)	Sym-bol	Labels	Shift + Ctrl + A		Opens the [Labels] window.
		Watch	Ctrl + W		Opens the [Watch] window.
		Locals	Shift + Ctrl + W		Opens the [Locals] window.
	Code	Eventpoints	Ctrl + E		Opens the [Event] window.
		Trace	Ctrl + T		Opens the [Trace] window.
		Stack Trace	Ctrl + K		Opens the [Stack Trace] window.
	Gra-phic	Image…	Shift + Ctrl + G		Opens the [Image] window.
		Waveform…	Shift + Ctrl + V		Opens the [Waveform] window.
	Per-form-ance	Performance Analysis	Shift + Ctrl + P		Opens the [Performance Analysis] window.
		Profile	Shift + Ctrl + F		Opens the [Profile] window.
		Realtime Profile	Shift + Ctrl + R		Opens the [Realtime Profile] window.
Setup	Radix	Hexadecimal			Uses a hexadecimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Decimal			Uses a decimal for displaying a radix in which the numerical values will be displayed and entered by default.
		Octal			Uses an octal for displaying a radix in which the numerical values will be displayed and entered by default.

Table A.1 GUI Menus (cont)

Menu	Option	Shortcut	Toolbar Button	Remarks
Setup (cont)	Radix (cont) Binary			Uses a binary for displaying a radix in which the numerical values will be displayed and entered by default.
	Emu-lator System…			Opens the [Configuration] dialog box allowing the user to modify the debugging platform settings.
Debug	Debug Sessions…			Opens the [Debug Sessions] dialog box to list, add, or remove the debug session.
	Debug Settings…			Opens the [Debug Settings] dialog box to set the debugging conditions or download modules.
	Reset CPU			Resets the target hardware and sets the PC to the reset vector address.
	Go	F5		Starts executing the user program at the current PC.
	Reset Go	Shift + F5		Resets the target microcomputer and executes the user program from the reset vector address.
	Go To Cursor			Starts executing the user program at the current PC until the PC reaches the address indicated by the current text cursor position.
	Set PC To Cursor			Sets the PC to the address at the row of the text cursor.
	Run…			Launches the [Run Program] dialog box allowing the user to enter the PC or PC breakpoint during executing the user program.
	Step In	F11		Executes a block of user program before breaking.

Table A.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
Debug (cont)	Step Over		F10		Executes a block of user program before breaking. If a subroutine call is reached, then the subroutine will not be entered.
	Step Out		Shift + F11		Executes the user program to reach the end of the current function.
	Step…				Launches the [Step Program] dialog box allowing the user to modify the settings for stepping.
	Step Mode	Auto			Steps only one source line when the [Source] window is active. When the [Disassembly] window is active, stepping is executed in a unit of assembly instructions.
		Assembly			Executes stepping in a unit of assembly instructions.
		Source			Steps only one source line.
	Halt Program		Esc		Stops the execution of the user program.
	Connect				Connects the debugging platform.
	Initialize				Disconnects the debugging platform and connects it again.
	Disconnect				Disconnects the debugging platform.
	Download Modules				Downloads the object program.
	Unload Modules				Unloads the object program.

Appendix B Command-Line Functions

The emulator supports the commands that can be used in the command-line window.

For details, refer to the online help.

Appendix C Notes on High-performance Embedded Workshop

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 downloading the load module, operating [Verify Memory] or [Save Memory] in the [Memory] window, or saving in the [Trace] or [Load Coverage] window because this will not allow correct file accessing to be performed.

4. 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 =

5. 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.

6. 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

   — File specification by commands

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

   Example: FILE_LOAD C:\HEW3\Tools\Renesas\DebugComp\Platform \E200F\Tutorial\Tutorial\Debug_SHxxxx_E200F_SYSTEM \tutorial.abs

7. Memory Save During User Program Execution

   Do not execute memory save or verifying during user program execution.

8. Load of Motorola S-type Files

   This High-performance Embedded Workshop 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.

9. Note on [Register] Window Operation During Program Execution

   The register value cannot be changed in the [Register] window during program execution. Even if the changed value is displayed, the register contents are not changed actually.

10. Break Function

   — BREAKPOINT cancellation

   When the contents of the BREAKPOINT address is modified during user program execution, the following message is displayed when the user program stops.

   BREAKPOINT IS DELETED A=xxxxxxxx

   If the above message is displayed, cancel all BREAKPOINT settings with the [Delete All] or [Disable] button in the [Eventpoint] window.

11. Number of BREAKPOINT and [Stop At] Settings in the [Run...] Menu

   The maximum number of BREAKPOINTs and [Stop At] settings allowed in the [Run...] menu is 1000. Therefore, when 1000 BREAKPOINTs are set, specification by [Stop At] in the [Run...] menu becomes invalid. Use the BREAKPOINTs and [Stop At] in the [Run...] menu with 1000 or less total settings.

12. Note on Displaying `Timeout error`

   If `Timeout error` is displayed, the emulator cannot communicate with the target microcomputer. Turn off the user system and connect the USB connector of the emulator again by using the High-performance Embedded Workshop.

13. Note on Using the [Run Program] Dialog Box

   When [Run...] is selected from the [Debug] 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.

14. BREAKPOINT Setting for SLEEP Instruction

   When a break is set for the SLEEP instruction, use the Break Condition not the BREAKPOINT.

15. Note on Session Save in the [Configuration] Dialog Box

   The following settings are not saved as a session:

   — JTAG clock in the [General] page

   — Loading flash memory in the [Loading flash memory] page

16. 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.

17. Memory Test Function

This product does not support the memory test function, which is used by selecting [Test…] from the [Memory] menu.

Appendix D Repair Request Sheet

Thank you for purchasing the E200F emulator (R0E0200F0EMU00 or R0E0200F2EMU00).

In the event of a malfunction, fill in the repair request sheet on the following pages and send it to your distributor.

Repair Request Sheet

To Distributor

Your company name:

Person in charge:

Tel.:

Item	Symptom
1. Date and time when the malfunction occurred	Month/Day/Year {at system initiation, in system operation} *Circle either of items in the braces { }.
2. Frequency of generation of the malfunction	( ) times in ( ) {day(s), week(s), or month(s)} *Enter the appropriate numbers in the parentheses ( ) and circle one of the three items in the braces { }.
3. System configuration when the malfunction occurred	System configuration of the emulator: • E200F emulator (R0E0200F0EMU00 or R0E0200F2EMU00): Serial No.: Revision: The side of the emulator unit is labeled with the above items; the serial no. is the four-digit number and the revision is the string of letters following the number. • Trace unit (R0E0200F0ETU00): Serial No.: Revision: These are impressed on the circuit board. • Expansion profiling unit (R0E0200F0EPU00): Serial No.: Revision: These are impressed on the circuit board. • Provided CD-R (R0E0200F0EMU00S): Version: V. Shown as 'V.x.xx release' on the CD-R (x: numeral). • Host computer in use: Manufacturer: Type number: OS: (Windows® 2000 or Windows® XP)

Item	Symptom
4. Settings when the malfunction occurred	(1) Operating mode: Mode (2) Voltage of the target system: V (3) Clock in use: (input from Xtal oscillation or external clock) -circle one item. (4) Operating frequency: MHz
5. Failure phenomenon	
6. Error in debugging	
7. Error in the diagnostic program	
8. The High-performance Embedded Workshop does not link-up with the emulator.	Content of the error message

For errors other than the above, fill in the box below.

Renesas Microcomputer Development Environment System
User's Manual
SH-4A, SH4AL-DSP E200F Emulator

Publication Date:	Rev.1.00, June 2, 2004
	Rev.6.00, July 25, 2007
Published by:	Sales Strategic Planning Div.
	Renesas Technology Corp.
Edited by:	Customer Support Department
	Global Strategic Communication Div.
	Renesas Solutions Corp.

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

RENESAS SALES OFFICES

http://www.renesas.com

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

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

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

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

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

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

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

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

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

SH-4A, SH4AL-DSP E200F Emulator
User's Manual

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

REJ10B0137-0600