User's Manual

SuperH™ Family E10A-USB Emulator

User's Manual
(HS0005KCU01H, HS0005KCU02H)

SuperH™ Family

E10A-USB H0005KCU01HE

Renesas Electronics
www.renesas.com

Rev. 10.01 Mar. 2016

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

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

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 systems manufactured by you.
8. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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.

Regulatory Compliance Notices

European Union regulatory notices

This product complies with the following EU Directives. (These directives are only valid in the European Union.)

CE Certifications:

• Electromagnetic Compatibility (EMC) Directive 2014/30/EU

EN 55022:2010 Class A

WARNING: This is a Class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

EN 55024:2010

• Information for traceability
  • Authorised representative
    Name: Renesas Electronics Corporation
    Address: Toyosu Foresia, 3-2-24, Toyosu, Koto-ku, Tokyo, 135-0061, Japan
  • Manufacturer
    Name: Renesas System Design Co.,Ltd.
    Address: 5-20-1, Josuihon-cho, Kodaira-shi, Tokyo 187-8588, Japan
  • Person responsible for placing on the market
    Name: Renesas Electronics Europe GmbH
    Address: Arcadiastrasse 10, 40472 Dusseldorf, Germany
  • Trademark and Type name
    Trademark: Renesas
    Product name: E10A-USB Emulator
    Type name: HS0005KCU01H / HS0005KCU02H

Environmental Compliance and Certifications:

• Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment (RoHS)

Directive 2002/95/EC

• Waste Electrical and Electronic Equipment (WEEE) Directive 2002/96/EC

● United States Regulatory notices

This product complies with the following EMC regulation. (This is only valid in the United States.)

FCC Certifications:

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:
(1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

CAUTION: Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

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 Electronics Corp. excluding all subsidiary products.

- Emulator
- User system interface cable

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

Purpose of the Emulator Product:

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

Limited Applications:

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

Improvement Policy:

Renesas Electronics 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.

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:

Other Important Things to Keep in Mind:

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.

Figures:

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

Device names:

This user's manual uses SHxxxx as an example of the device names.

Limited Anticipation of Danger:

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.

SAFETY PAGE

READ FIRST

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

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

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

DEFINITION OF SIGNAL WORDS

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

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

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

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

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

NOTE emphasizes essential information.

⚠ WARNING

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

1. **Do not repair or remodel the emulator product by yourself for electric shock prevention and quality assurance.**
2. **Always switch OFF the host computer and user system before connecting or disconnecting any CABLES or PARTS.**
3. **Connect the connectors in the user system and in the user interface cable by confirming the correct direction.**

CAUTION

Caution to Be Taken for Disposal:

Penalties may be applicable for incorrect disposal of this waste, in accordance with your national legislation.

European Union regulatory notices:

The WEEE (Waste Electrical and Electronic Equipment) regulations put responsibilities on producers for the collection and recycling or disposal of electrical and electronic waste. Return of WEEE under these regulations is applicable in the European Union only. This equipment (including all accessories) is not intended for household use. After use the equipment cannot be disposed of as household waste, and the WEEE must be treated, recycled and disposed of in an environmentally sound manner.

Renesas Electronics Europe GmbH can take back end of life equipment, register for this service at " http://www.renesas.eu/weee ".

Warnings on Emulator Usage

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

⚠ WARNING

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

CAUTION

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

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 install debugger software, a text file for user registration is created on your PC. Fill it in and email it to your local distributor. If you have replaced an emulator main unit or emulation probe, rewrite an emulator name and serial number in the text file you filled in earlier to register your new hardware products.

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, please contact your local distributor.

Table of Contents

Section 1 Overview

The E10A-USB emulator (hereafter referred to as the emulator) is a support tool for developing application systems to run 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. The emulator enables debugging anywhere indoors or out. The host computer for controlling the emulator must be an IBM PC compatible machine with USB 1.1/2.0 (Full-Speed).

Figure 1.1 shows the configuration of a system where the emulator is in use.

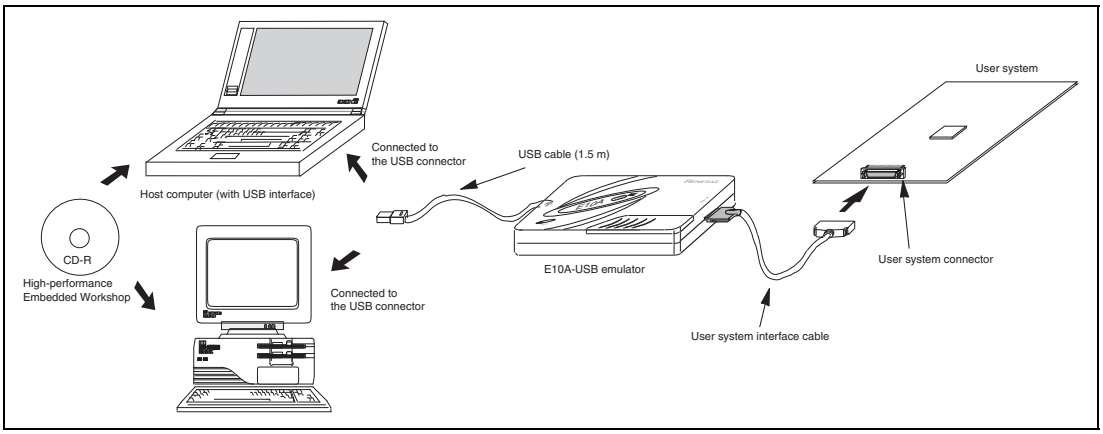


Figure 1.1 System Configuration with the Emulator

The emulator provides the following features:

- Excellent cost-performance emulator
  Compactness and connection to the USB are implemented.
- 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 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.
- Various debugging functions
  Various break and trace functions enable efficient debugging. Breakpoints and break conditions can be set by the specific window, trace information can be displayed on a window, and command-line functions can be used.
- 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 computer, creating a debugging environment in any place.
- AUD trace function*
  The AUD trace function enables realtime trace.

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

1.1 Warnings

CAUTION

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

1. Check all components against the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Protect the emulator from excessive impacts and stresses. For details, refer to section 1.2, Environmental Conditions.
4. When moving the host computer 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 3, Preparation before Use.
6. Supply power to the connected equipment after connecting all cables. Cables must not be connected or removed while the power is on.

1.2 Environmental Conditions

CAUTION

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

Table 1.1 Environmental Conditions

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

Table 1.2 lists the acceptable operating environments.

Table 1.2 Operating Environments

Item	32-Bit Editions of Windows® XP	32-Bit Editions of Windows Vista® or 32-Bit or 64-Bit Editions of Windows® 7
Host computer	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).	
CPU	Pentium® III (1 GHz) or higher recommended	Pentium® 4 (3 GHz), Core™ 2 Duo (1 GHz), or higher recommended
Minimum memory capacity	1 Gbyte or more recommended (at least 10 times the size of the load module file)	1.5 Gbyte or more recommended (at least 10 times the size of the load module file)
Hard-disk capacity	Installation disk capacity: 600 Mbytes or more. (Prepare an area at least double the memory capacity (four-times or more recommended) as the swap area.)	
Pointing device such as mouse	Connectable to the host computer; compatible with Windows® XP, Windows Vista®, or Windows® 7	
Display	Monitor resolution: 1024 x 768 or higher	
Power voltage	5.0 ± 0.25 V (USB-bus power type)	
Current consumption	HS0005KCU01H: 260 mA (max.) HS0005KCU02H: 420 mA (max.)	
CD-ROM drive	Required to install the High-performance Embedded Workshop for the emulator or refer to the emulator user's manual.	

Microsoft, Windows, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and or other countries. All other brand and product names are trademarks, registered trademarks or service marks of their respective holders.

1.3 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.

Section 2 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 6, Tutorial.

2.1 Overview

Table 2.1 gives a functional overview of the emulator.

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

Table 2.1 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	Trace functions	• Trace function incorporated in the device • AUD trace function: Branch trace or memory access trace • Memory output function of trace data
4	Break functions	• Hardware break function (conditions and the number of conditions differ according to the device) • PC break function (255 points) • Forced break function

Table 2.1 Emulator Functions (cont)

No.	Item	Function
5	Performance measurement function	• Uses a counter in the device to measure the number of cycles that passes during point-to-point execution. • Measures the number of cycles that pass in executing individual functions and lists them at the end of execution from a 'Go' command.
6	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 address)
7	General/control register access function	Reads or writes the general/control registers.
8	Internal I/O register access function	Reads or writes the internal I/O registers.*
9	Source-level debugging function	Various source-level debugging functions.
10	Command line function	Supports command input. Batch processing is enabled when a file is created by arranging commands in input order.
11	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 6, I/O File Format, in the High-performance Embedded Workshop V.4.09 User's Manual.
The following directory contains [SHxxxx.io] (xxxx means the name of emulator device group.):
<High-performance Embedded Workshop folder>:
\Tools\Renesas\DebugComp\Platform\E10A-USB\xxxx\IOFiles

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

2.2 Trace Functions

The emulator has two trace functions.

2.2.1 Internal Trace Function

The branch source and branch destination addresses, mnemonics, operands, and source lines are displayed. This function uses the trace buffer built into the device.

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 extended for some products. For details on the specifications of each product, refer to the online help.

2.2.2 AUD Trace Function

This is the large-capacity trace function that is enabled when the AUD pins are connected to the emulator. When an event that starts trace acquisition occurs, the trace information is output in realtime from the AUD pins. This function is only available on the E10A-USB emulator with model name HS0005KCU02H.

Counting each set of a branch source instruction and branch destination instruction as one branch, the maximum amount of information acquired by a trace is 32,767.

(1) Trace acquisition event

The following events can be acquired by the AUD trace function.

(a) Branch generation information

The branch source and branch destination addresses are acquired.

(b) Memory access information within the specified range

Memory access in the specified range can be acquired by trace.

Two memory ranges can be specified for channels A or B. The read, write, or read/write cycle can be selected as the bus cycle for trace acquisition.

This function is called the window trace function.

(c) Software trace

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 SHC/C++ compiler manual.

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

Notes: 1. This function cannot be supported with some versions of the SHC/C++ compiler. Since the supported version differs depending on the device to be debugged, refer to section 2.2.2, Trace Functions, in the additional document, Supplementary Information on Using the SHxxxx.

2. The types of events acquired by a trace differ depending on the product. For details on the specifications of each product, refer to the online help.

(2) Trace acquisition mode

The AUD trace function has the following modes to acquire a trace.

Table 2.2 shows the AUD trace acquisition mode that can be set in each trace function.

Table 2.2 AUD Trace Acquisition Mode

Type	Mode	Description
Continuous trace occurs	Realtime trace	When the trace information is being generated intensively that the output from the AUD pin incapable of keeping up, the CPU temporarily suspends the acquisition of trace information. Therefore, although the user program is run in real time, the acquisition of some trace information might not be possible.
	Non realtime trace	When trace information is being generated so intensively that the output from the AUD pin is incapable of keeping up, CPU operations are temporarily suspended and the output of trace information takes priority. In such cases, the realtime characteristics of the user program are lost.
Trace buffer full	Trace continue	This function overwrites the latest trace information to store the oldest trace information.
	Trace stop	After the trace buffer becomes full, the trace information is no longer acquired. The user program is continuously executed.

(3) Trace 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.
- Type: Displays the type of trace acquisition information.
- Address: Displays the addresses from which the trace data was acquired.
- Data: Displays the data acquired in the trace. For information without data, displays '********'.
- 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, Address, and Data columns have different meanings according to the type of AUD trace that has been selected.

Table 2.3 [Trace Window] Display Contents

Trace Type	Type Column	Address Column	Data Column
Branch trace	BRANCH	Branch source address	No display
	DESTINATION	Branch destination address	No display
Window trace*1	MEMORY	Memory access address	Memory access data
Software trace*1	S_TRACE	Trace(x) function execution address	Variable x data
Data lost*1, *2	LOST	No display	No display
CPU wait generation*1, *2	CPU-WAIT	No display	No display

Notes: 1. Not displayed in the internal trace.

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.

The following items will be displayed, according to the device to be debugged.

For specifications of the individual products, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help.

- 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 AUD trace that has been selected.

Table 2.4 [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.

2.2.3 Memory Output Function of Trace Data

In some devices to be debugged, trace data can be written to the specified memory range. The data is read from the memory range written in the [Trace] window and the result is then displayed.

Note: Do not specify the program area as the memory in the specified range is overwritten.

2.2.4 Useful Functions of the [Trace] Window

The trace window provides the following useful functions.

(1) Searches for the specified data.
(2) Extracts the specified data.
(3) Filters and displays again the specified data.
(4) Supplements the information from the branch destination address to the next branch source address.

For the usage of those functions, refer to section 5.6, Viewing the Trace Information.

(5) Changes the trace settings during user program execution.
In some devices to be debugged, trace settings can be changed during user program execution. For details on the specifications of each product, refer to the online help.

2.3 Break Function

The emulator has the following three break functions.

(1) Hardware break function

Uses a break controller incorporated in the device.

The access address, instruction fetch address, data, or bus cycle condition can be set. The logical address is the address condition.

This function can be also set from the [Event] column in the [Editor] or [Disassembly] window. For the setting, refer to section 5.2, Downloading a Program.

Note: In some devices to be debugged, hardware break settings can be changed during user program execution. For details on the specifications of each product, refer to the online help.

(2) PC break function (BREAKPOINT)

Breaks when the dedicated instruction at the specified address that has been replaced is executed. This function cannot be set at a place other than RAM or internal flash memory area since a memory write occurs.

It can also be set when the [S/W breakpoint] column for the line to be set is double-clicked in the [Editor] or [Disassembly] window.

(3) Forced break function

Forcibly breaks the user program.

2.4 Performance Measurement Function

The emulator has two types of performance measurement functions.

2.4.1 Function for Measuring the Number of Cycles from Point to Point

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.

This function is hereafter called the performance measurement function or PA1.

Notes: 1. Supplemental Explanation on Performance Measurement for Products of the SH-2A Device Group

Regarding Measurement of Numbers of Exceptions and Interrupts

Even when [Exception/interrupt counts (EA)] is selected as the item for measurement, trap-instruction exceptions due to TRAPA instructions will not be counted.

2. Items to be measured differ according to the product and some products do not support this function. For details on the specifications of each product, refer to the online help.

2.4.2 Profiling Function

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.

Notes: 1. Use of the profiling and performance measurement 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. Items to be measured differ according to the product and some products do not support this function. For details on the specifications of each product, refer to the online help.

2.5 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 that has not been within the MCU, select [Emulator] from the [Setup] 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

In some devices to be debugged, memory contents can be monitored during user program execution. For details on the specifications of each product, refer to the online help.

(7) 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 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.

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 and the internal flash memory. Therefore, an operation such as memory write or BREAKPOINT should be set only for the RAM area and the internal flash memory. 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 may change the cache data when it accesses memory. For details, refer to section 2.1 in the additional document, Supplementary Information on Using the SHxxxx.

2.6 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. 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 6.20, Stack Trace Function.

2.7 Function for Releasing Interrupts to the User during User Program Breaks

On some devices, all interrupts are open to the user during the execution of user programs. A mode setting is available to specify whether or not interrupt processing is executed during breaks in execution of the user program.

2.8 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.

Section 3 Preparation before Use

3.1 Emulator Preparation

Unpack the emulator and prepare it for use as follows:

⚠ WARNING

READ the reference sections shaded in figure 3.1 before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

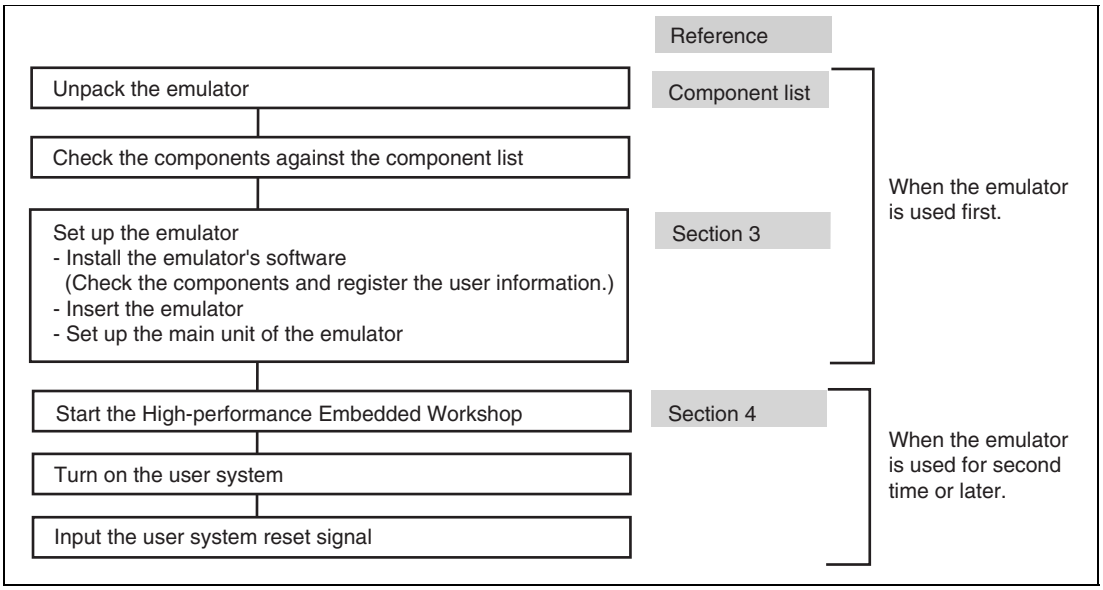


Figure 3.1 Emulator Preparation Flow Chart

3.2 Emulator Hardware Configuration

As shown in figure 3.2, the emulator consists of an emulator, a USB cable, and a user system interface cable. The emulator is connected to the host computer via USB 1.1, and also to the USB port conforming to USB 2.0.

Figure 3.2 Emulator Hardware Configuration (when the 36-pin Type Connector is Used)

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

Emulator Top View:

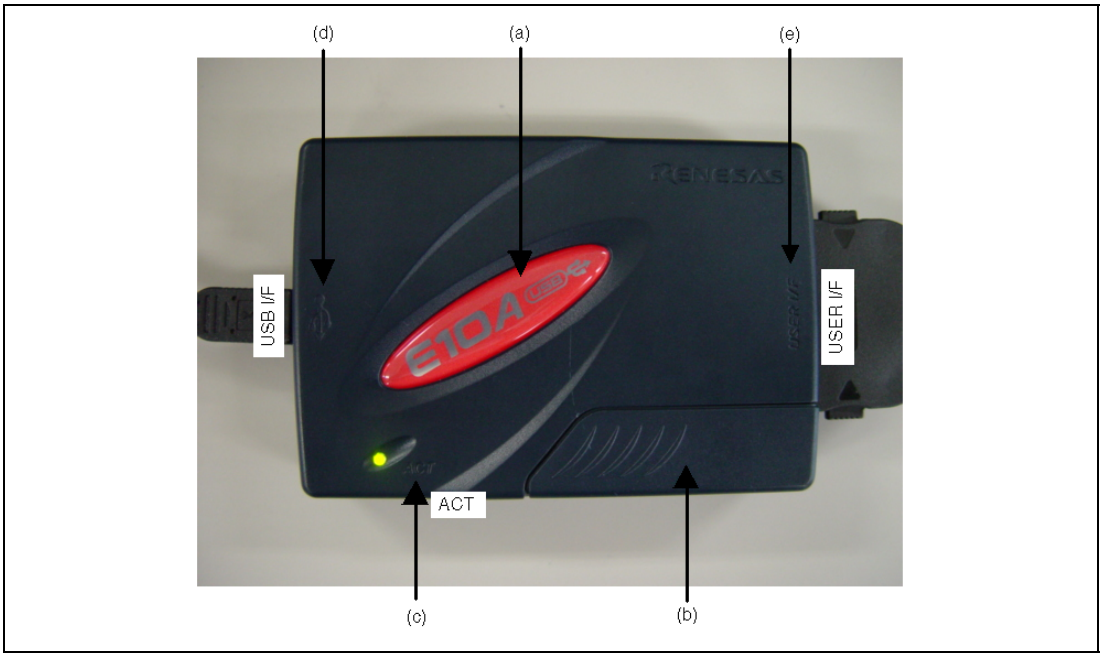


Figure 3.3 Emulator Top View

(a) E10A-USB logo plate: A yellow plate (for HS0005KCU01H) or a red plate (for HS0005KCU02H) dedicated for the emulator is provided to be easily distinguished from other E-series emulators.

(b) Sliding switch cover: A cover to protect switches for setting the emulator, which is closed to prevent incorrect operation. Be sure to close this cover during emulation.

(c) ACTION LED: Marked ‘ACT’. When this LED is lit, the E10A-USB control software is in operation.

(d) Host connector: Marked ‘’. A connector for the host computer is provided at the side of this mark.

(e) H-UDI port connector: Marked ‘USER I/F’. A connector for the user system interface cable is provided at the side of this mark.

Note: Even if the LED is not lit, the USB is not disconnected or malfunctioned.

Emulator Host-side View:

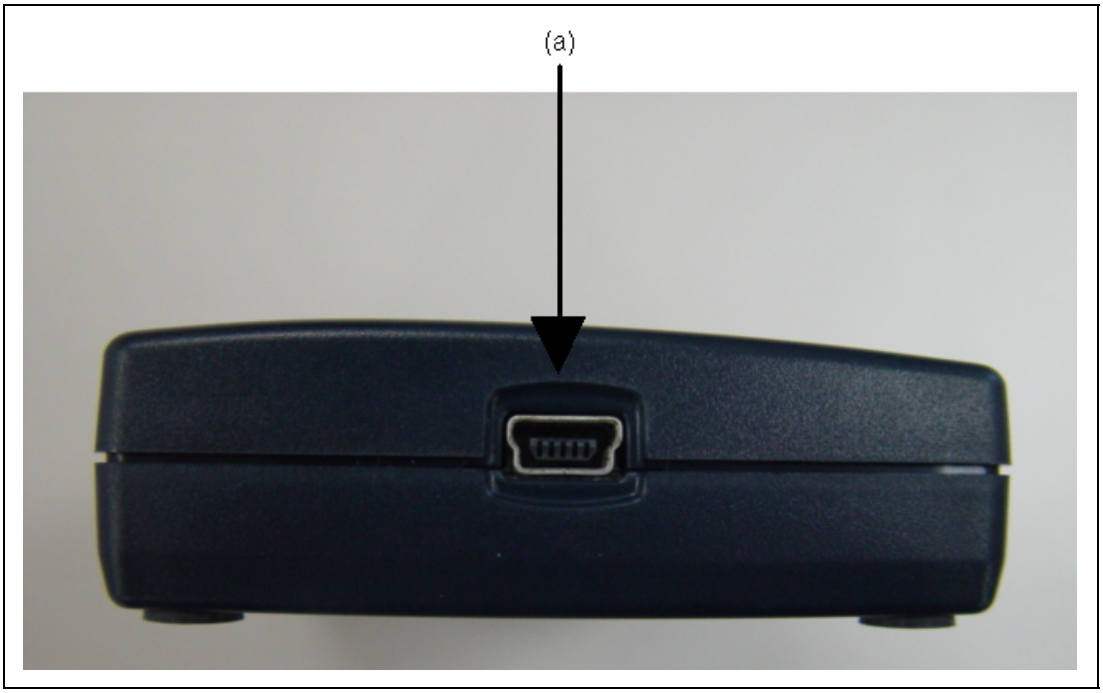


Figure 3.4 Emulator Host-side View

(a) Host-side connector: A USB connector for the host computer. Be sure to connect the provided USB cable.

Emulator User-side View:

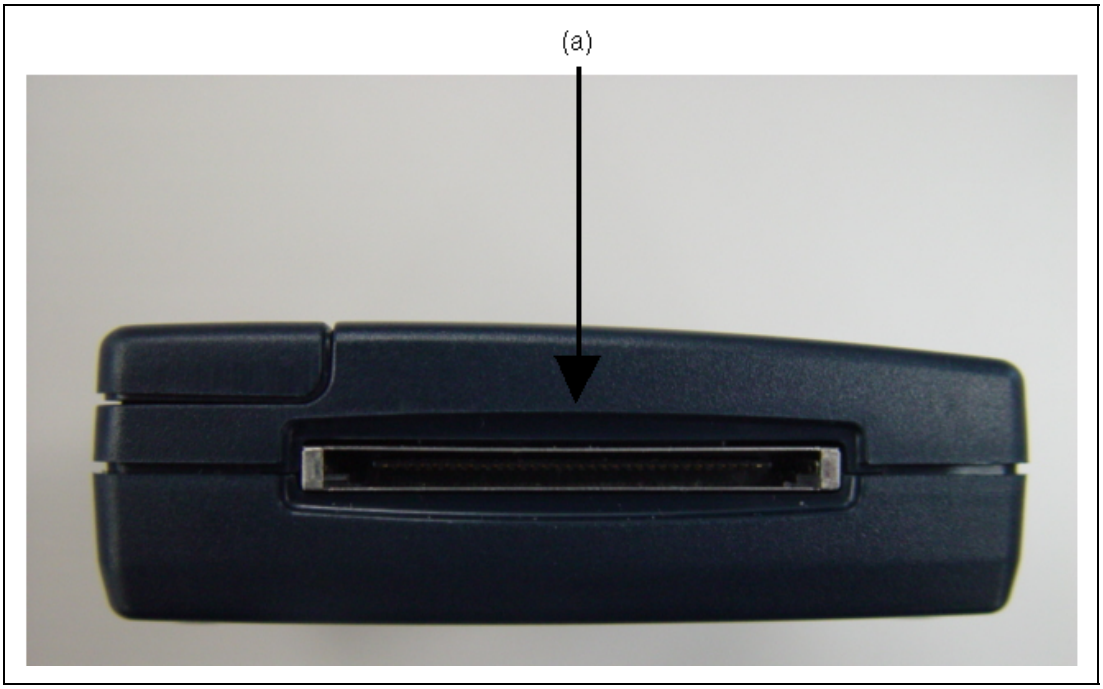


Figure 3.5 Emulator User-side View

(a) User-side connector: A user system interface cable is connected.

Emulator Bottom View:

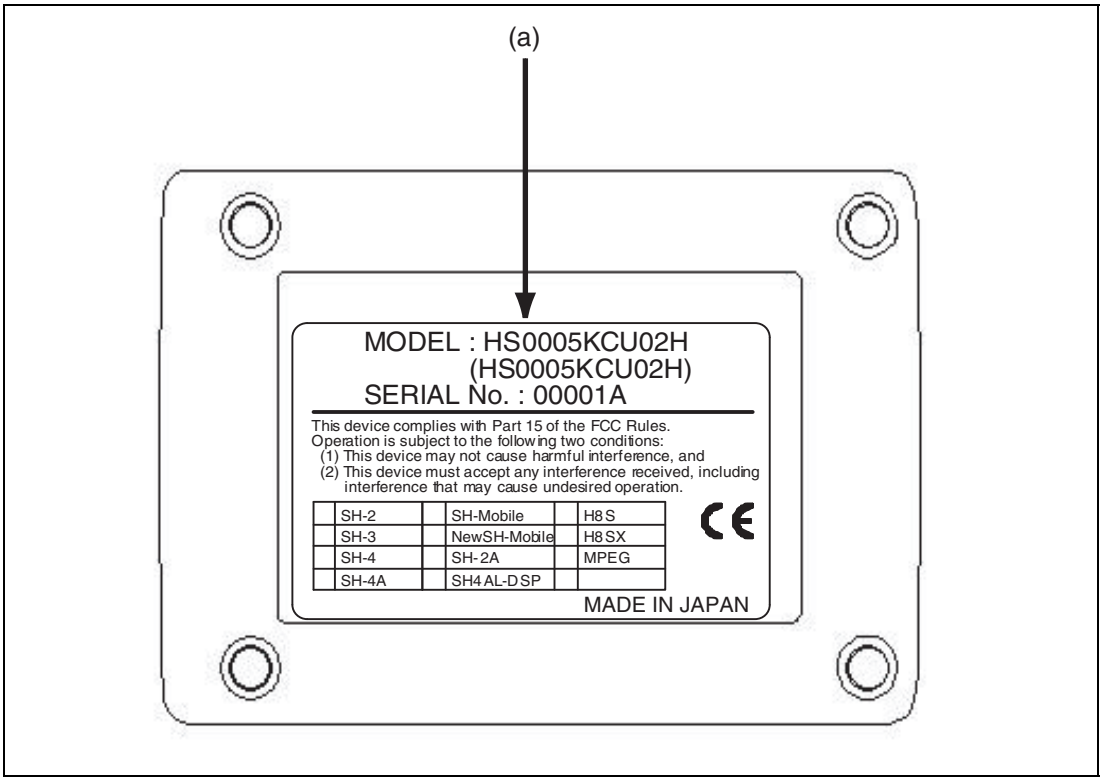


Figure 3.6 Emulator Bottom View

(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.
Only one device group can be set up using the setup tool when the emulator is purchased. Be sure to check the device group you have selected on the label for product management.

3.3 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 3.1 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	E10A-USB emulator driver	USB drivers for the E10A-USB emulator.
Help	Online help for the E10A-USB emulator	An online help file. This is copied to the hard disk as part of the installation process.
Manuals	E10A-USB emulator manuals	E10A-USB emulator user's manuals. They are provided as PDF files.

3.4 Installing Emulator's Software

Execute HewInstMan.exe from the root directory of the CD-R and follow the cues shown on screen to install the software.

Note: 1. 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. When installing a driver, the [Select Device Group] message will be displayed in the [Select Device] dialog box. Only the device group needs to be selected; that is, the device name does not have to be selected. The listed devices are those supported by the E10A USB. If the target device is not listed, the version of the emulator software you are using is old or support for the device may not be available.
3. The latest version of the emulator software is provided on the website. If the device to be used is not listed, it is not supported. In that case, please contact a distributor or agency since providing a preliminary version of the emulator software may be possible as an alternative.

3.5 Connecting the Emulator to the Host Computer

This section describes how to connect the emulator to the host computer. For the position of each connector of the emulator, refer to section 3.2, Emulator Hardware Configuration.

Note: 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 computer via the USB 1.1, and also to the USB port conforming to USB 2.0. Figure 3.7 shows the system configuration.

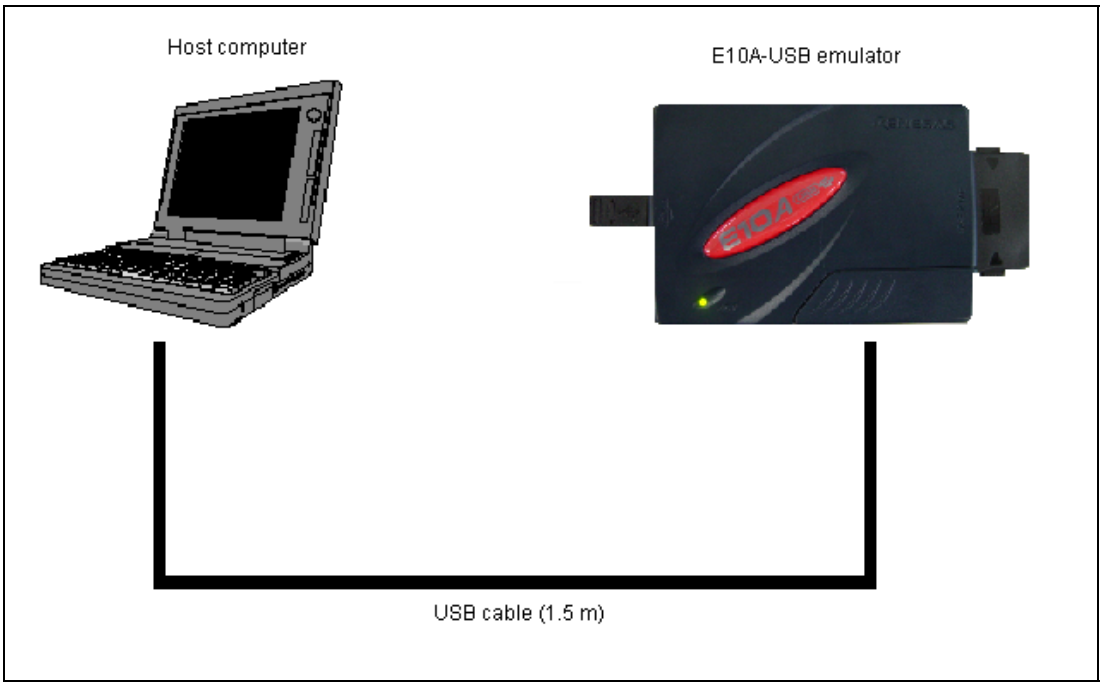


Figure 3.7 System Configuration when Connecting the Emulator to the Host Computer

3.6 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 host computer is turned off or the emulator is not connected to the host computer with the USB cable.
2. Connect the user system interface cable to the user-side connector of the emulator.
3. Connect the USB cable to the host-side connector of the emulator.

Figure 3.8 shows the position of the connector.

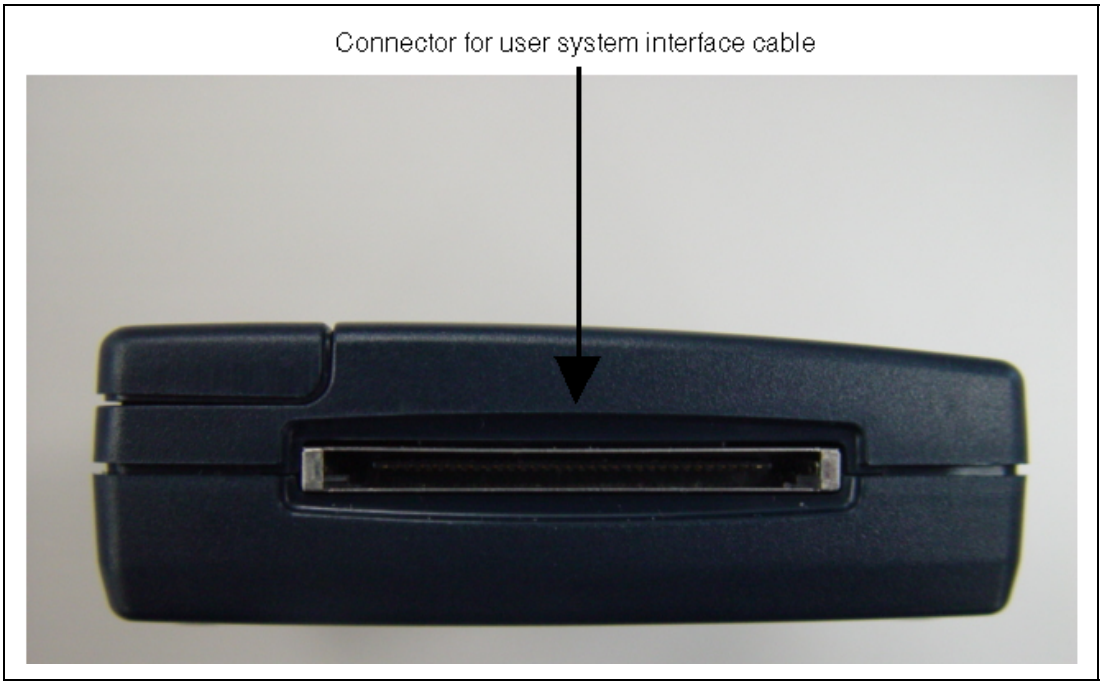


Figure 3.8 Position of the Connector

(1) The connector must be installed to the user system. Table 3.2 shows the recommended connector for the emulator.

Table 3.2 Recommended H-UDI Port Connector

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

Notes: 1. When designing the 14-pin connector layout on the user board, do not place any components within 3 mm of the H-UDI port connector.
When designing the 36-pin connector layout on the user board, do not connect other signal lines to the H-UDI port connector.
2. The H-UDI is an interface compatible with the Joint Test Action Group (JTAG) specifications.

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

(3) Connect pins 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 33, 34, and 36 (when using the 36-pin user system interface cable) and pins 9, 10, 12, 13, and 14 (when using the 14-pin user system interface cable) of the H-UDI port connector to GND firmly on the PCB. 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.

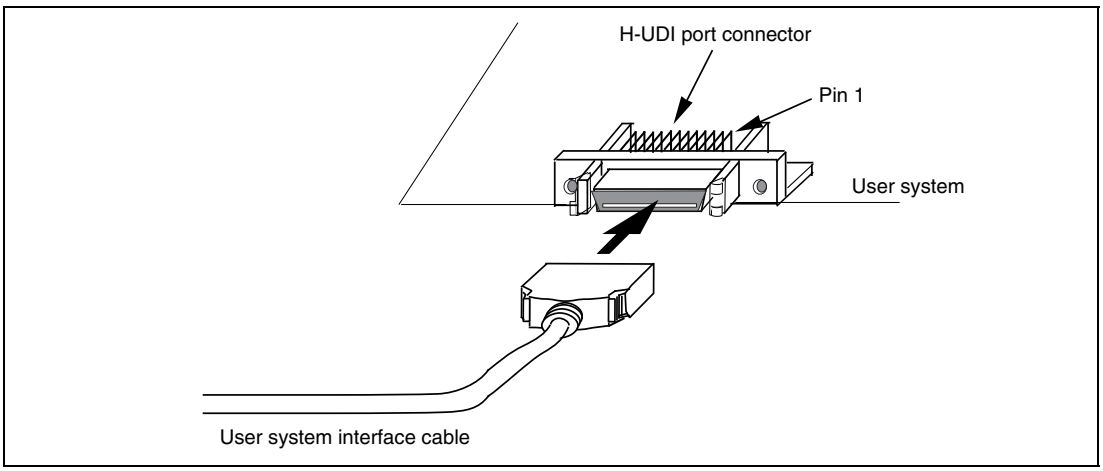


Figure 3.9 Connecting the User System Interface Cable to the User System when the 36-pin Type Connector is Used

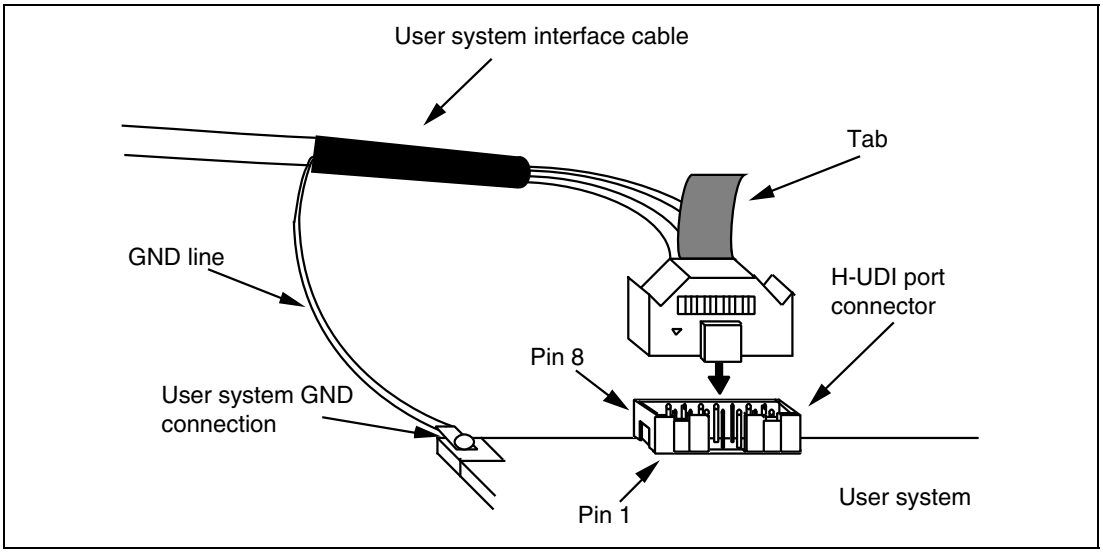


Figure 3.10 Connecting the User System Interface Cable to the User System when the 14-pin Type Connector is Used

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 device's pin assignments.
2. To remove the 14-pin type user system interface cable from the user system, pull the tab on the connector upward.
3. The range of communication that the emulator operates at is different depending on the device used.
4. To connect the signals from the connector, refer to section 1 in the additional document, Supplementary Information on Using the SHxxxx.
5. When developing user systems, do not connect the TDI and TDO signals of the device to the boundary scan loop, or separate them by using a switch (figure 3.11).

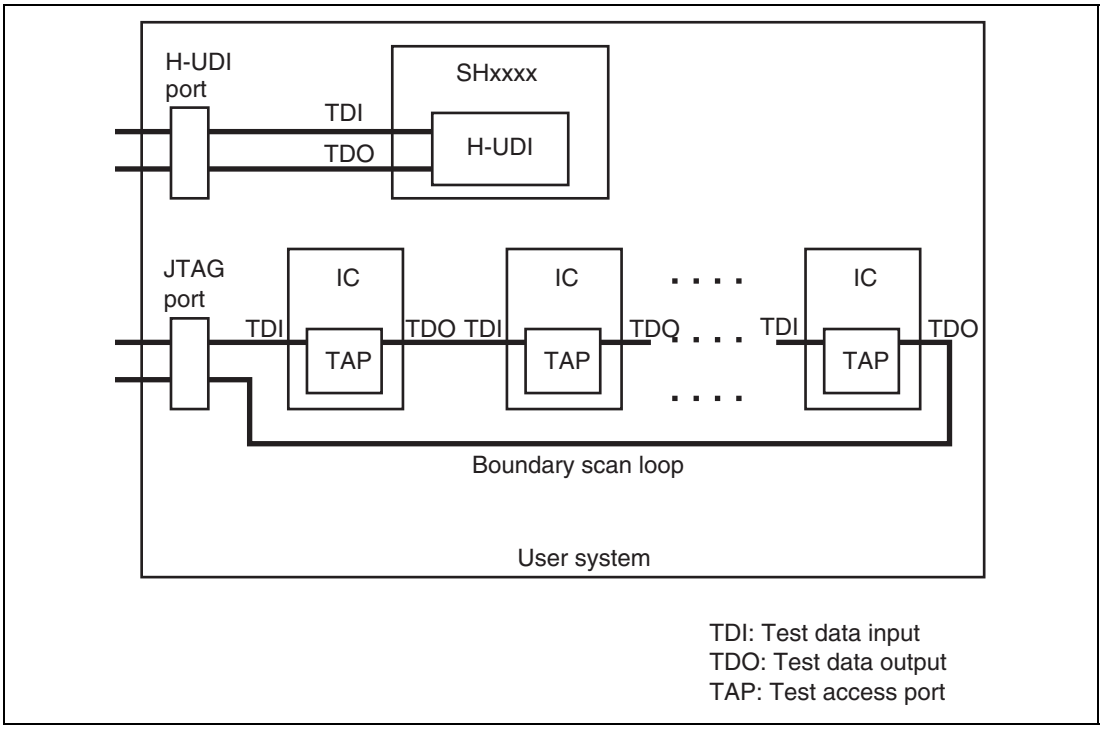


Figure 3.11 User System Example

3.7 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 3.12).

If it is difficult to separate the frame ground from the signal ground in the user system, set the GND for DC power input (AC adapter) of the host computer and the frame ground of the user system as the same potential. If the GND potential is different between the host computer and the target system, an overcurrent will flow in the low-impedance GND line and thin lines might be burned.

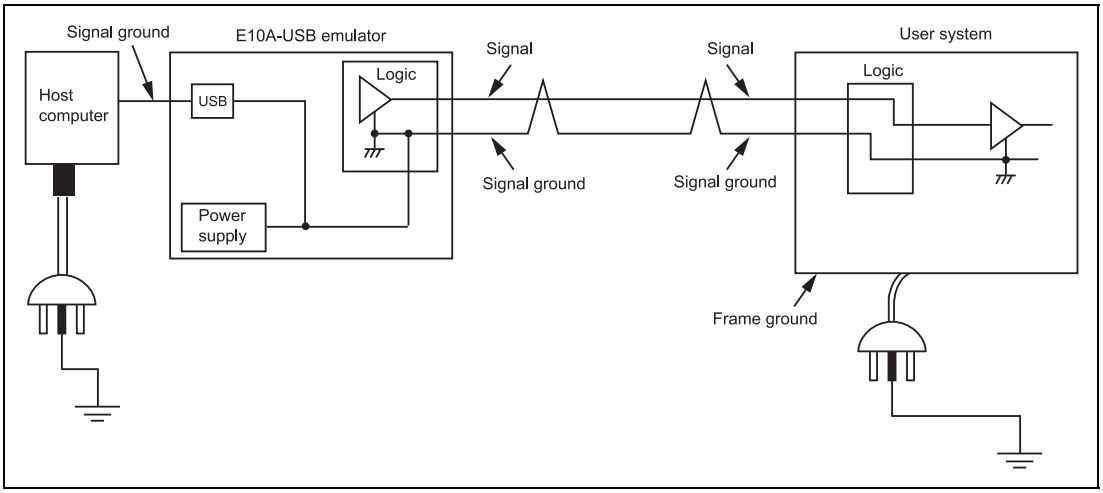


Figure 3.12 Connecting System Ground

3.8 Setting the DIP Switches

⚠ WARNING

Do not change switches (SW2 and SW3) while the emulator and the user sytem are turned on. The changing of switches (SW2 and SW3) will result in a FIRE HAZARD and will damage the user system and the emulator product. The USER PROGRAM will be LOST.

The emulator incorporates a switch (SW1) for setting up the emulator, a switch (SW2) for determining whether or not UVCC*[1] is connected, and a switch (SW3) for determining which pins of the H-UDI port connector are assigned to the /CA pin*[2] and the /AUDMD pin*[3]. To change these settings, use the DIP switches that are attached to the lower right of the emulator's upper side. To open the sliding switch cover, slide it to the right. The DIP switches consist of three switches (SW1 to SW3) as shown in figure 3.13. When they are in the upper position, the emulator is turned on. When they are in the lower position, the emulator is turned off.

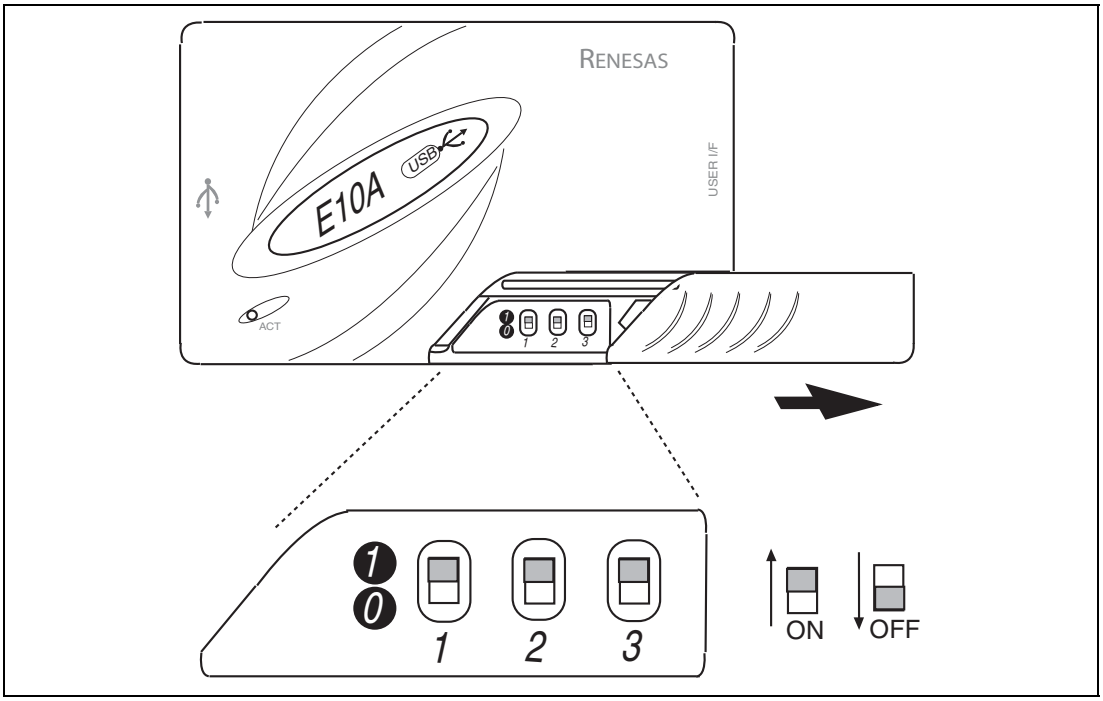


Figure 3.13 DIP Switches

Notes: 1. When the VCC pin (I/O power supply) on the user system is connected to the UVCC pin, the emulator is able to operate at the same voltage level of the user interface as VCC.
2. The /CA pin is only supported by the SH-Mobile microcomputers.
3. The /AUDMD pin is only supported by microcomputers with the SH2-series RAM monitoring function.

Tables 3.3 through 3.8 show the relationships between settings and functions of DIP switches 1 to 3. Use the settings depending on the usage of the user system.

CAUTION

Use only the settings shown in tables 3.3 to 3.8. Use of other settings will not activate the emulator. The USER PROGRAM will be LOST.

- Settings for use of the 36-pin interface cable

Description: When the VCC (I/O power supply related to the H-UDI) of the user system is connected to the UVCC pin of the H-UDI port connector, set the UVCC with the power supplied. Here, the I/O voltage of the user interface applies to the ranges between 1.8 V to 5.0 V. If the VCC is not connected, set the UVCC as disconnected.

Table 3.3 Switch Settings of the E10A-USB (Using SH-Mobile Series 36-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	0 (off)	3.3 V fixed	N.C.	/CA	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	/CA	UVCC	Power is supplied to UVCC

Table 3.4 Switch Settings of the E10A-USB (Using SH-2 Series 36-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	1 (on)	3.3 V fixed	/AUDMD	N.C.	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	/AUDMD	UVCC	Power is supplied to UVCC

Note: The /AUDMD and CK pins are only supported by microcomputers with the SH2-series RAM monitoring function. When SW2 = 0 and SW3 = 1, the CK pin can be connected to pin 29.

Table 3.5 Switch Settings of the E10A-USB (Using SH-3/SH-4 Series 36-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	1 (on)	3.3 V fixed	N.C.	N.C.	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	N.C.	UVCC	Power is supplied to UVCC

Table 3.6 Switch Settings of the E10A-USB (Using New_SH-Mobile Series/ SH-4A Series Products with 4-bit AUD Bus Width/SH-2A Series 36-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	1 (on)	1 (on)	1.8 V to 5.0 V	N.C.	UVCC	Normally used

Table 3.7 Switch Settings of the E10A-USB (Using SH-4A Series 36-pin Interface for Products with 8-bit AUD Bus Width)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	1 (on)	1 (on)	3.3 V fixed	AUDATA5	AUDATA6	Normally used

- Settings for use of the 14-pin interface cable

Description: When the VCC (I/O power supply related to the H-UDI) of the user system is connected to the UVCC pin of the H-UDI port connector, set the UVCC with the power supplied. Here, the I/O voltage of the user interface applies to the ranges between 1.8 V to 5.0 V. If the VCC is not connected, set the UVCC as disconnected.

Table 3.8 Switch Settings of the E10A-USB (Using SH-Mobile Series 14-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 8	Signal to be Connected to Pin 11	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	0 (off)	3.3 V fixed	N.C.*1	/CA	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	/CA	UVCC	Power is supplied to UVCC

Note: Pin 8 can be connected to GND.

Table 3.9 Switch Settings of the E10A-USB (Using SH-2 Series 14-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 8	Signal to be Connected to Pin 11	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	1 (on)	3.3 V fixed	N.C.*1	N.C.	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	N.C.	UVCC	Power is supplied to UVCC

Note: Pin 8 can be connected to GND.

Table 3.10 Switch Settings of the E10A-USB (Using SH-3/SH-4 Series 14-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 8	Signal to be Connected to Pin 11	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	0 (off)	1 (on)	3.3 V fixed	N.C.*1	N.C.	UVCC is disconnected
	1 (on)	1 (on)	1.8 V to 5.0 V	N.C.	UVCC	Power is supplied to UVCC

Note: Pin 8 can be connected to GND.

Table 3.11 Switch Settings of the E10A-USB (Using New_SH-Mobile Series/SH-4A Series Products with 4-bit AUD Bus Width/SH-2A Series 14-pin Interface)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	1 (on)	1 (on)	1.8 V to 5.0 V	N.C.	UVCC	Normally used

Table 3.12 Switch Settings of the E10A-USB (Using SH-4A Series 14-pin Interface for Products with 8-bit AUD Bus Width)

Switch Settings			State of the E10A-USB			
SW1	SW2	SW3	User-interface I/O Voltage	Signal to be Connected to Pin 15	Signal to be Connected to Pin 29	Condition
0 (off)	-	-	-	-	-	The emulator is only set up
1 (on)	1 (on)	1 (on)	3.3 V fixed	N.C.	N.C.	Normally used

3.9 Interface Circuits in the Emulator

Figures 3.14 through 3.17 show interface circuits in the emulator. Use them as a reference to determine the value of the pull-up resistance.

Note: The 74LVC2G125 operates at 3.3 V or VCC (1.8 to 5.0 V) from the H-UDI port connector (changed by the switch).

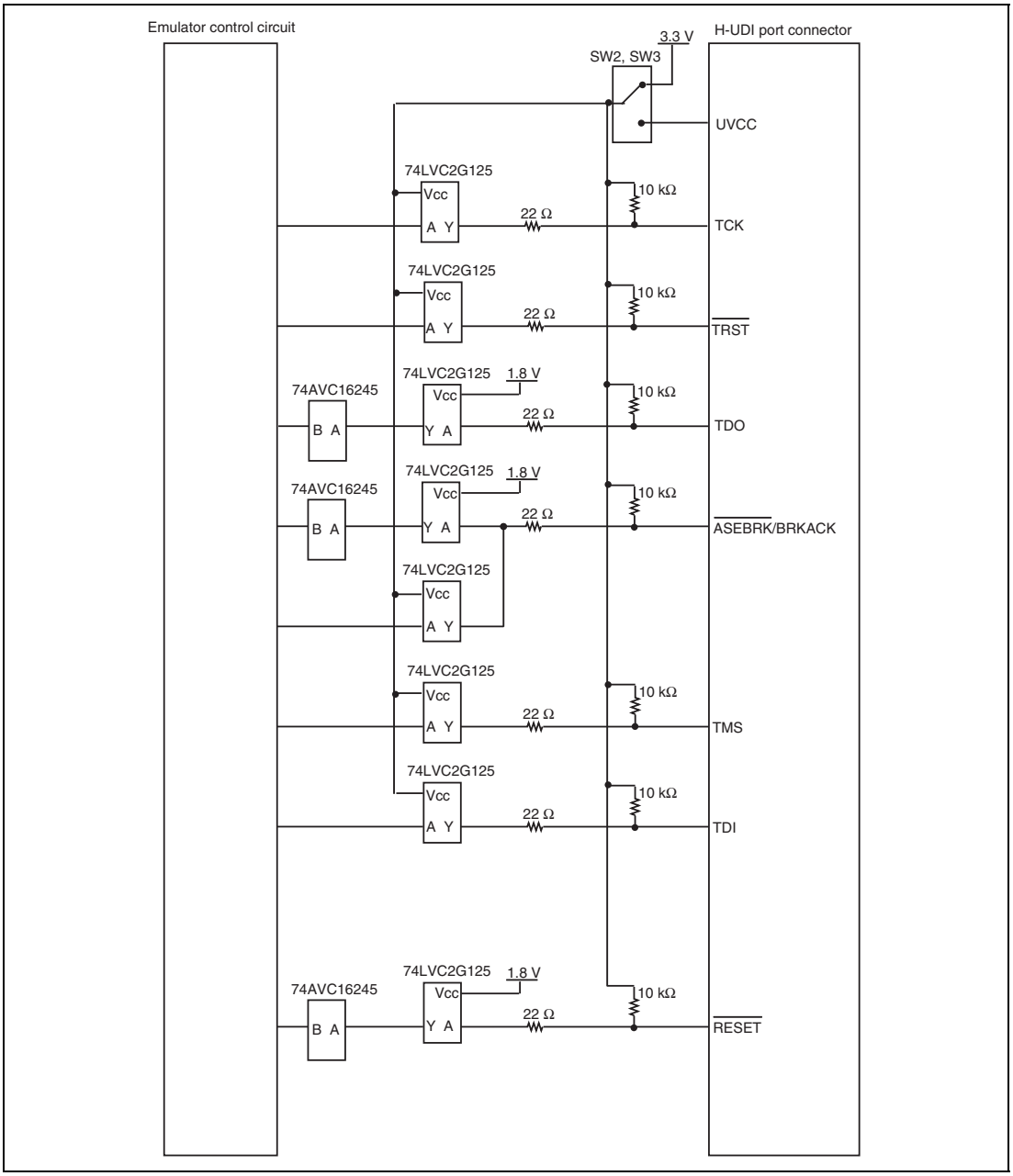


Figure 3.14 Interface Circuits in the Emulator (H-UDI)
(New_SH-Mobile Series / SH-4A Series / SH-2A Series)

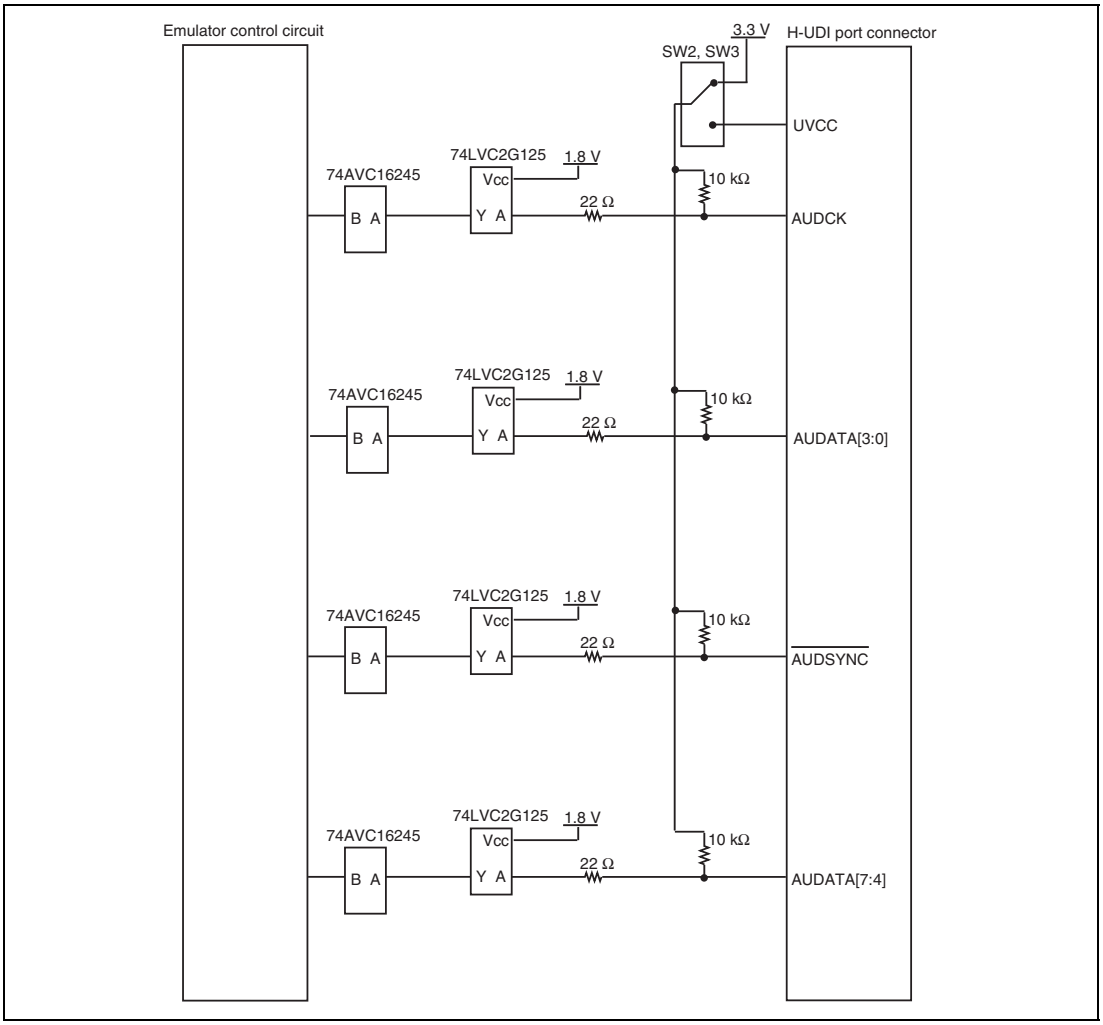


Figure 3.15 Interface Circuits in the Emulator (AUD)
(New_SH-Mobile Series / SH-4A Series / SH-2A Series)

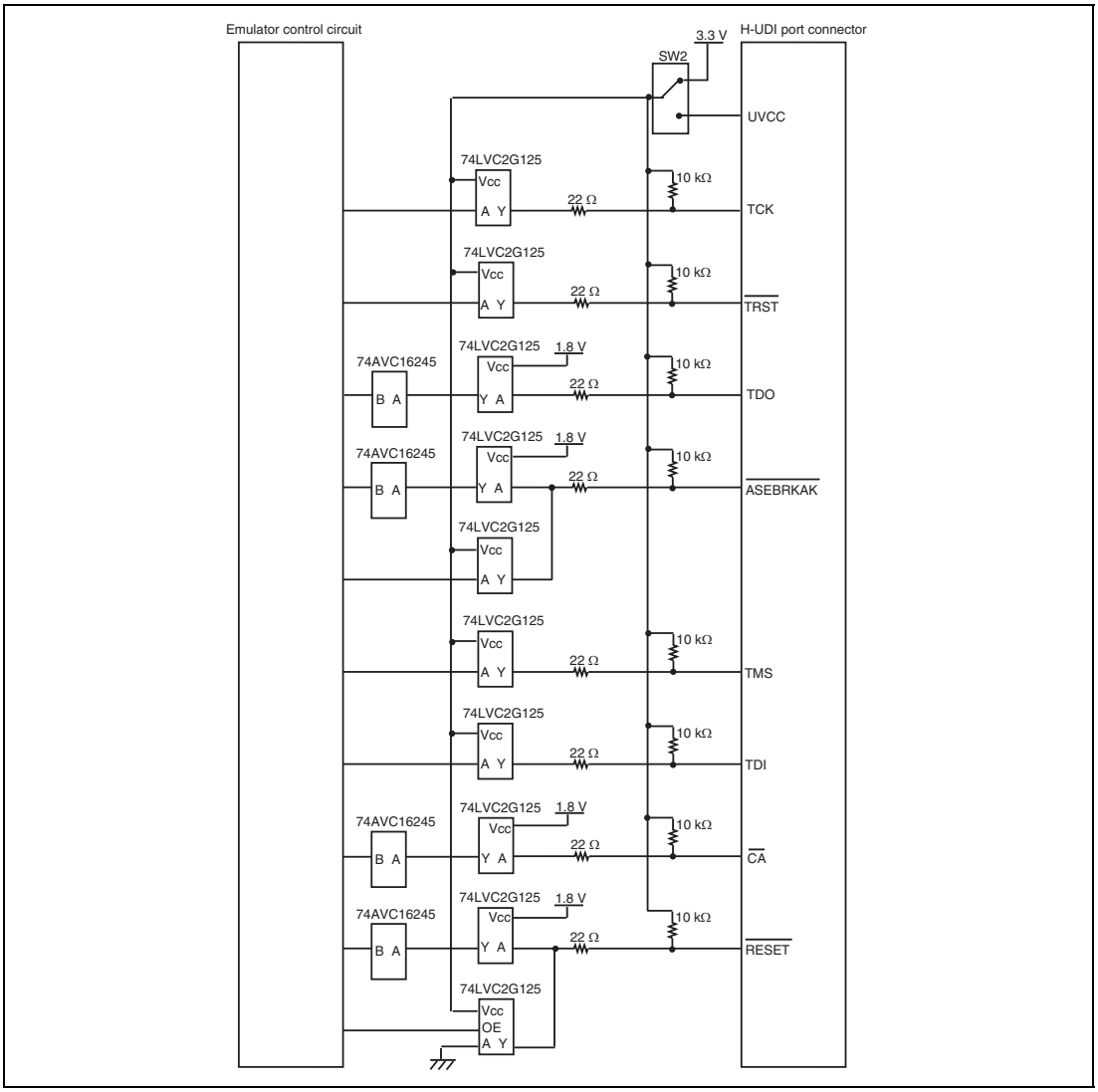


Figure 3.16 Interface Circuits in the Emulator (H-UDI)
(SH-Mobile Series / SH-4 Series / SH-3 Series / SH-2 Series / MPEG Series)

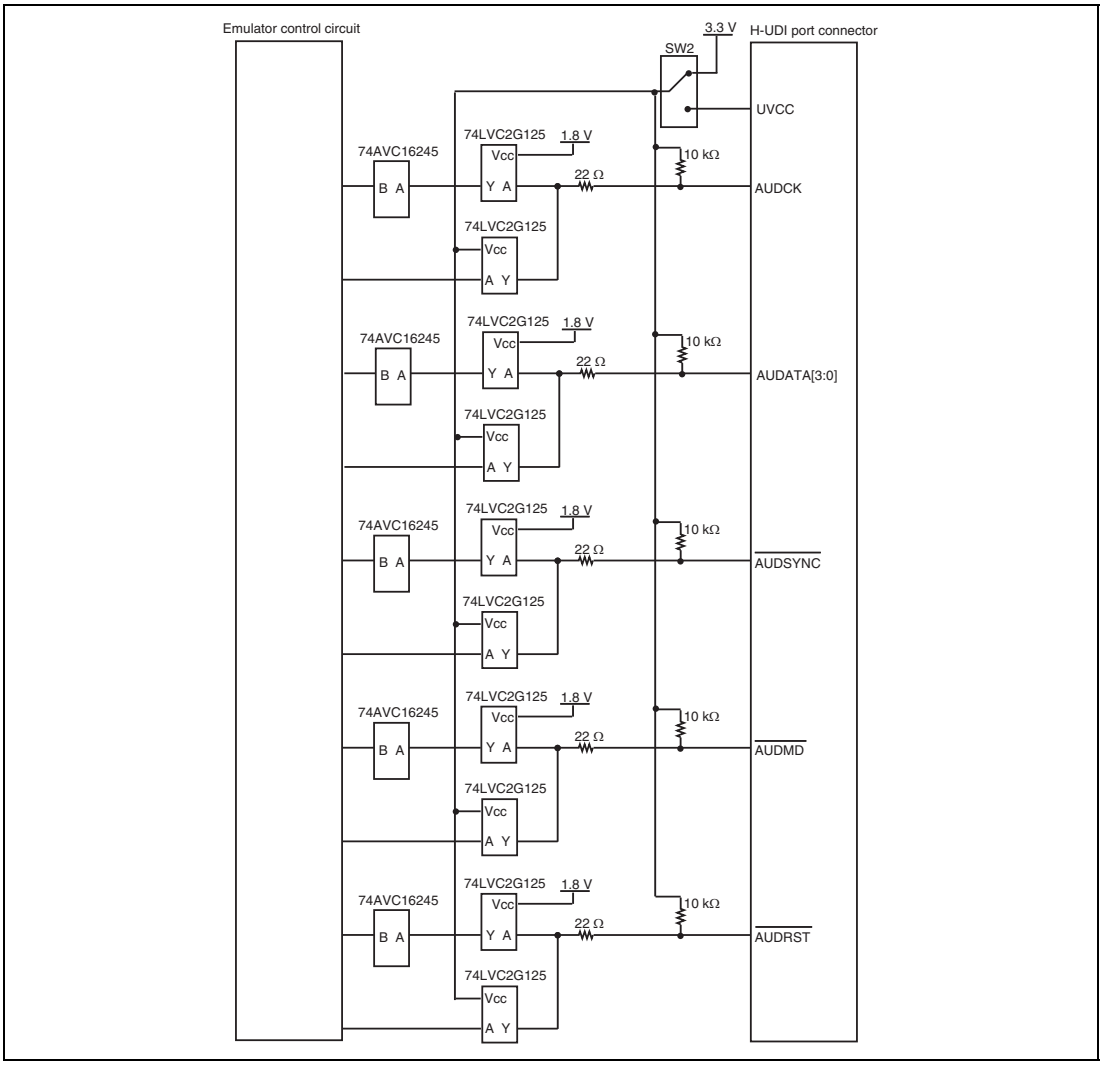


Figure 3.17 Interface Circuits in the Emulator (AUD)
(SH-Mobile Series / SH-4 Series / SH-3 Series / SH-2 Series / MPEG Series)

3.10 Setting up the Emulator

Set up the emulator's firmware using the following procedures.

Note: Only one device group can be set up using the setup tool when the emulator is purchased. Be sure to check the device group you have selected on the label for product management attached to the back of the emulator box. To use the emulator for another device group after set up, purchase the license tool to add a device group.

CAUTION

Do not disconnect the USB cable unless instructed to do so by an on-screen message. Incorrect operation will damage the emulator product.

3.10.1 Setting up at Purchasing the Emulator or Updating the Version of Software

Note: If you are using the HS0005KCU01H (serial No.: 03311C or later) or HS0005KCU02H (serial No.: 04146E or later) emulator hardware, the below procedure may not be required; follow the procedure only when the dialog box shown in figure 3.18 or 3.19 is displayed by using the procedure described in section 3.11, System Check.

Figure 3.18 [The product currently connected] Dialog Box

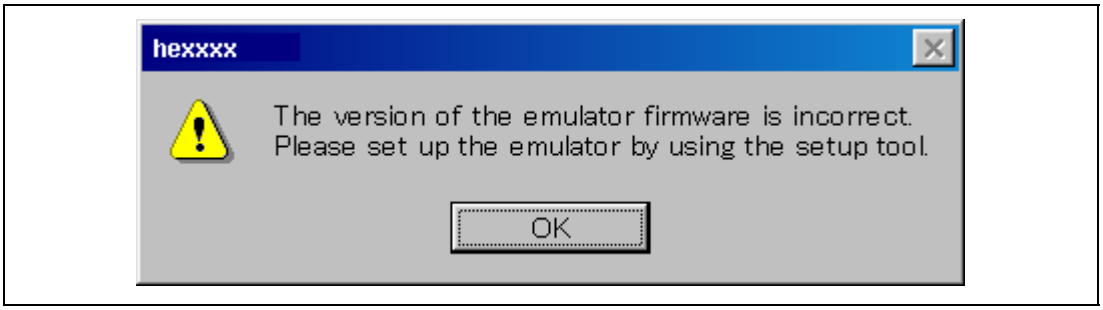


Figure 3.19 [The version of the emulator firmware is incorrect] Dialog Box

1. Open the sliding switch cover and check that the switch (SW1) for setting the emulator is turned to '1'.

2. Select [Renesas] -> [High-performance Embedded Workshop] -> [Tools] -> [Setup tool for E10A-USB Emulator] -> [SHxxxx Device Group] from [Programs] in the [Start] menu. A tool for setting up the emulator is activated.

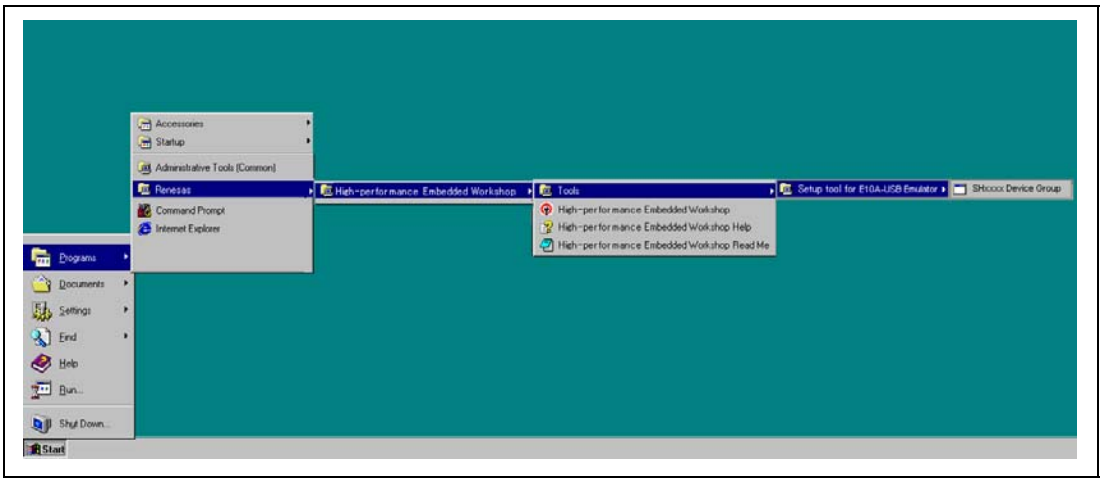


Figure 3.20 [Start] Menu

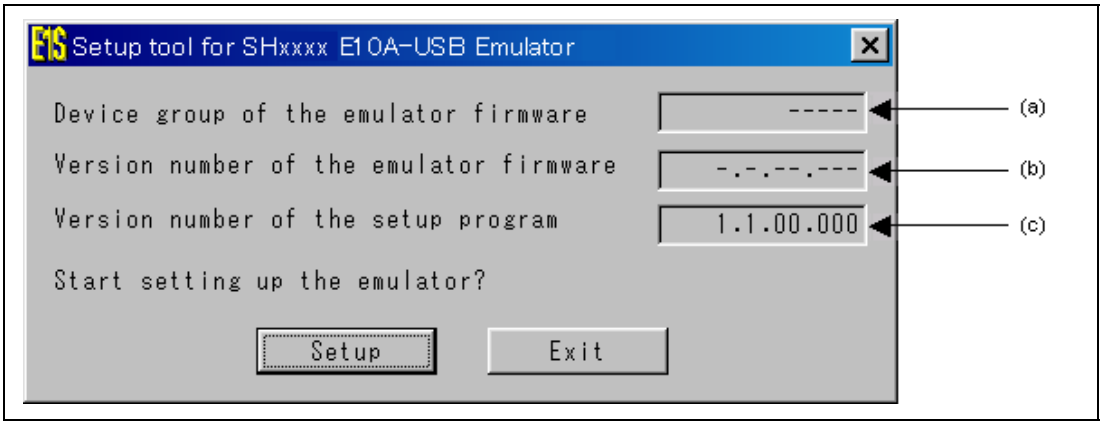


Figure 3.21 Setup Tool for Emulator

(a) Device group of the emulator firmware: Name of the device group currently set.

(b) Version number of the emulator firmware: The version number of software for controlling the SHxxxx group in the emulator. This item is displayed only when the SHxxxx group is available.

(c) Version number of the setup program: The version number of the setup program.

Notes: 1. If the version numbers shown in (b) and (c) are the same, setup of the emulator is not required. Setup the emulator only when "-.-.--.---" is shown in (b) or the version number of (b) is older than that of (c).

2. If an emulator other than the SHxxxx E10A-USB is connected, the following error message will be displayed to exit the setup tool.

Figure 3.22 Error Message

3. If the following error message is displayed, the host computer is not connected to the emulator or the setup switch (SW1) is turned to '0'.

Figure 3.23 Error Message

If the setup switch (SW1) is turned to '0', set it to '1' and connect the USB cable again.

3. Clicking the [Setup] button displays the following dialog box.

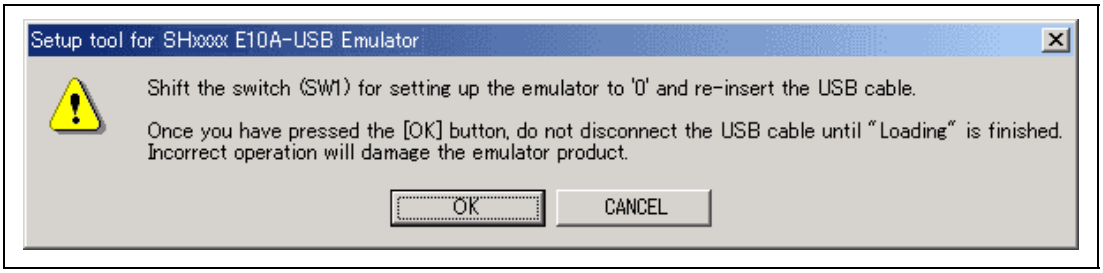


Figure 3.24 [Setup tool for SHxxxx E10A-USB Emulator] Dialog Box

4. Turn the setup switch (SW1) to ‘0’, connect the USB cable again, and click the [OK] button. Setting up the emulator’s firmware is started.

Notes: 1. If the following dialog message is displayed, insert the USB cable again.

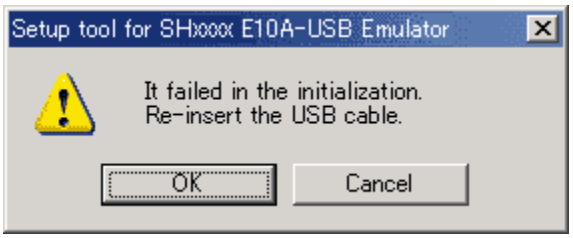


Figure 3.25 [Setup tool for SHxxxx E10A-USB Emulator] Dialog Box

2. When [Add New Hardware Wizard] is displayed, select [Install the software automatically].
3. Although a dialog box will be displayed to indicate disconnection of the USB, this is not a problem.

CAUTION

Do not turn off the host computer or disconnect the USB cable while setting up the emulator. The emulator may be damaged.

Figure 3.26 Start of Setting up the Emulator

5. When the following dialog box is displayed, setting up the emulator is completed.

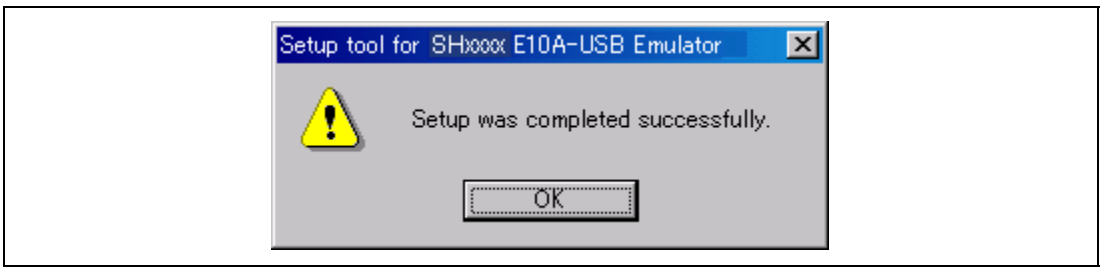


Figure 3.27 Message for Completion of Setting up the Emulator

6. When setting up the emulator has been completed, the following message will be displayed. Turn the setup switch (SW1) to '1', connect the USB cable again, and click the [OK] button.

Figure 3.28 [Setup Tool for SHxxxx E10A-USB Emulator] Dialog Box

Notes: 1. Be sure to turn the setup switch (SW1) to '1' except when the setup tool is used.
2. To use the license tool for another device group, it is necessary to set up the firmware by using the setup tool or license tool that corresponds to the device group. To use the setup tool, however, the license tool must also be installed. For details on the license tool, refer to the following section.

3.10.2 Setting up the Emulator by Using the License Tool to Add a Device Group

In the license tool, the emulator for the current product group can be used for debugging another product group (device group such as SHxxxx Device Group, H8S Device Group, or H8SX Device Group that is supported by the emulator).

In this section, the names of the device group that have already been installed and another device group to be added by the license tool are shown as SHxxxx Device Group and H8S Device Group, respectively. Replace these names according to your environment when you read this section.

The license tool to add a device group does not include software for the E10A-USB emulator.

Install the software for your product group by using the CD-R provided for the emulator or accessing the data on the web site.

Note: If no product groups have been installed in the emulator at the time of purchase, do not use the license tool.
Refer to section 3.10.1, Setting up at Purchasing the Emulator or Updating the Version of Software, and use the setup tool.

(1) Installing the emulator

Inserting the CD-R into the CD-ROM drive automatically activates the installation wizard (to prevent automatic activation, insert the CD-R by pressing the Shift key). If the installation wizard is not automatically activated, execute setup.exe from the root directory of the CD-R.

Follow the instructions by the installation wizard.

(2) Setting up the emulator

1. Open the sliding switch cover and check that the switch (SW1) for setting the emulator is turned to '1'.
2. Select [Renesas] -> [License tool for E10A-USB] -> [H8S Device Group] from [Programs] in the [Start] menu. This starts up the license tool to add a device group to the emulator.

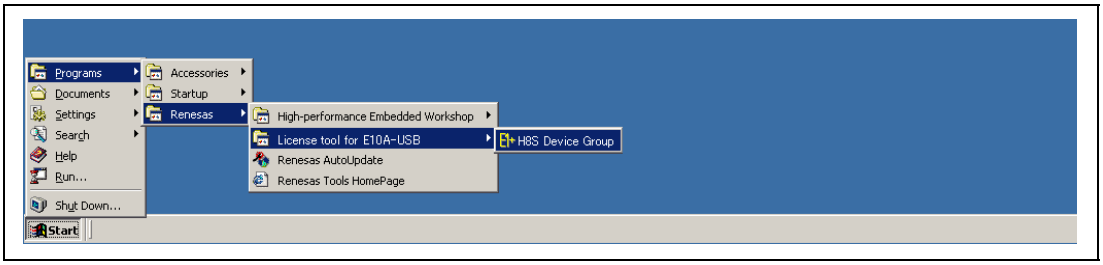


Figure 3.29 [Start] Menu

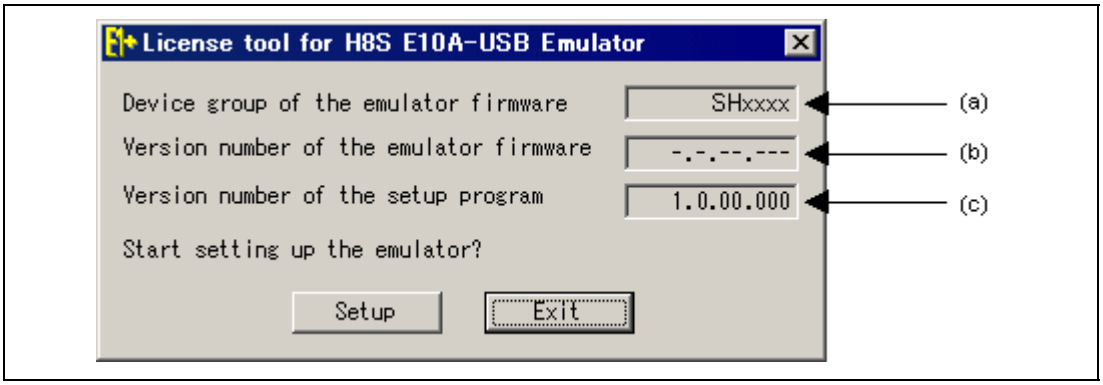


Figure 3.30 License Tool for Emulator

(a) Device group of the emulator firmware: Name of the device group currently set.

(b) Version number of the emulator firmware: The version number of software for controlling the H8S Device Group in the emulator. This item is displayed only when the H8S Device Group is available.

(c) Version number of the setup program: The version number of the setup program.

Notes: 1. If the version numbers shown in (b) and (c) are the same, setup of the emulator is not required. Set up the emulator only when "-.-.--.---" is shown in (b) or the version number of (b) is older than that of (c).

2. If the following error message is displayed, the host computer is not connected to the emulator or the setup switch (SW1) is turned to '0'.

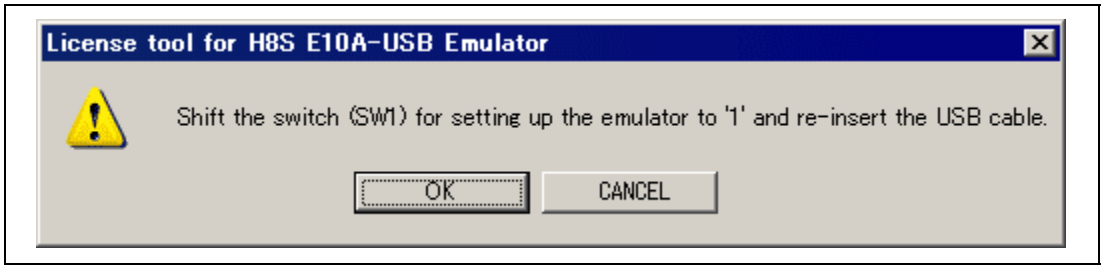


Figure 3.31 Error Message

If the setup switch (SW1) is turned to '0', set it to '1' and connect the USB cable again.

3. Click the [Setup] button.
   When the following dialog box is displayed, turn the setup switch (SW1) to '0', connect the USB cable again, and click the [OK] button.

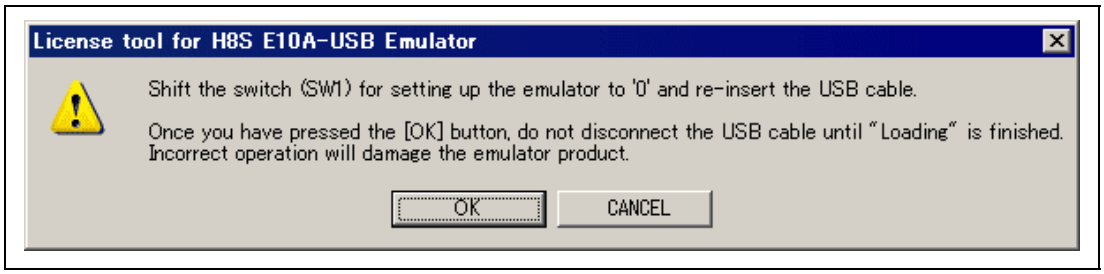


Figure 3.32 [License tool for H8S E10A-USB Emulator] Dialog Box

4. Setting up the emulator's control software is started.

Notes: 1. If the following dialog message is displayed, insert the USB cable again.

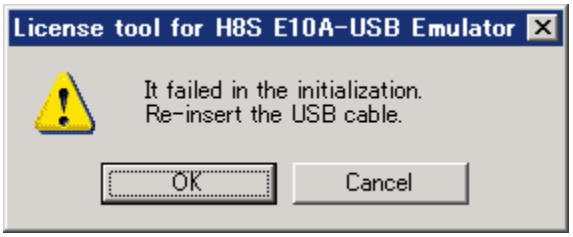


Figure 3.33 [License tool for H8S E10A-USB Emulator] Dialog Box

2. When [Add New Hardware Wizard] is displayed, select [Install the software automatically].
3. Although a dialog box will be displayed to indicate disconnection of the USB, this is not a problem.

CAUTION

Do not turn off the host computer or disconnect the USB cable while setting up the emulator. The emulator may be damaged.

Figure 3.34 Start of Setting up the Emulator

5. When the following dialog box is displayed, setting up the emulator is completed.

Figure 3.35 Message for Completion of Setting up the Emulator

6. When setting up the emulator has been completed and the following message is displayed, turn the setup switch (SW1) to ‘1’, connect the USB cable again, and click the [OK] button.

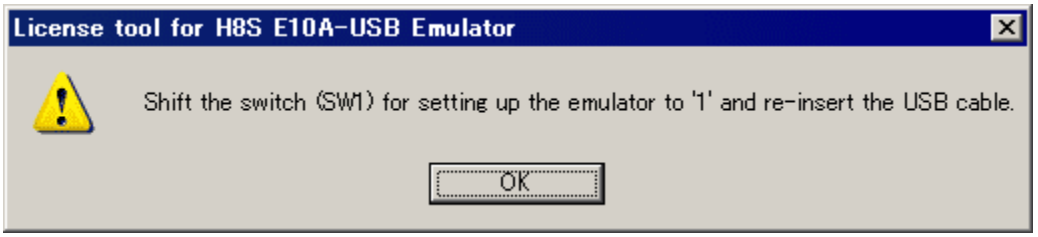


Figure 3.36 [License tool for H8S E10A-USB Emulator] Dialog Box

Notes: 1. Be sure to turn the setup switch (SW1) to ‘1’ except when the license tool for adding device groups is used.

2. To use the license tool for another device group, it is necessary to set up the firmware by using the setup tool or license tool that corresponds to the device group. To use the setup tool, however, the license tool must also be installed. For details on the setup tool, refer to section 3.10, Setting up the Emulator, in the SuperH™ Family E10A-USB Emulator User’s Manual, or H8S, H8SX Family E10A-USB Emulator User’s Manual.
3. After you have added a device group by using the license tool, place the attached device-group sticker onto the back of the emulator box. Otherwise, the emulator may not be considered as a target product when repair is required. The license tool provides a license to use the target device group. Be sure to acquire a license for each of the E10A-USB emulators being used with that device group.

3.11 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.

Refer to section 4, Preparations for Debugging, for the other activating method to create a new project or use an existing workspace.

1. Connect the emulator to the host computer.
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. Select [Renesas] -> [High-performance Embedded Workshop] -> [High-performance Embedded Workshop] from [Programs] in the [Start] menu.

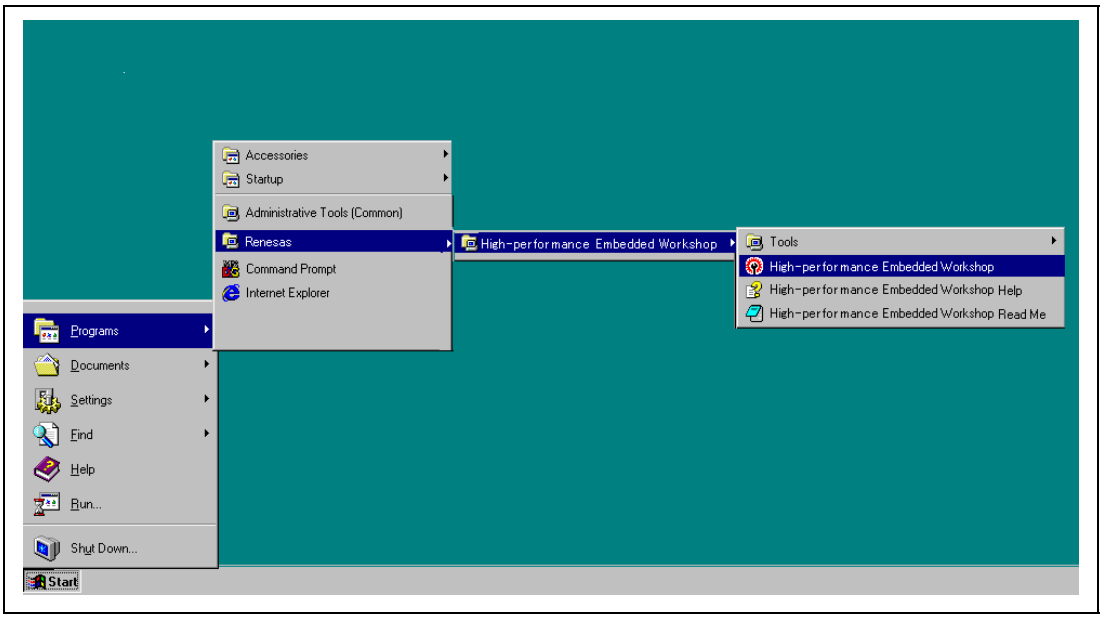


Figure 3.37 [Start] Menu

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

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

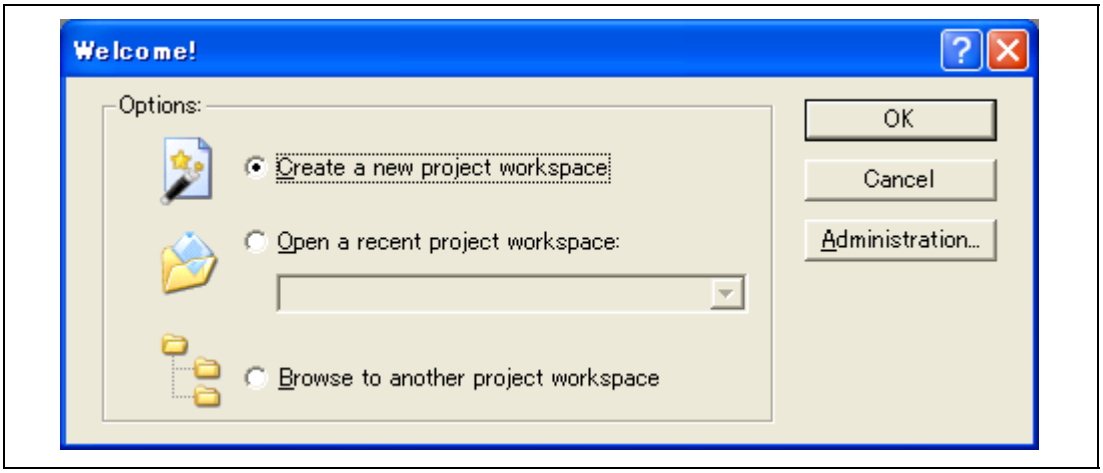


Figure 3.38 [Welcome!] Dialog Box

[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\E10A-USB\xxxx\Tutorial

Here, ‘xxxx’ means the target product group.

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

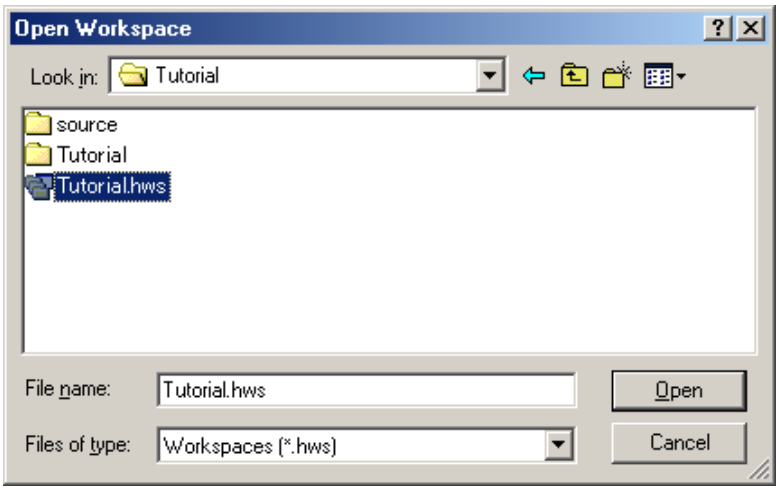


Figure 3.39 [Open Workspace] Dialog Box

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

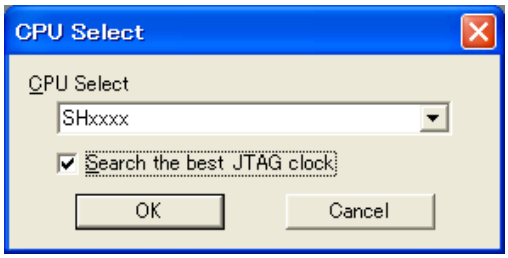


Figure 3.40 [CPU Select] Dialog Box

The [CPU Select] dialog box has the following options.

- [Search the best JTAG clock] check box
  Search the JTAG clock values and start up with the highest available value as the initial value.

Note: Opening this dialog box may not be possible. This depends on the device in use.

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

7. The [Select Emulator mode] dialog box is displayed depending on the MCU/MPU used.

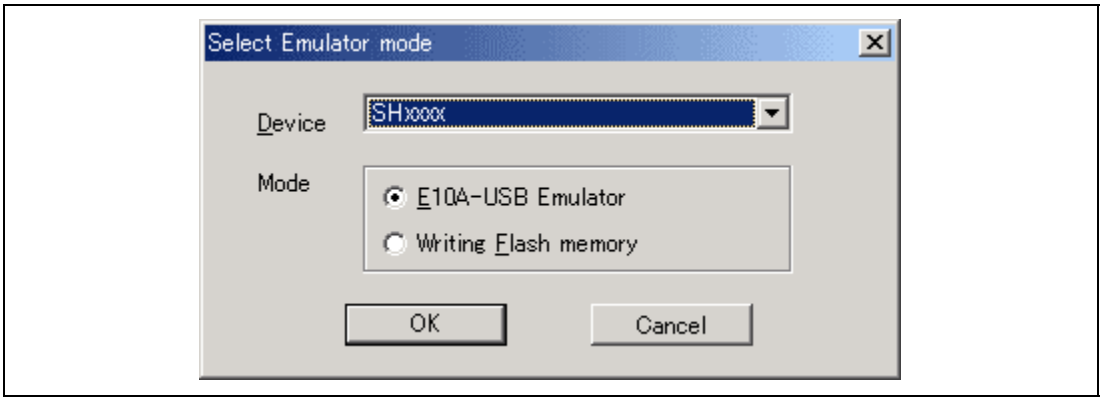


Figure 3.41 [Select Emulator mode] Dialog Box

Select the MCU/MPU name in use from the [Device] drop-down list box. The following items are selectable in the [Mode] group box.

— E10A-USB Emulator

The E10A-USB emulator for the specified MCU/MPU is activated. Debugging the program is enabled.

— Writing Flash memory

The user program is programmed to the internal flash memory. To download a load module, register it in the workspace. Do not attempt anything other than downloading of the load module.

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

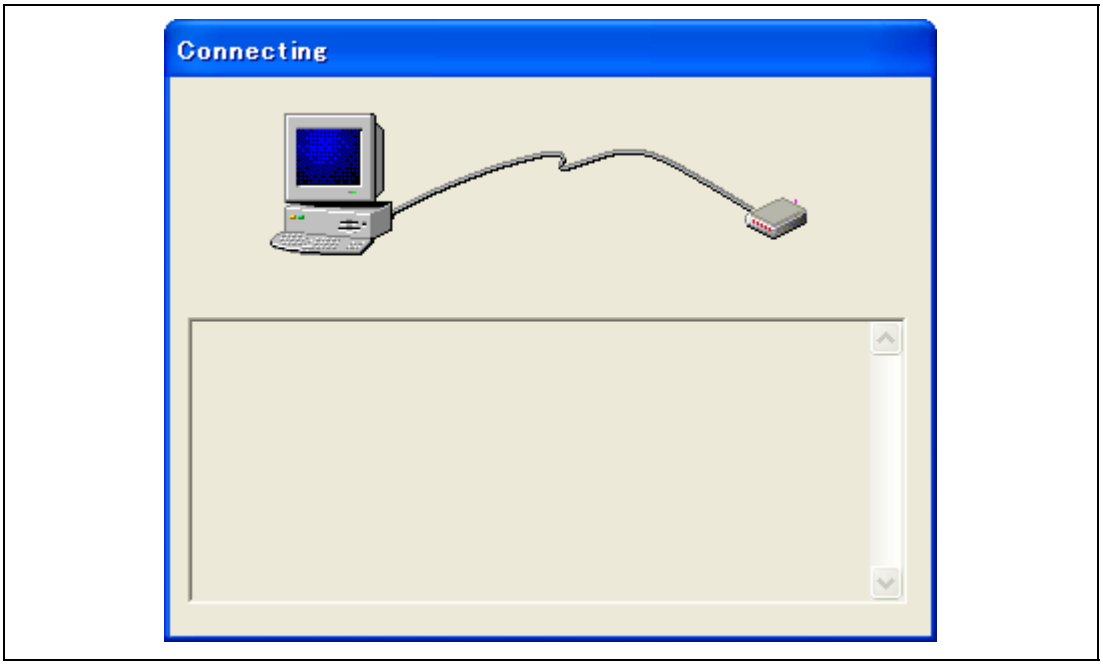


Figure 3.42 [Connecting] Dialog Box

9. The dialog box shown in figure 3.43 is displayed if no product groups have been installed in the emulator at the time of purchase or if the SHxxxx license has been installed in the emulator but the emulator firmware has been set up for a different device group. The dialog box shown in figure 3.44 is displayed if an old version of the emulator firmware has been set up in the emulator. Clicking the [OK] button sets up the emulator firmware.

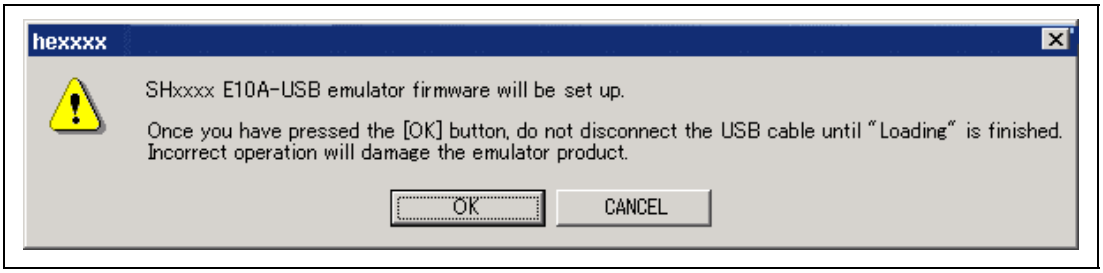


Figure 3.43 Dialog Box to Confirm Setting up of the Emulator Firmware

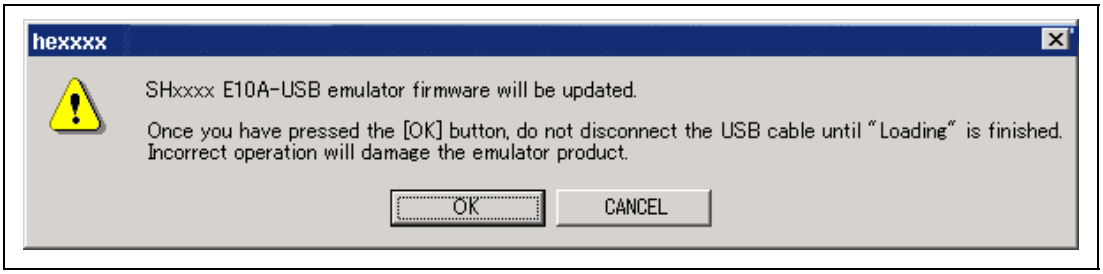


Figure 3.44 Dialog Box to Confirm Updating of the SHxxxx Emulator Firmware

CAUTION

The USB cable must not be disconnected until writing is complete. Early disconnection may damage the emulator.

Note: The above dialog boxes are only displayed if you are using the HS0005KCU01H (serial No.: 03311C or later) or HS0005KCU02H (serial No.: 04146E or later) emulator hardware. In this case, follow the procedure described in section 3.10.1, Setting up at Purchasing the Emulator or Updating the Version of Software.

10. The dialog box shown in figure 3.45 is displayed.

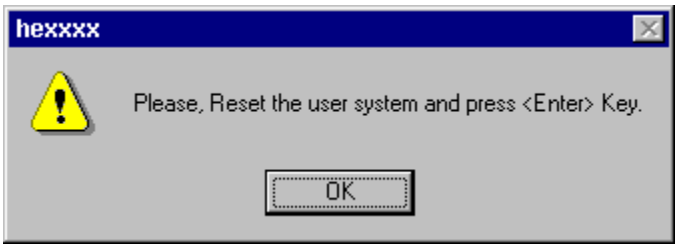


Figure 3.45 Dialog Box of the RESET Signal Input Request Message

11. Power on the user system.
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.

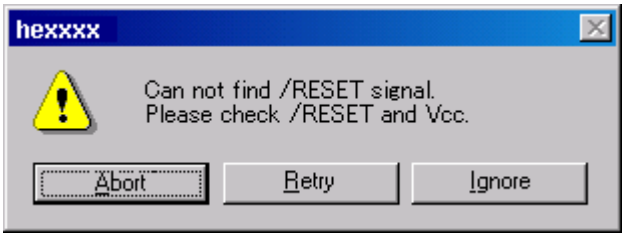


Figure 3.46 [Can not find /RESET signal] Dialog Box

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 using the MCU with flash memory, the [Clock] dialog box shown in figure 3.47 is opened.
For the [Clock] dialog box, set the frequency of the crystal oscillator which has been connected to or the external clock which has been input to the target microcomputer (MCU).

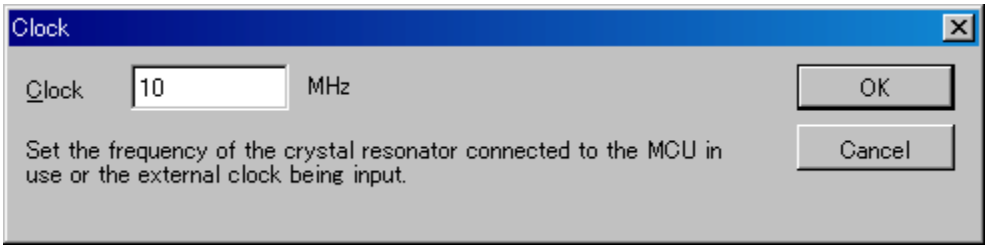


Figure 3.47 [Clock] Dialog Box

15. After the following dialog box is displayed, input the ID code as a security code for the flash memory. However, H'FFFFFFFF is disabled as the ID code.
    Input this ID code when [E10A-USB Emulator] is selected and the [New ID code] check box is unselected on activating the emulator. If the ID code is not matched or the [New ID code] check box is selected, the flash memory contents are erased.

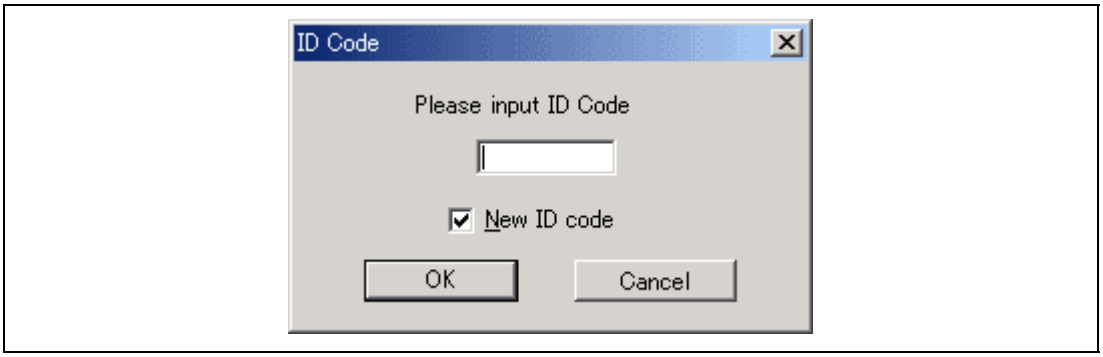


Figure 3.48 [ID Code] Dialog Box

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

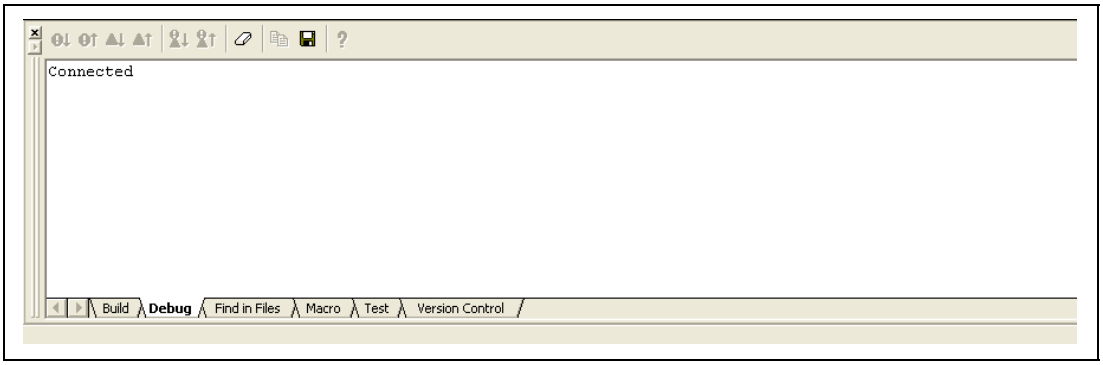


Figure 3.49 [Output] Window

Note: When the user program has already been downloaded to the flash memory, source-level debugging cannot be executed because there is no debugging information on the user program after the emulator has been activated. Be sure to load the debugging information file. For details, refer to section 4.2.1, Setting at Emulator Activation.

Notes: 1. If the emulator is not initiated, the following dialog boxes shown in figures 3.50 through 3.56 will be displayed.

(a) If the following dialog box is displayed and the method 11 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.

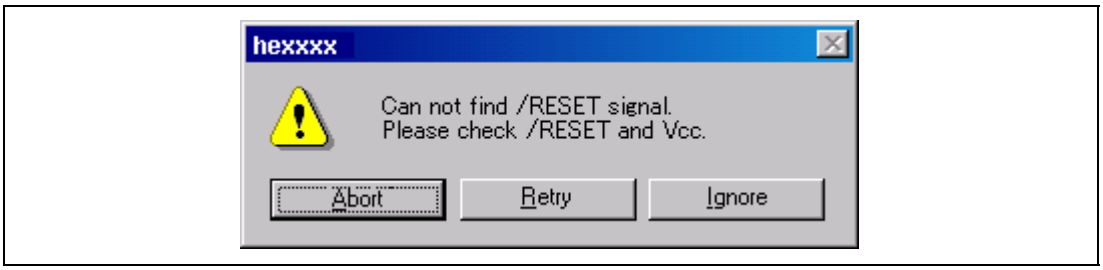


Figure 3.50 [Can not find /RESET signal] Dialog Box

(b) If the following dialog box is displayed, check that the H-UDI port connector on the user system is correctly connected.

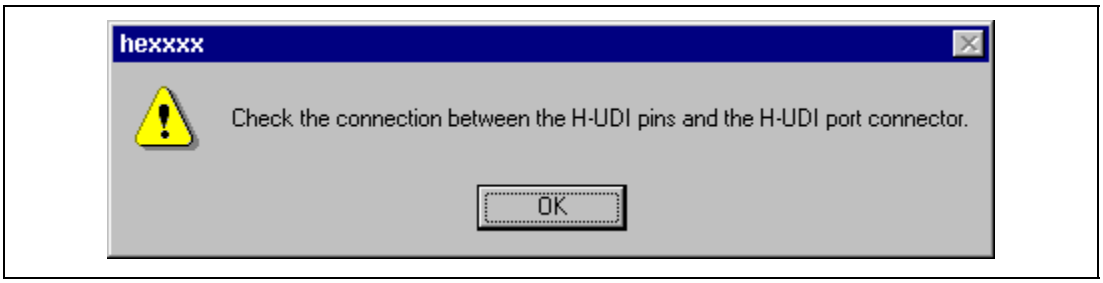


Figure 3.51 [Check the connection] Dialog Box

(c) If the following dialog box is displayed, the emulator's firmware may not be set up correctly. Set up the firmware of the device group that is used for the setup tool or license tool.

Figure 3.52 [The product currently connected] Dialog Box

(d) If the following dialog box is displayed, the version of the firmware in the emulator may be old. Set up the firmware by using the setup tool.

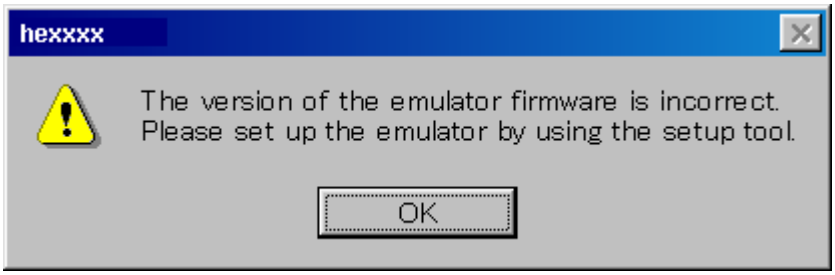


Figure 3.53 [The version of the emulator firmware is incorrect] Dialog Box

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

Figure 3.54 [COMMUNICATION TIMEOUT ERROR] Dialog Box

Figure 3.55 [INVALID ASERAM FIRMWARE!] Dialog Box

Figure 3.56 [Error JTAG boot] Dialog Box

(f) The following dialog box is displayed when the flash memory cannot be erased. Change the MCU since the flash memory has been reprogrammed more times than the limitation.

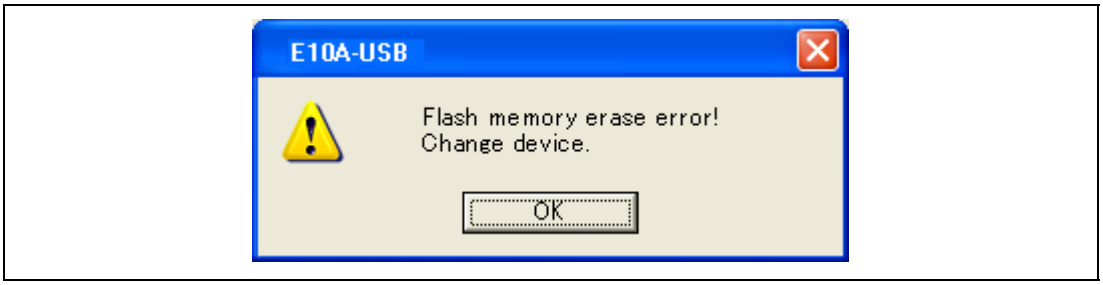


Figure 3.57 [Flash memory erase error!] Dialog Box

Note: If a mode is illegally set, the error message shown in figure 3.57 will be displayed.

(g) The following dialog box is displayed when the flash memory cannot be reprogrammed. An incorrect system clock value has been input to the [Clock] dialog box or the flash memory has been reprogrammed more times than the limitation.

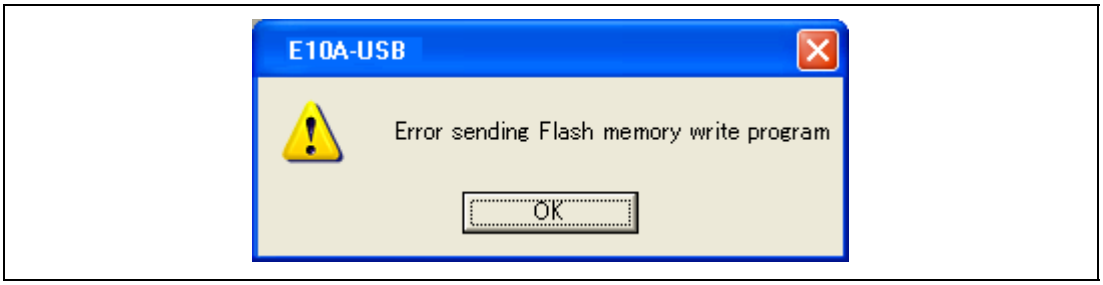


Figure 3.58 [Error sending Flash memory write program] Dialog Box

(h) The following dialog box is displayed when an incorrect ID code has been input. For security, the flash memory is completely erased.

Figure 3.59 [ID code error!] Dialog Box

(i) The following dialog box is displayed when the MCU cannot communicate with the emulator. The MCU may not operate correctly; check the MCU settings.

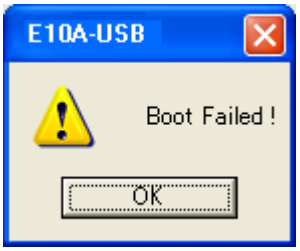


Figure 3.60 [Boot Failed!] Dialog Box

2. If an incorrect driver has been selected, the following dialog box will appear.

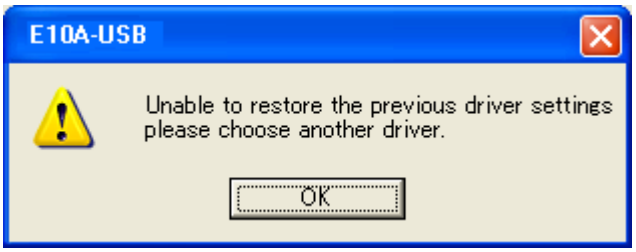


Figure 3.61 [Unable to restore the previous driver settings] Dialog Box

3. 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.

Section 4 Preparations for Debugging

4.1 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 computer 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.

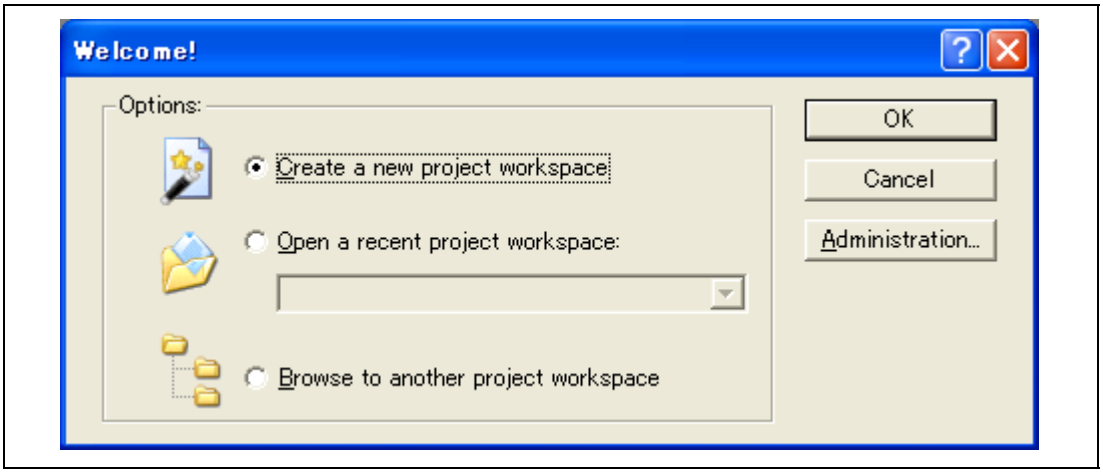


Figure 4.1 [Welcome!] Dialog Box

[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 operation of [Open a recent project workspace] radio button is same as the operation without specifying the workspace file when [Browse to another project workspace] is selected.

4.1.1 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.

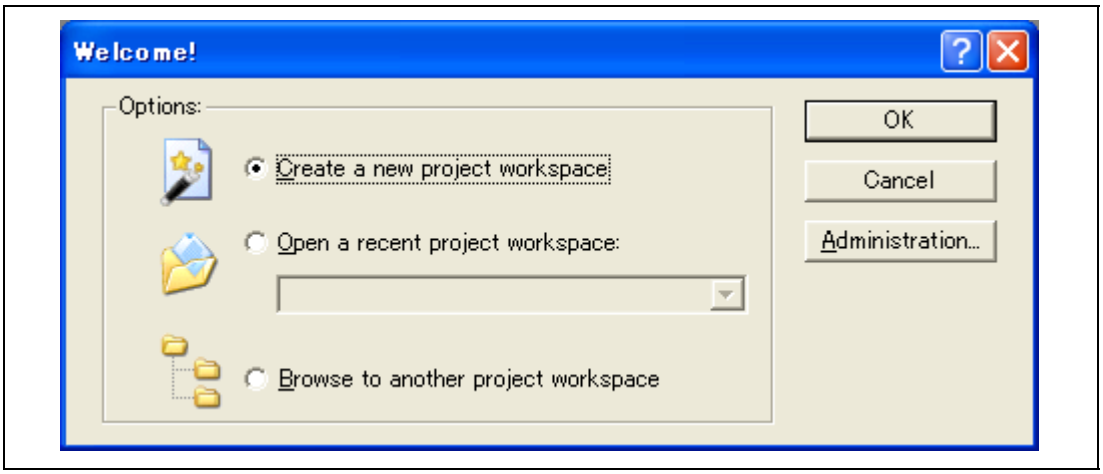


Figure 4.2 [Welcome!] Dialog Box

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.

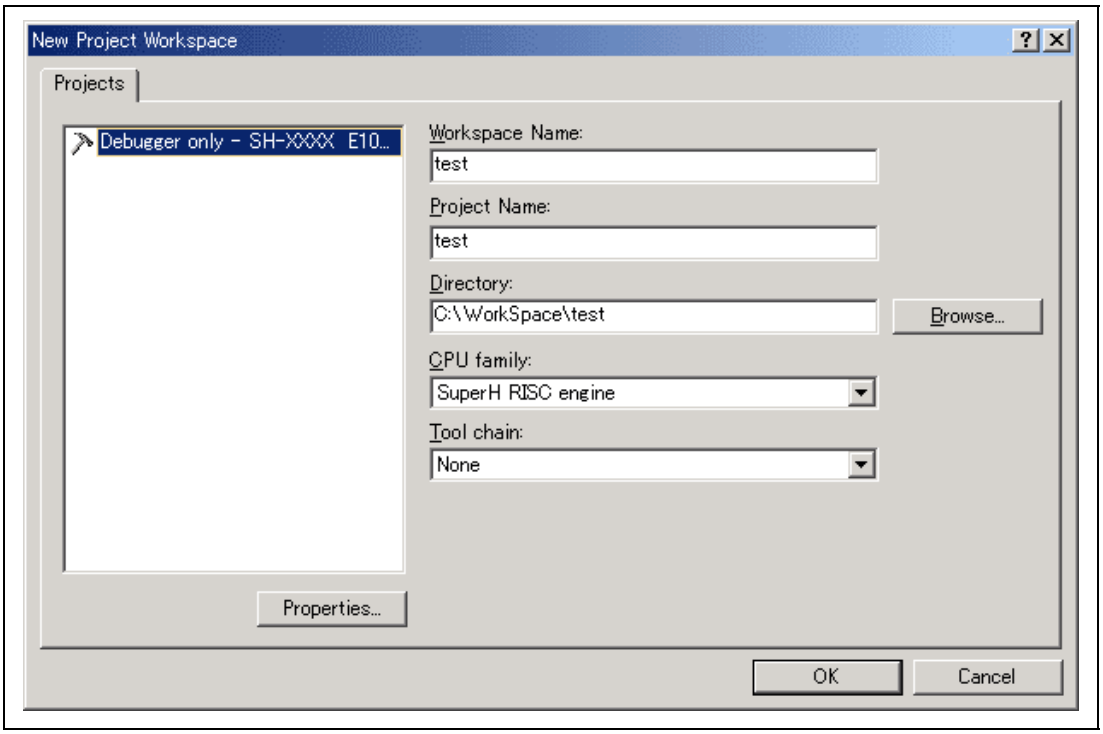


Figure 4.3 [New Project Workspace] Dialog Box

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

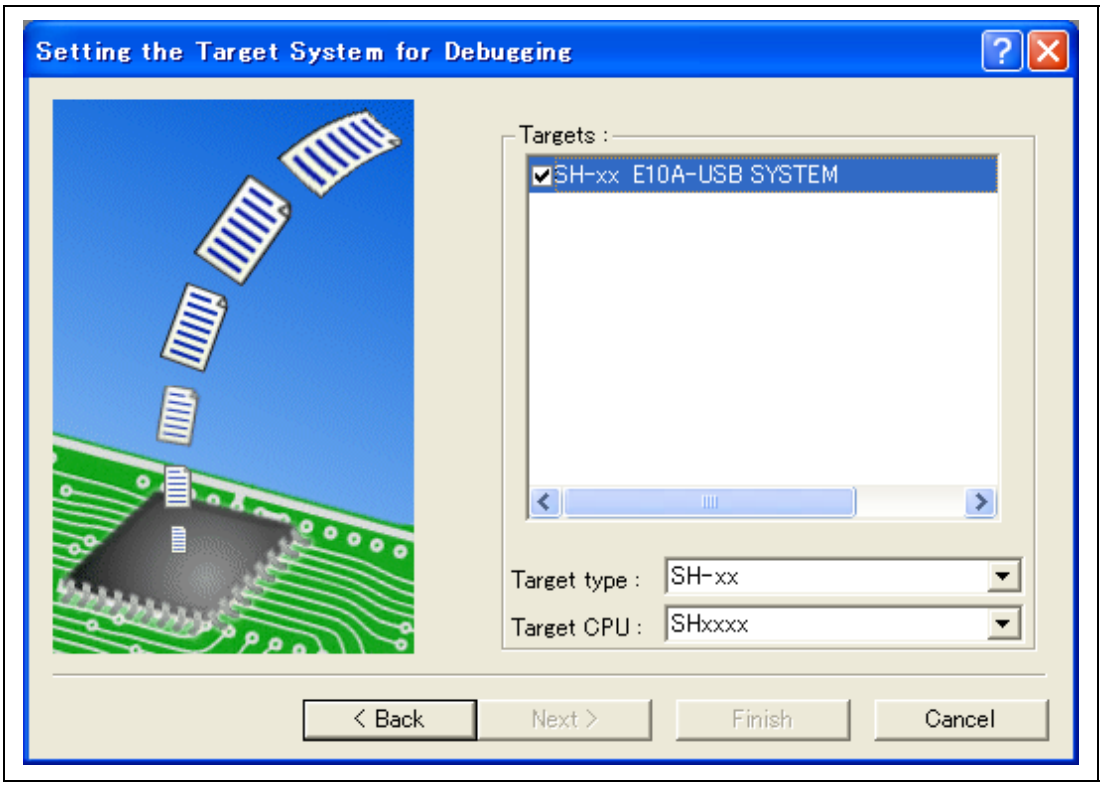


Figure 4.4 [Setting the Target System for Debugging] Dialog Box

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

For details on the items that can be selected in this dialog box, refer to the file titled “MPUs and MCUs Supported by the E10A-USB Emulator” (esupportdevice.pdf), which is included on the CD of the E10A-USB emulator software.

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

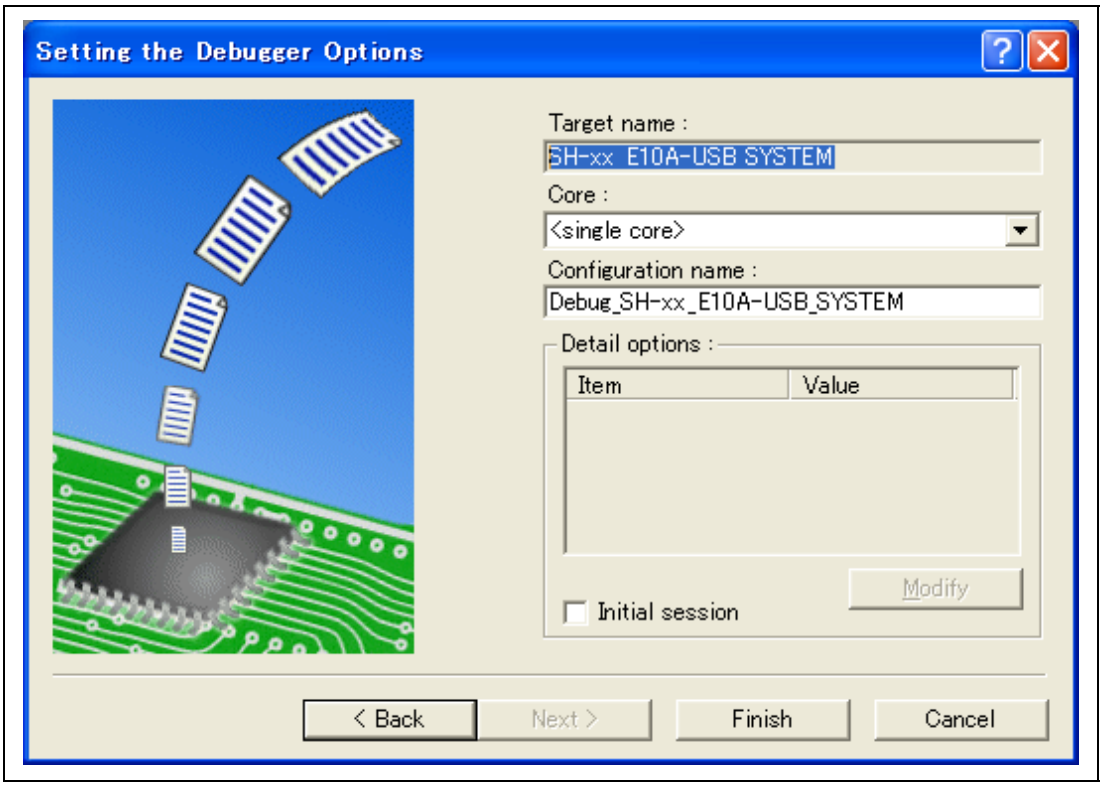


Figure 4.5 [Setting the Debugger Options] Dialog Box

This is the end of the emulator setting.

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

After the High-performance Embedded Workshop has been activated, the emulator is automatically connected. For operation during connection, refer to section 3.11, System Check.

4.1.2 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.

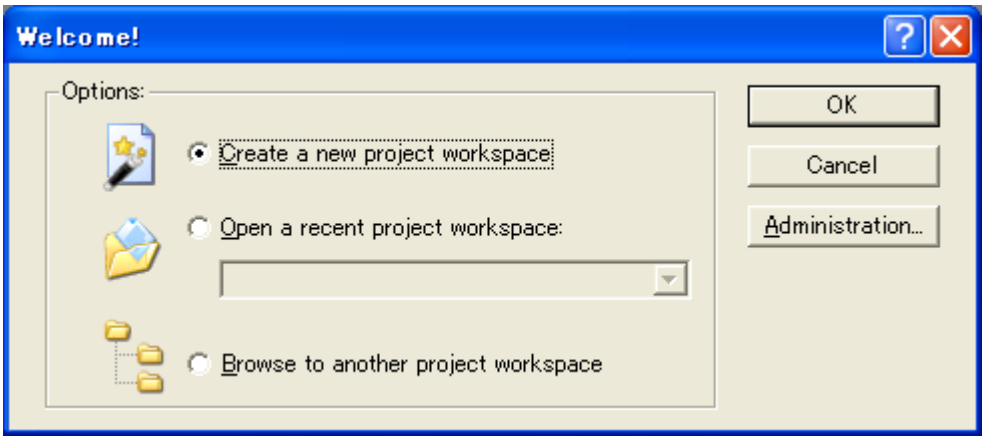


Figure 4.6 [Welcome!] Dialog Box

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

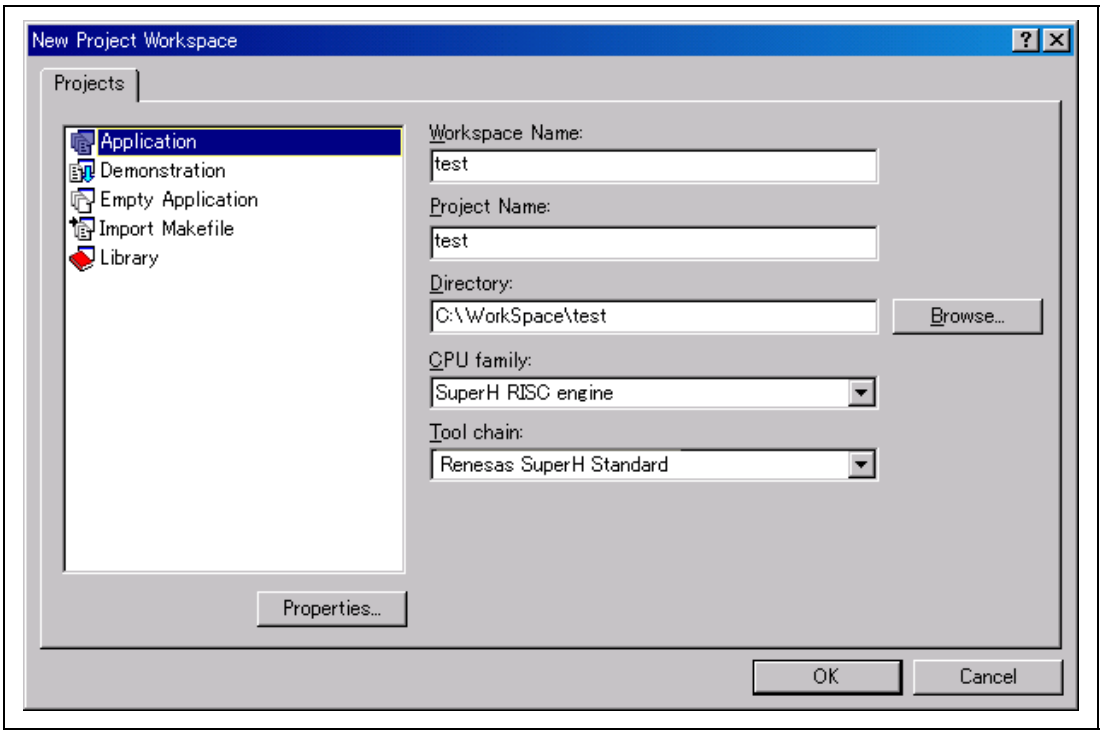


Figure 4.7 [New Project Workspace] Dialog Box

[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 the generated program is used by the emulator, delete the Printf statement.

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

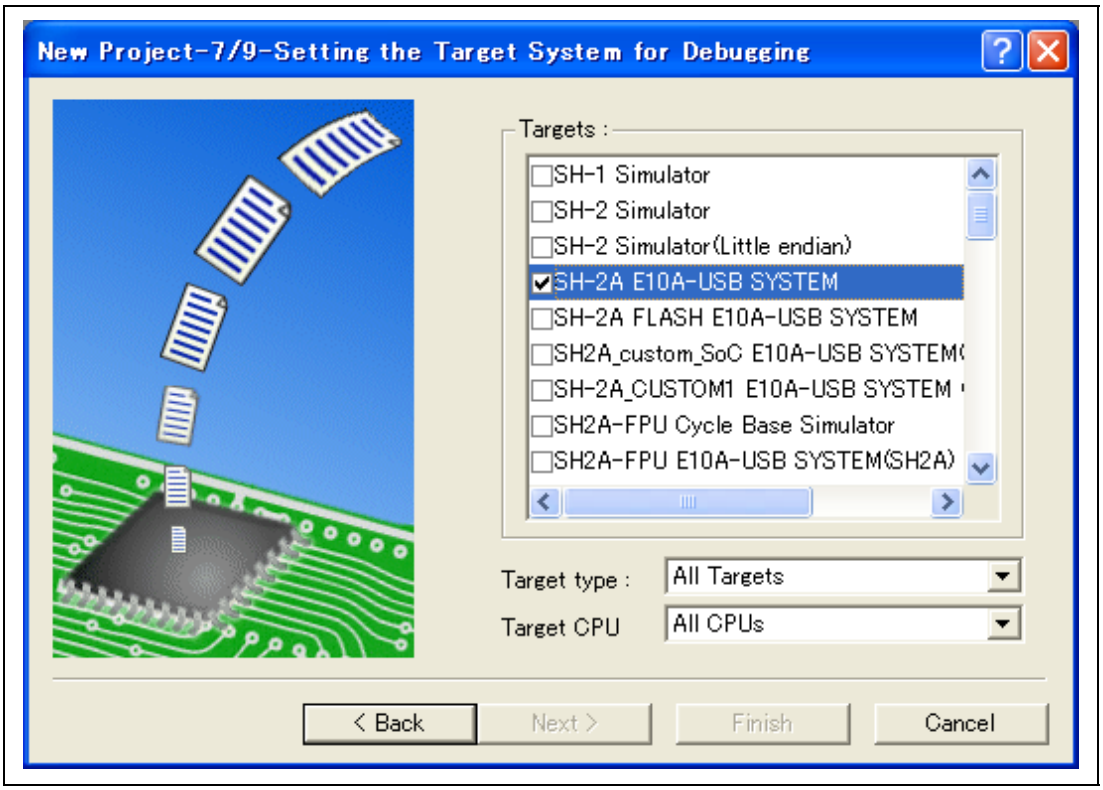


Figure 4.8 [New Project –7/9– Setting the Target System for Debugging] Dialog Box

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

For details on the items that can be selected in this dialog box, refer to the file titled “MPUs and MCUs Supported by the E10A-USB Emulator” (esupportdevice.pdf), which is included on the CD of the E10A-USB emulator software.

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.

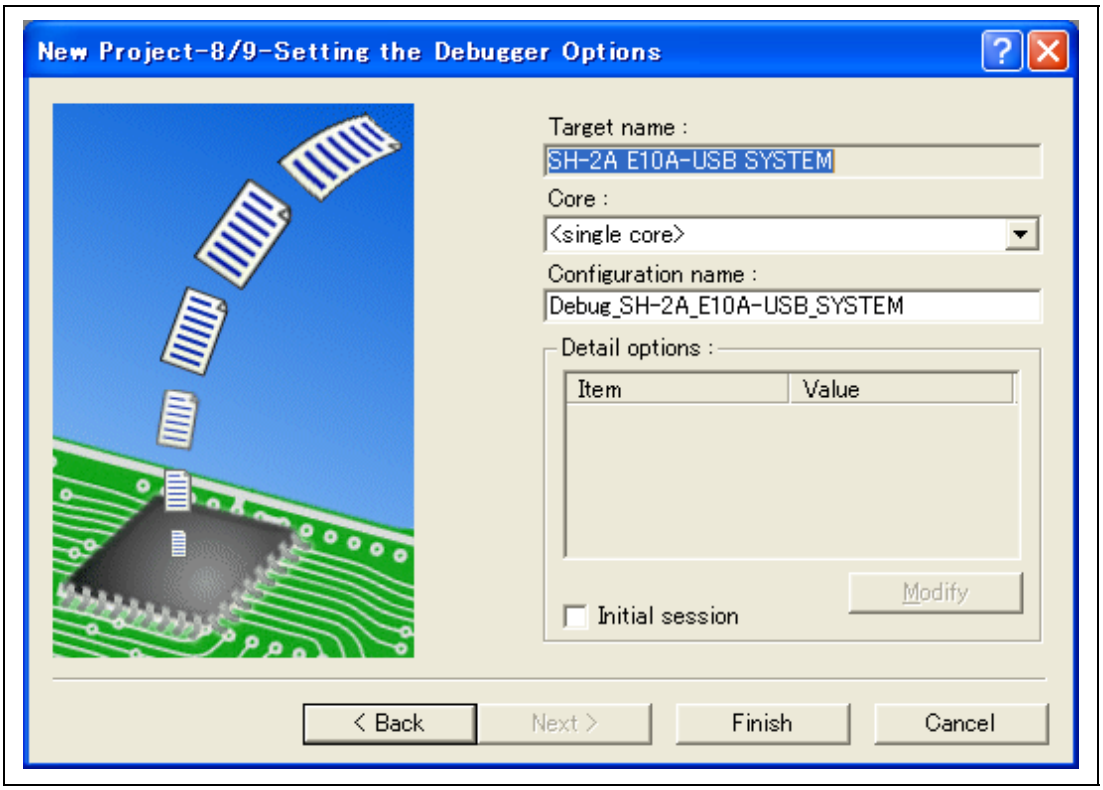


Figure 4.9 [New Project –8/9– Setting the Debugger Options] Dialog Box

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 3.11, System Check.

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

   Select [Debug settings] from the [Debug] 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.

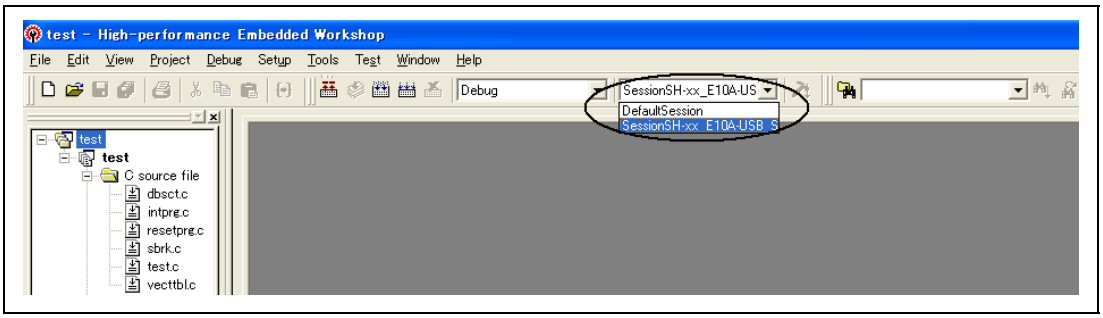


Figure 4.10 Selecting the Session File

In the list box that is circled in figure 4.10, select the session file name including the character string that has been set in the [Target name] text box in figure 4.9, [New Project –8/9– Setting the Debugger Options] dialog box. The setting for using the emulator has been registered in this session file.

After selected, the emulator is automatically connected.

4.1.3 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.

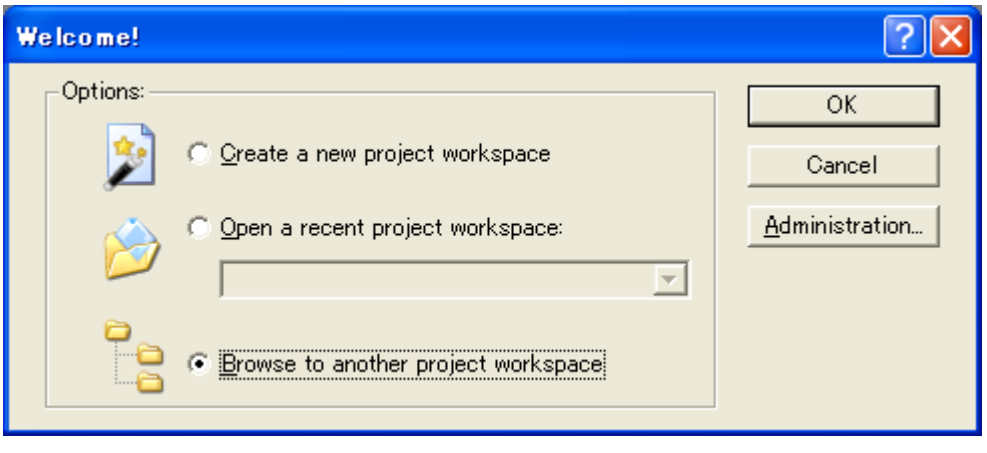


Figure 4.11 [Welcome!] Dialog Box

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.

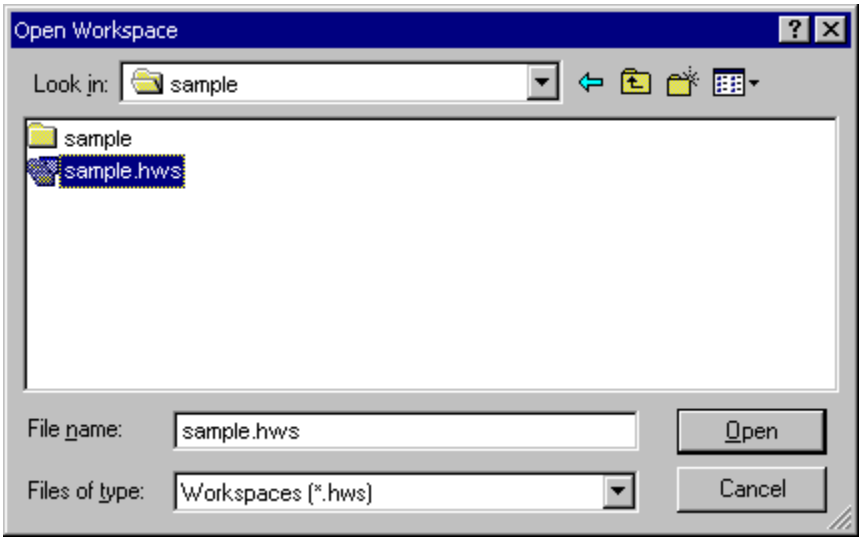


Figure 4.12 [Open Workspace] Dialog Box

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 4.4, Connecting the Emulator.

4.2 Setting at Emulator Activation

4.2.1 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 [Debug] menu to open the [Debug Settings] dialog box.

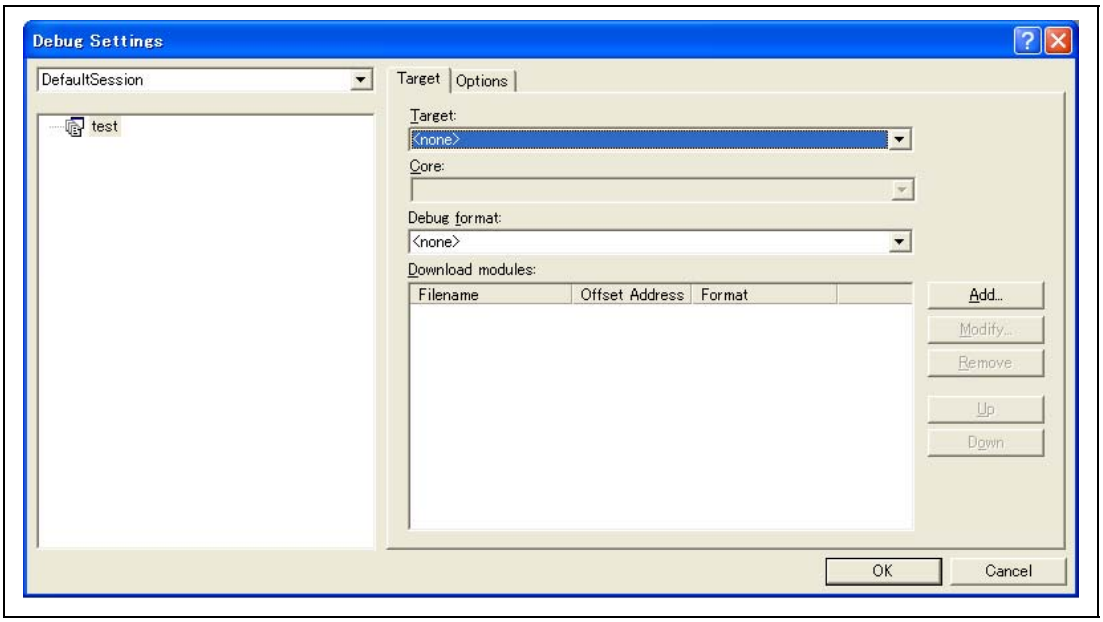


Figure 4.13 [Debug Settings] Dialog Box ([Target] Page)

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.

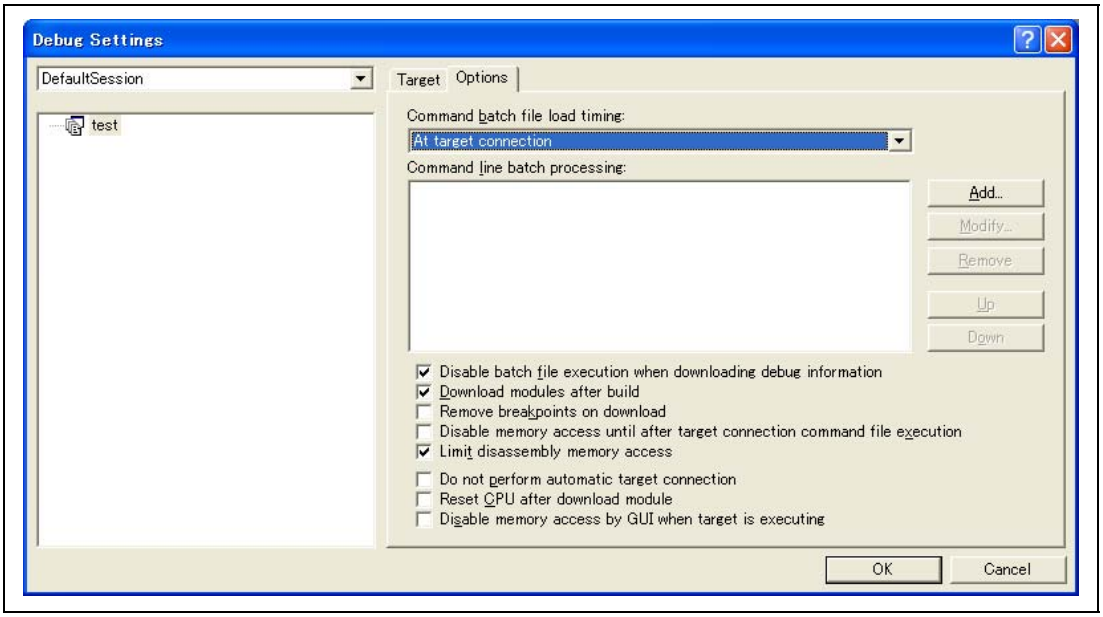


Figure 4.14 [Debug Settings] Dialog Box ([Options] Page)

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

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

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.

4.2.2 Downloading a Program

A download module is added under [Download modules] in the [Workspace] window.

Open the load module of [Download modules] in the [Workspace] window by clicking the right-hand mouse button and select [Download module] to start downloading the module.

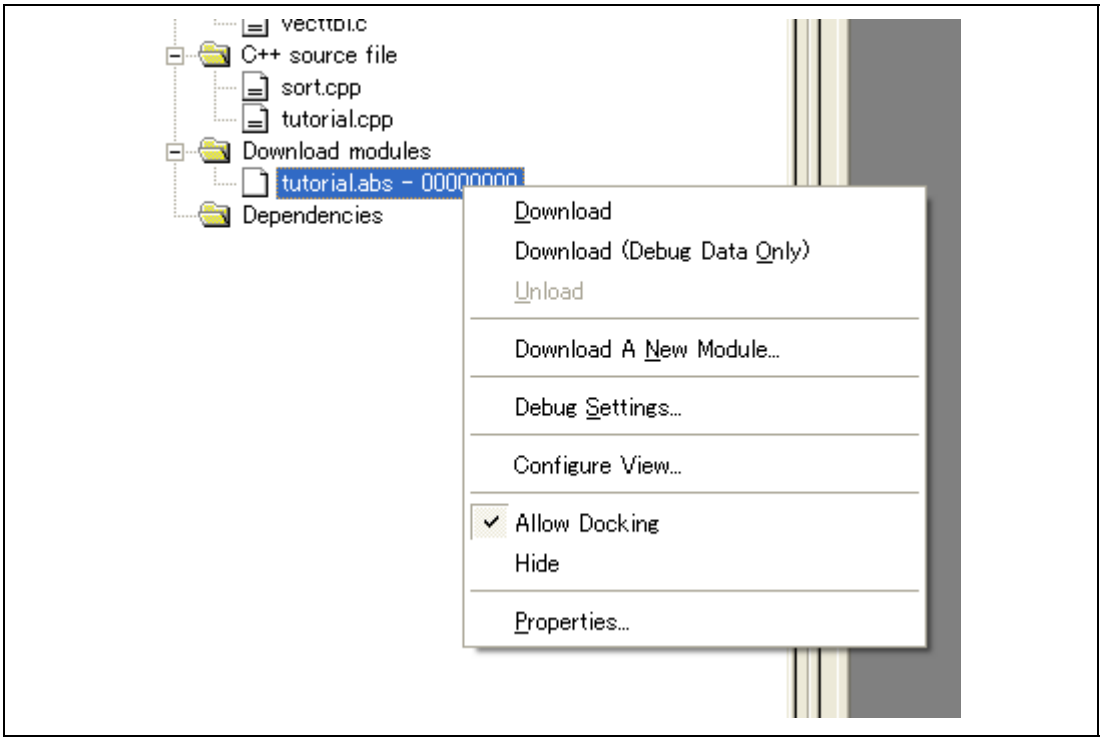


Figure 4.15 Download Menu of the [Workspace] Window ([Projects])

Notes: 1. When load modules are downloaded, select [Debug] -> [Download] -> [All DownLoad Modules].

2. The emulator downloads programs to the flash memory just before execution of the user program.

4.2.3 Setting the Writing Flash Memory Mode

The following describes the procedures when the emulator is used as a tool for programming of the internal ROM. The load module to be downloaded to the new workspace is registered and programmed.

Note: 1. When the [Writing Flash memory] mode is selected, command chains registered on the [Options] page of the [Debug Settings] dialog box must be removed. The [Reset CPU after download module] checkbox must also be deselected.

2. The emulator is also usable as a programming tool in mass production when it is operating in the [Writing Flash Memory] mode.
However, the flash memory of an actual MCU which has been used in connection with the E10A-USB Emulator for debugging will have been programmed at emulation and subjected to stress accordingly. Do not use an MCU that has been used for debugging in a mass-produced product.

(a) Select the new project workspace.

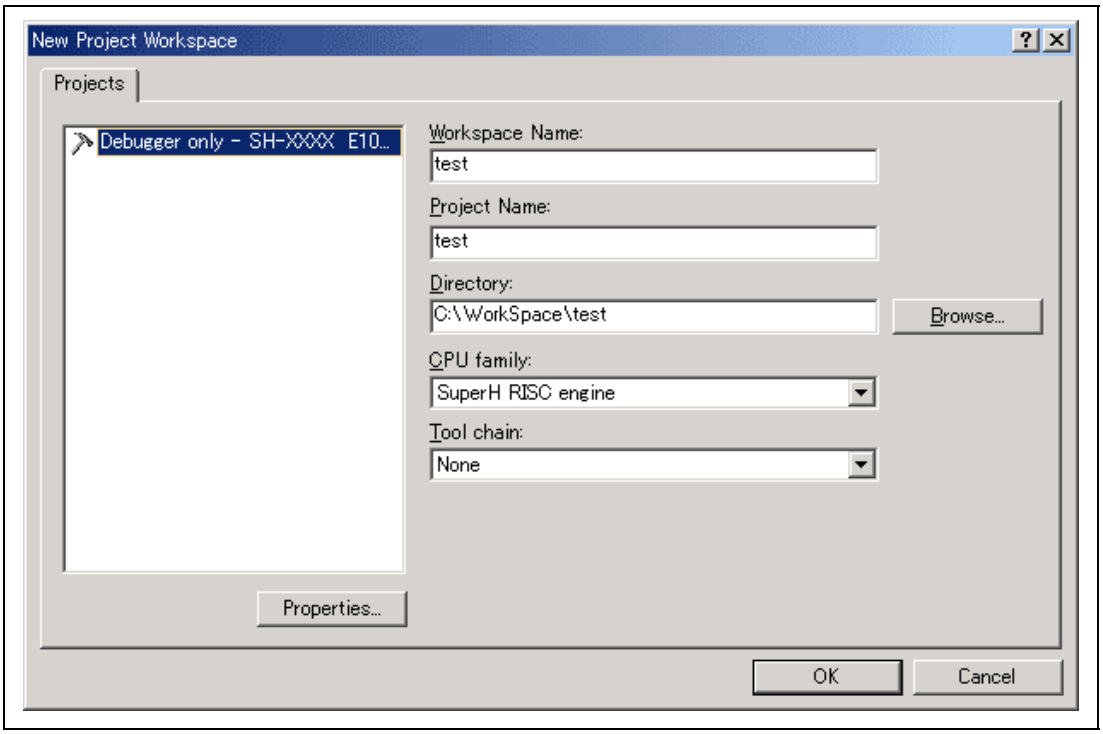


Figure 4.16 [New Project Workspace] Dialog Box

(b) Select the target MCU and click the [Next] button.

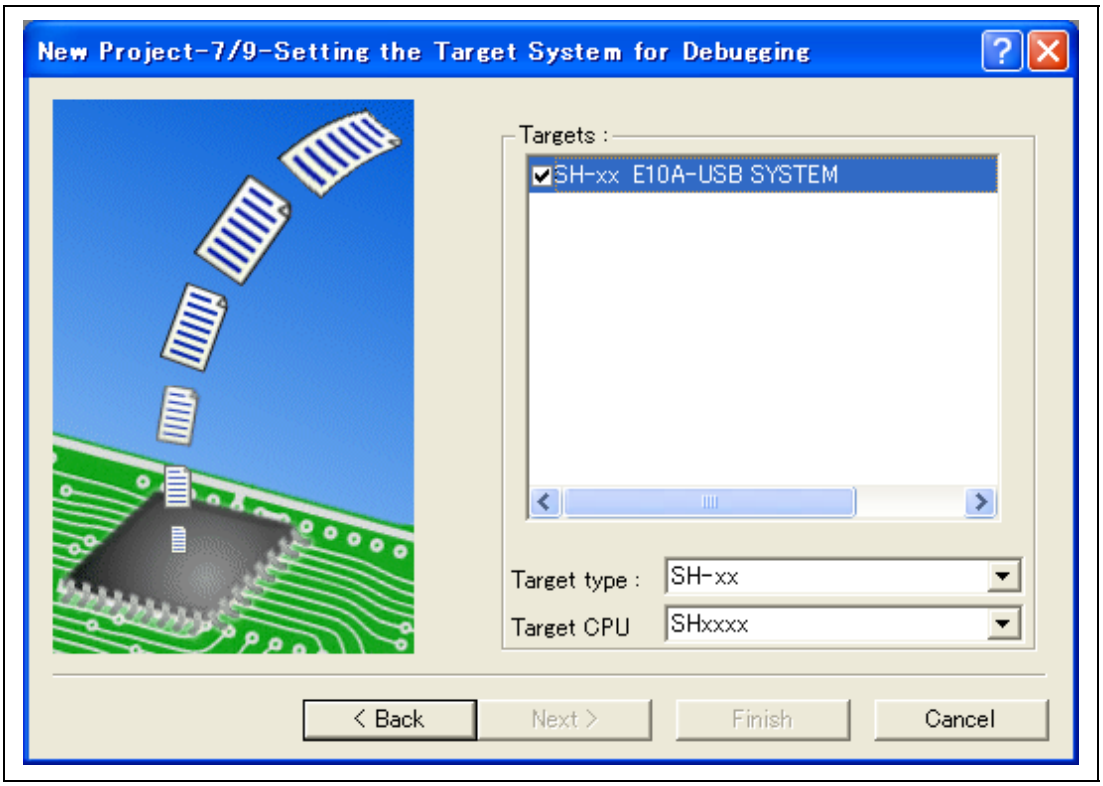


Figure 4.17 [New Project –7/9– Setting the Target System for Debugging] Dialog Box

(c) The [Select Emulator mode] dialog box is displayed.

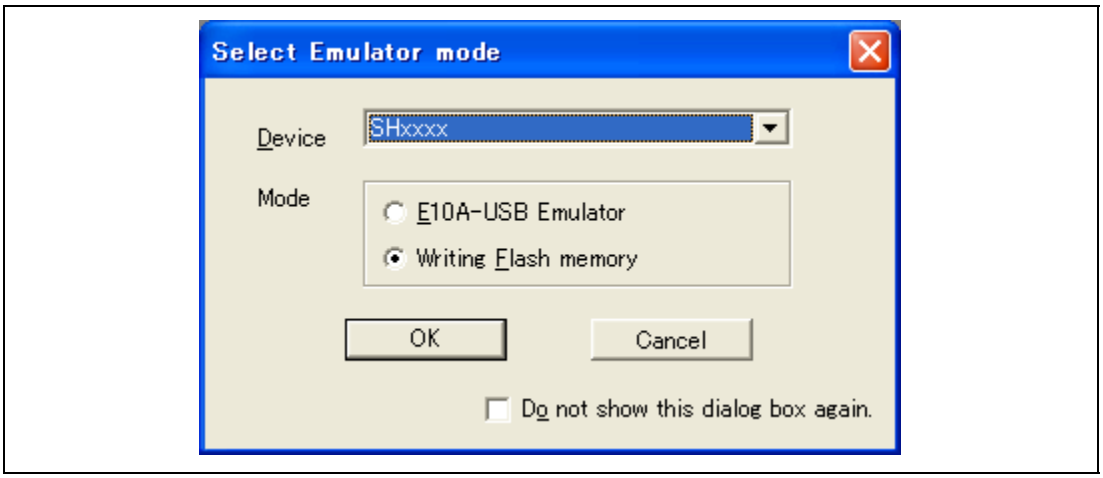


Figure 4.18 [Select Emulator mode] Dialog Box

Select the [Writing Flash memory] mode.

(d) Turn on the target board, input the reset signal from the user system, and press the [OK] button.

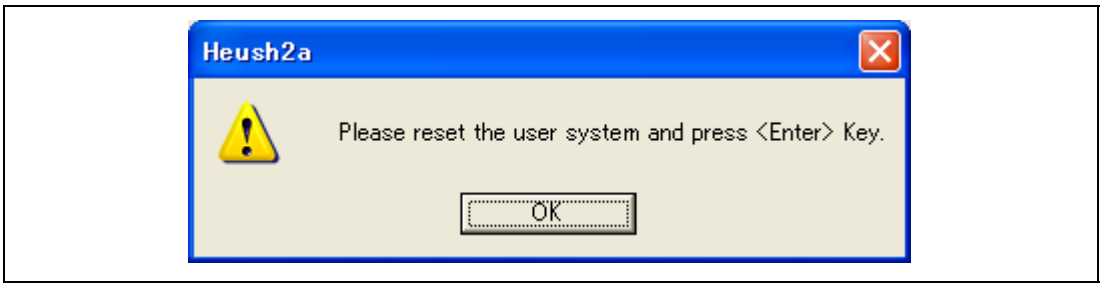


Figure 4.19 Dialog Box of the RESET Signal Input Request Message

(e) For the [Clock] dialog box, set the frequency of the crystal oscillator which has been connected to or the external clock which has been input to the target microcomputer (MCU).

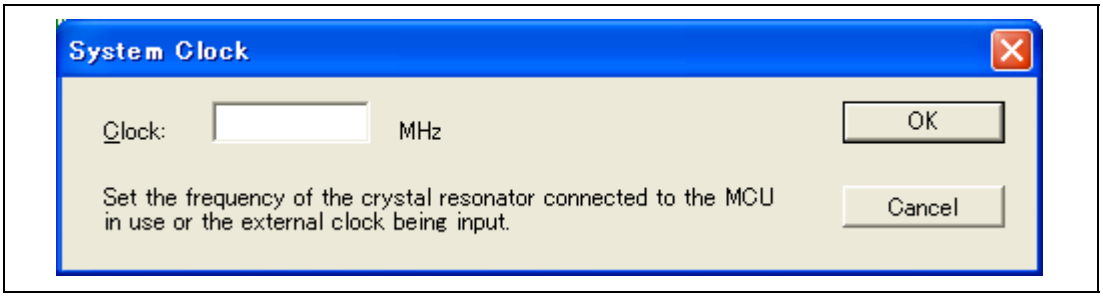


Figure 4.20 [Clock] Dialog Box

(f) Select [Debug Setting] from the [Debug] menu.

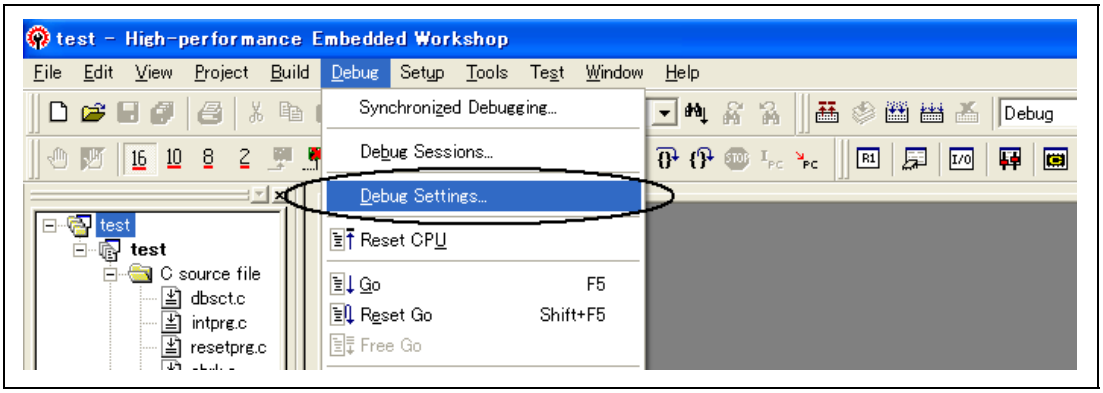


Figure 4.21 High-performance Embedded Workshop Window

(g) Select the target MCU and then the download module with the [Add…] button.

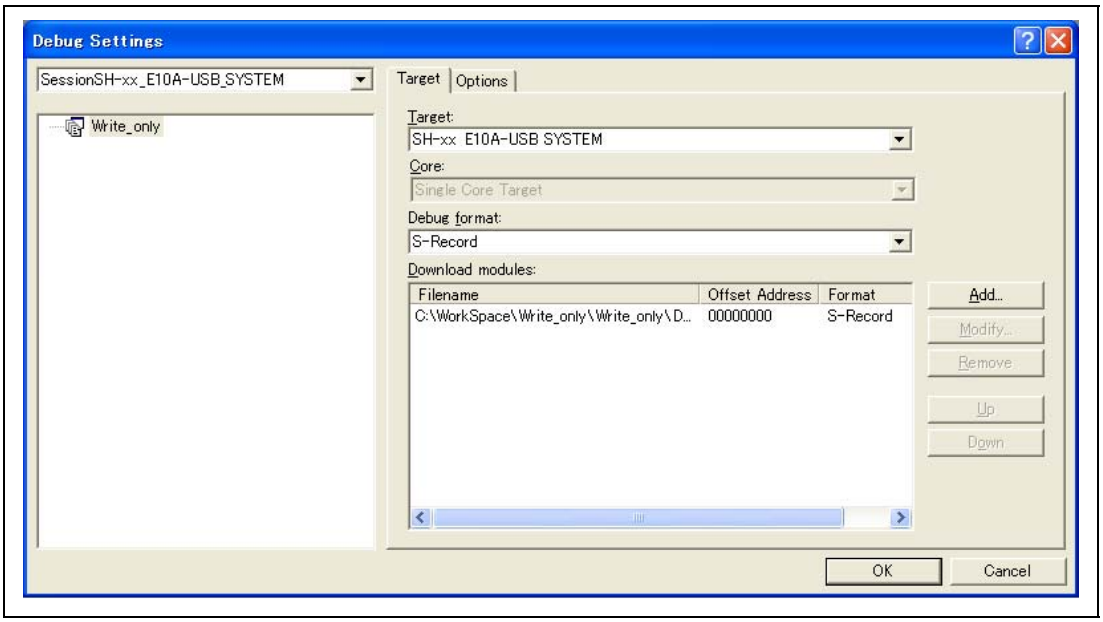


Figure 4.22 [Debug Setting] Dialog Box ([Target] Page)

(h) The download file is displayed on [Project Files].

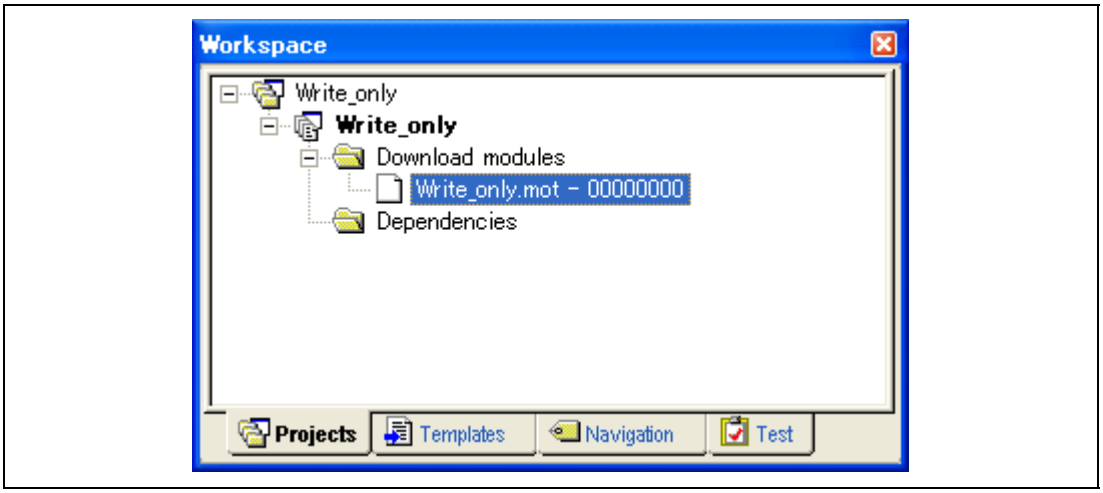


Figure 4.23 [Workspace] Window ([Project Files])

(i) Select and download the file with the right-hand mouse button.

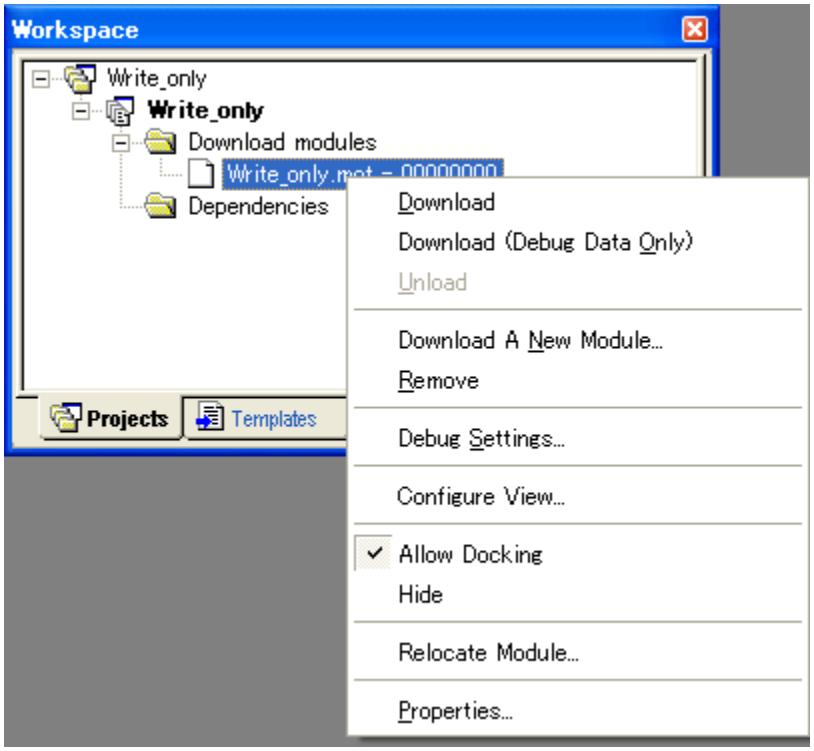


Figure 4.24 Download Menu of the [Workspace] Window ([Project Files])

(j) The dialog box for sum checking is displayed and programming is completed. Sum data is a value that data in the internal ROM area has been added by byte. If no user program exists, the value is calculated by H'FF.

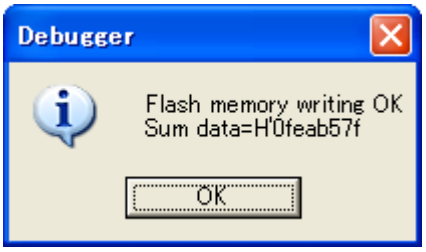


Figure 4.25 Message for Completion of Flash Memory Programming

(k) When the following dialog box is displayed, click on the [OK] button. The debugging platform is automatically disconnected.

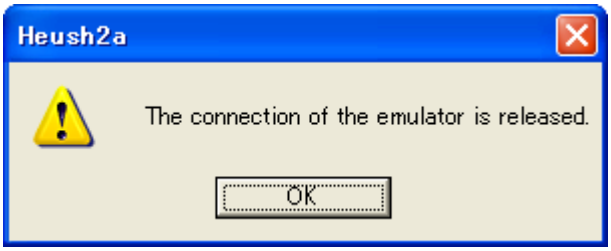


Figure 4.26 Message for Exiting Writing Flash Memory Mode and Automatic Disconnection

4.3 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.

4.3.1 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 4.27) in the toolbar.

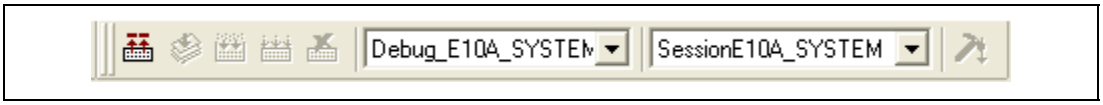


Figure 4.27 Toolbar Selection

- From the dialog box
  1. Select [Debug -> Debug Sessions…]. This will open the [Debug Sessions] dialog box (figure 4.28).

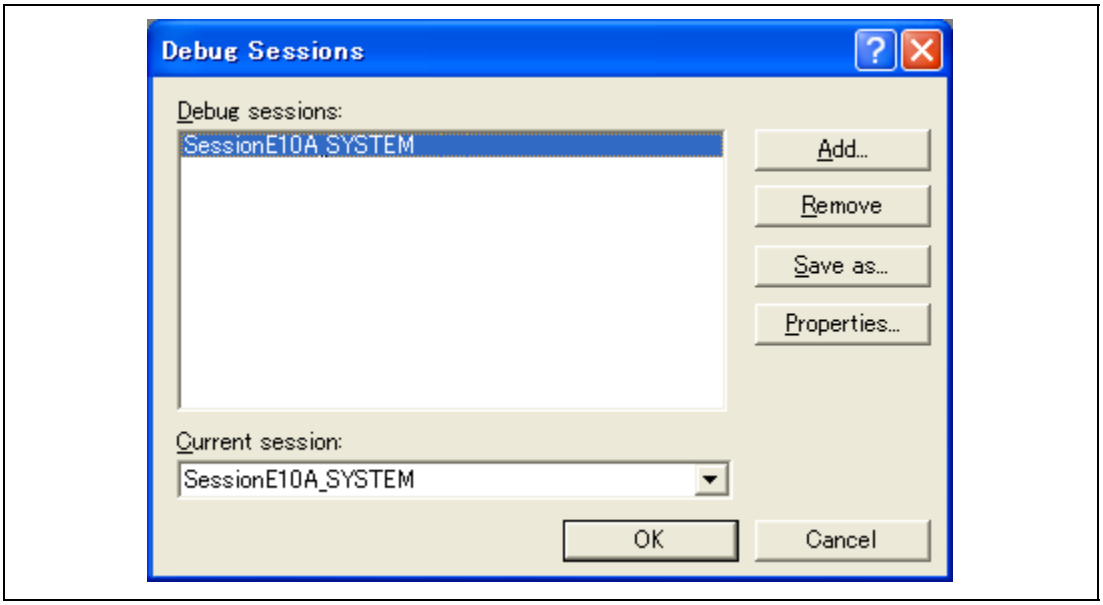


Figure 4.28 [Debug Sessions] Dialog Box

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

4.3.2 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 [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  2. Click the [Add…] button to display the [Add new session] dialog box (figure 4.29).
  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.

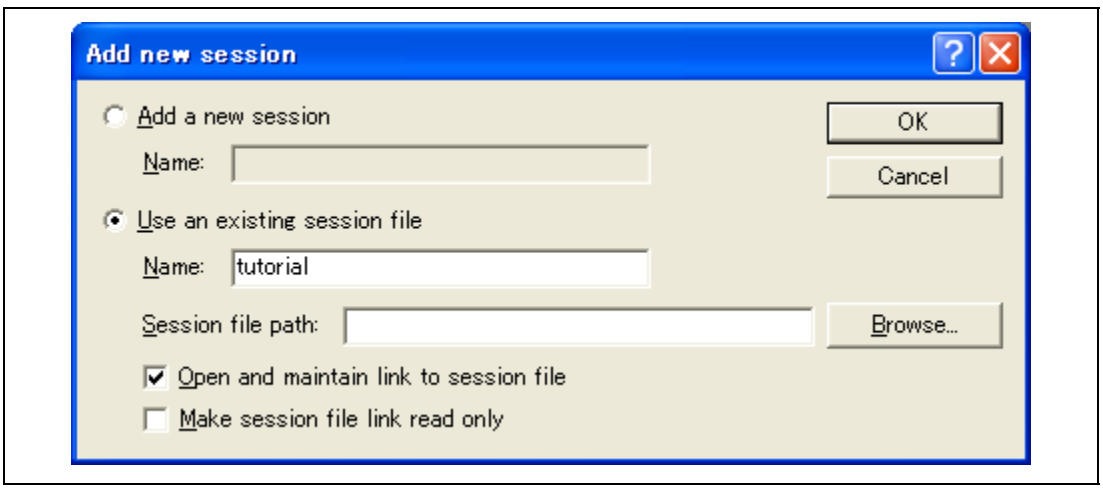


Figure 4.29 [Add new session] Dialog Box

- To import an existing session into a new session file
  1. Select [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  2. Click the [Add…] button to display the [Add new session] dialog box (figure 4.29).
  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 existing 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 [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  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 [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  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 4.30).

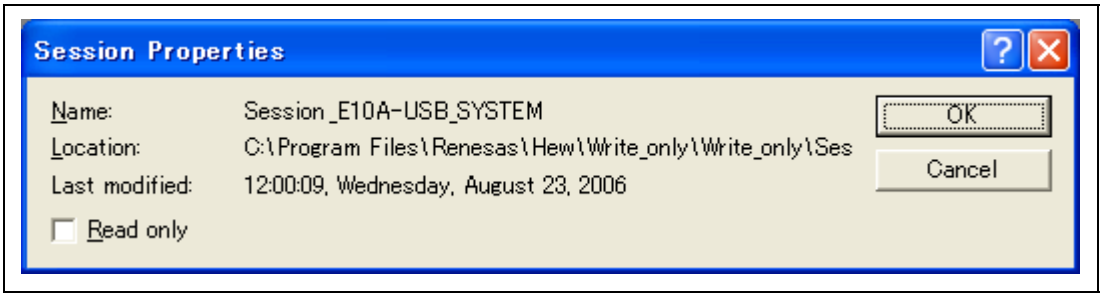


Figure 4.30 [Session Properties] Dialog Box

- To make a session read-only
  1. Select [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  2. Select the session you would like to make read-only.
  3. Click the [Properties] button to display the [Session Properties] dialog box (figure 4.30).
  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 [Debug -> Debug Sessions…] to display the [Debug Sessions] dialog box (figure 4.28).
  2. Select the session you would like to save.
  3. Click the [Save as…] button to display the [Save Session] dialog box (figure 4.31).
  4. Specify the location to save the new file.
  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.

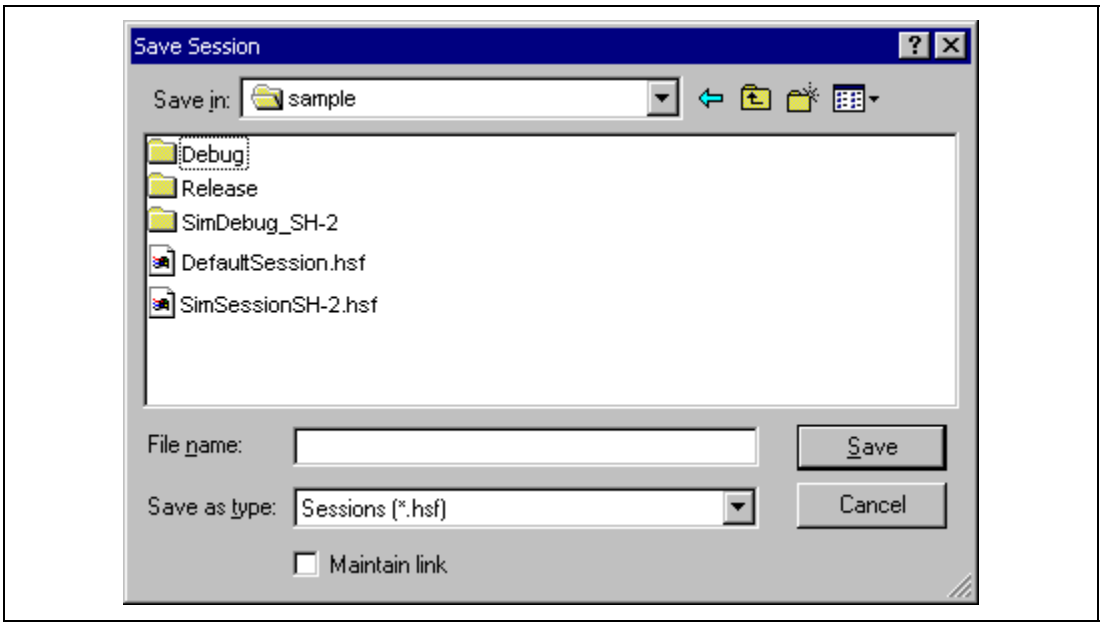


Figure 4.31 [Save Session] Dialog Box

4.3.3 Saving Session Information

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

4.4 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 [Debug] 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.

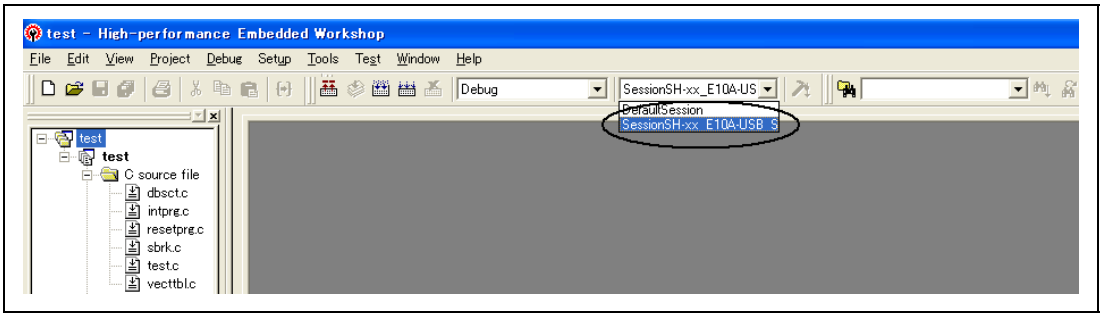


Figure 4.32 Selecting the Session File

In the list box that is circled in figure 4.32, select the session file name including the character string that has been set in the [Target name] text box in figure 4.9, [New Project –8/9– Setting the Debugger Options] 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 4.3, Debug Sessions.

4.5 Reconnecting the Emulator

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

Select [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 4.13, [Debug Settings] Dialog Box ([Target] Page)) that is opened by selecting [Debug settings] from the [Debug] menu.

4.6 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.

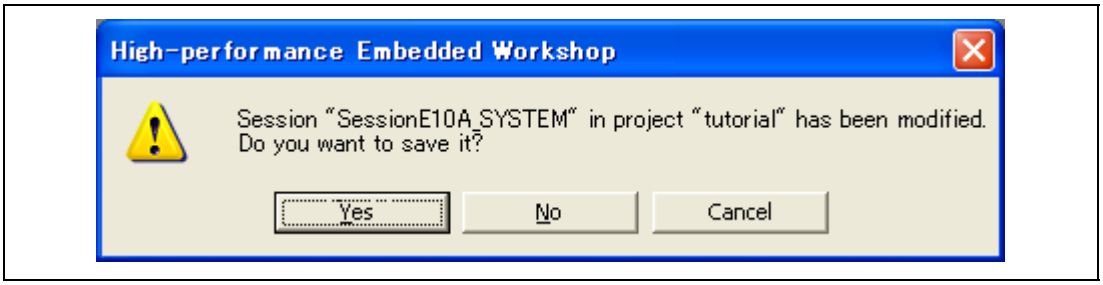


Figure 4.33 [Session has been modified] Message Box

Section 5 Debugging

This section describes the debugging operations and their related windows and dialog boxes.

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.

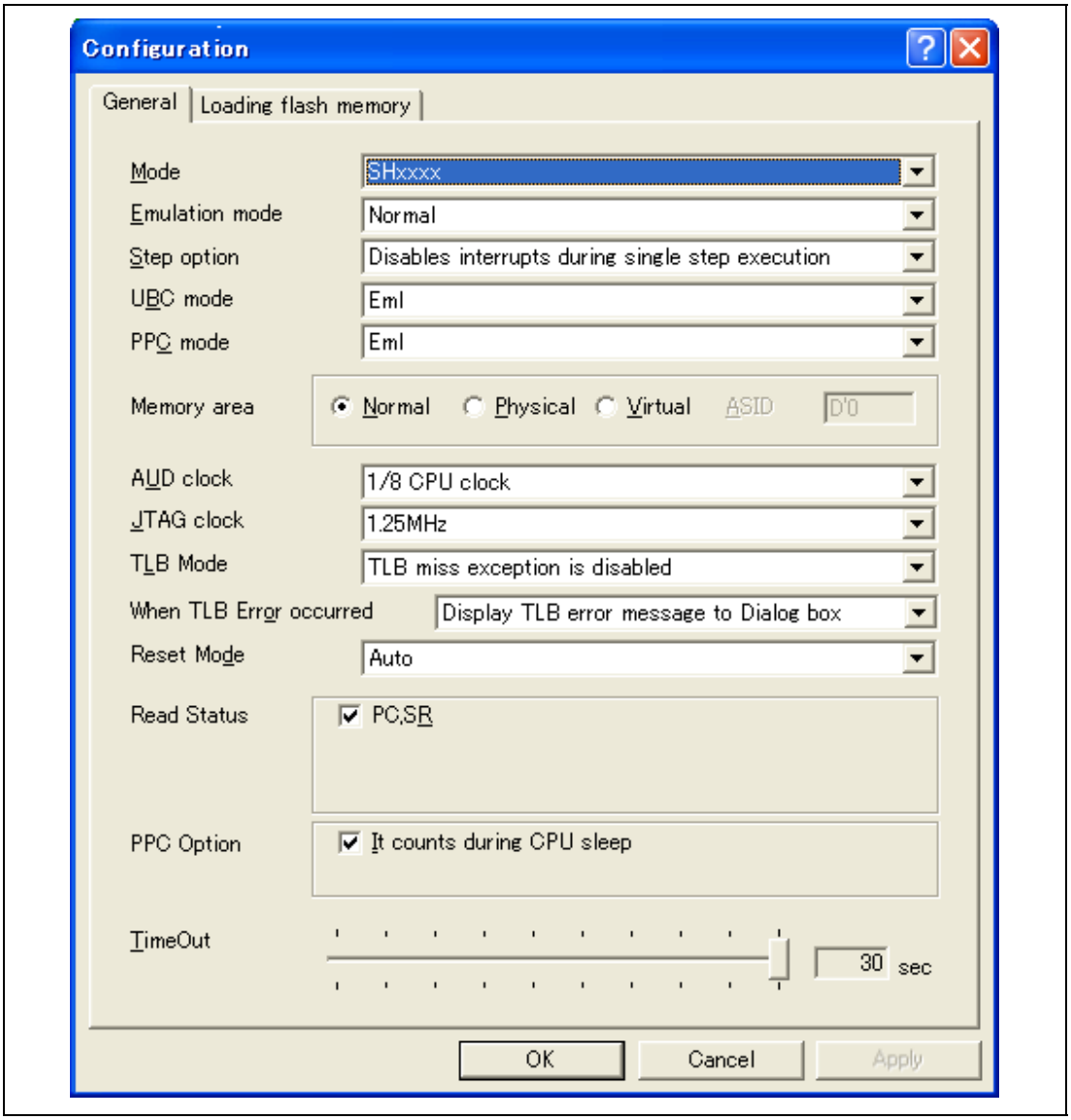


Figure 5.1 [Configuration] Dialog Box ([General] Page)

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

[Mode]	Displays the MCU/MPU name.
[Emulation mode]	Selects the emulation mode at user program execution. Select Normal to perform normal emulation. Select No break to disable PC breakpoint or break condition settings during emulation.
[Step option]	Sets the step interrupt option. Disable interrupts during single step execution: Disables interrupts* during step execution. Enable interrupts during single step execution: Enables interrupts during step execution. *Note: Include interrupts in a break.
[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 MCU/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 2.2.3, 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 MCU/MPU's guaranteed TCK range. For the upper limit for TCK, refer to section 2.2.3, Notes on Using the JTAG (H-UDI) Clock (TCK) and AUD Clock (AUDCK), in the additional document, Supplementary Information on Using the SHxxxx.

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 Downloading to the Flash Memory

Sets the emulator operation conditions for downloading the external flash memory. This function is not available when the SH7047F, SH7144F, or SH7145F is in use.

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

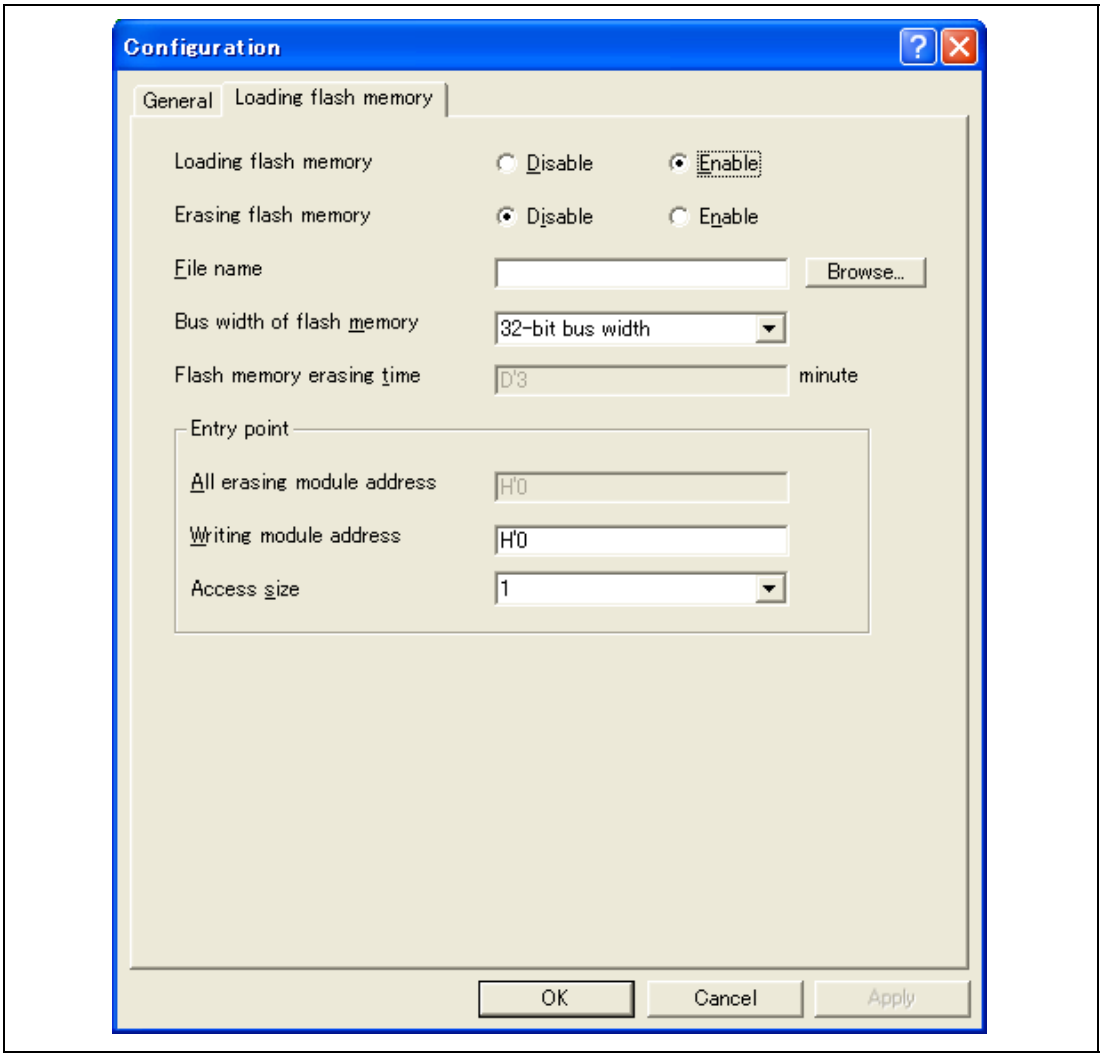


Figure 5.2 [Configuration] Dialog Box ([Loading flash memory] Page)

Items that can be displayed in the sheet 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 programmed. 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 or access size of the write/erase module. (It must be RAM address.) All erasing module address: Inputs the calling destination address of the erase module. Writing module address: Inputs 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.

5.2 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 been detected, 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 4.2, 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 target 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.

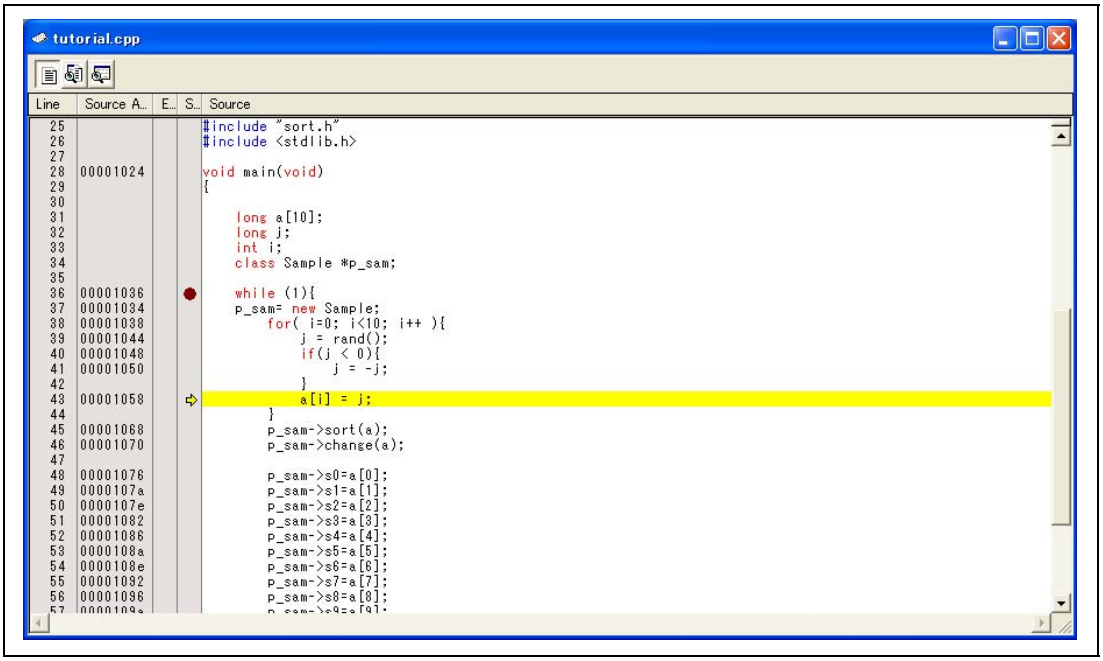


Figure 5.3 [Source] Window

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 (Event column): Event information (event condition)

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

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.

Event column

The Event column displays the following item:

●: An address condition for the event condition is set. The number of address conditions that can be set is the same as that of event condition channels at which the address condition can be set, but it differs depending on the product.

This is also set by using the popup menu.

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

Figure 5.4 Popup Menu

Note: The contents of the 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

S/W breakpoint column displays the following items:

: A bookmark is set.

: A PC Break is set.

: PC location

➲ To switch off a column in all source files

1. Click the right-hand mouse button on the [Source] window or select the [Edit] menu.
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.

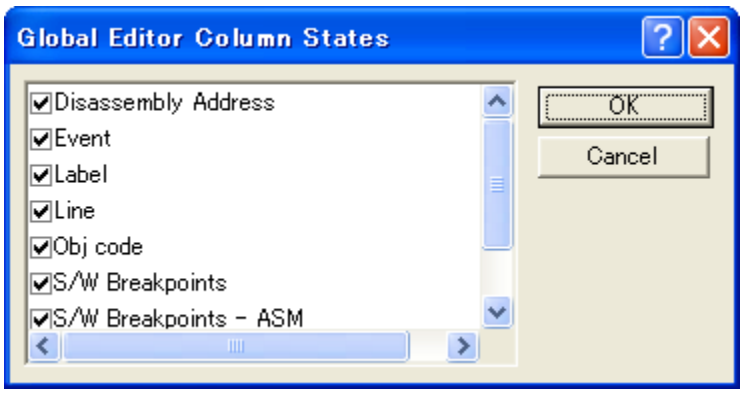


Figure 5.5 [Global Editor Column States] Dialog Box

➲ 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 [Edit] menu.
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 [Disassembly] toolbar button at the top of the window when a source file is opened to show the assembly-language code 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).

Selecting the [Mixed display] toolbar button ( ) displays both the source and the code. The following shows an example in this case.

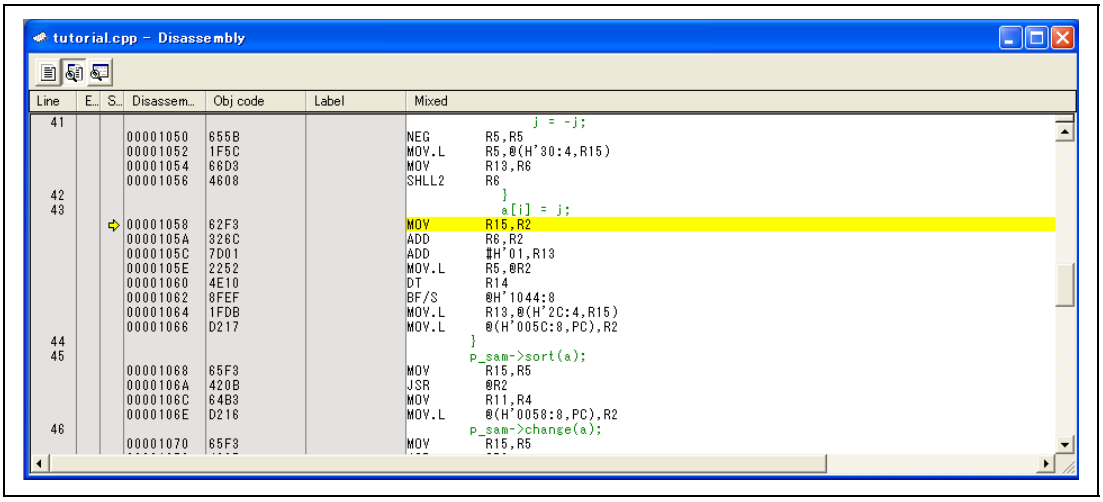


Figure 5.6 [Disassembly] Window

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.

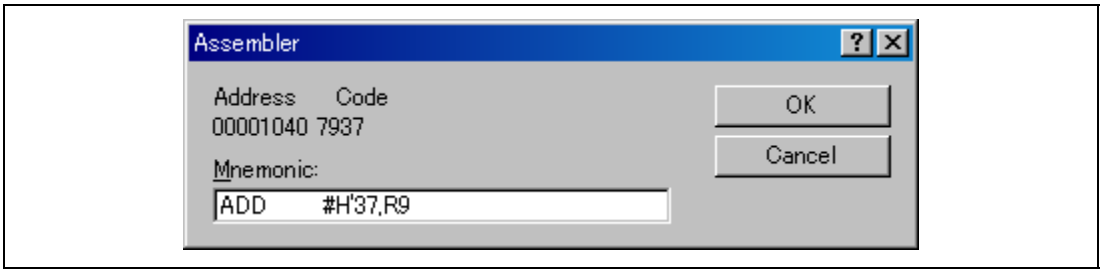


Figure 5.7 [Assembler] Dialog Box

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 machine code on the actual 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 [Editor] window will not be changed. This is the same even if the source file contains 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. Double-click on the address in the [Disassembly] window or select [Set Address…] from the popup menu, and the dialog box shown in figure 5.8 is displayed.

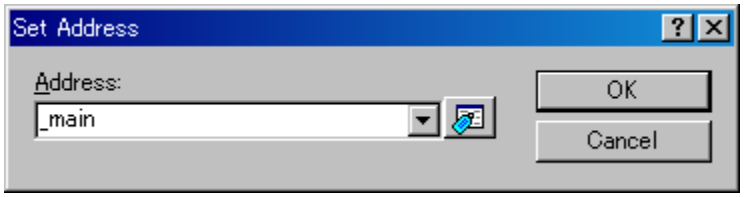


Figure 5.8 [Set Address] Dialog Box

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 [Editor] 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.

Note: This function is not supported in some devices to be debugged. For details on the specifications of each product, refer to the online help.

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.

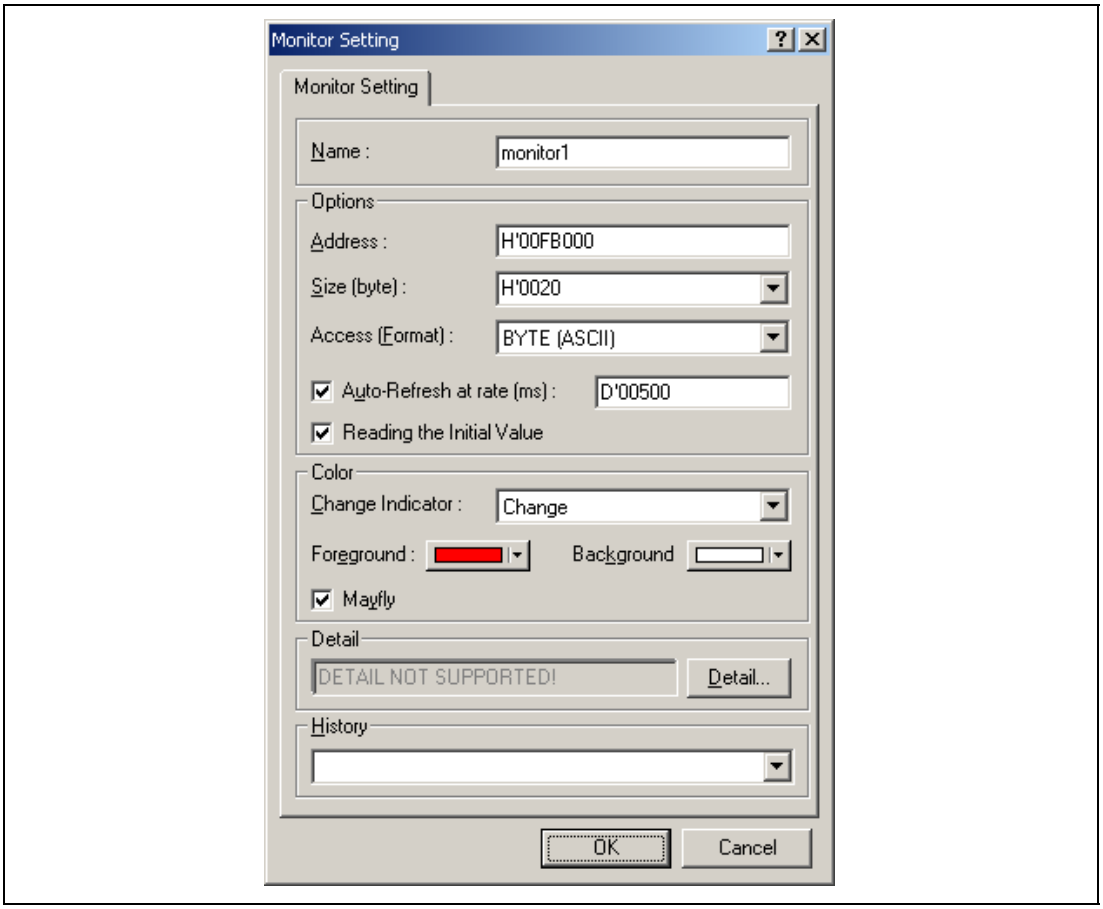


Figure 5.9 [Monitor Setting] Dialog Box

[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]: Not supported in the emulator.

[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.

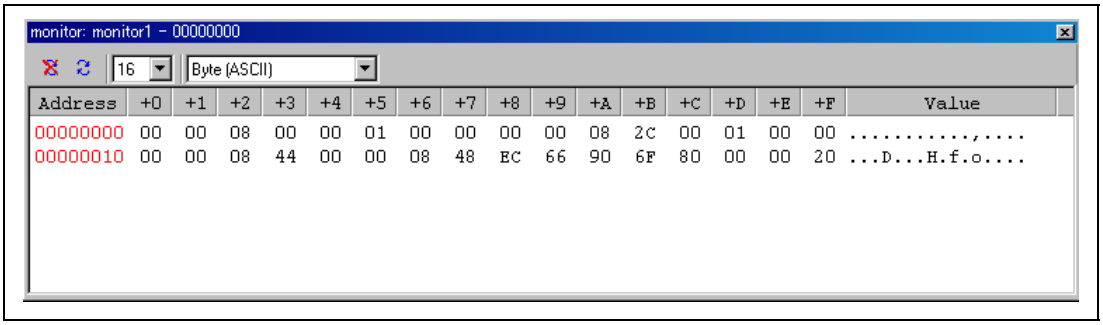


Figure 5.10 [Monitor] Window

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.

When the address of the variable registered in the [Watch] window exists within the monitoring range that has been set by the Monitor function, the value of the variable can be updated and displayed.

This function allows checking the content of a variable without affecting the realtime operation.

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.

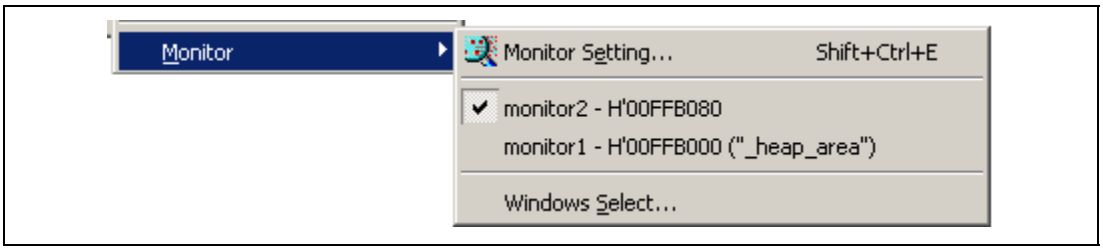


Figure 5.11 Monitor Setting List

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.

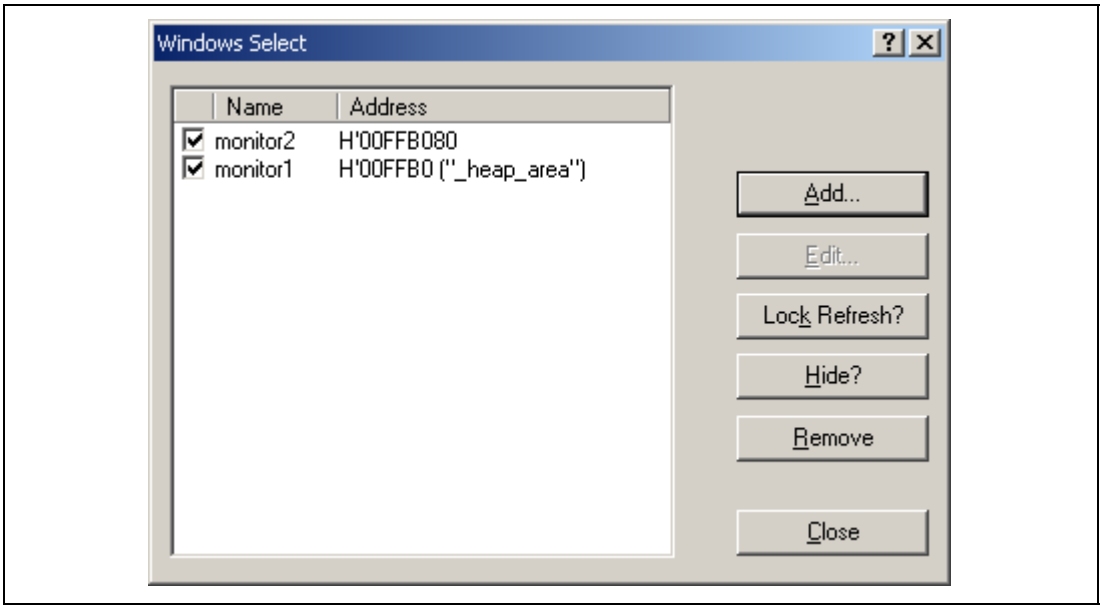


Figure 5.12 [Windows Select] Dialog Box

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.

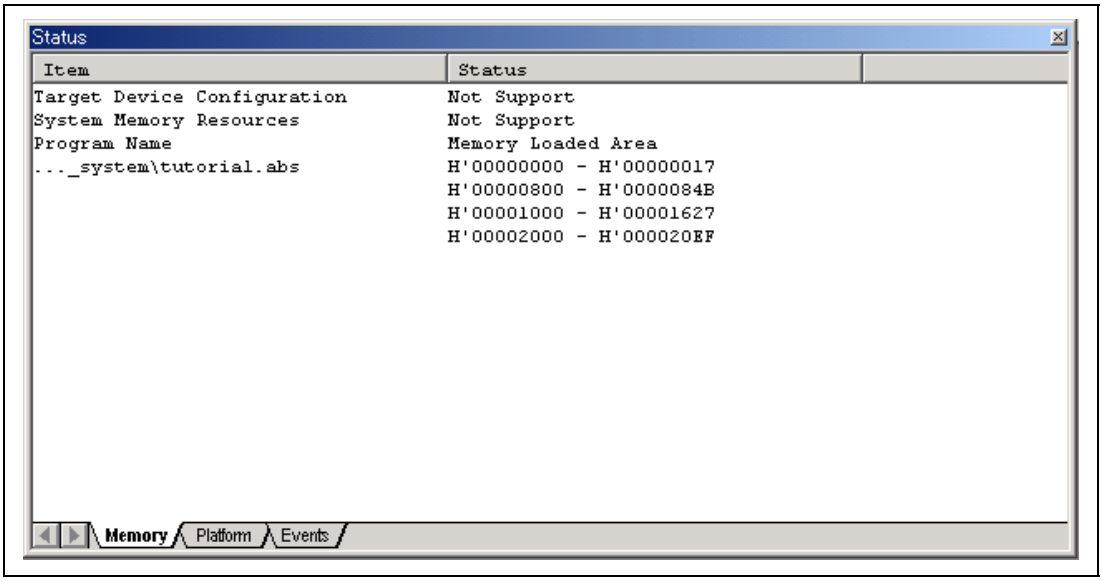


Figure 5.13 [Status] Window

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.
- [Platform] sheet
  Contains information about the status of the emulator, typically including the CPU type and emulation mode, the state of execution, and the statistic information of execution.
- [Events] sheet
  Contains information about the event information, including resource information and breakpoints.

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 Using the Event Points

The emulator has the event point 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.

5.5.1 PC Breakpoints

When the instruction of the specified address is fetched, the user program is stopped. Up to 255 points can be set.

5.5.2 Event Conditions

Event conditions can be used for more complex conditions such as the data condition as well as specification of the single address.

When the condition is satisfied, event conditions are also used as the start/end conditions for performance measurement in addition to halting the user program. When event conditions are used as the start/end conditions for performance measurement, start from setting in the [Performance Analysis] window.

Several event conditions can be used to set more complex conditions.

Notes: 1. When event conditions are used as the start/end conditions for performance measurement, step operation cannot be performed. In addition, when execution is restarted from the address where step operation has been stopped by the BREAKPOINT, the single step function is used and operation is disabled. Restart execution after the BREAKPOINT has been canceled.

2. 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 performance measurement start/end conditions are set, the settings of the break conditions will be disabled.

3. The event conditions that can be set vary according to the emulator in use. For details, refer to the online help.

5.5.3 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 two sheets:

[Breakpoint] sheet: Displays the settings made for PC breakpoints. It is also possible to set, modify, and cancel PC breakpoints.

[Event condition] sheet: Displays or sets the settings made for event condition channels.

5.5.4 Setting PC Breakpoints

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

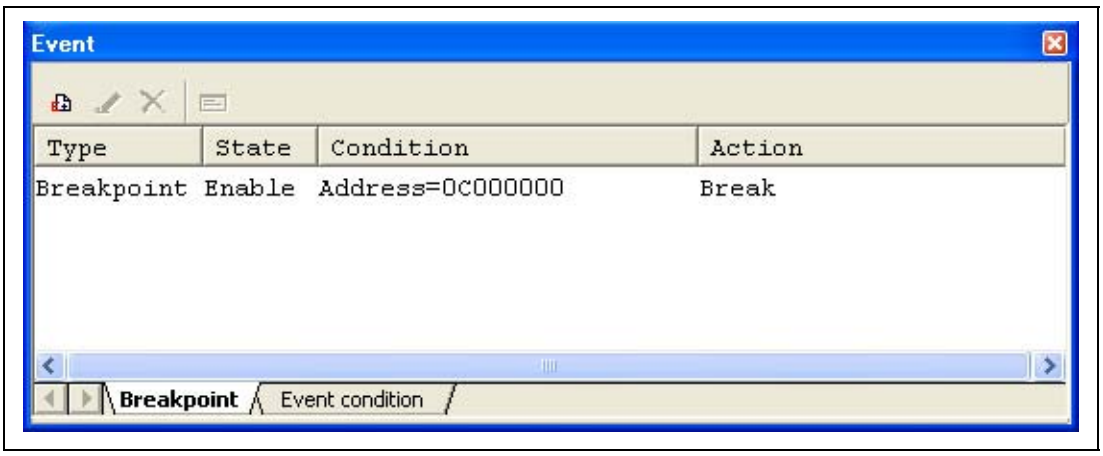


Figure 5.14 [Event] Window ([Breakpoint] Sheet)

This window displays and sets the breakpoints. 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

When a breakpoint is double-clicked in this window, the [Breakpoint] dialog box is opened and break conditions can be modified.

A popup menu containing the following options is available by right-clicking within the window.

5.5.5 Add

Sets breakpoints. Clicking this item will open the [Breakpoint] dialog box and break conditions can be specified.

5.5.6 Edit

Only enabled when one breakpoint is selected. Select a breakpoint to be edited and click this item. The [Breakpoint] dialog box will open and break conditions can be changed.

5.5.7 Enable

Enables the selected breakpoint(s).

5.5.8 Disable

Disables the selected breakpoint(s). When a breakpoint is disabled, the breakpoint will remain in the list; when specified conditions have been satisfied, a break will not occur.

5.5.9 Delete

Removes the selected breakpoint. To retain the details of the breakpoint but not have it cause a break when its conditions are met, use the Disable option (see section 5.5.8, Disable).

5.5.10 Delete All

Removes all breakpoints.

5.5.11 Go to Source

Only enabled when one breakpoint is selected. Opens the [Source] window at the address of the breakpoint.

5.5.12 [Breakpoint] Dialog Box

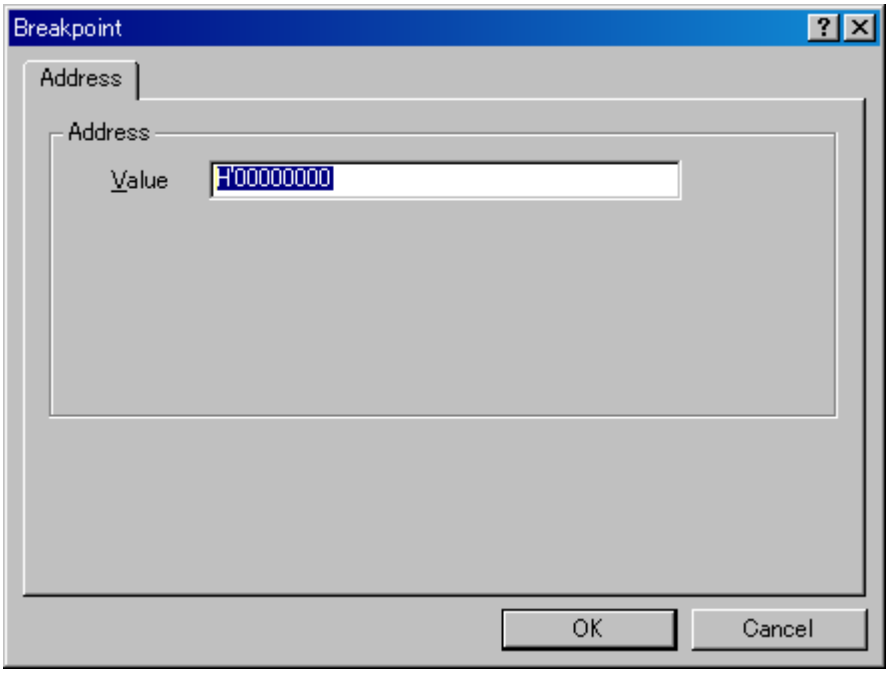


Figure 5.15 [Breakpoint] Dialog Box

This dialog box specifies break conditions.

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 255 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 break conditions. Clicking the [Cancel] button closes this dialog box without setting the break conditions.

5.5.13 Setting Event Conditions

On the [Event condition] sheet, the settings for event conditions are displayed, modified, and added.

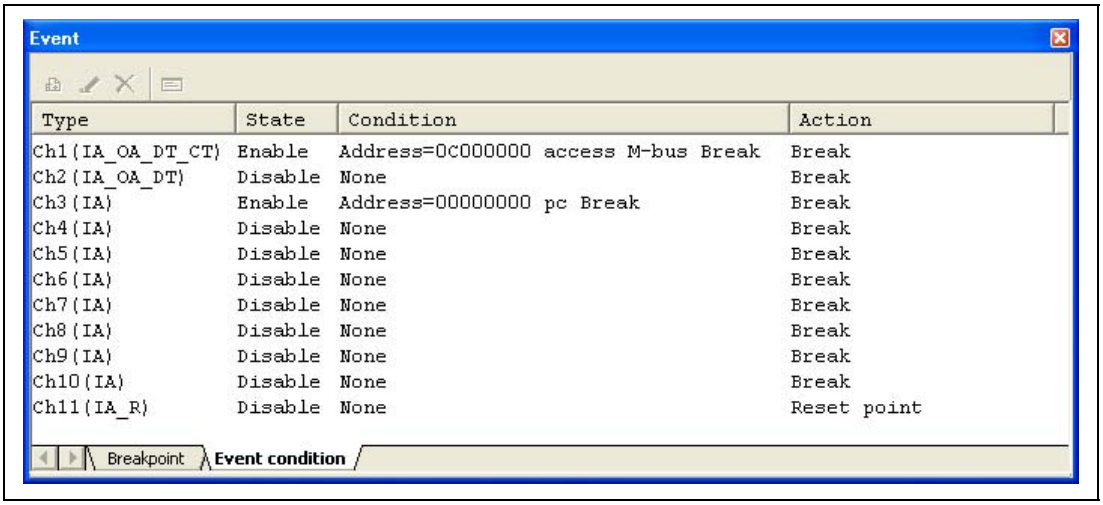


Figure 5.16 [Event] Window ([Event condition] Sheet)

This window displays and sets the break condition. Since the number of channels for detecting conditions and the contents to be set differ depending on the product, refer to the on-line help for each product.

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

[Type] Event channel number

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

[Condition] A condition that satisfies a break. The displayed contents differ depending on the break type.

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

When a breakpoint is double-clicked in this window, the [Event condition] dialog box is opened and break conditions can be modified. For details on the [Event condition] dialog box, refer to the on-line help for each product.

A popup menu containing the following options is available by right-clicking within the window.

5.5.14 Edit...

Only enabled when one event channel is selected. Select a breakpoint to be edited and click this item. The [Event condition] dialog box will open and break conditions can be changed.

5.5.15 Enable

Enables the selected event channel(s). An event channel that the condition has not been set is not enabled.

5.5.16 Disable

Disables the selected event channel(s). When an event channel is disabled, a break will not occur even if specified conditions have been satisfied.

5.5.17 Delete

Initializes the condition of the selected event channel. To retain the details of the event channel but not have it cause a break when its conditions are met, use the Disable option (see section 5.5.16, Disable).

5.5.18 Delete All

Initializes conditions of all event channels.

5.5.19 Go to Source

Only enabled when one event channel is selected. Opens the [Source] window at address of event channel.

If an address value has not been set to the event channel, this option cannot be used.

5.5.20 [Combination action(Sequential PtoP)]

Sets a sequential condition of event channels or points where performance measurement or internal tracing starts and stops.

5.5.21 Editing Event Conditions

Handlings for settings other than PC breakpoints and event conditions are common. The following describes examples of such handling.

5.5.22 Modifying Event Conditions

Select an event condition to be modified, and choose [Edit...] from the popup menu to open the dialog box for the event, which allows the user to modify the event conditions. The [Edit...] menu is only available when one event condition is selected.

5.5.23 Enabling Event Conditions

Select an event condition and choose [Enable] from the popup menu to enable the selected event condition.

5.5.24 Disabling Event Conditions

Select an event condition and choose [Disable] from the popup menu to disable the selected event condition. When an event condition is disabled, the event condition will remain in the list, but an event will not occur when the specified conditions have been satisfied.

5.5.25 Deleting Event Conditions

Select an event condition and choose [Delete] from the popup menu to remove the selected event condition. To retain the event condition but not have it cause an event when its conditions are met, use the [Disable] option (see section 5.5.24, Disabling Event Conditions).

5.5.26 Deleting All Event Conditions

Choose [Delete All] from the popup menu to remove all event conditions.

5.5.27 Viewing the Source Line for Event Conditions

Select an event condition and choose [Go to Source] from the popup menu to open the [Source] or [Disassembly] window at the address of the event condition. The [Go to Source] menu is only available when one event condition that has the corresponding source file is selected.

5.6 Viewing the Trace Information

For the description on the trace function, refer to section 2.2, Trace Functions.

5.6.1 Opening the [Trace] Window

To open the [Trace] window, choose [View -> Code -> Trace] or click the [Trace] toolbar button ( ).

5.6.2 Acquiring Trace Information

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 [Trace] window.

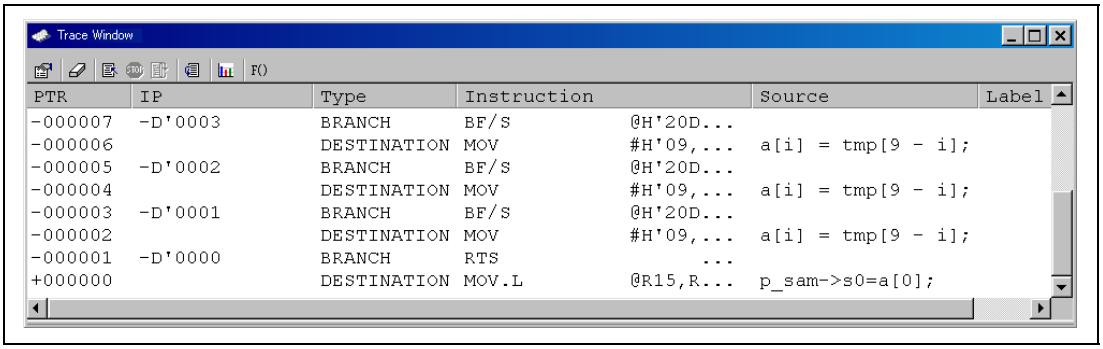


Figure 5.17 [Trace] Window (Type 1) (Internal Trace)

This window displays the following trace information items:

[PTR] Pointer to a location in the trace buffer (+0 for the last executed instruction)

[IP] The amount of acquired trace information

[Type] Type of branch:
BRANCH: Branch source
DESTINATION: Branch destination

[Address] Instruction address

[Instruction] Instruction mnemonic

[Source] The C/C++ or assembly-language source program

[Label] Label information

Selecting the [Set…] menu in the popup menu of the [Trace] window displays the [Acquisition] dialog box. When [AUD function] is selected in [Trace Type] within the dialog box, the trace information is acquired by using the AUD trace function.

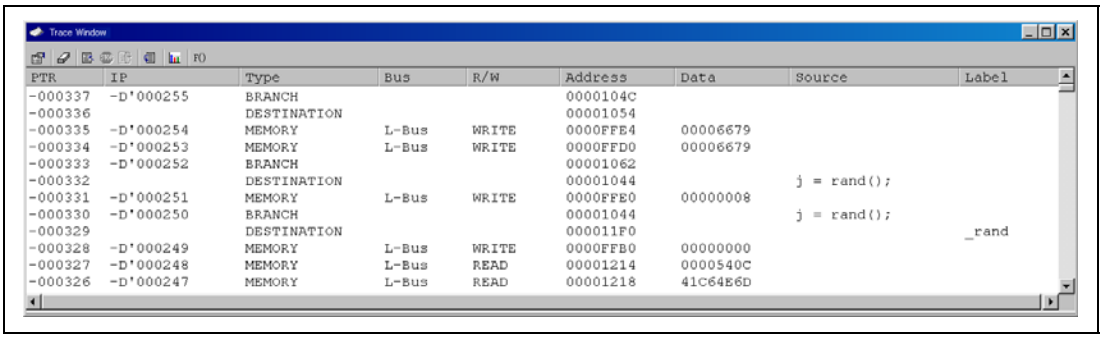


Figure 5.18 [Trace] Window (Type 1) (AUD Trace)

This window displays the following trace information items (some of this information will not be displayed in some products):

[PTR] The 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)
CPU-WAIT: CPU was waiting for the output of the trace information (only in the non-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.

[Instruction] Instruction mnemonic

[Repeat] Displayed only when the Repeat filter is used. This item shows the number of consecutive branch operations.

[Probe] State of the input probe

[Timestamp] Timestamp value

[Source] The C/C++ or assembly-language source program

[Label] Label information

Note: Since the displayed contents differ depending on the product, refer to each product's online help. Some MCUs/MPUs supported may not have the AUD trace function.

Some devices 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.

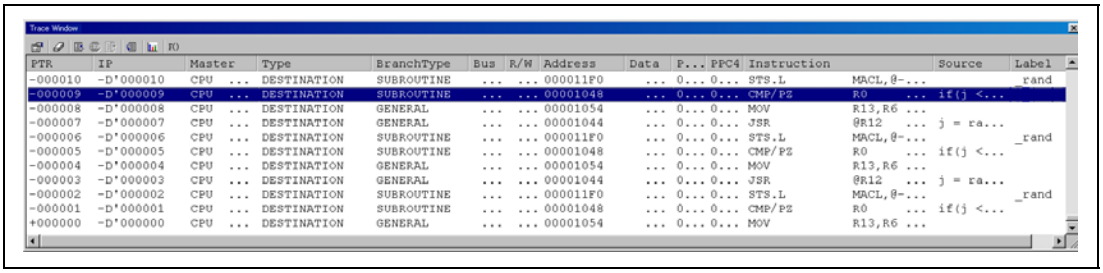


Figure 5.19 [Trace] Window (Type 2)

[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)
CPU-WAIT: CPU was waiting for the output of the trace information (only in the non-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.
[PPC]	Performance-counter output
[Instruction]	Instruction mnemonic
[Source]	The C/C++ or assembly-language source program
[Label]	Label information

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 can change the display order.

5.6.3 Specifying Trace Acquisition Conditions

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 acquisition condition is set in the [Acquisition] dialog box that is displayed by selecting [Acquisition…] from the popup menu.

The [Acquisition] dialog box has the following pages:

Table 5.1 [Acquisition] Dialog Box Pages

Page	Item
[Trace mode]	Sets trace acquisition conditions.
[Window trace]	Sets window trace acquisition conditions.*2
[AUD Branch trace]	Sets branch conditions acquired by the AUD trace function.*1
[Branch Trace]	Sets branch conditions acquired by the internal trace function.

Notes: 1. This dialog box is not supported by the products that do not support the AUD trace function.
2. Some products do not support this page. For details, refer to the online help for each product.

(1) [Trace mode] page

Sets trace acquisition conditions.

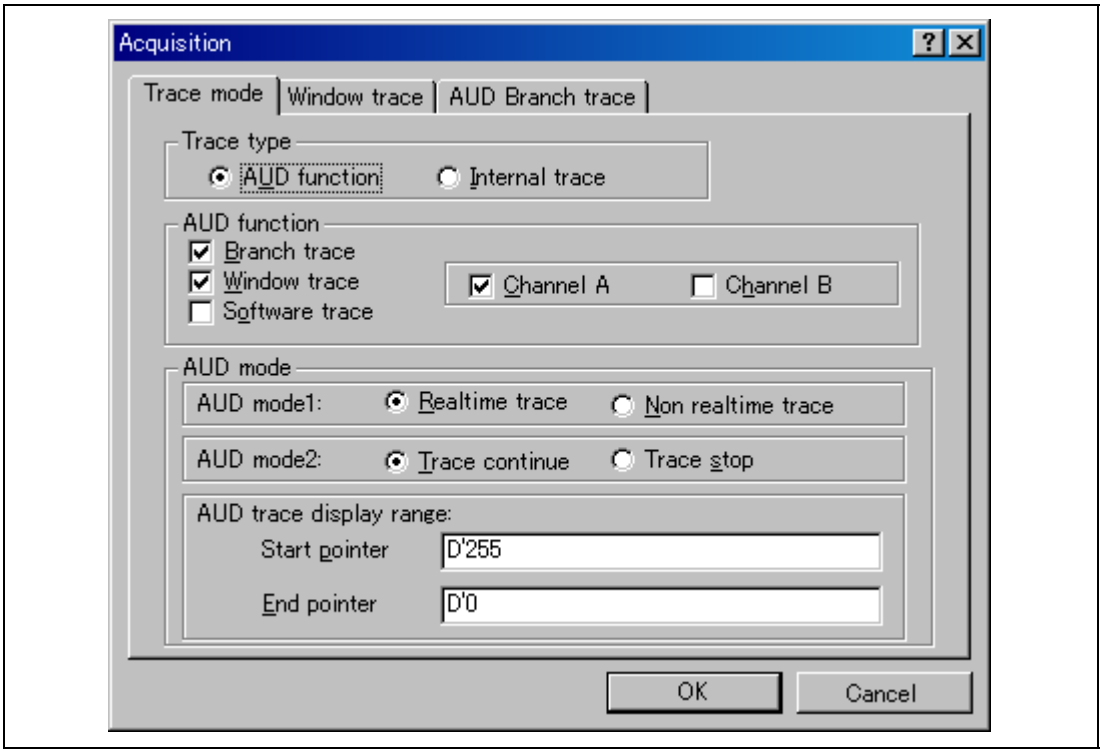


Figure 5.20 [Acquisition] Dialog Box ([Trace mode] Page)

This dialog box specifies the methods and conditions for the acquisition of trace information.

[Trace type]: Selects the type of the trace function.

[AUD function] Uses the AUD trace function.
[Internal trace] Uses the internal trace function.

The following can only be set when the AUD trace function is used.

[AUD function]: Sets the trace acquisition condition.

For a description of the conditions for trace acquisition, refer to section 2.2.2, AUD Trace Function.

[AUD mode]: Sets the acquisition mode of the AUD trace.

[AUD mode 1]: An option to determine the operation when the trace information is continuously generated.

[Realtime trace] Some of the trace information will not be output.

[Non realtime trace] Where necessary, the CPU waits until each item of trace information has been output.

[AUD mode 2]: An option that determines the operation when the trace buffer of the emulator becomes full.

[Trace continue] The oldest trace information is overwritten by the latest information.

[Trace stop] Trace information is not acquired after the buffer has been filled.

[AUD trace display range]: Set the display range of the [Trace] window.

[Start pointer] The trace is displayed from the specified value.

[End pointer] The trace is displayed to the specified value.

Clicking the [OK] button stores the settings. Clicking the [Cancel] button closes this dialog box without modifying the settings.

Some devices to be debugged display the following dialog box. For details on the specifications of each product, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help.

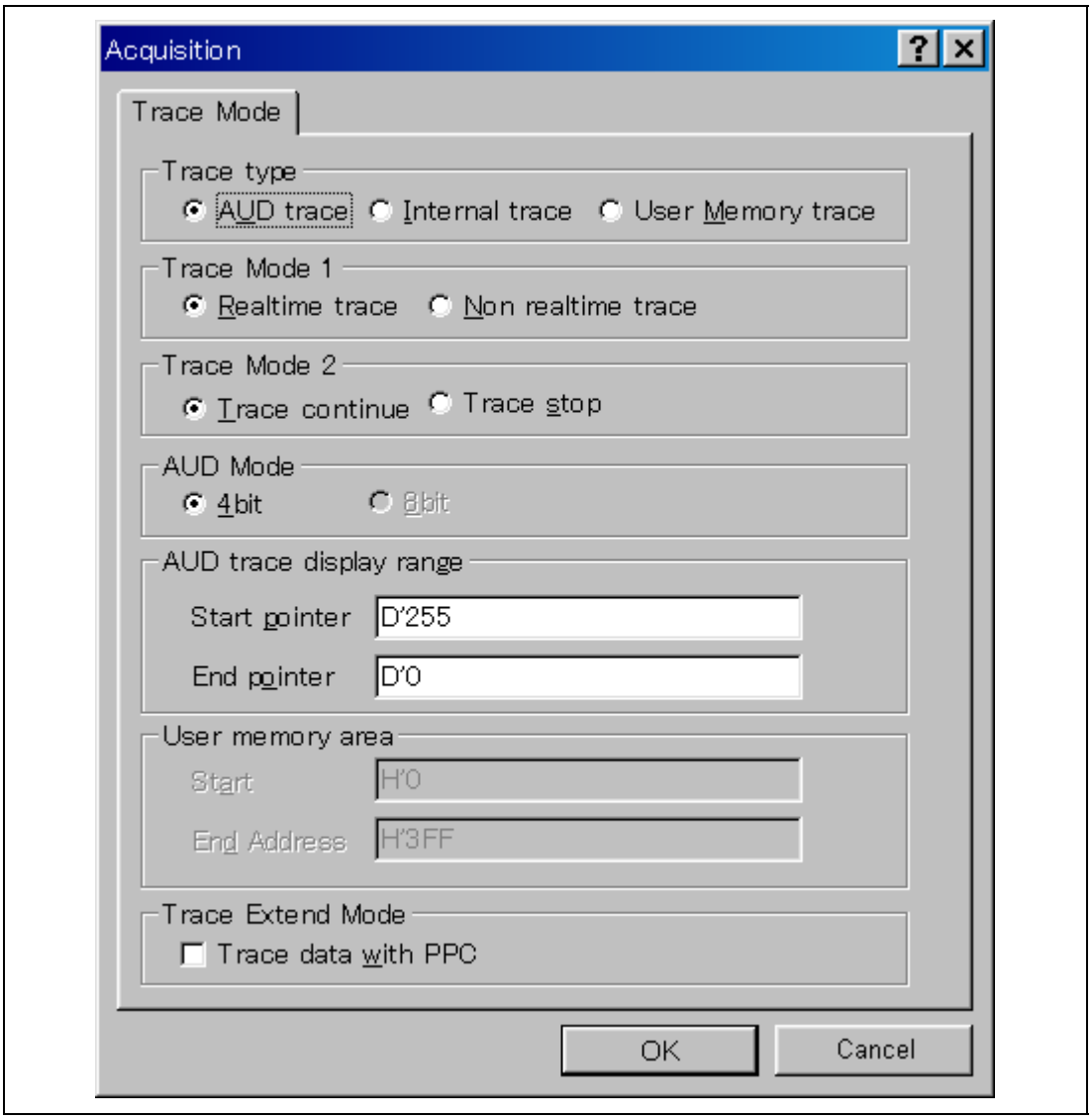


Figure 5.21 [Acquisition] Dialog Box ([Trace mode] Page)

[Trace type]: Selects the type of trace function.

- [AUD function] Uses the AUD trace function.
- [Internal trace] Uses the internal trace function.
- [User Memory trace] Uses the memory output function of trace data.

[Trace Mode 1]: An option to determine the operation when the trace information is continuously generated; only available when selecting [AUD trace] or [User Memory trace].

- [Realtime trace] Some of the trace information will not be output.
- [Non realtime trace] Where necessary, the CPU waits until each item of trace information has been output.

[Trace mode 2]: An option that determines the operation when the trace buffer of the emulator becomes full; only available when selecting [AUD trace] or [User Memory trace].

- [Trace continue] The oldest trace information is overwritten by the latest information.
- [Trace stop] Previous trace information is not acquired.

[AUD Mode]: Some devices to be debugged can select the 8-bit mode of the AUD pin. For details on the specifications of each product, refer to the additional document, Supplementary Information on Using the SHxxxx, or the online help. This is only available when [AUD trace] is selected.

[AUD trace display range]: An option to set the display range of the [Trace] window; this is only available when [AUD trace] is selected.

- [Start pointer] The trace is displayed from the specified value.
- [End pointer] The trace is displayed to the specified value.

[User Memory area]: An option to set the display range of the [Trace] window; this is only available when [User Memory trace] is selected.

- [Start] Specifies the start address of the memory range that the trace result is written to.
- [End address] Specifies the end address of the memory range that the trace result is written to.

[Trace Extend Mode]:

- [Trace data with PPC] Outputs a performance counter in the [Trace] window. (If this function is enabled, the source of the branch trace will not be displayed.)

Clicking the [OK] button stores the settings. Clicking the [Cancel] button closes this dialog box without modifying the settings.

(2) [Window trace] page

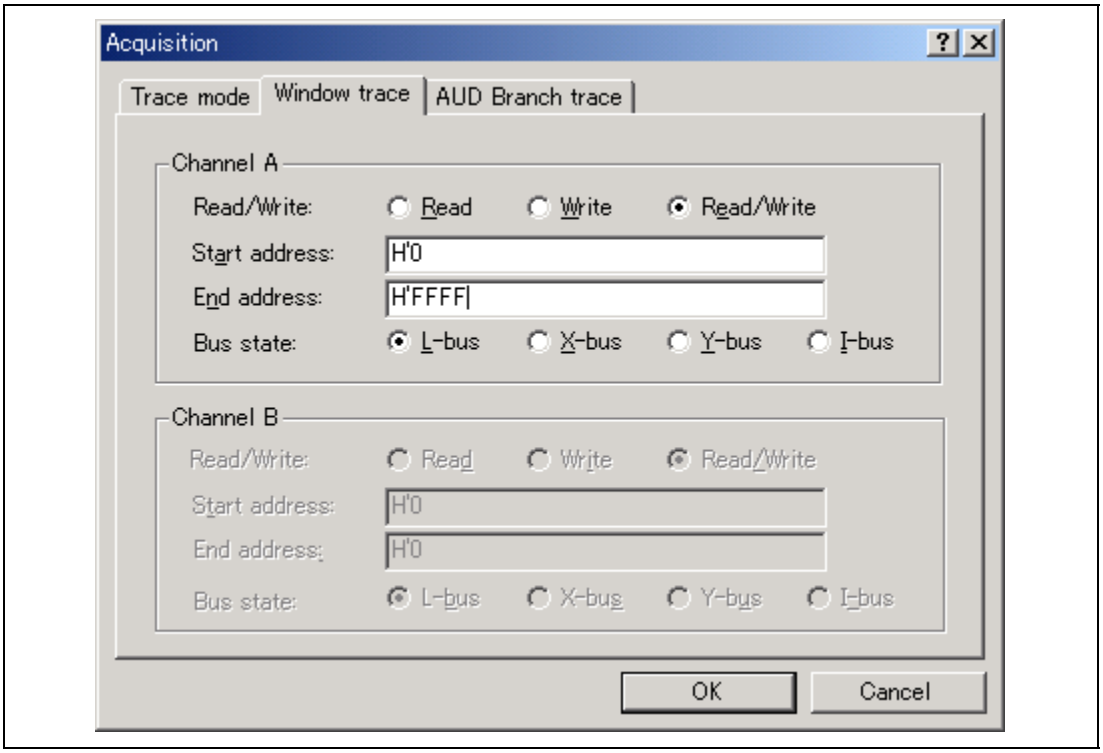


Figure 5.22 [Acquisition] Dialog Box ([Window trace] Page)

This dialog box is used to specify conditions for the acquisition of trace information. For a description of the conditions for trace acquisition, refer to section 2.2.2, AUD Trace Functions.

[Channel A]: Enables channel A of the window trace.

[Bus state]	Selects a bus for trace acquisition.
[Read/Write]	Sets tracing of read or write access or both.
[Trace start address]	Sets an address range for the tracing of data access. The start address is set here.
[Trace end address]	Sets an address range for the tracing of data access. The end address is set here.

[Channel B]: Enables channel B of the window trace.

[Bus state]	Selects a bus for trace acquisition.
[Read/Write]	Sets tracing of read or write access or both.
[Trace start address]	Sets an address range for the tracing of data access. The start address is set here.
[Trace end address]	Sets an address range for the tracing of data access. The end address is set here.

(3) [AUD Branch trace] page

Selects the type of branches to be acquired.

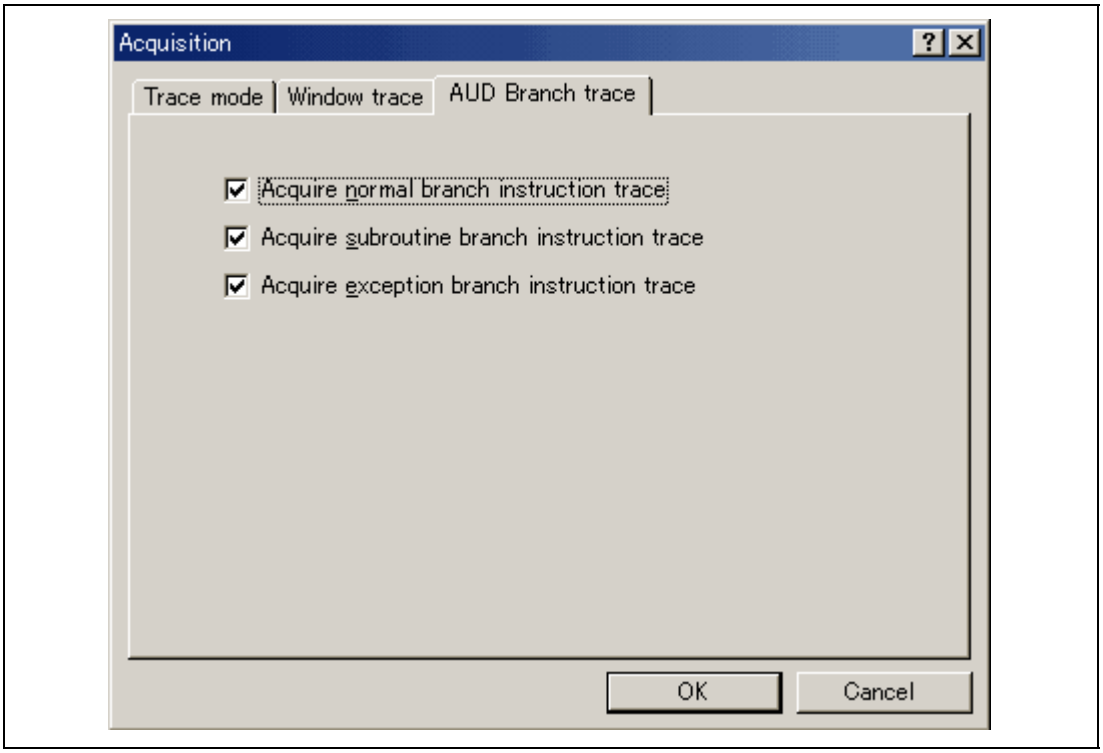


Figure 5.23 [Acquisition] Dialog Box ([AUD Branch trace] Page)

5.6.4 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 5.2 [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.

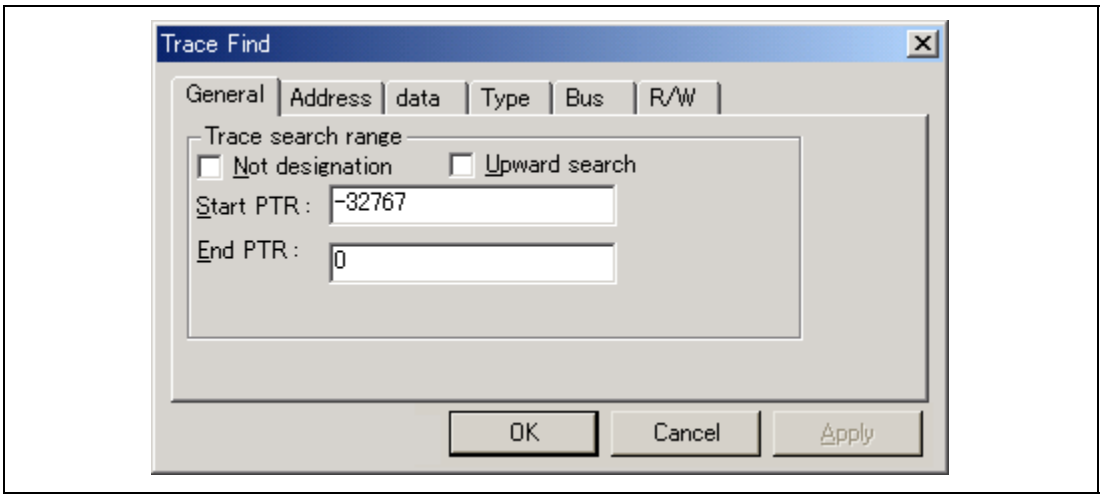


Figure 5.24 [Trace Find] Dialog Box ([General] Page)

[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.

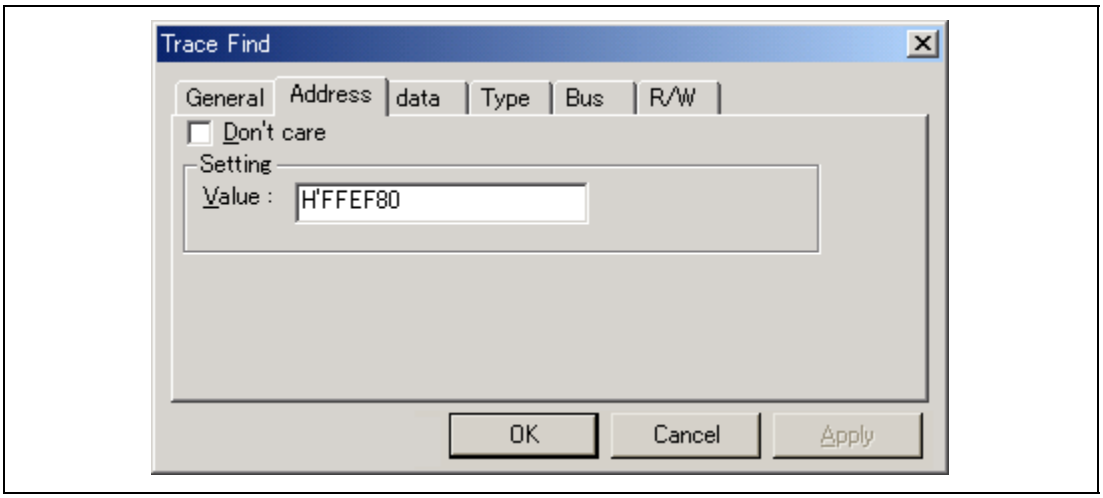


Figure 5.25 [Trace Find] Dialog Box ([Address] Page)

[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.

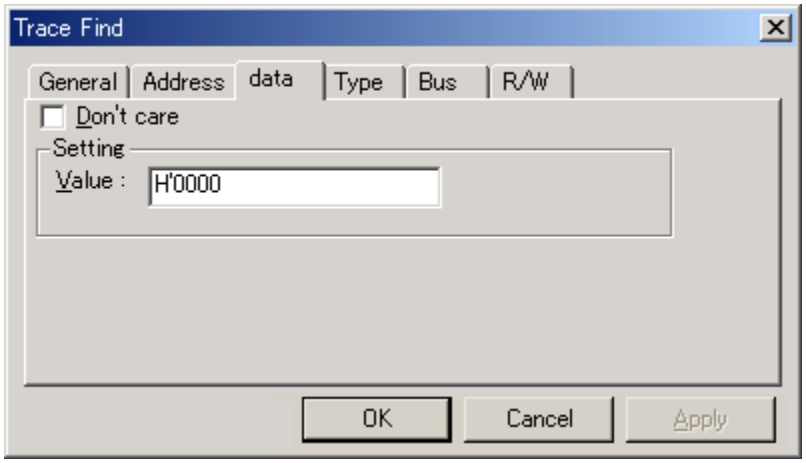


Figure 5.26 [Trace Find] Dialog Box ([data] Page)

[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.

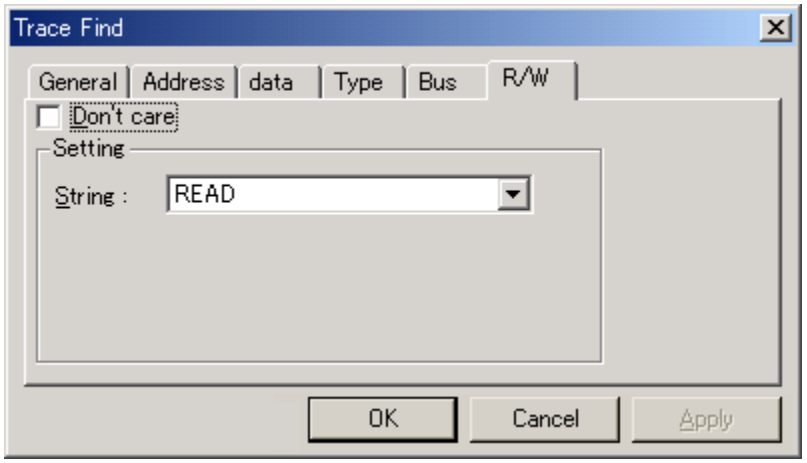


Figure 5.27 [Trace Find] Dialog Box ([R/W] Page)

[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.

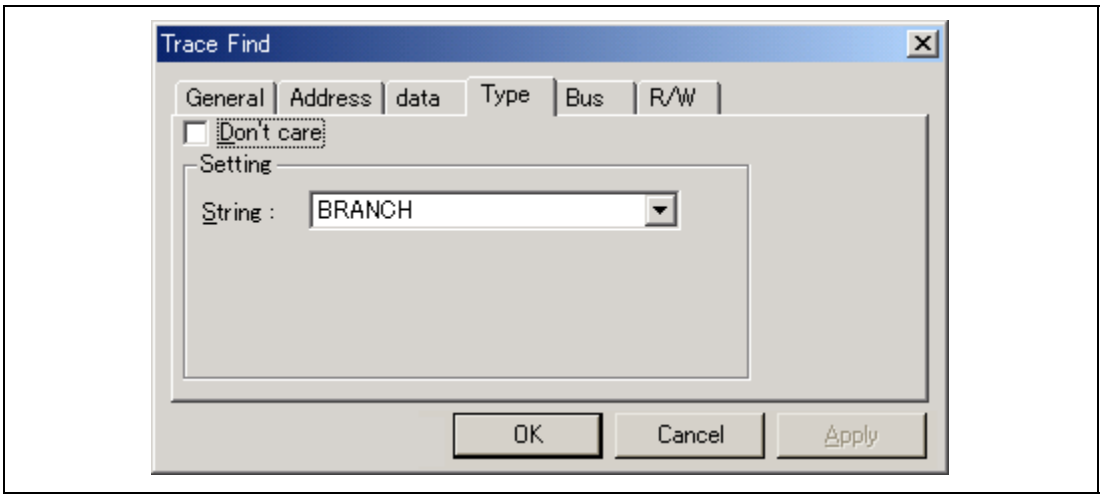


Figure 5.28 [Trace Find] Dialog Box ([Type] Page)

[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.

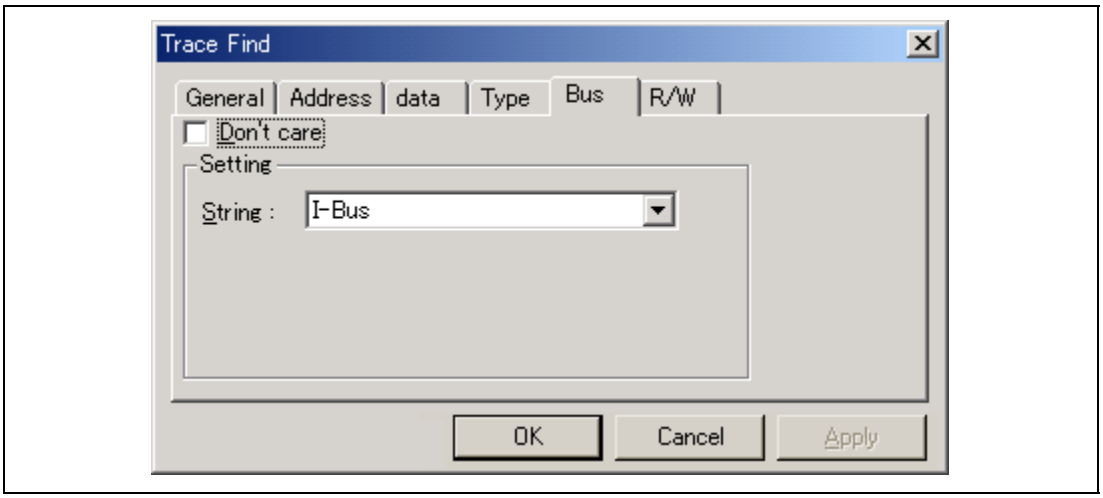


Figure 5.29 [Trace Find] Dialog Box ([Bus] Page)

[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).

5.6.5 Clearing the Trace Information

When [Clear] is selected from the popup menu, the trace buffer that stores the trace information becomes empty. If several [Trace] windows are open, all [Trace] windows will be cleared as they all access the same buffer.

5.6.6 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 5.6.10, Extracting Records from the Acquired Information.

5.6.7 Viewing the [Editor] Window

The [Editor] 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 [Editor] or [Disassembly] window opens and the selected line is marked with a cursor.

5.6.8 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.

5.6.9 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.

5.6.10 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 5.3 [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.

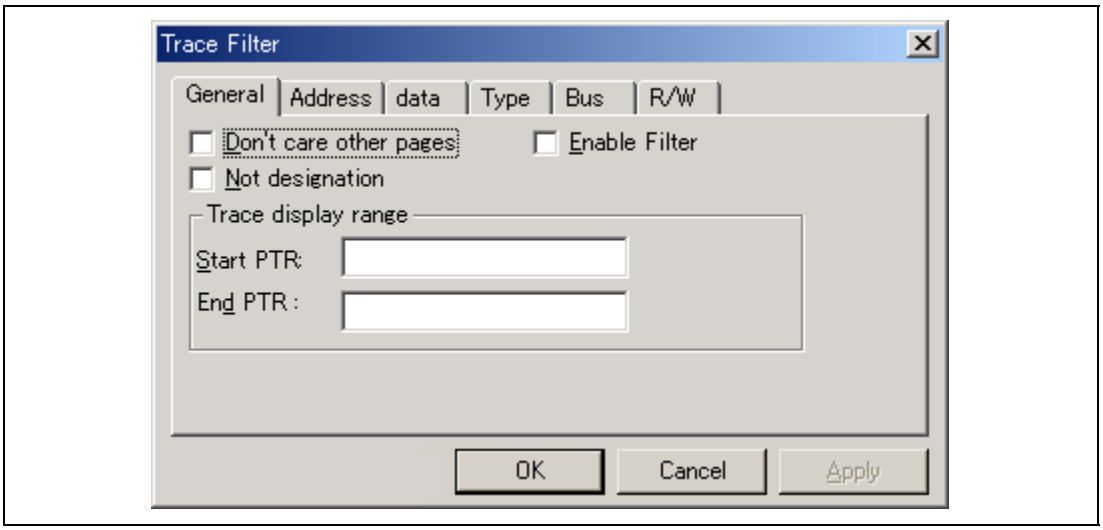


Figure 5.30 [Trace Filter] Dialog Box ([General] Page)

[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.

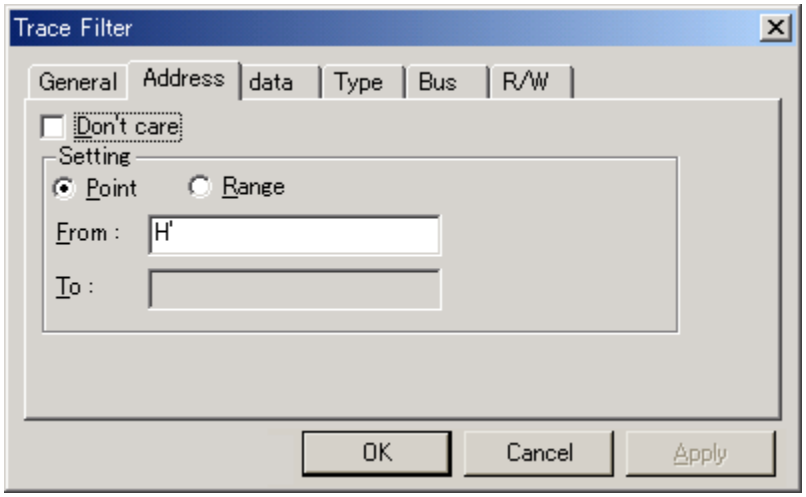


Figure 5.31 [Trace Filter] Dialog Box ([Address] Page)

[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.

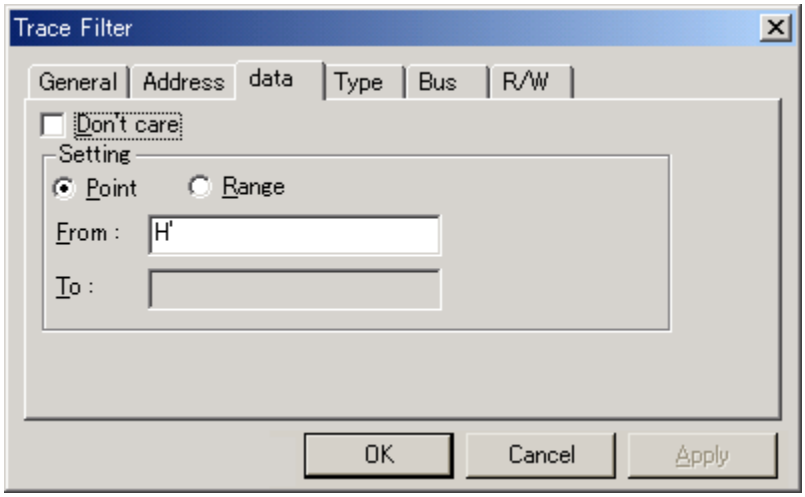


Figure 5.32 [Trace Filter] Dialog Box ([data] Page)

[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.

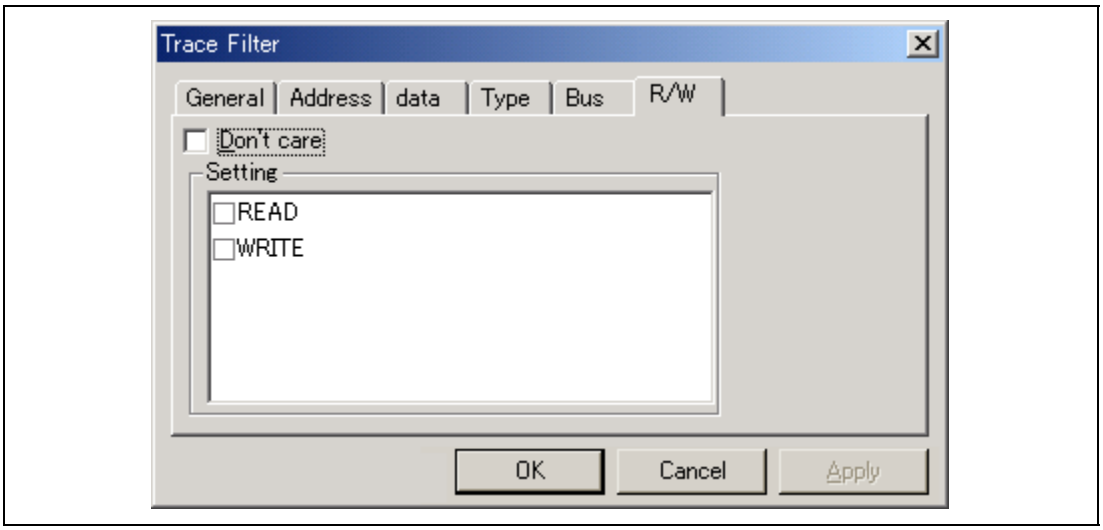


Figure 5.33 [Trace Filter] Dialog Box ([R/W] Page)

[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.

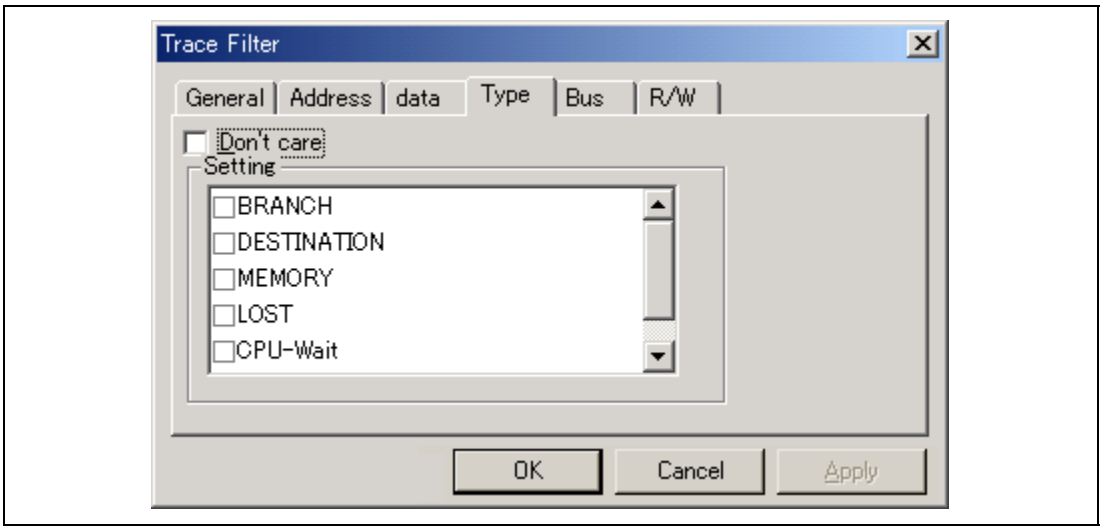


Figure 5.34 [Trace Filter] Dialog Box ([Type] Page)

[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.

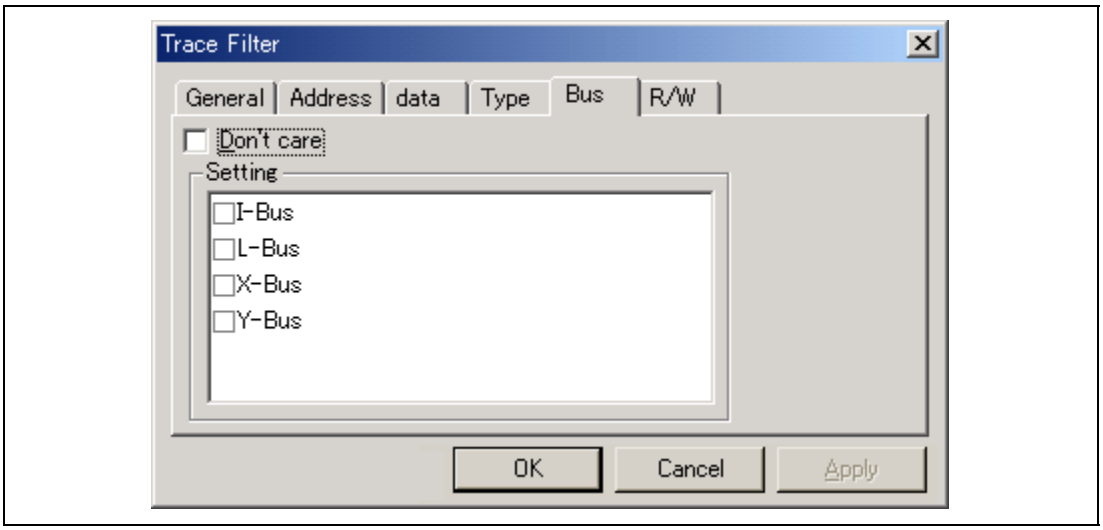


Figure 5.35 [Trace Filter] Dialog Box ([Bus] Page)

[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).

5.6.11 Analyzing Statistical Information

Choose [Statistic…] from the popup menu to open the [Statistic] dialog box and analyze statistical information under the specified conditions.

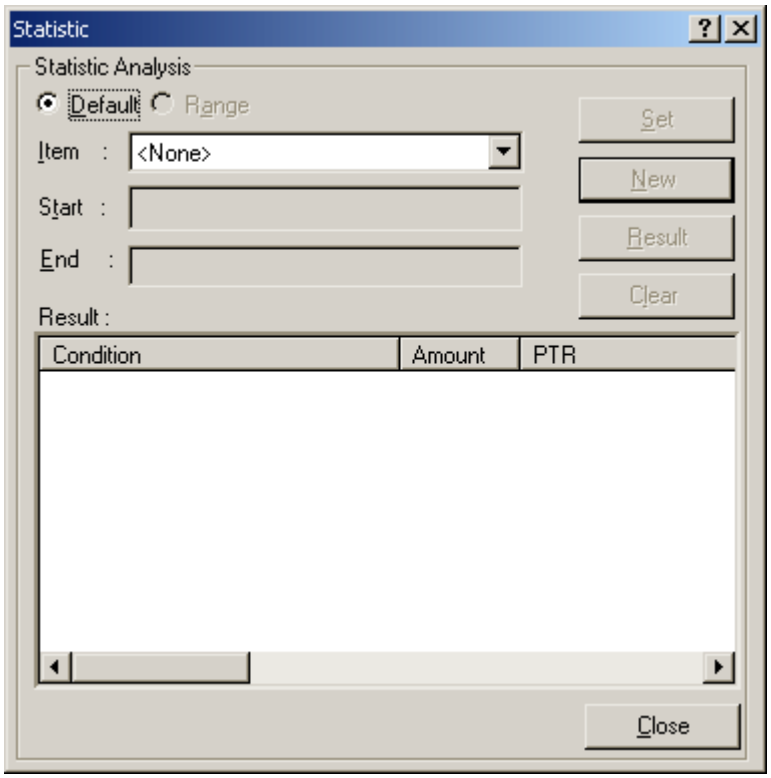


Figure 5.36 [Statistic] Dialog Box

[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.

5.6.12 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.

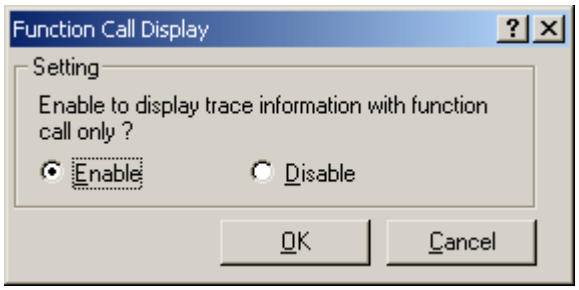


Figure 5.37 [Function Call Display] Dialog Box

[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.

5.7 Analyzing Performance

Use the performance analysis function to measure execution performance. The performance analysis function does not affect the realtime operation because it measures execution performance in the specified range by using the on-chip circuit for performance measurement.

Note: The measurement conditions and the number of channels differ depending on the product. Refer to the E10A-USB emulator's additional document that is specific to the MCU/MPU you are using.

5.7.1 Opening the [Performance Analysis] Window

Choose [View -> Performance -> Performance Analysis] or click the [PA] toolbar button ( ) to open the [Select Performance Analysis Type] dialog box.

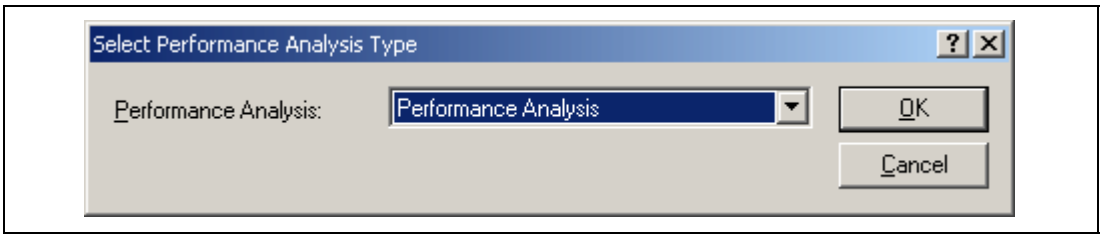


Figure 5.38 [Select Performance Analysis Type] Window

Click the [OK] button to open the [Performance Analysis] window.

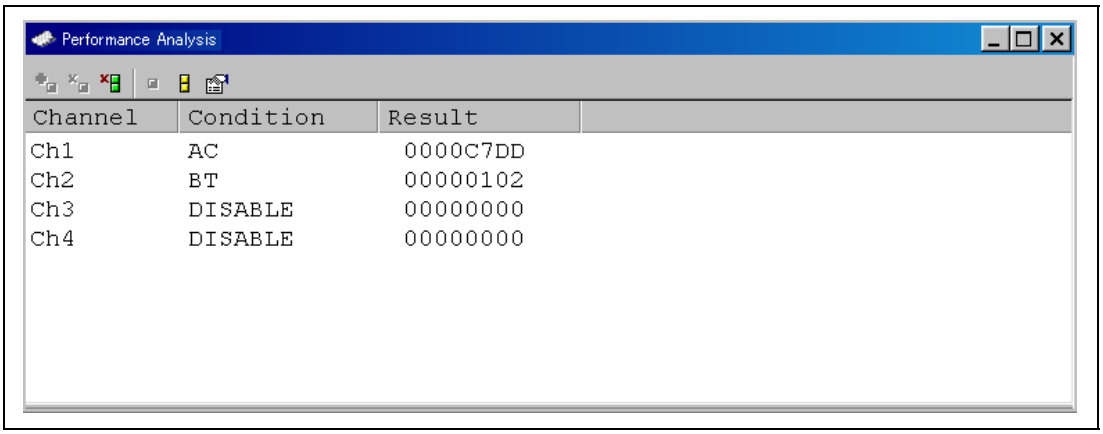


Figure 5.39 [Performance Analysis] Window

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.

5.7.2 Setting Conditions for Measurement

Conditions for measurement can be displayed and changed in the [Performance Analysis] window. Select a point where a condition is to be set, and then select [Set…] from the popup menu to display the [Performance Analysis Properties] dialog box.

5.7.3 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.

5.7.4 Deleting a Measurement Condition

Select [Reset] from the popup menu with a measurement condition selected to delete the condition.

5.7.5 Deleting All Measurement Conditions

Choose [Reset All] from the popup menu to delete all the conditions that have been set.

5.8 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.

5.8.1 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.

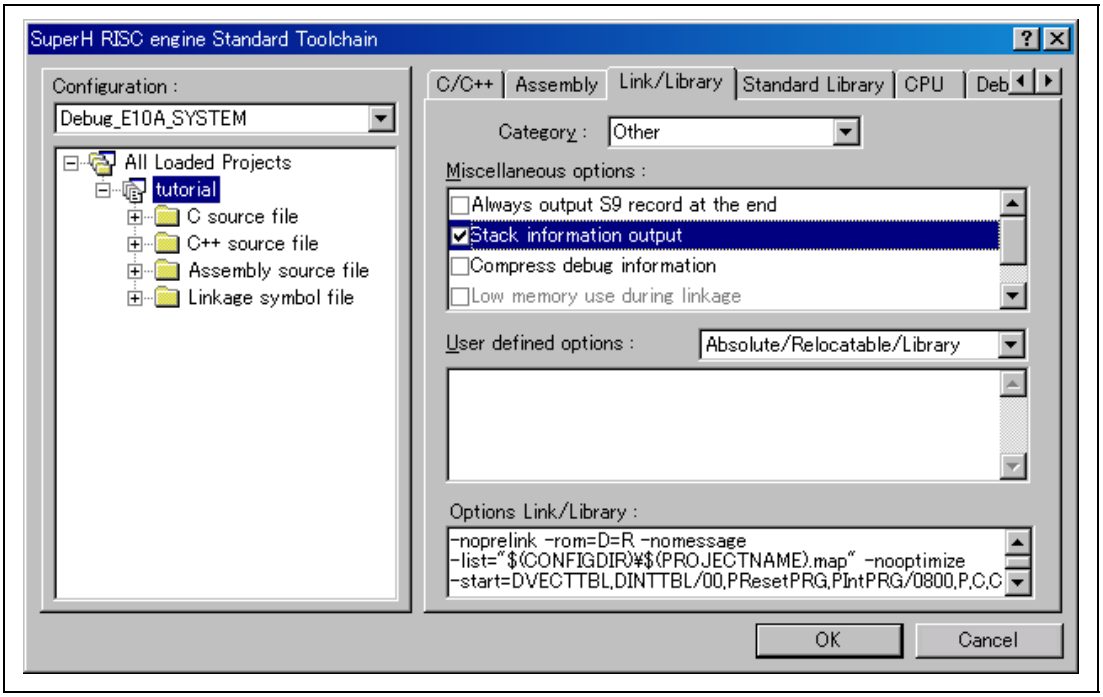


Figure 5.40 [Standard Toolchain] Dialog Box (1)

5.8.2 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.

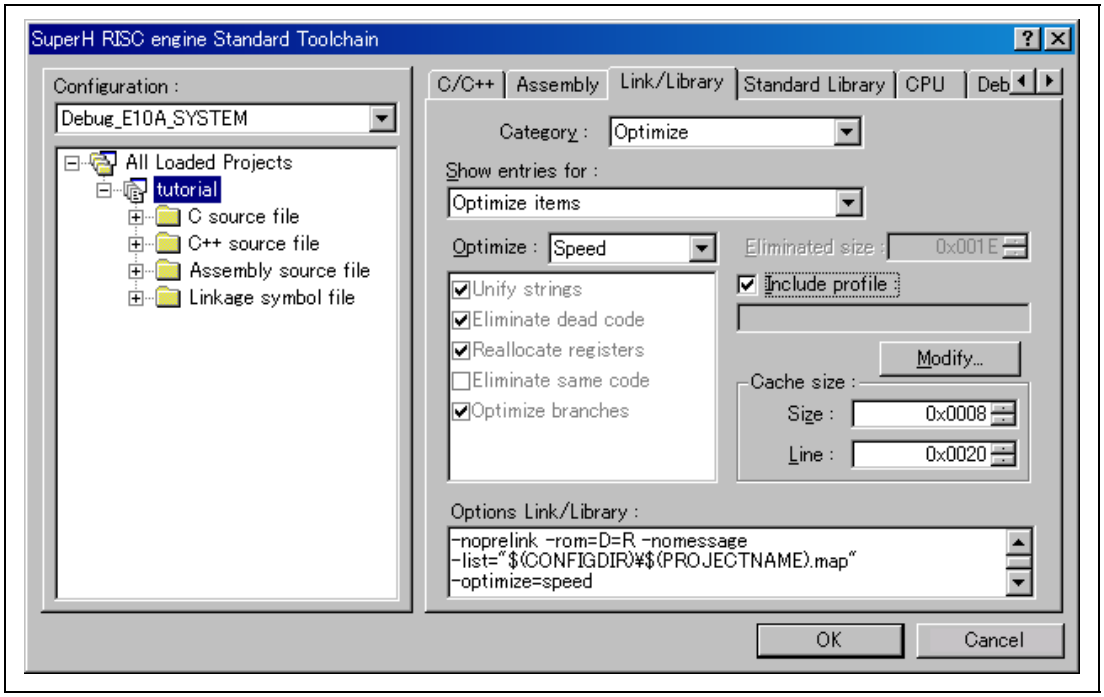


Figure 5.41 [Standard Toolchain] Dialog Box (2)

To enable the settings in the [Include profile] box, specify the [Optimize] list box as some setting other than [None].

5.8.3 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] sheet of the [Options] dialog box (figure 5.42) that can be opened by choosing [Setup ->Options…] from the main menu.

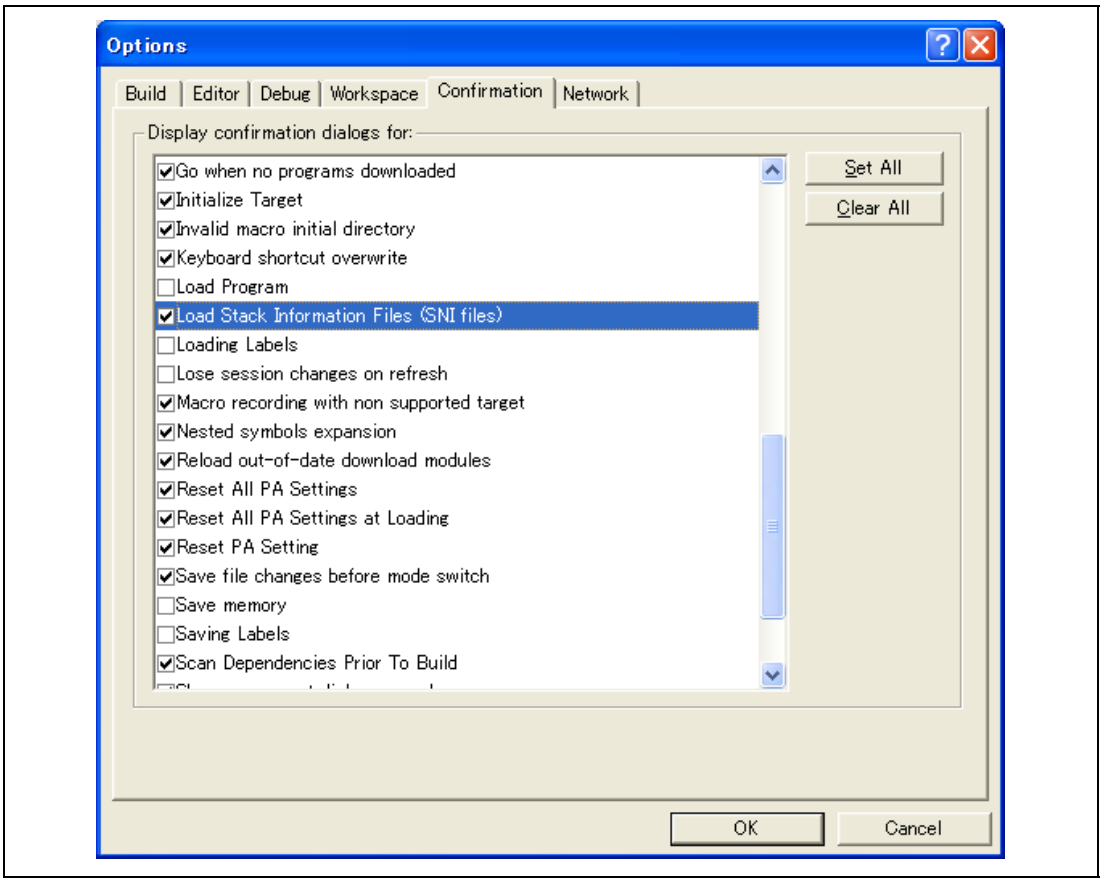


Figure 5.42 [Options] Dialog Box

5.8.4 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.

5.8.5 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.

5.8.6 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.

5.8.7 [List] Sheet

This sheet lists functions and global variables and displays the profile data for each function and variable.

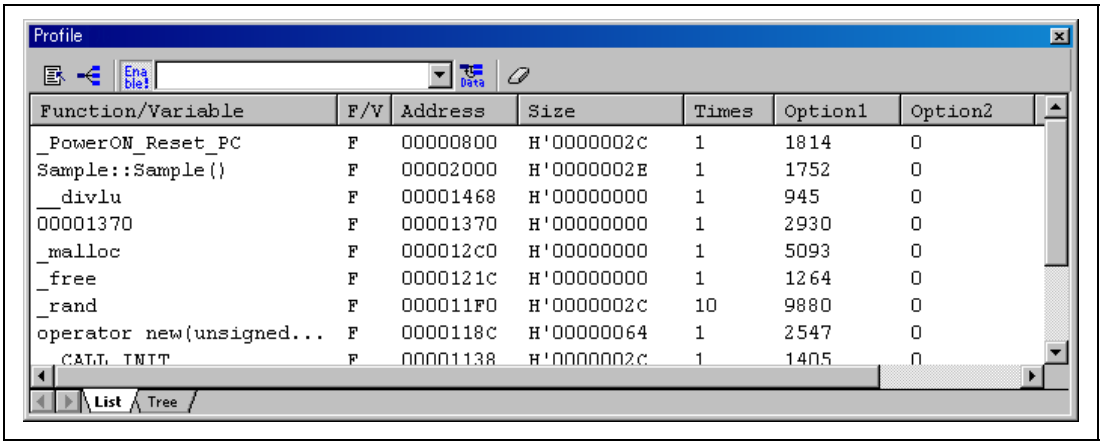


Figure 5.43 [List] Sheet

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 5.8.8, [Tree] Sheet.

5.8.8 [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.

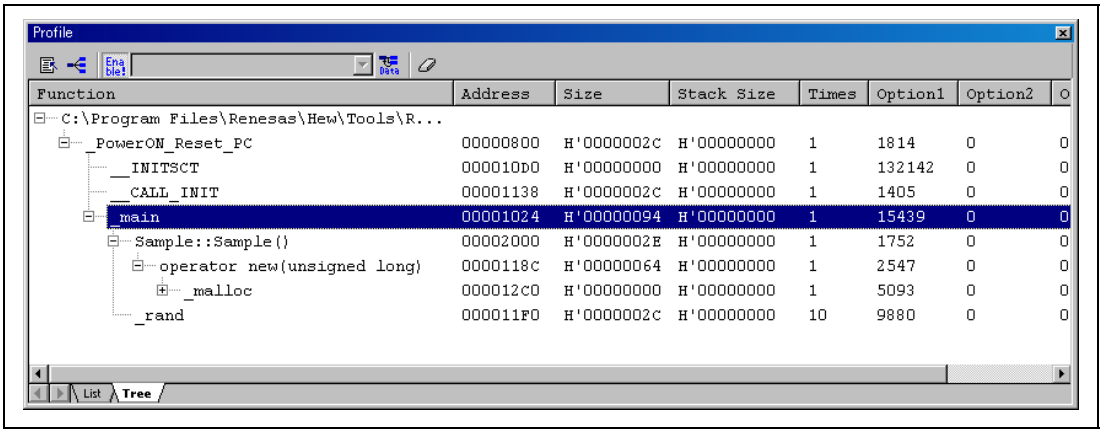


Figure 5.44 [Tree] Sheet

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.

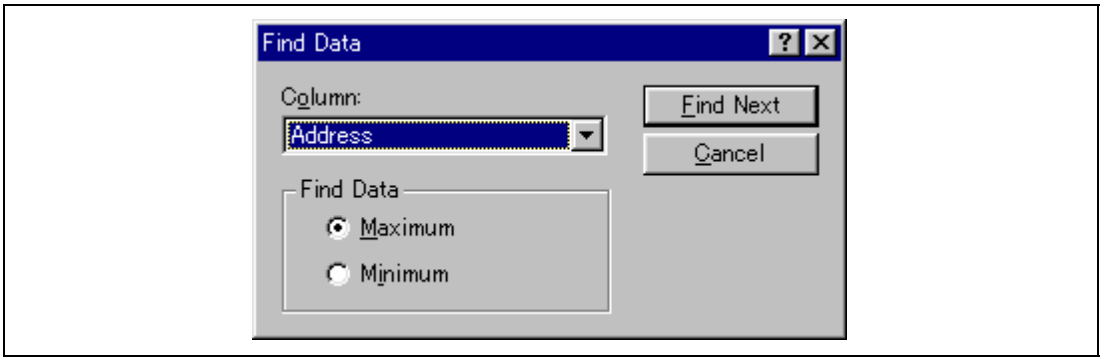


Figure 5.45 [Find Data] Dialog Box

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.

5.8.9 [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.

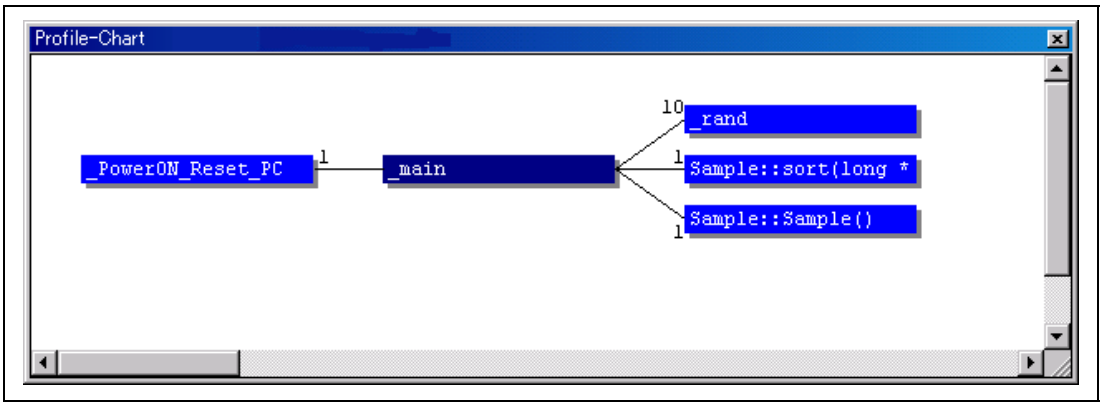


Figure 5.46 [Profile-Chart] Window

5.8.10 Types and Purposes of Displayed Data

The profile 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 are using a microcomputer which incorporates a cache memory, more of the cache memory will need to be updated when you execute larger functions. 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.

5.8.11 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.

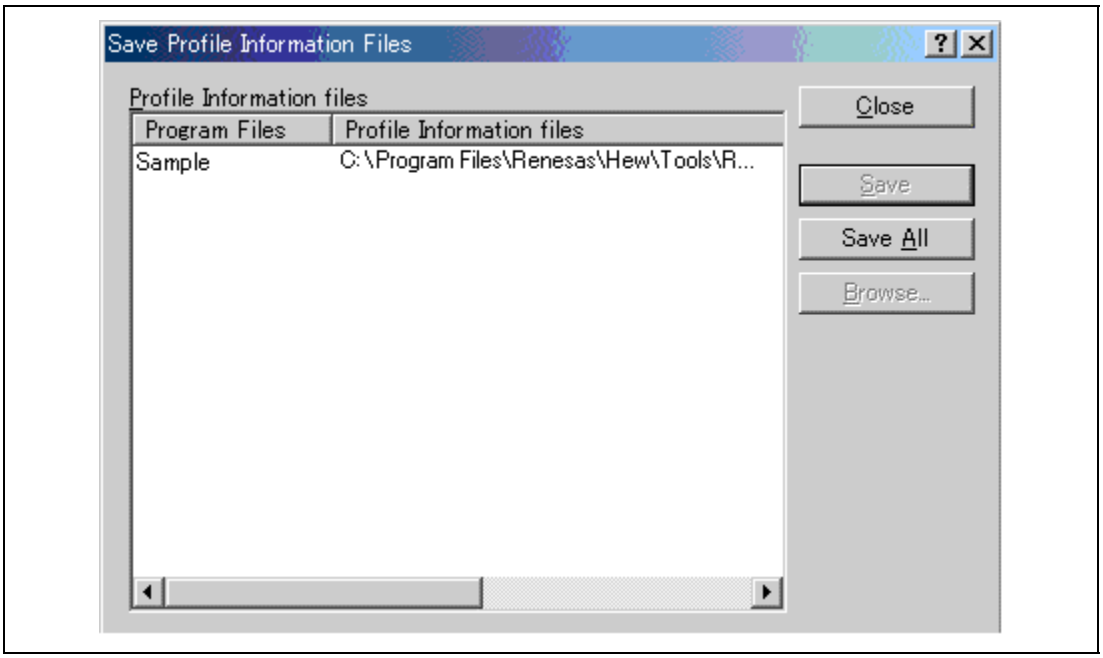


Figure 5.47 [Save Profile Information Files] Dialog Box

5.8.12 Notes

1. Tolerances

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

   Since the function also counts when the measured item is generated at break or re-execution, 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 profile function is being used

   (a) Performance measurement function

   The profile function is implemented by using the performance measurement function described in section 2.4, Performance Measurement Function. This function cannot be used when the profile function is enabled.

   (b) Step function

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

   (c) Internal trace function

   When the profile 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 profile function is enabled.

   (d) Continuous trace function (only for the supported devices)

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

   (e) Halt function

   When the profile function is enabled, do not use the halt function of the internal and AUD traces.

   (f) Memory access during user program execution

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

   (g) When the profile 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 [Trace] window when the execution is returned to the user program from the emulator firmware. In this case, **EML** is displayed.

   (h) When the profile function is enabled, do not use the function of Event Condition 3.

3. Others

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

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

Environment:

Host computer: 3.00 GHz (Pentium® 4)
Memory: 1.00 Gbyte
SH72633: 5 MHz (TCK clock)
OS: Windows® XP
Execution program: 10,000 nested calls

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

(ii) When the profile function is used in the setting without including a child function: 17 seconds

(iii) When the profile function is used in the setting including a child function: 20 seconds

5.9 Using Multiple Debugging Platforms

Multiple debugging platforms can be operated at the same time in the 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.

5.9.1 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 5.48 Message for Driver Selection

Click the [OK] button. The following dialog box will appear.

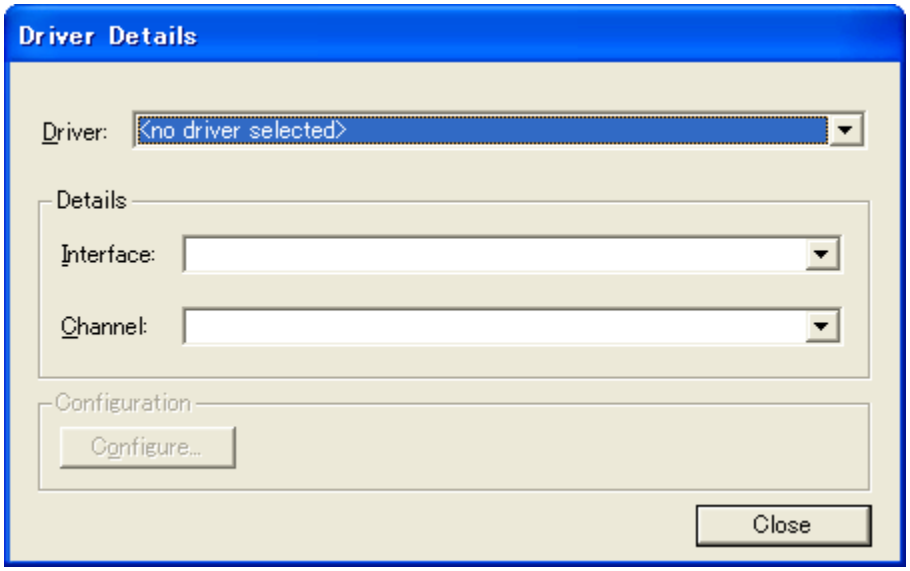


Figure 5.49 [Driver Details] Dialog Box (1)

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 5.50.

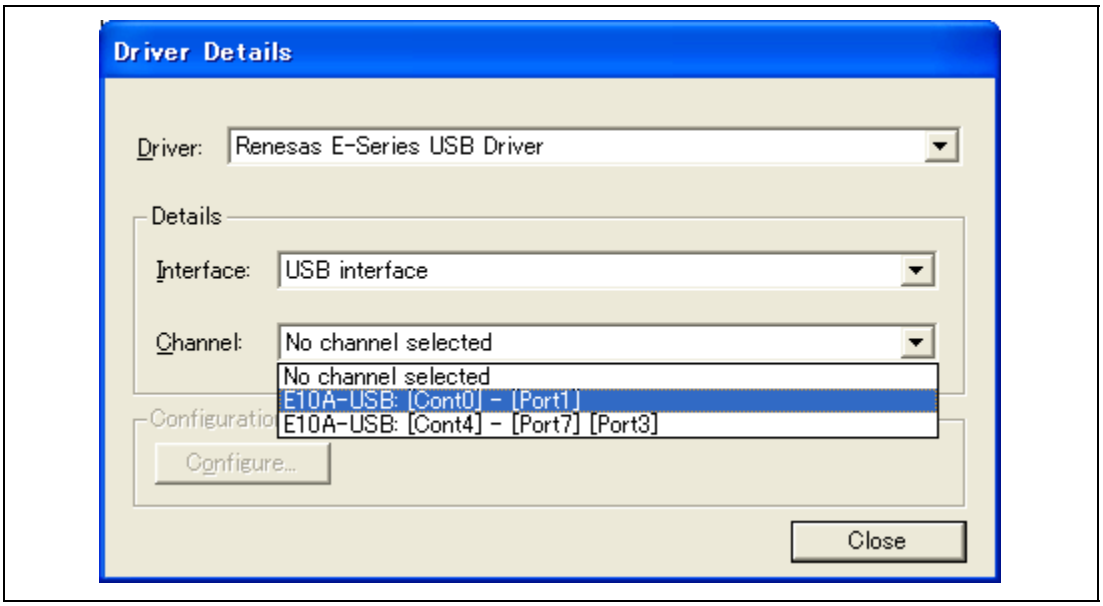


Figure 5.50 [Driver Details] Dialog Box (2)

Information displayed in figure 5.50 is the information of the USB connector to which the emulators are connected.

Note: The displayed character strings of the information differ according to the host computer's environment.

Check which of the character strings of information show emulators.

Select [<no driver selected>] in the [Driver] drop-down list box and disconnect one emulator from the USB connector. After that, select [Renesas E-Series USB Driver] in the [Driver] drop-down list box. Only information on the USB connector that the emulator is connected to is displayed in the [Channel] drop-down list box.

The above procedures are used to discern which of the emulators are indicated by the displayed character strings of information in the [Channel] drop-down list box.

Examples

E10A-USB: [Cont0] – [Port1]:
The emulator is connected to port 1 of USB controller 0.

E10A-USB: [Cont4] – [Port7][Port3]:
The emulator is connected to general hub port 3 connected to port 7 of USB controller 4 (only selectable when a USB hub is connected).

5.10 Start/Stop Function

The start/stop function is useful if you wish to control a user system in synchronization with starting and stopping of user program execution.

To open the [Start/Stop Function Setting] dialog box, select [Setup -> Emulator -> Start/Stop function setting…] or click on the [Start/Stop Function Setting dialog box] toolbar button.

Figure 5.51 [Start/Stop Function Setting dialog box] Toolbar Button

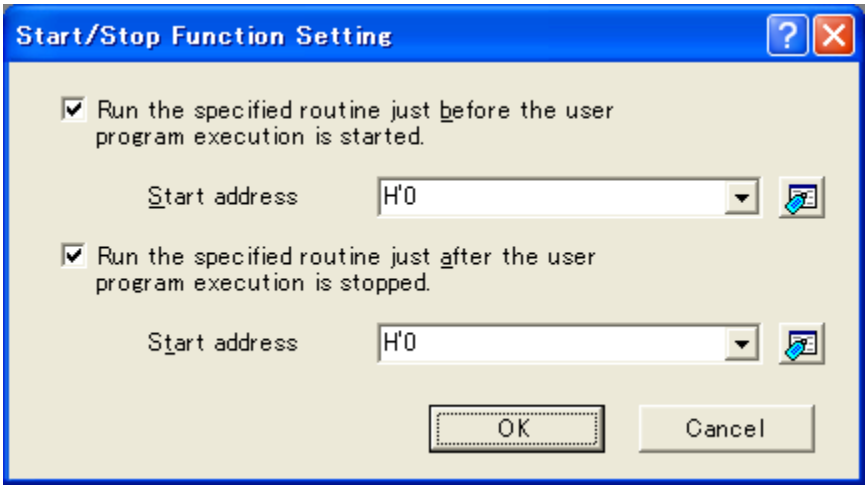


Figure 5.52 [Start/Stop Function Setting] Dialog Box

The [Start/Stop Function Setting] dialog box contains the following options.

[Run the specified routine just before the user program execution is started.] checkbox:

Select this box if you wish to run a user routine just before the user program execution starts.

[Start address]: Specify an address where a routine will start to run just before the user program execution starts. You can also specify a label. Do not specify an odd-numbered address.

[Run the specified routine just after the user program execution is stopped.] checkbox:

Select this box if you wish to run a user routine just after the user program execution stops.

[Start address]: Specify an address where a routine will start to run just after the user program execution stops. You can also specify a label. Do not specify an odd-numbered address.

Notes: 1. The start/stop function is not supported for all products. For details on the specifications of each product, refer to the online help.
2. For details, refer to the START_FUNCTION_BEFORE and STOP_FUNCTION_AFTER command pages in the online help information.
3. The start/stop function is subject to the following limitations.
   - An RTS instruction must be written at the end of the user routine.
   - The user routine must end.
   - Ensure that no break occurs within the user routine.
   - No interrupts should be accepted in the user routine.
   - The user routine to run before the user program execution starts must end within 2 ms.
   - The user routine to run after the user program execution stops must end within 30 s.
   - User stack should be used by the user routine to run after user program execution.
   - If the user routine is to use a general-purpose register, saving the contents of the register, making the initial setting, and restoring the contents should all be done within the routine.
   - If values are set via the [Registers] window of the HEW just before user program execution starts, these values will not be reflected to the user routine.
   - If values of general-purpose registers are changed by the user routine, execution of the user program being debugged will not be affected.
   - While profiling is in progress, the user routine (start or stop) runs whenever branch information is acquired.
   - The specified user routine must not include the sleep() intrinsic function in the case of C/C++ or the SLEEP instruction in the case of assembly.

- The specified user routine must not include division or modular multiplication in the case of C/C++ or DIVS or DIVU instructions in the case of assembly.
- The specified user routine must not include interrupt functions declared by #pragma interrupt in the case of C/C++ or the RESBANK instruction in the case of assembly.
- Ensure that FPU exceptions do not occur within the user routine.

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 Elf/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).
4. When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP) and re-compile the program. The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
5. The displayed addresses and data may differ from those given in this section depending on the MCU/MPU to be used.

6.2 Running the High-performance Embedded Workshop

To run the High-performance Embedded Workshop, refer to section 3.11, 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 MCU/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 MCU/MPU's guaranteed TCK range.
  For details of the limitations on both clocks, refer to section 2.2.3, 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 [Setup] menu to set a communication clock. The [Configuration] dialog box is displayed.

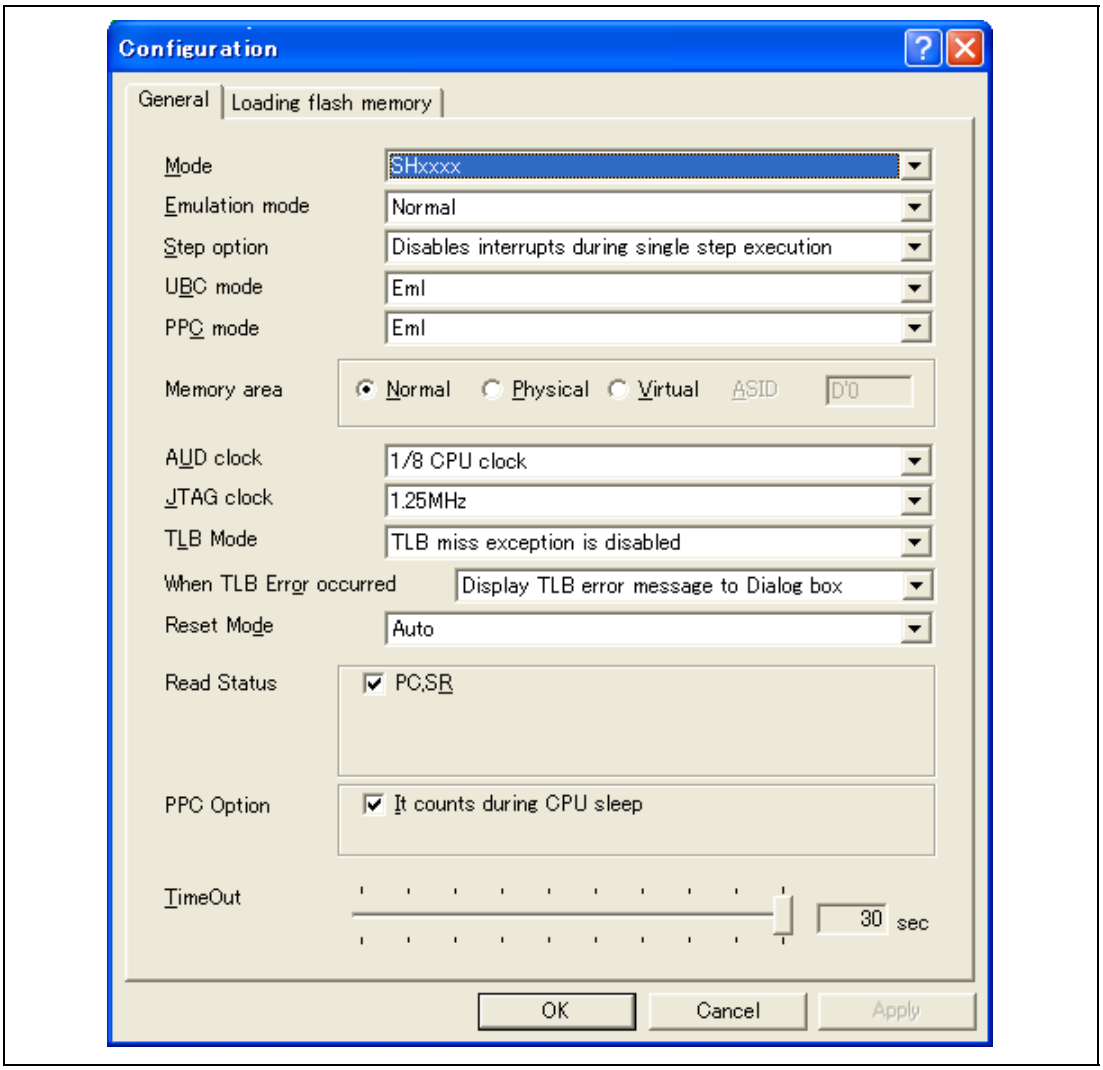


Figure 6.1 [Configuration] Dialog Box

- 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 device 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*** and ***H'000000FF*** in the [Begin] and [End] edit boxes, respectively.

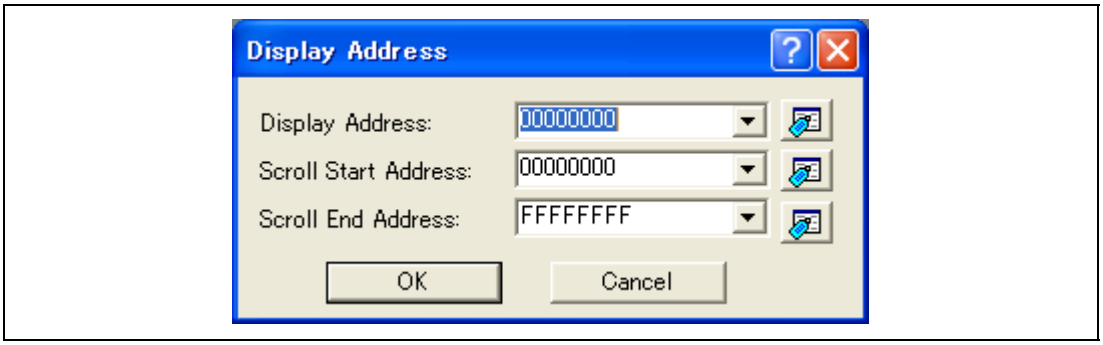


Figure 6.2 [Display Address] Dialog Box

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

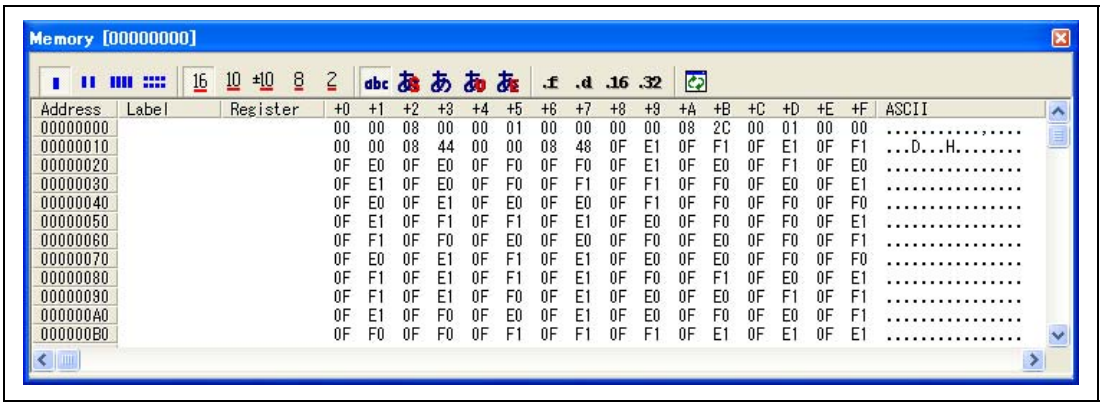


Figure 6.3 [Memory] Window

- 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.

6.6 Downloading the Tutorial Program

6.6.1 Downloading the Tutorial Program

Download the object program to be debugged.

In this emulator, it is enabled to download the program and set the PC breakpoint in the internal flash memory area. For the method to set the PC breakpoint, refer to section 6.17.1, PC Break Function.

- Select [Download module] from [Tutorial.abs] under [Download modules].

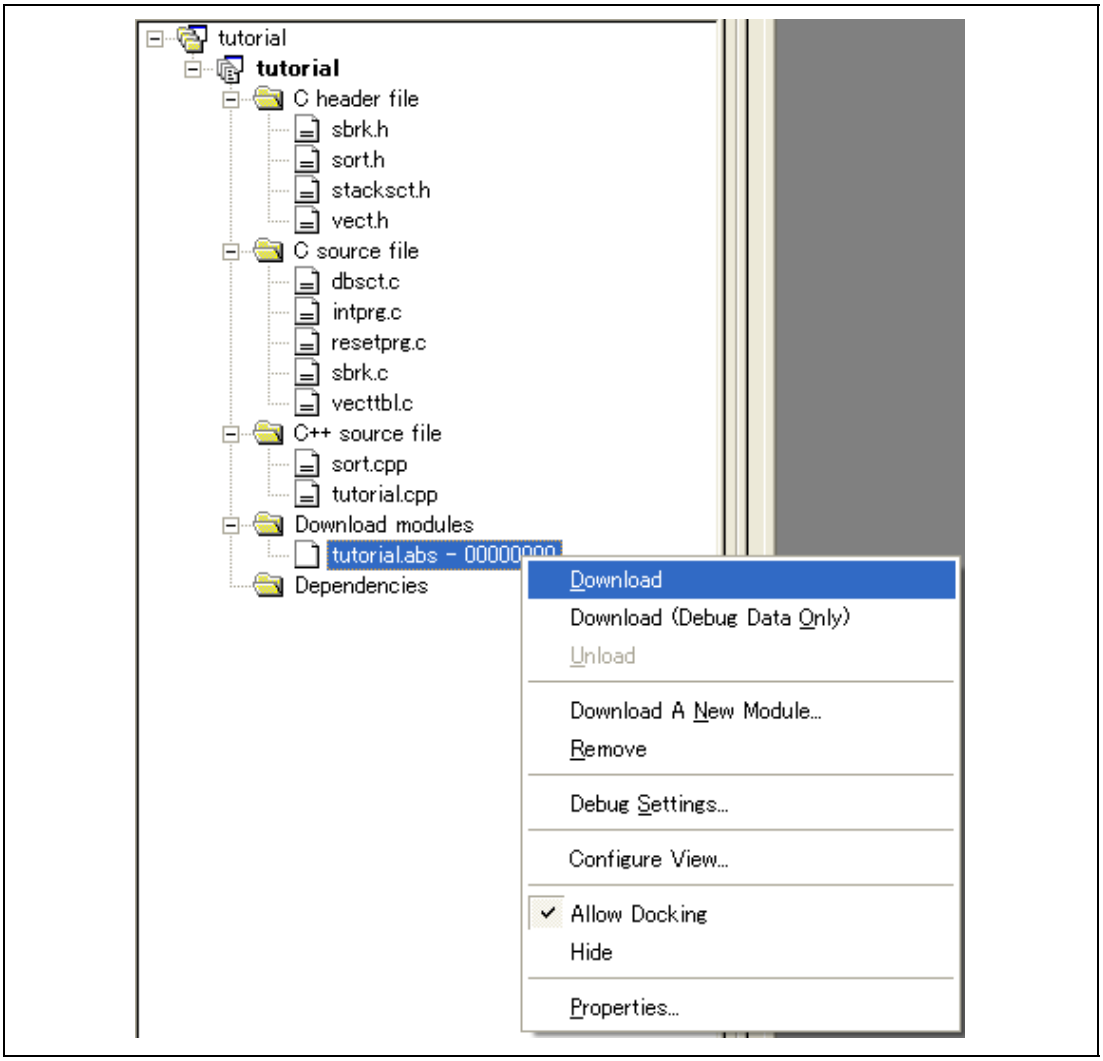


Figure 6.4 Downloading the Tutorial Program

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].

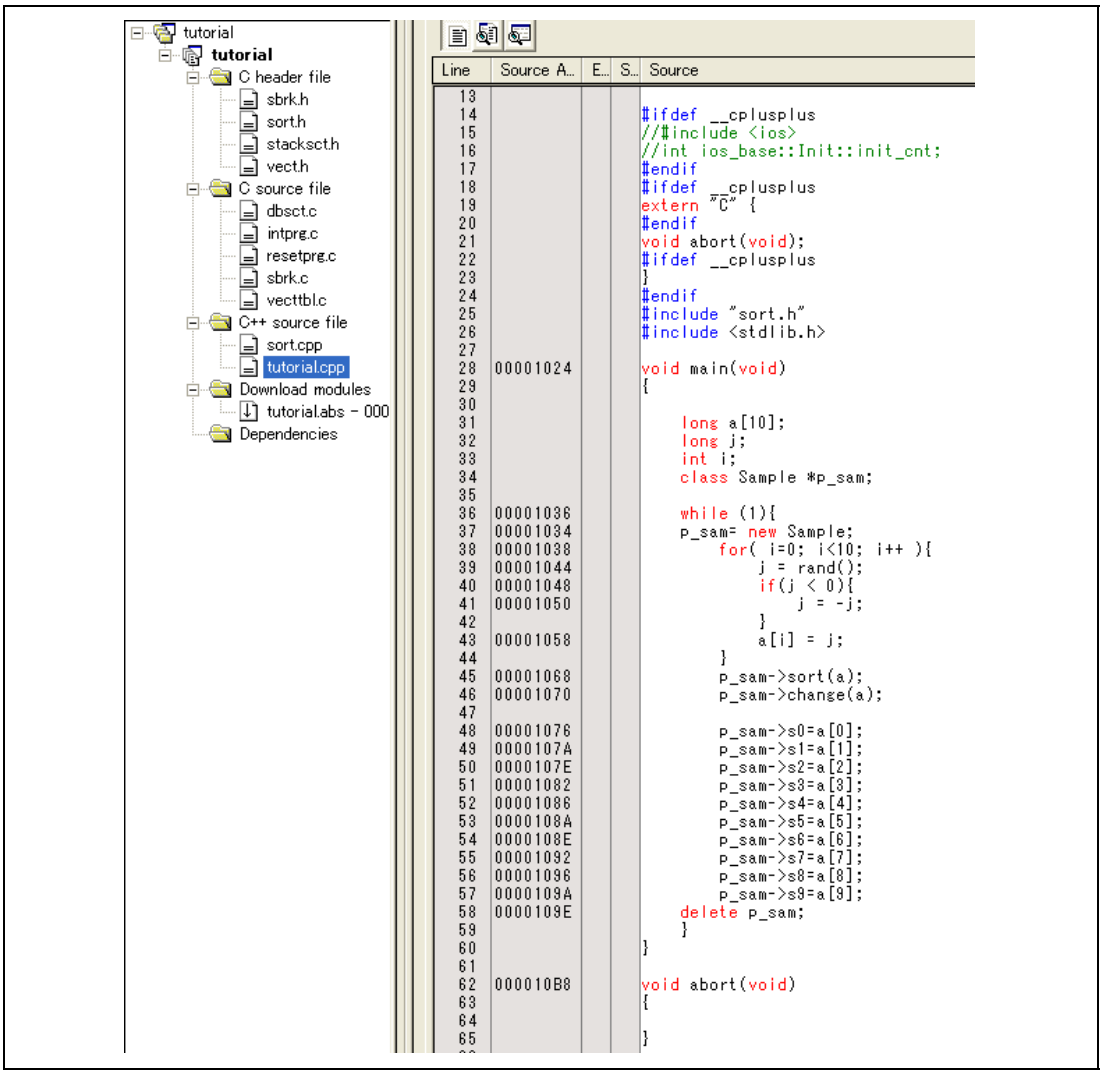


Figure 6.5 [Editor] Window (Displaying the Source Program)

- Select a font and size that are legible from the [Format…] option in the [Setup] menu if necessary.

Initially the [Editor] 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 [Editor] 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 breakpoint] column on the line containing the `sort` function call.

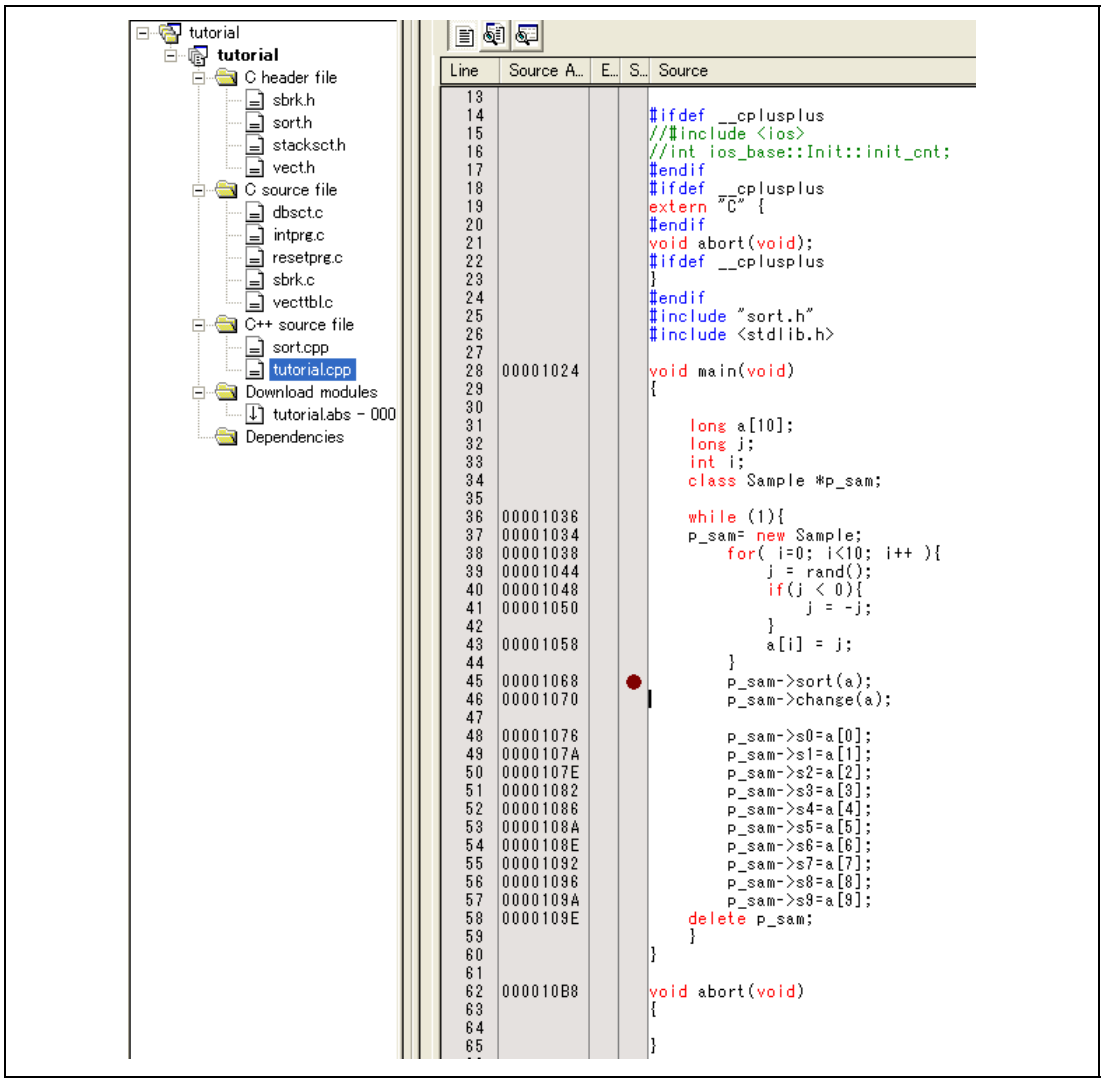


Figure 6.6 [Editor] Window (Setting a PC Breakpoint)

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.

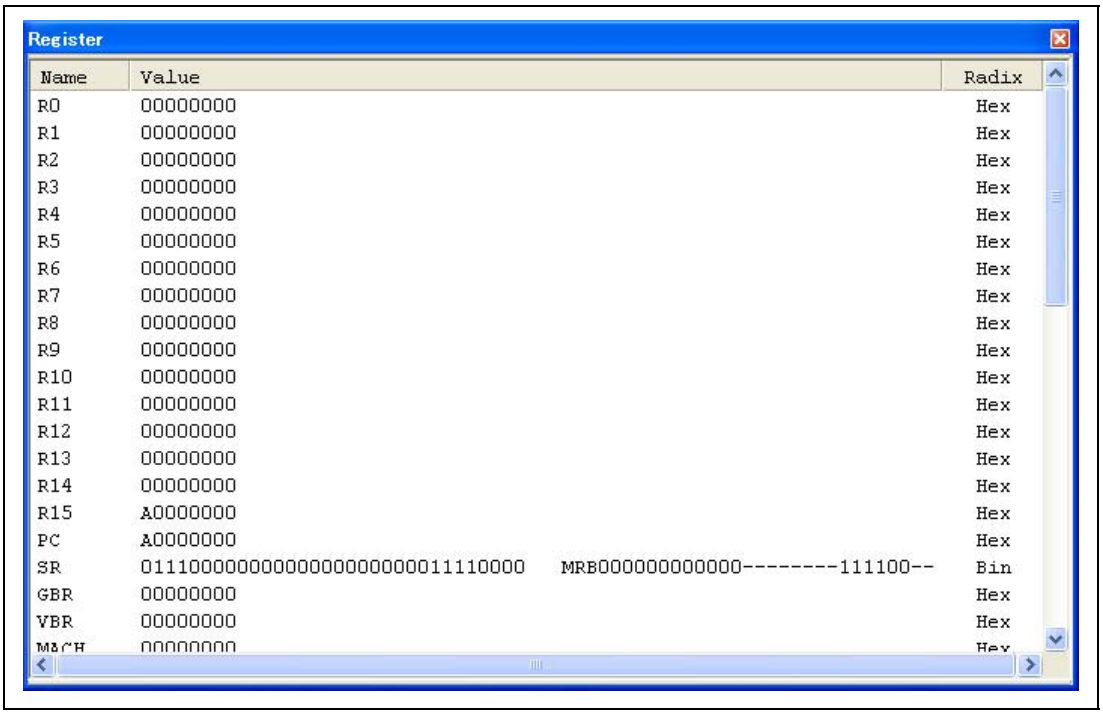


Figure 6.7 [Register] Window

- 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.

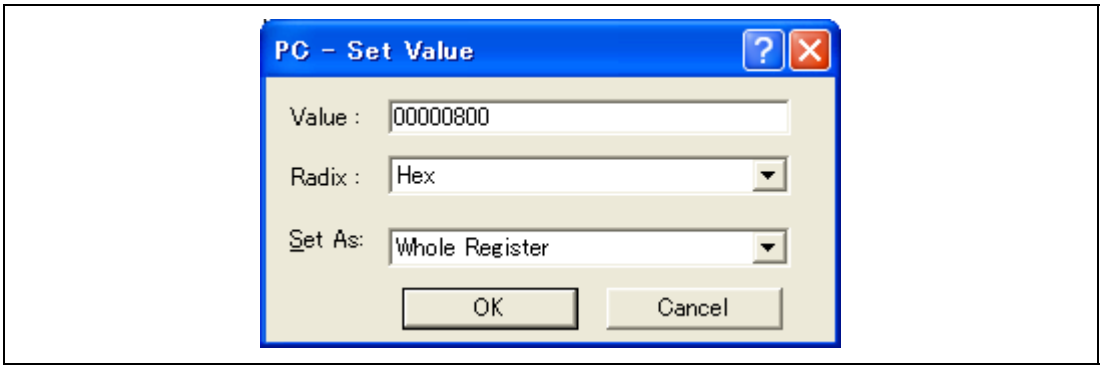


Figure 6.8 [Register] Dialog Box (PC)

- 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.
  When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.

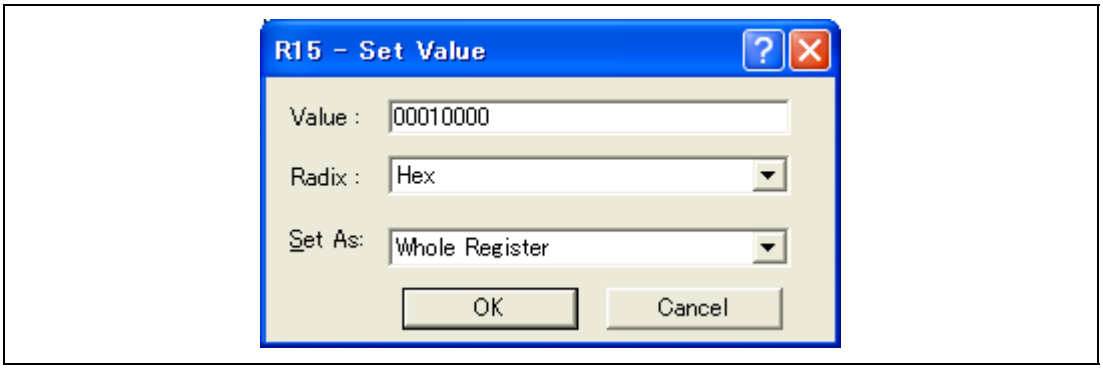


Figure 6.9 [Register] Dialog Box (R15)

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 in the products that support the CPU status acquisition function. The program will be executed up to the breakpoint that has been set, and an arrow will be displayed in the [S/W breakpoint] 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\E10A-USB\xxxx\Tutorial\source.
Here, 'xxxx' means the target product group.

2. If program execution is failed, select [Reset CPU] from the [Debug] menu, reset the device, and restart the procedure from figure 6.8.

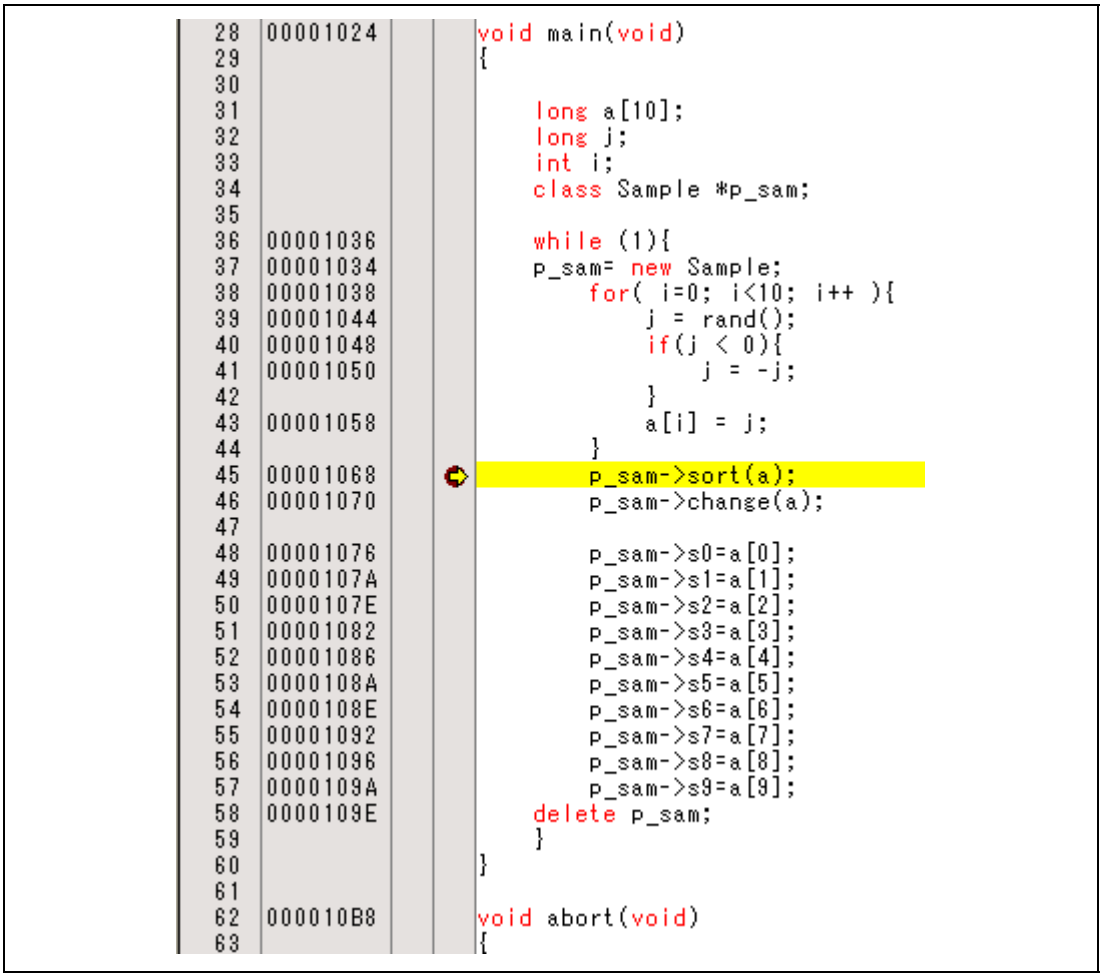


Figure 6.11 [Editor] Window (Break State)

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`.

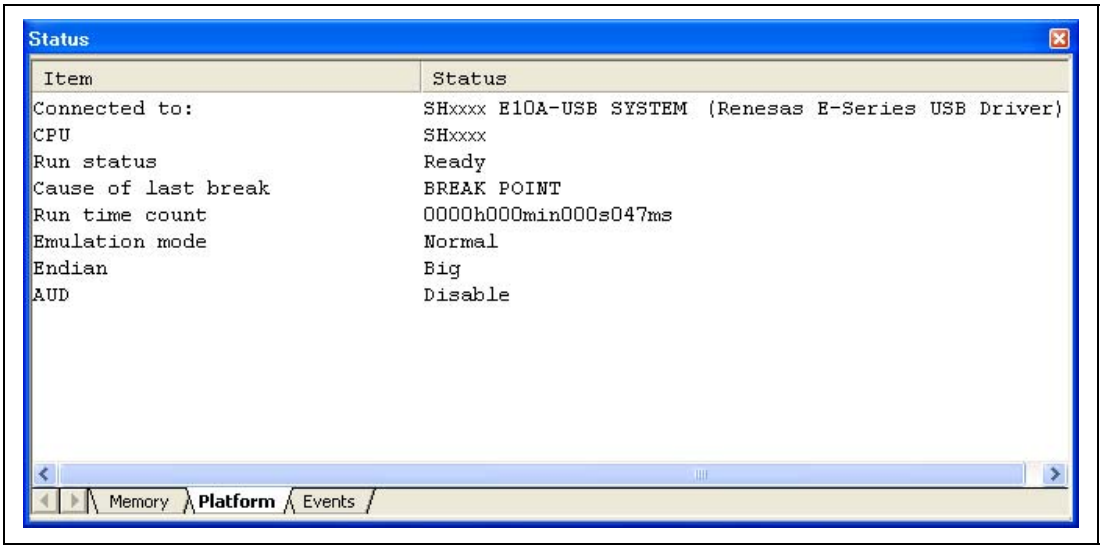


Figure 6.12 [Status] Window

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.

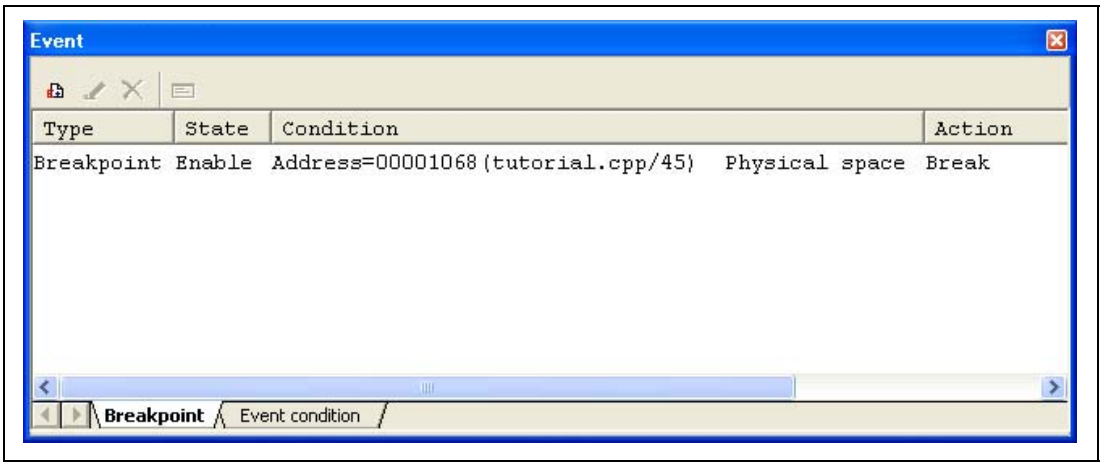


Figure 6.13 [Event] Window

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.

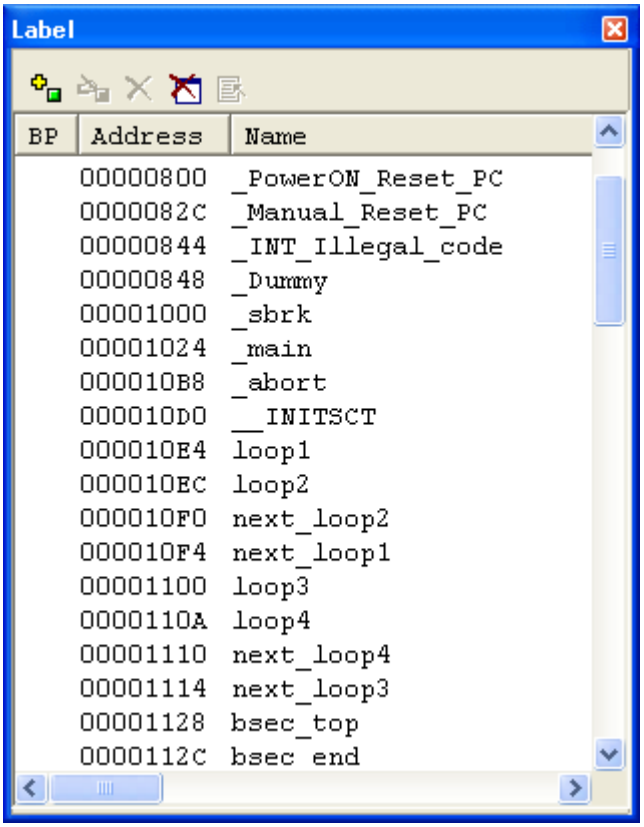


Figure 6.14 [Label] Window

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, ***00000000*** in the [Scroll Start Address] edit box, and ***FFFFFFFF*** in the [Scroll End Address] edit box.

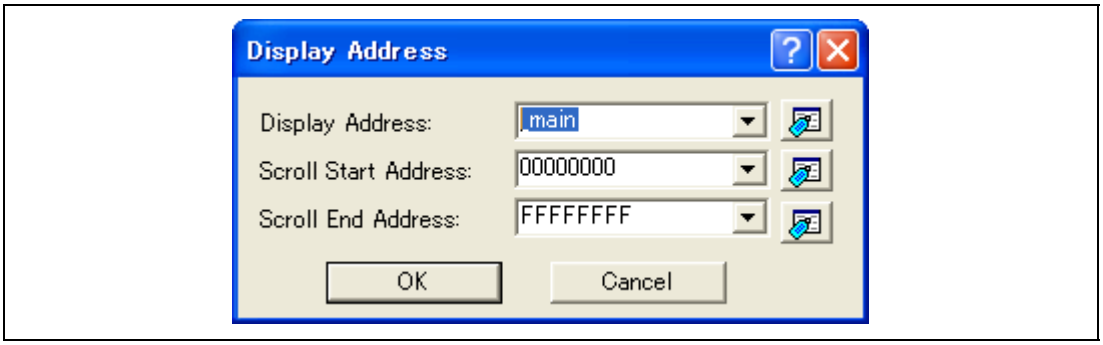


Figure 6.15 [Display Address] Dialog Box

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

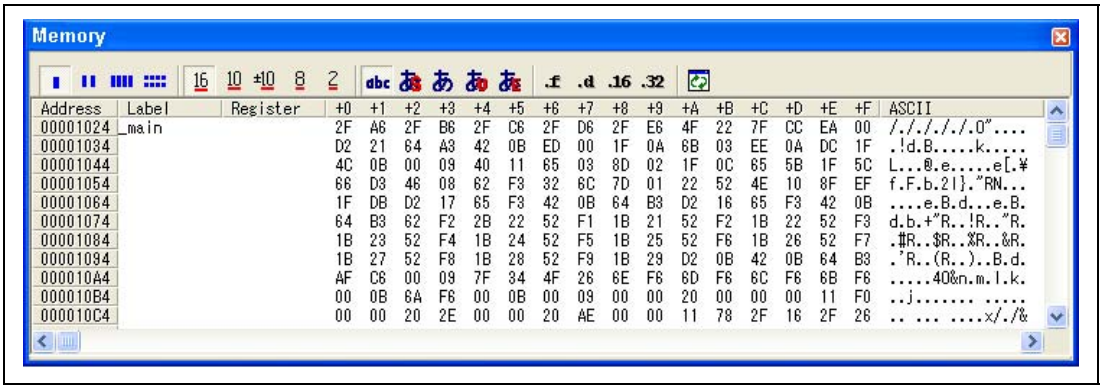


Figure 6.16 [Memory] Window

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.

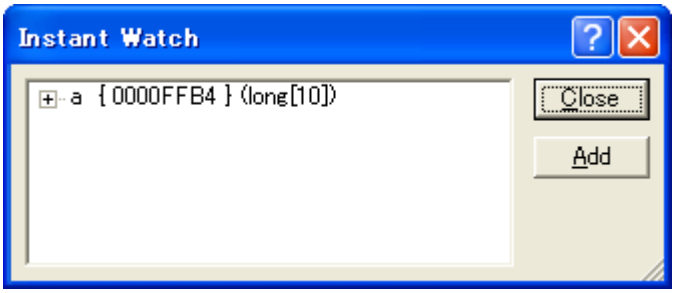


Figure 6.17 [Instant Watch] Dialog Box

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

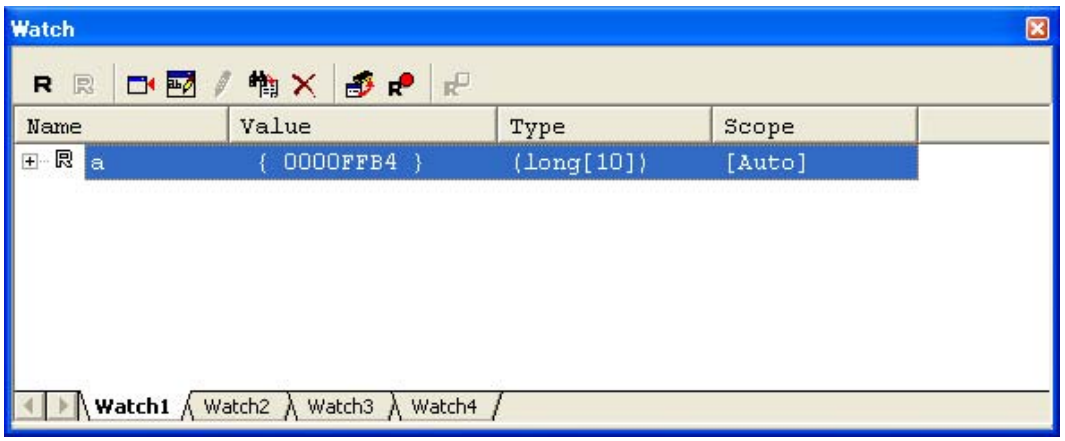


Figure 6.18 [Watch] Window (Displaying the Array)

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`.

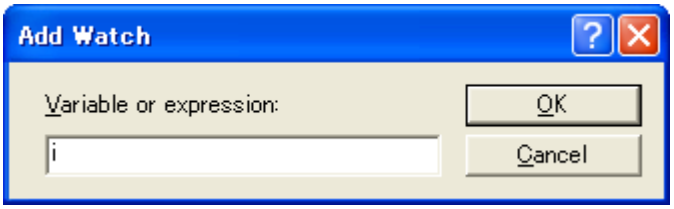


Figure 6.19 [Add Watch] Dialog Box

- Click the [OK] button.

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

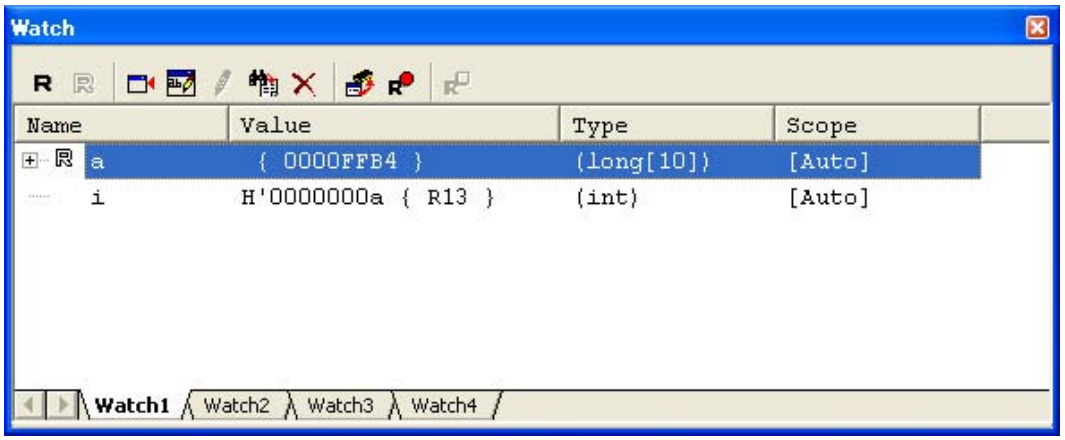


Figure 6.20 [Watch] Window (Displaying the Variable)

The user can click mark '+' at the left side of array a in the [Watch] window to watch all the elements.

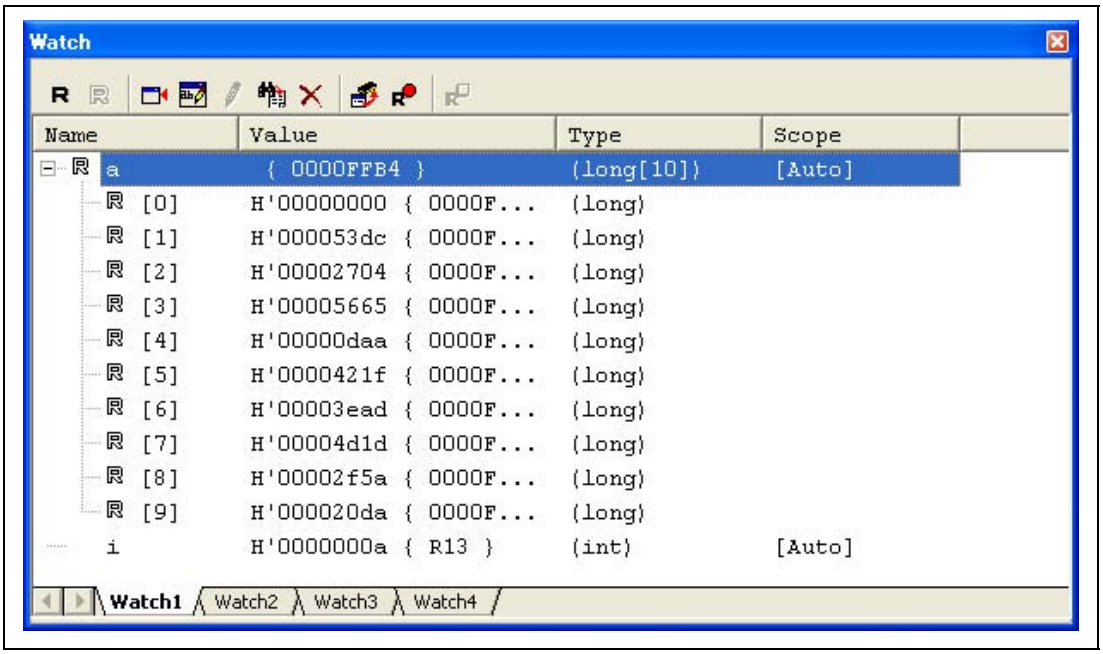


Figure 6.21 [Watch] Window (Displaying Array Elements)

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.

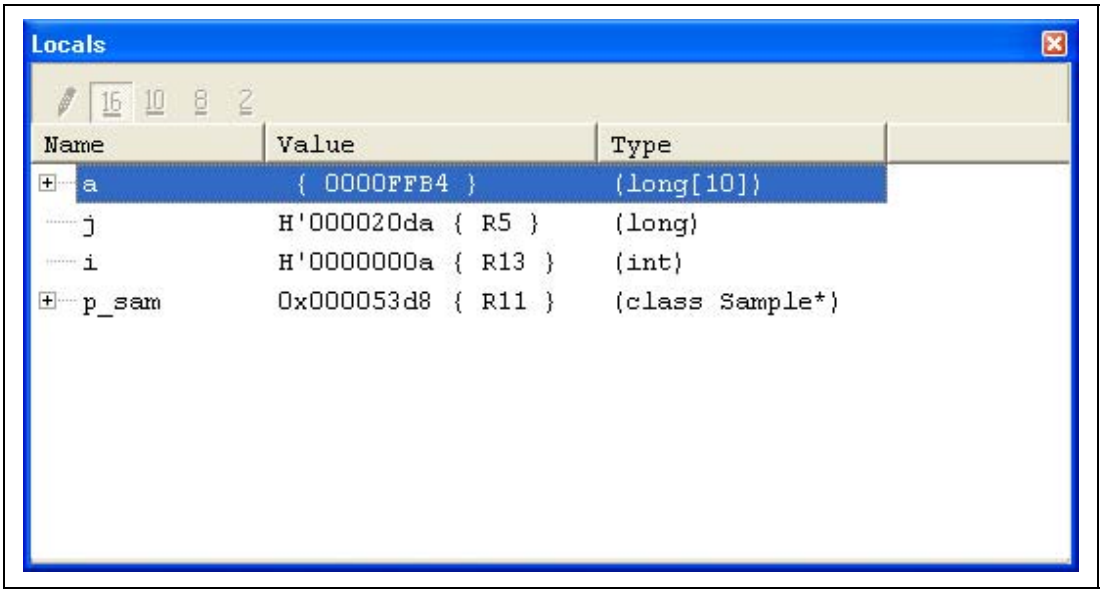


Figure 6.22 [Locals] Window

- 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

```
11                  //-----------------------------------------------------
12  00002000        Sample::Sample()
13  00002002        {
14  00002012            s0=0;
15  00002016            s1=0;
16  00002018            s2=0;
17  0000201A            s3=0;
18  0000201C            s4=0;
19  0000201E            s5=0;
20  00002020            s6=0;
21  00002022            s7=0;
22  00002024            s8=0;
23  00002026            s9=0;
24                  }
25
26  0000202E        void Sample::sort(long *a)
27                  {
28                      long t;
29                      int i, j, k, gap;
30
31  0000203E            gap = 5;
32                      while( gap > 0 ){
33  00002046                for( k=0; k<gap; k++){
34  00002054                    for( i=k+gap; i<10; i=i+gap ){
35  0000205C                        for(j=i-gap; j>=k; j=j-gap){
36  0000206C                            if(a[j]>a[j+gap]){
37                                          t = a[j];
38  00002078                                a[j] = a[j+gap];
39  0000207C                                a[j+gap] = t;
40                                      }
41                                      else
42                                          break;
43                                  }
44                              }
45                          }
46  00002090                gap = gap/2;
47                      }
48                  }
```

Figure 6.24 [Editor] Window (Step In)

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

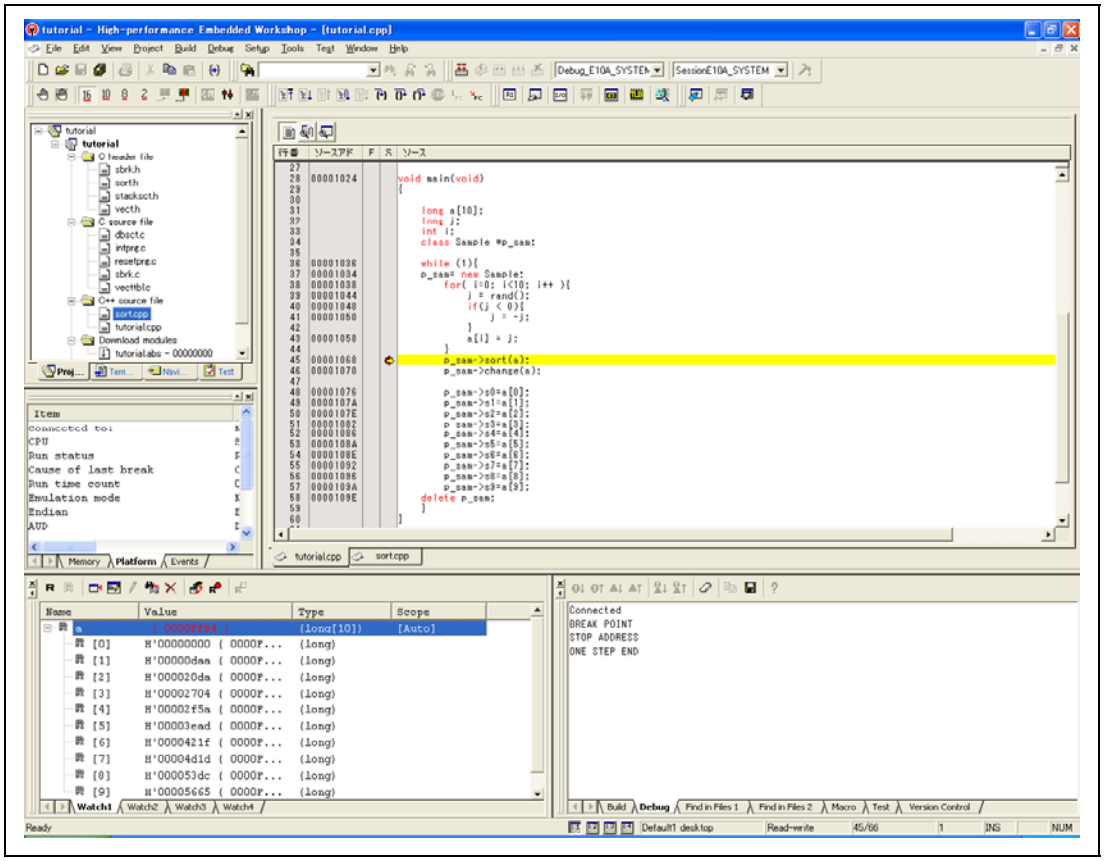


Figure 6.26 [High-performance Embedded Workshop] Window (Step Out)

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

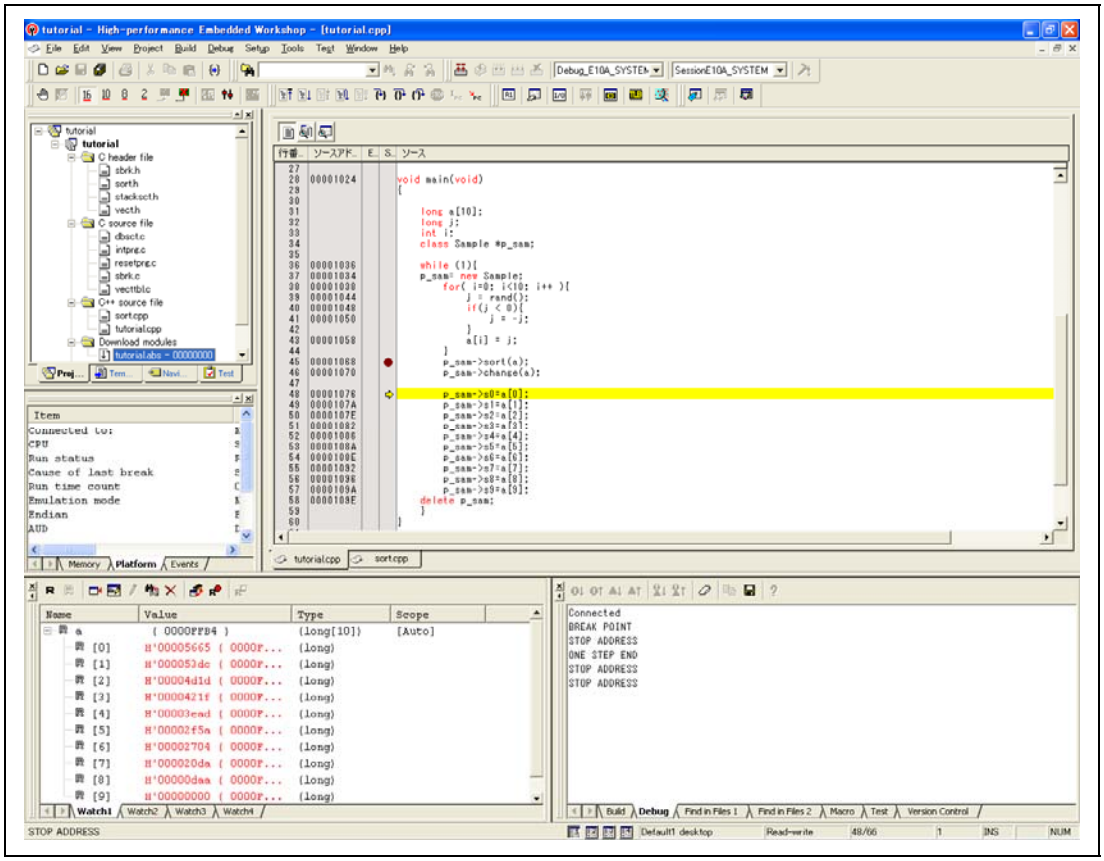


Figure 6.28 [High-performance Embedded Workshop] Window (Step Over)

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 and hardware break functions. With the High-performance Embedded Workshop, a PC breakpoint can be set using the [Breakpoint] sheet of the [Event] window, and a hardware break condition can be set using the [Event condition] sheet.

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

6.17.1 PC Break Function

The emulator can set up to 255 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.

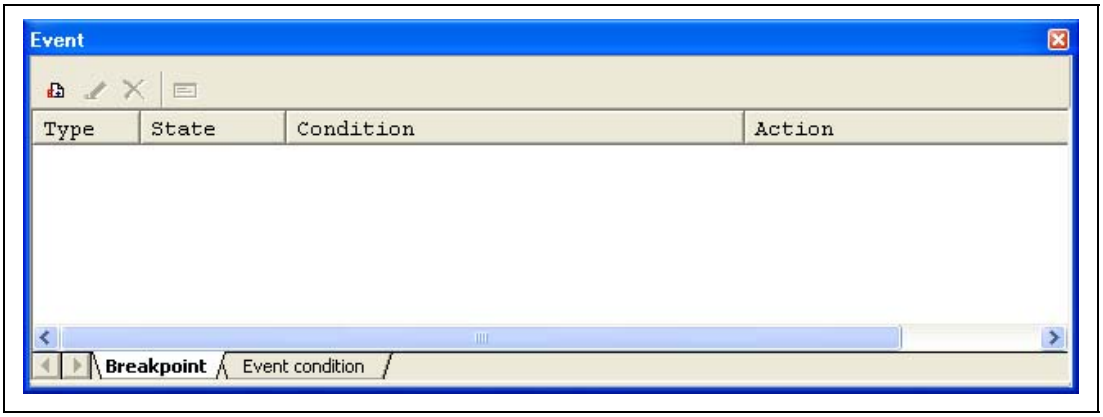


Figure 6.31 [Event] Window (Before PC Breakpoint Setting)

- Click the [Event] window with the right-hand mouse button and select [Add…] from the popup menu.
- Enter ***H'00001076*** in the [Address] edit box.

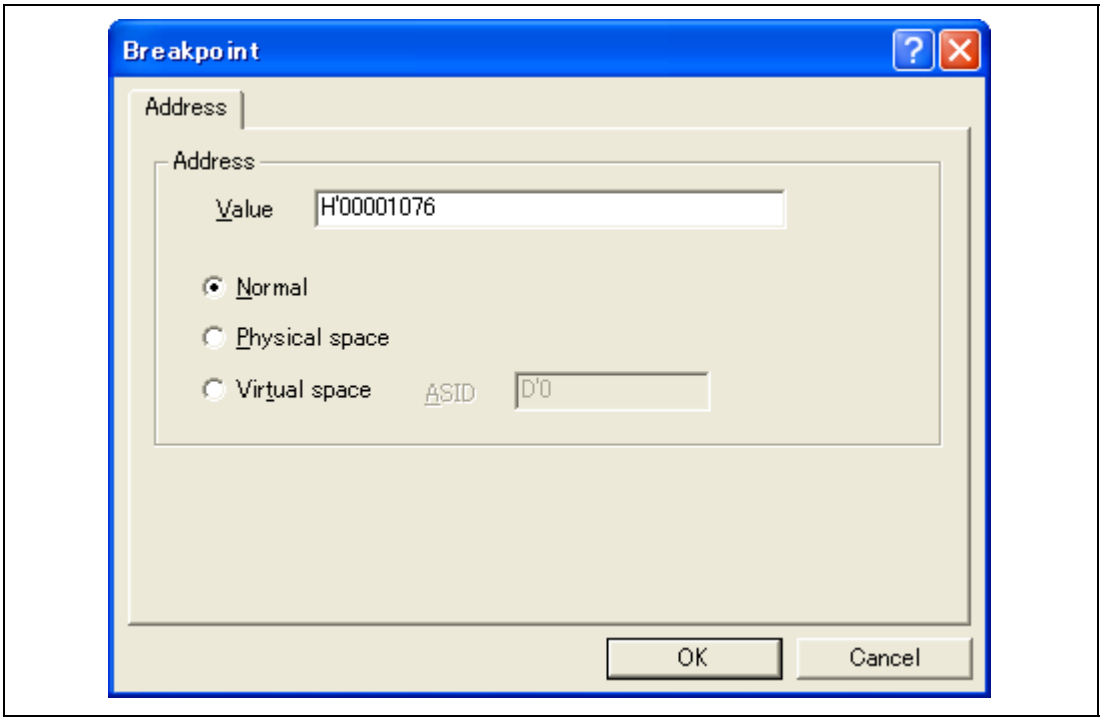


Figure 6.32 [Breakpoint] Dialog Box

Notes: 1. This dialog box differs according to the product. For the items of each product, refer to the online help.

2. For some MCUs/MPUs, the address value to be entered in the [Address] edit box is not ***H'00001076***. In such cases, specify the address where "p_sam->s0=a[0];" (tutorial.cpp/48) is located.

- Click the [OK] button.

The PC breakpoint that has been set is displayed in the [Event] window.

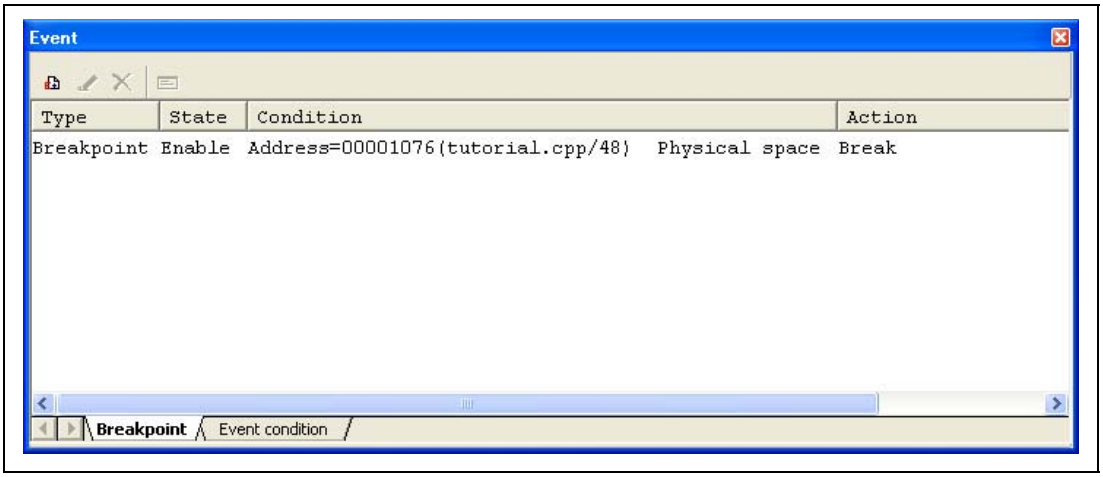


Figure 6.33 [Event] Window (PC Breakpoint Setting)

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. When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
- If program execution is failed, reset the device and execute again the procedures above.

The program runs, and stops at the set PC breakpoint.

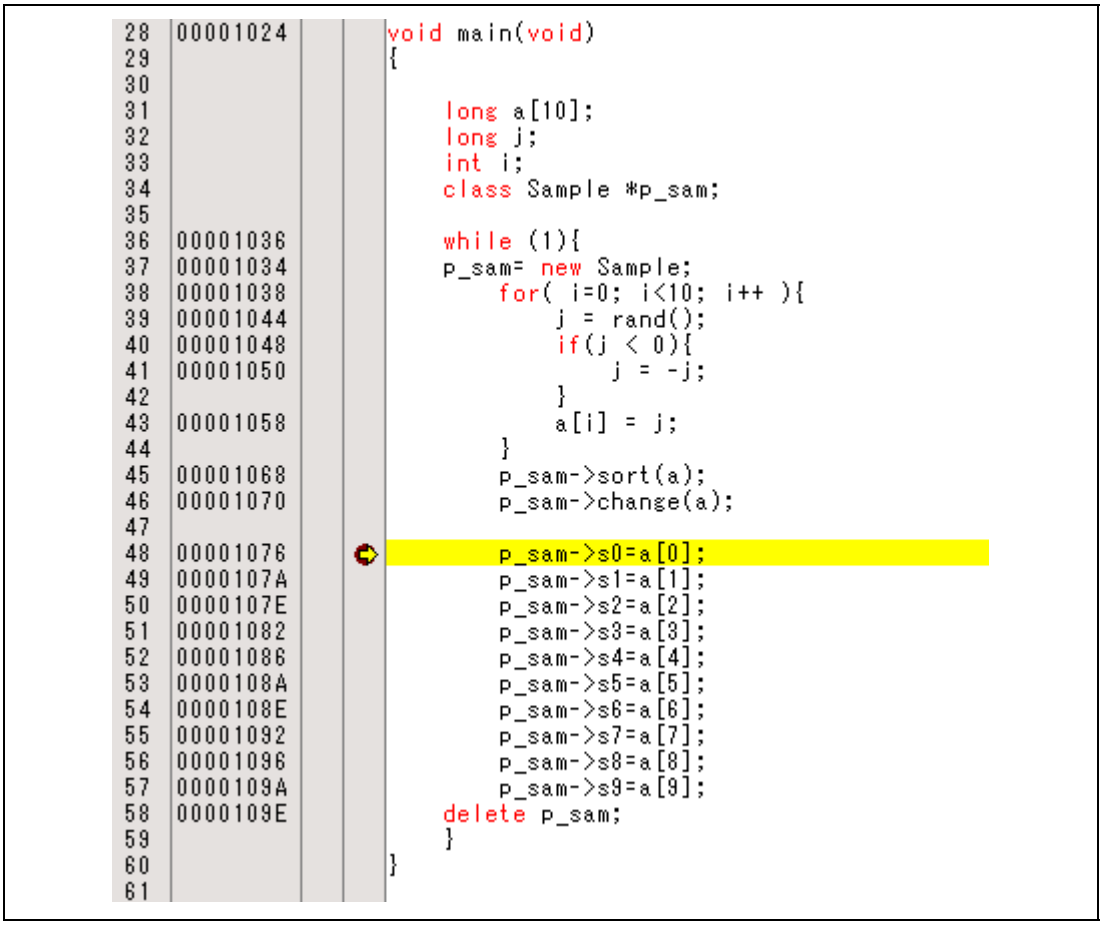


Figure 6.34 [Editor] Window at Execution Stop (PC Break)

The [Status] window displays the following contents.

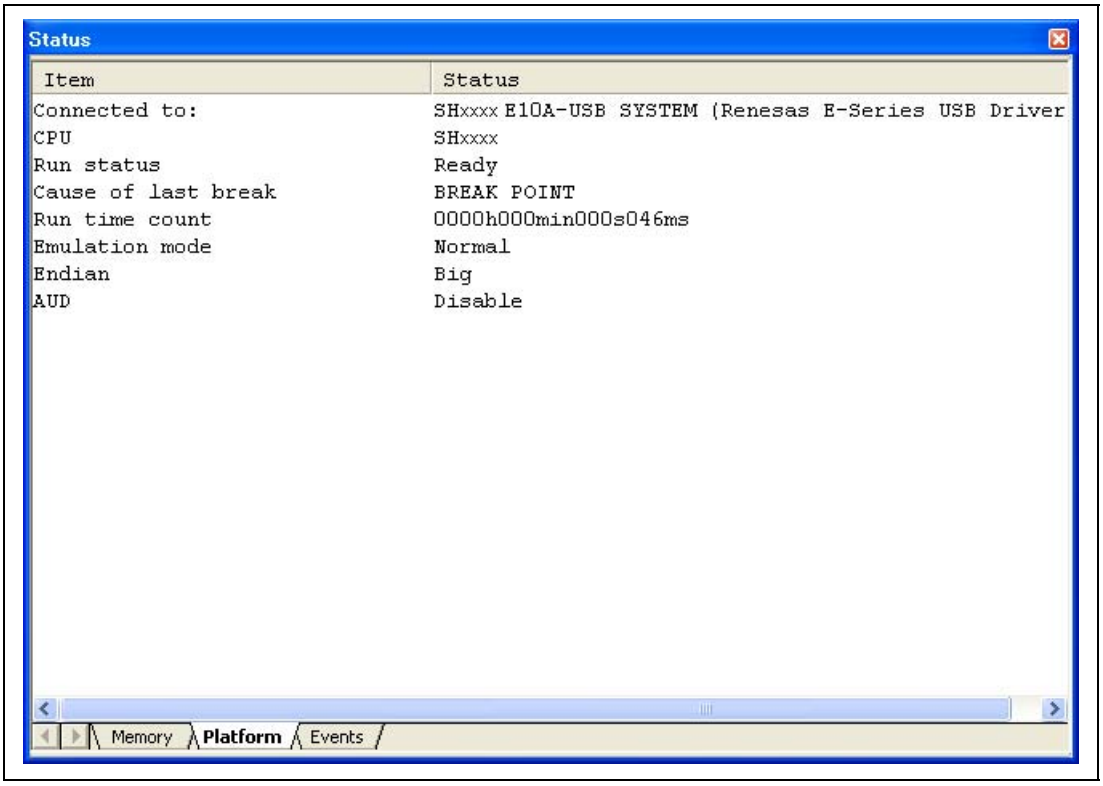


Figure 6.35 Displayed Contents of the [Status] Window (PC Break)

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 Hardware Break Function

Hardware break settings vary with the product.

A method is given below in which the address bus condition is set under an event condition “Ch1(IA_OA)”.

- 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 Ch1(IA_OA), click the [Event condition] tab.

A hardware break condition can be set independently for each of the event channels. In this example, set “Ch1(IA_OA)”.

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

- Select a line of Ch1(IA_OA) in the [Event] window. When highlighted, double-click this line.

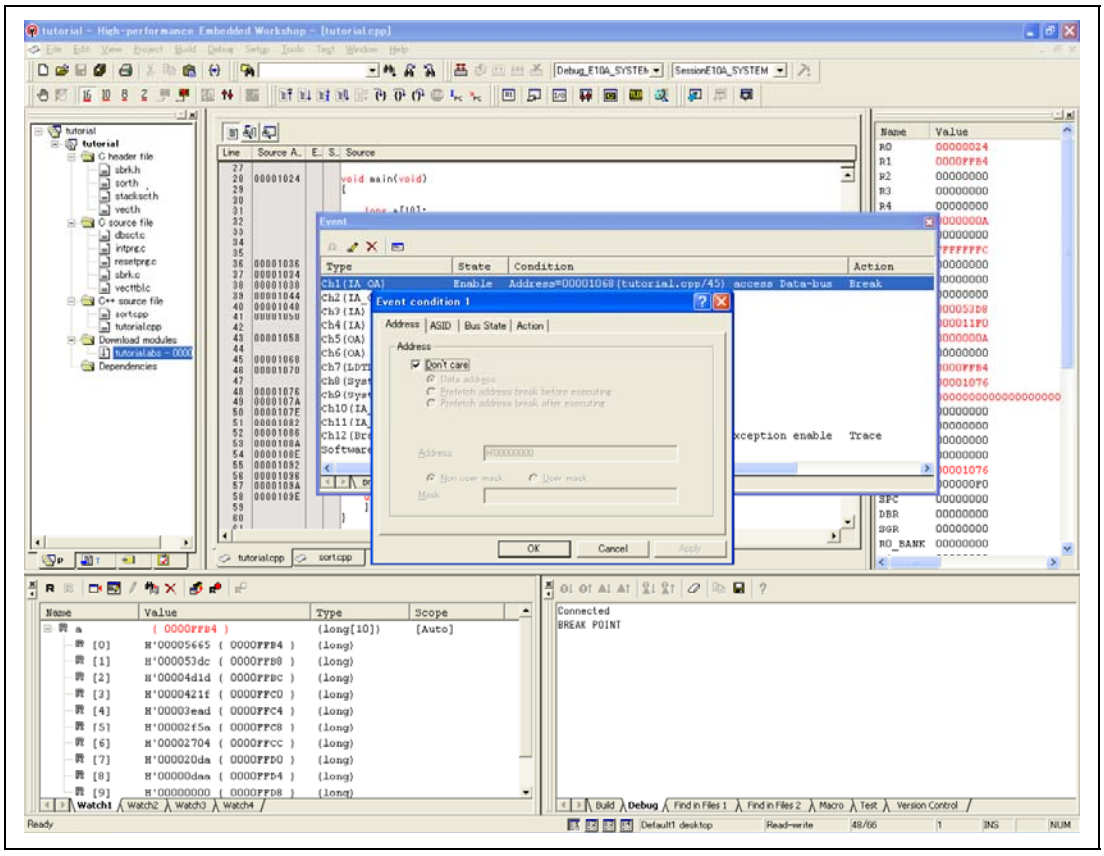


Figure 6.36 [High-performance Embedded Workshop] Window ([Event condition 1])

- The [Event condition 1] dialog box is displayed.
- Clear the [Don't care] check box in the [Address] page.
- Select the [Prefetch address break after executing] radio button and enter ***H'00001068*** as the value in the [Address] edit box.

Notes: 1. The number of hardware break conditions differs according to the product. For the number that can be specified for each product, refer to the online help.

2. For some MCUs/MPUs, the address value to be entered in the [Address] edit box is not ***H'00001068***. In such cases, specify the address where "p_sam->sort(a);" (tutorial.cpp/45) is located.

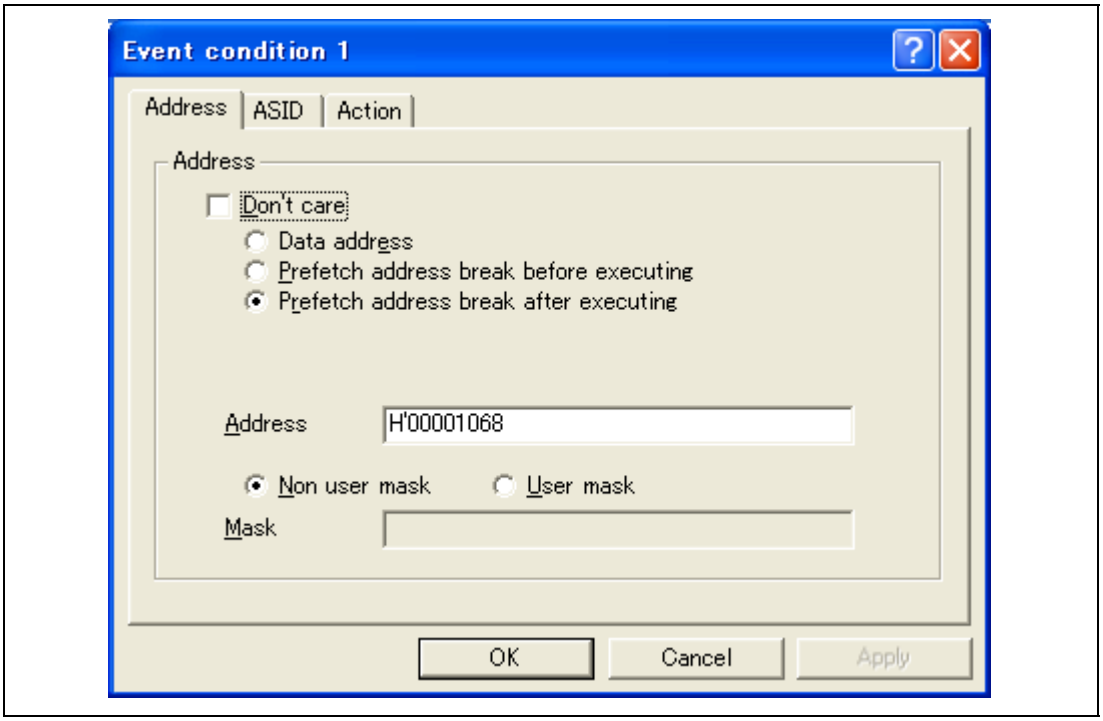


Figure 6.37 [Address] Page ([Event condition 1] Dialog Box)

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'00001068 (tutorial.cpp/45) pcafter`.
- 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. When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
- 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 Ch1(IA_OA).

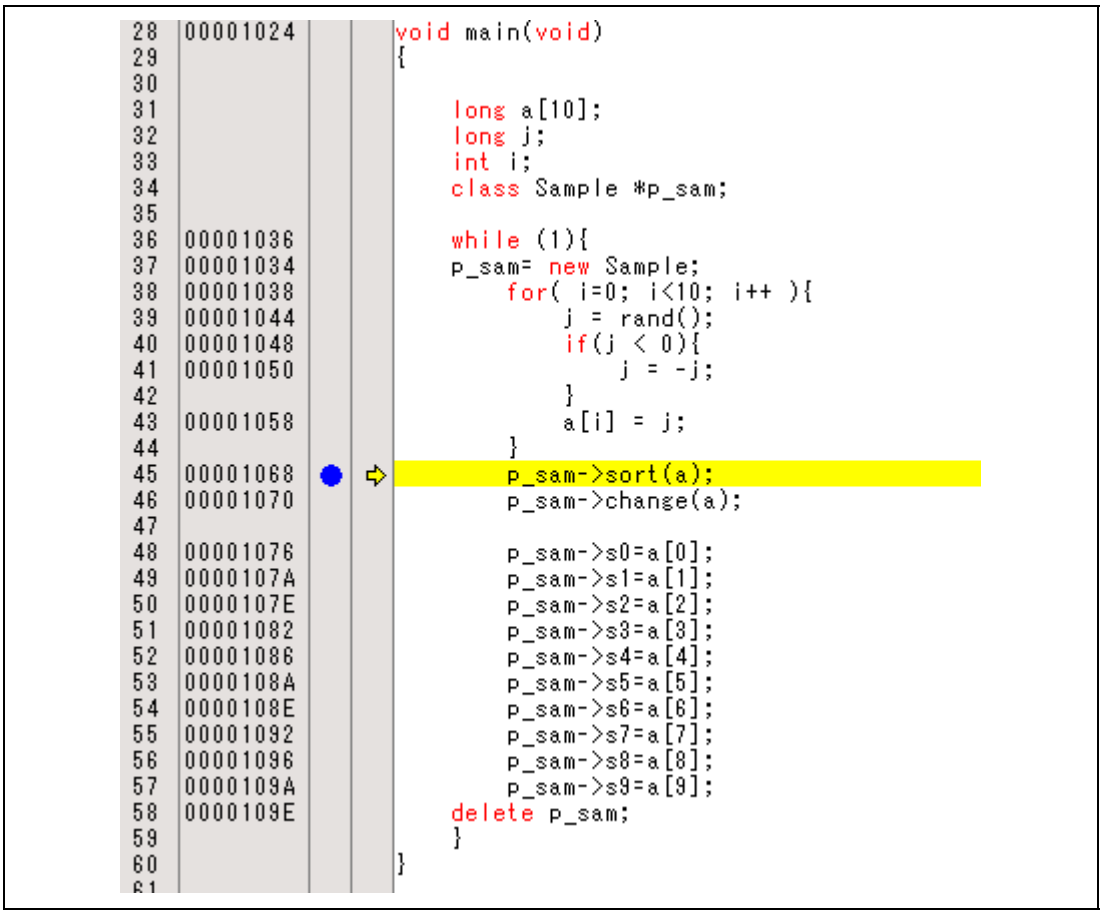


Figure 6.38 [Editor] Window at Execution Stop (Event Condition 1)

The [Status] window displays the following contents.

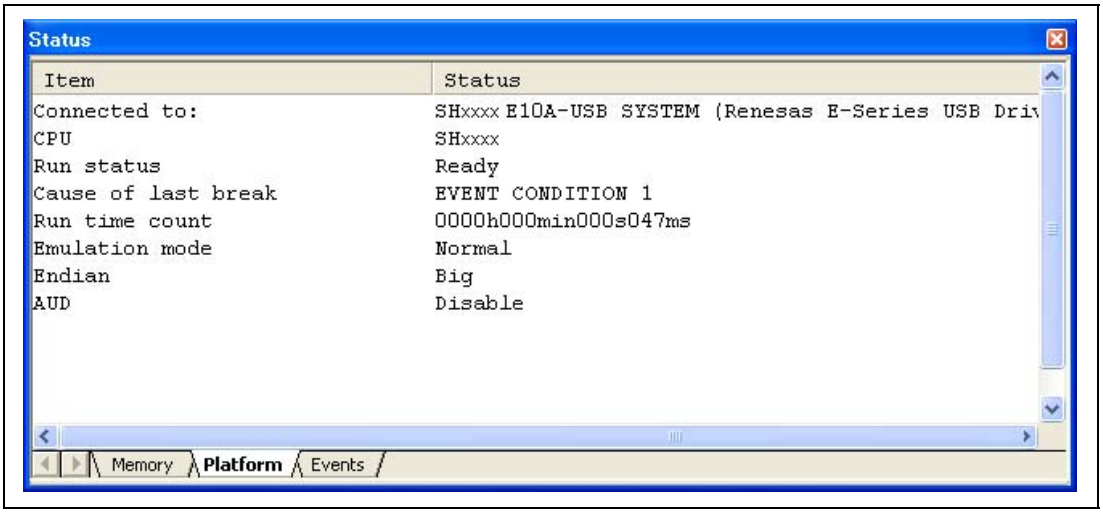


Figure 6.39 Displayed Contents of the [Status] Window (Event Condition 1)

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

6.18.1 Setting the Sequential Event Condition

The emulator has sequential event functions.

Set hardware break conditions as follows:

Ch1(IA_OA):
A break condition is satisfied immediately after address H'00001068 is accessed.

Ch2(IA_OA_DT_CT):
A break condition is satisfied immediately after address H'00001058 is accessed.

Note: For some MCUs/MPUs, the address values are different from those given above. In such cases, specify the address of “p_sam->sort(a);” (tutorial.cpp/45) for Ch1(IA_OA) and specify the address of “a[i]=j;” (tutorial.cpp/43) for Ch2(IA_OA_DT_CT).

Follow the setting method described in the previous section.

To set these breakpoints 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.

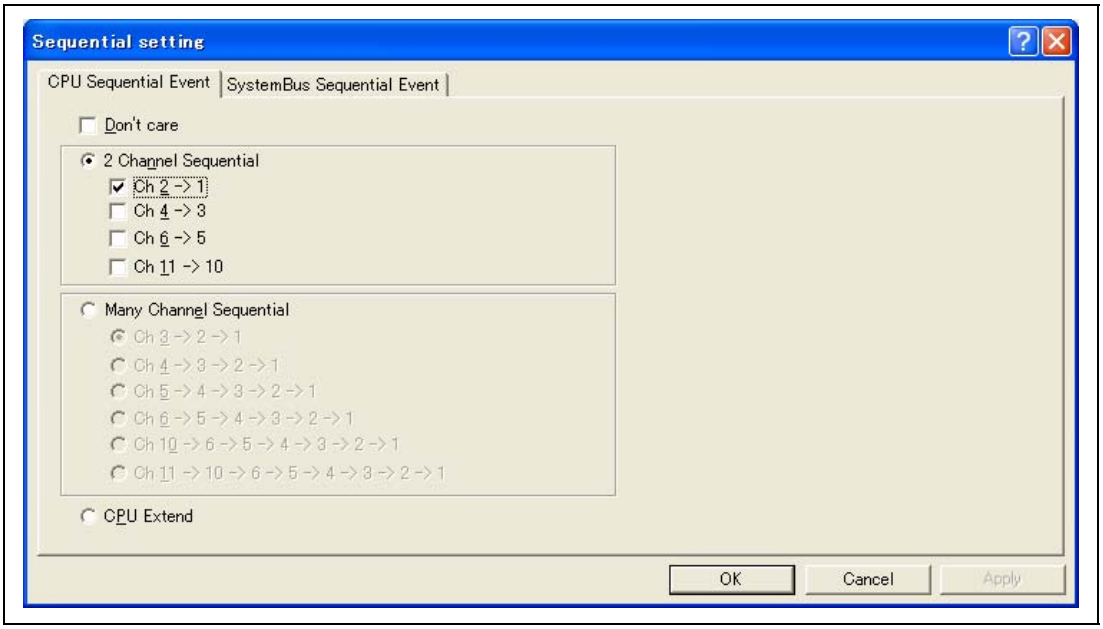


Figure 6.40 [Sequential setting] Dialog Box

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 [Ch 2 -> 1] radio button and click the [OK] button.

When the setting is completed, the [Event] window will be as shown in figure 6.41.

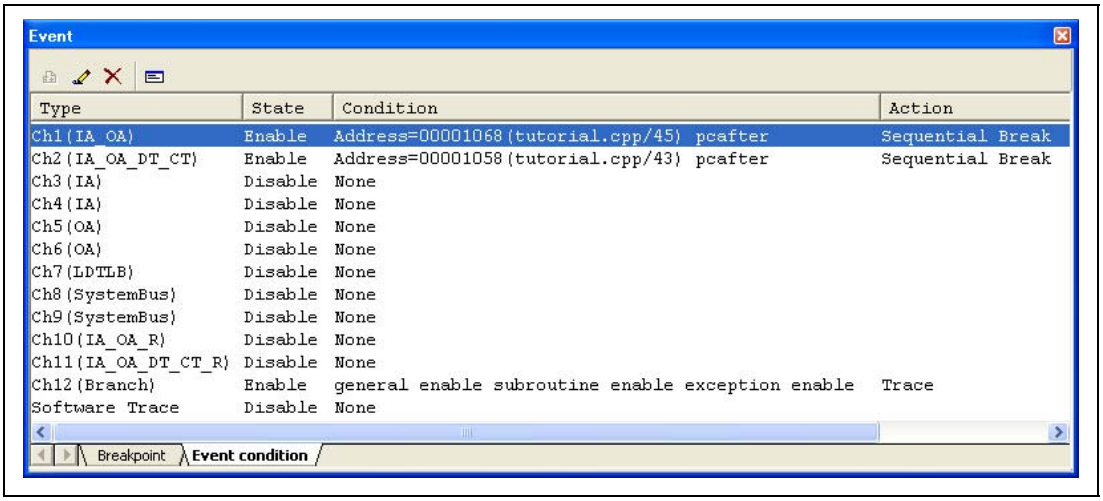


Figure 6.41 [Event condition] Page

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. When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
- 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 Ch1(IA_OA).

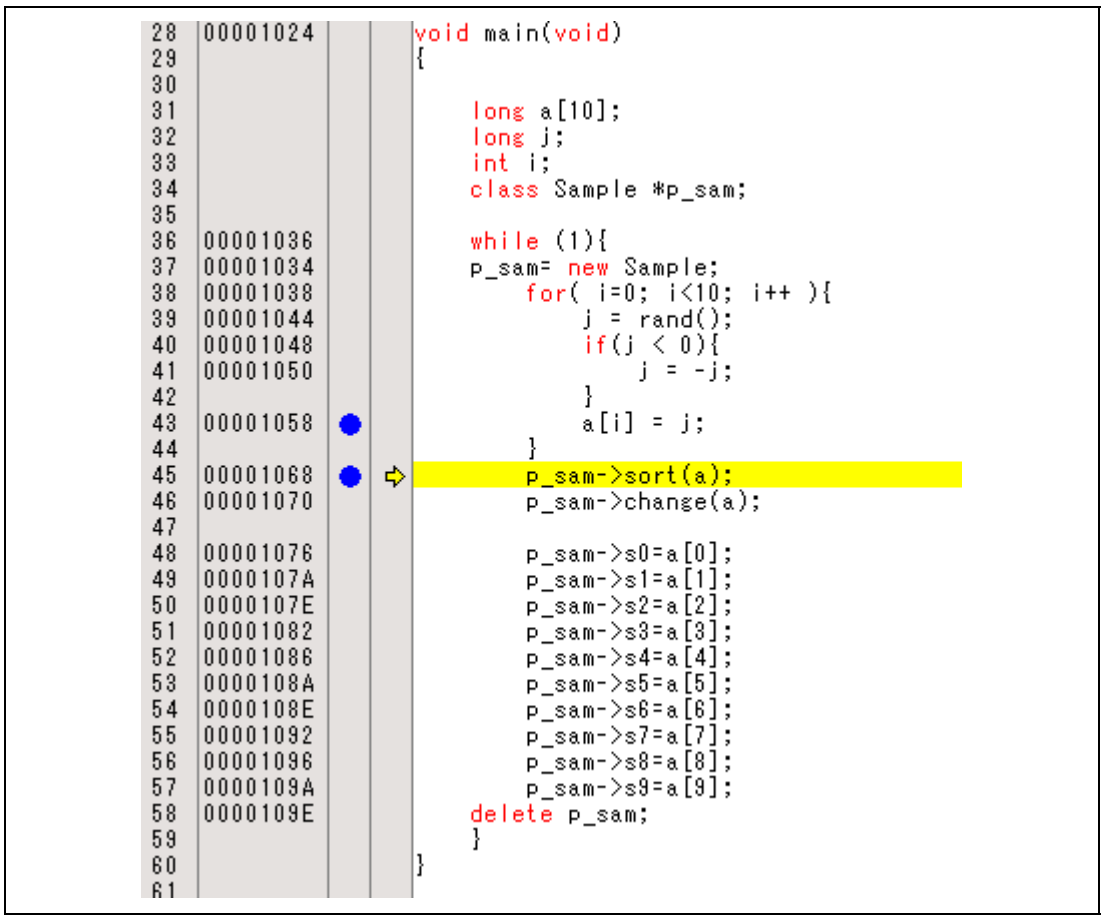


Figure 6.42 [Editor] Window at Execution Stop (Sequential Break)

The [Status] window displays the following contents.

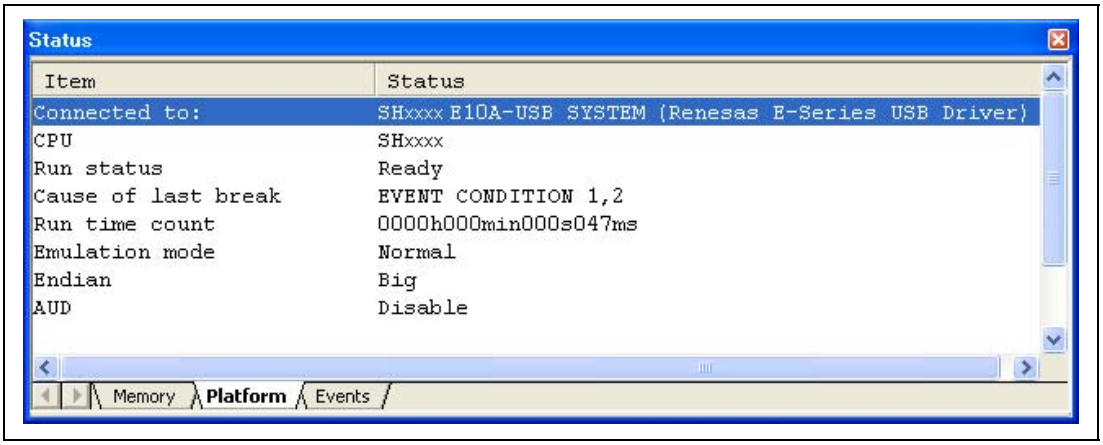


Figure 6.43 Displayed Contents of the [Status] Window (Sequential Break)

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 hardware break 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 three branch-instruction trace functions.

- Internal Trace Function

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

Notes: 1. The number of branch instructions that can be acquired by a trace differs according to the product. For the number that can be specified for each product, refer to the online help.
2. The internal trace function is not supported for all products. For 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 number of events acquired by a trace is 262,144.

  This function is only available on the E10A-USB emulator with model name HS0005KCU02H.

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 value
- Data value
- Mnemonic
- Operand
- Source line

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 valid 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.

6.19.1 Displaying the Trace Window

Select [Trace] from the [Code] submenu of the [View] menu. The result of the acquired trace is displayed.

6.19.2 Internal Trace Function

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

In the internal trace function, the type of the branch instruction can be specified and acquired.

The type is specified as follows:

- Select [Eventpoints] from the [Code] submenu of the [View] menu.
- Click on the [Event condition] tab.
- Select the line of Ch12(Branch) so that it is highlighted. Then double-click on this line.
- The [Event condition 12] dialog box appears. Deselect the [Don't care] checkbox on the [Branch event] page.

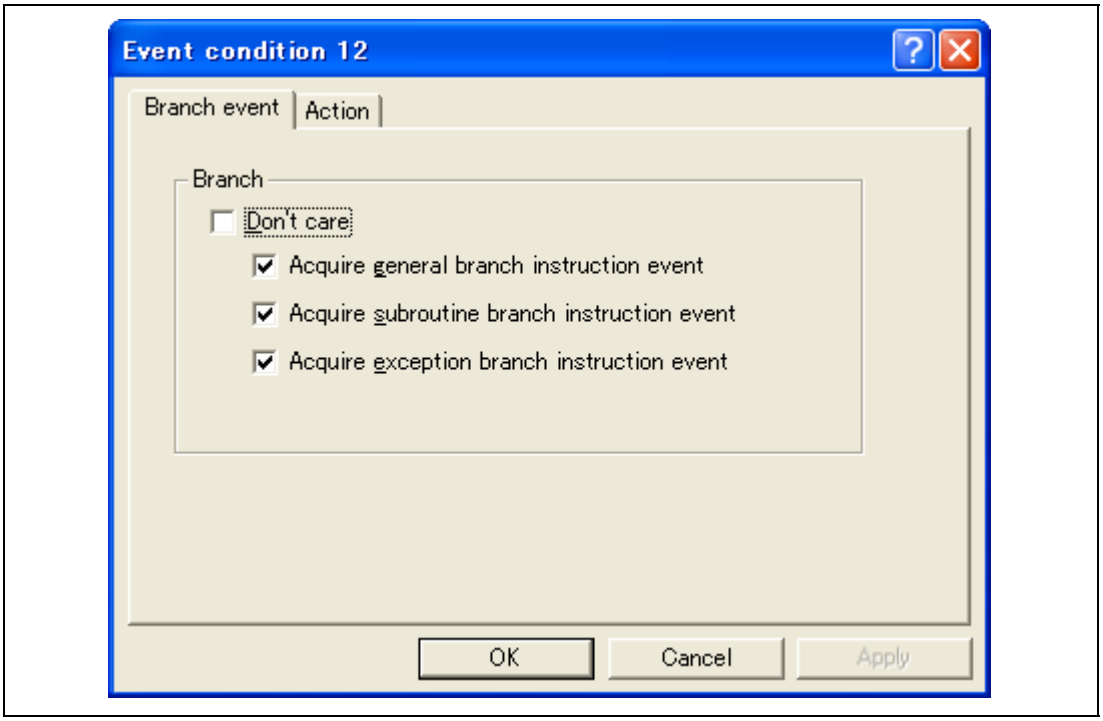


Figure 6.44 [Branch event] Page

Deselect the [Acquire break] checkbox and select the [Acquire trace] checkbox on the [Action] page.

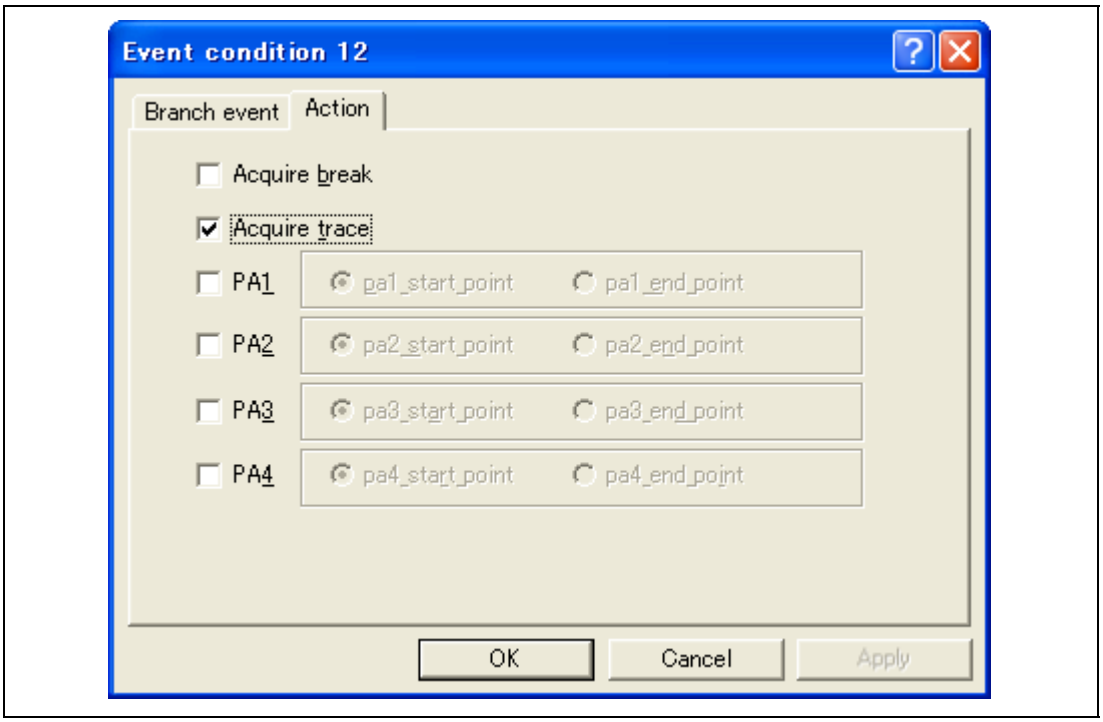


Figure 6.45 [Action] Page

Notes: 1. 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.

2. In some products, the branch settings are made in the [Branch trace] page of the [Acquisition] dialog box. For details on the specifications of each product, refer to the online help.

Run the program as shown in the example of section 6.17.1, PC Break Function. The trace results are displayed in the [Trace] window after the program execution is completed.

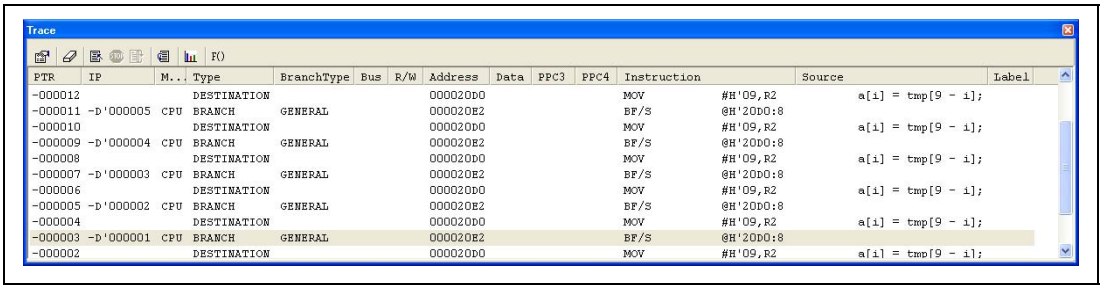


Figure 6.46 [Trace] Window

- 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.

6.19.3 AUD Trace Function

This function is available when the AUD pin of the MCU/MPU is connected to the emulator.

The following is the procedure for setting the AUD trace function.

(1) Setting the trace acquisition mode

- Display the [Trace] window.
- Click the [Trace] window with the right-hand mouse button and select [Acquisition] from the popup menu to display the [Trace Acquisition] dialog box.

The trace acquisition condition is set in the [Trace mode] page.

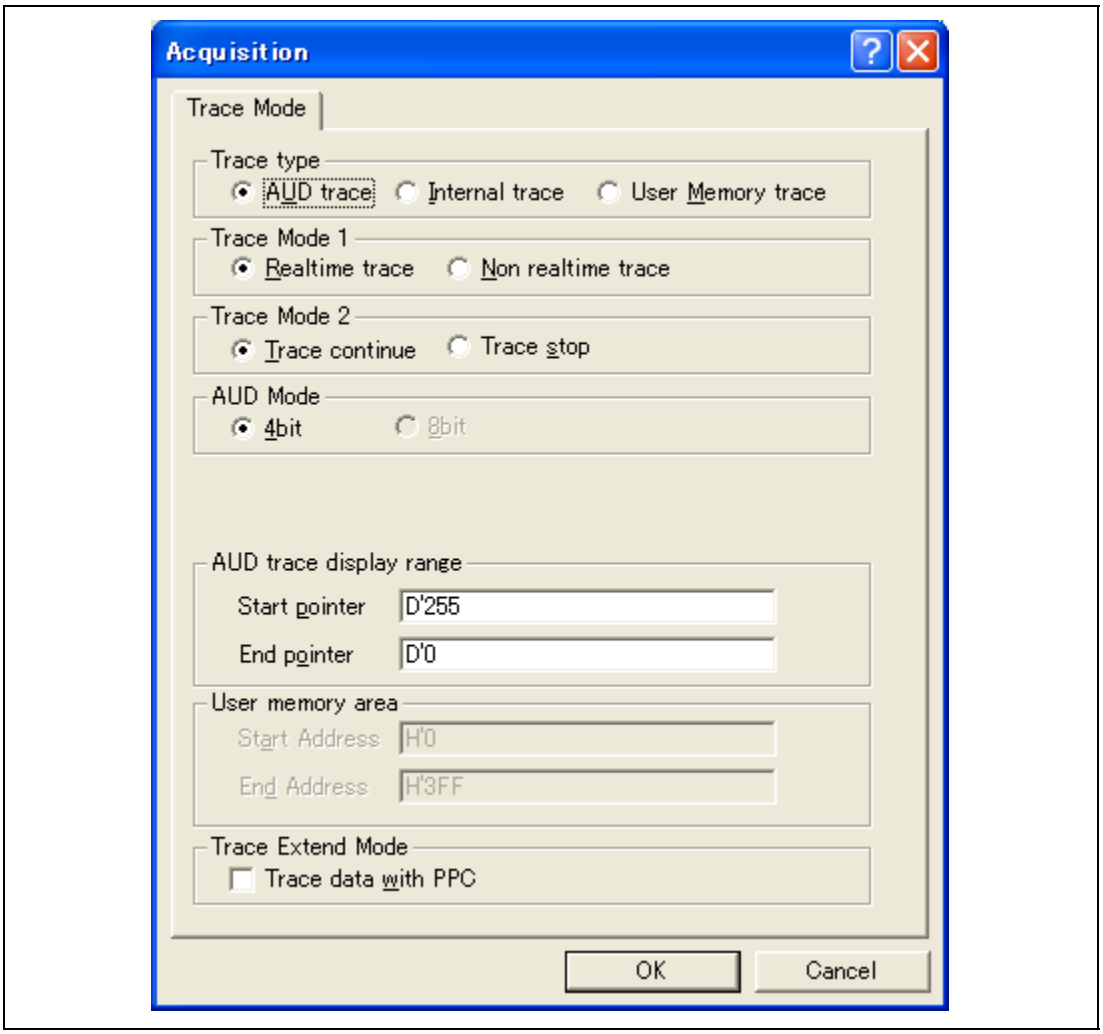


Figure 6.47 [Trace mode] Page

Note: This dialog box cannot be used in a product that does not support the AUD trace function. 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.

The following table shows the options.

AUD Trace Acquisition Mode

Type	Mode	Description
Continuous trace occurs	Realtime trace	When the trace information is being generated intensively that the output from the AUD pin incapable of keeping up, the CPU temporarily suspends the output of trace information. Therefore, although the user program is run in real time, the acquisition of some trace information might not be possible.
	Non realtime trace	When trace information is being generated so intensively that the output from the AUD pin is incapable of keeping up, CPU operations are temporarily suspended and the output of trace information takes priority. In such cases, the realtime characteristics of the user program are lost.
Trace buffer full	Trace continue	This function always overwrites the oldest trace information to acquire the latest trace information.
	Trace stop	When the trace buffer becomes full, the trace information is no longer acquired. The user program is continuously executed.

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.

(2) 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 [Trace] window after the program execution is completed.

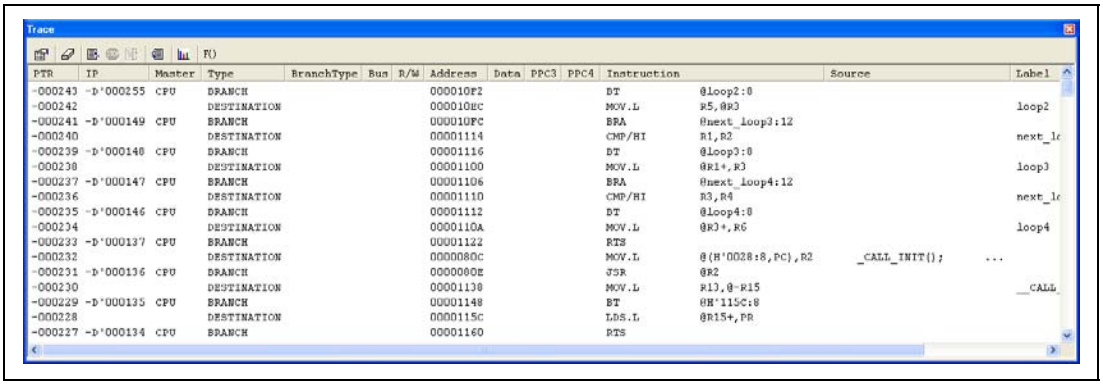


Figure 6.48 [Trace] Window (Example)

6.19.4 Memory Output Trace Function

This function is used to write the trace result to the specified memory range.

The following is the procedure for setting the memory output trace function.

(1) Setting the trace acquisition mode

- Display the [Trace] window.
- Click the [Trace] window with the right-hand mouse button and select [Acquisition] from the popup menu to display the [Acquisition] dialog box.

The trace acquisition condition is set in the [Trace mode] page.

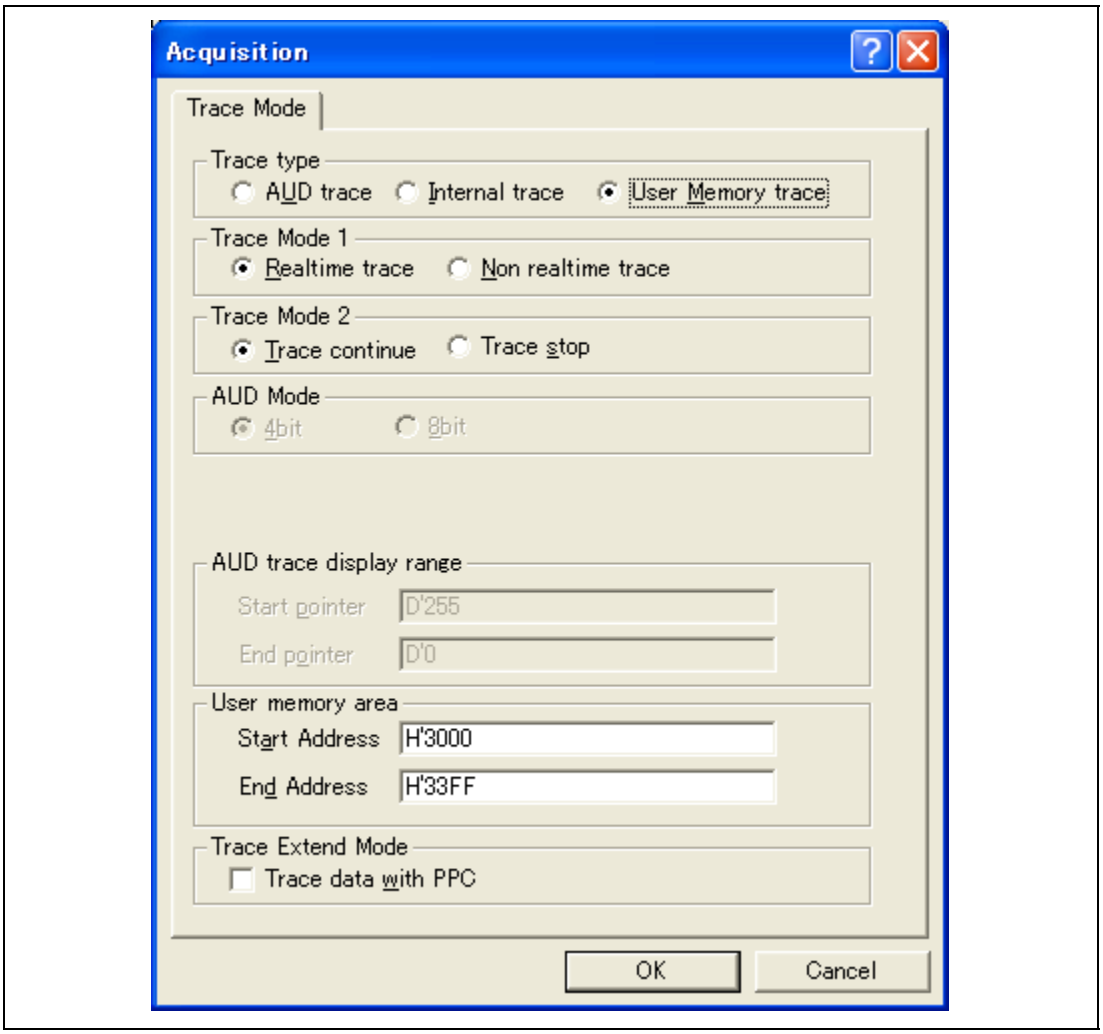


Figure 6.49 [Trace Mode] Page

The following table shows the options.

Trace Acquisition Mode

Type	Mode	Description
Continuous trace occurs	Realtime trace	When trace information is being generated so intensively that the output from the memory is incapable of keeping up, all the information may not be output. The user program can be executed in realtime, but some trace information will be lost.
	Non realtime trace	When trace information is being generated so intensively that the output from the memory is incapable of keeping up, the CPU stops operations. The user program is not executed in realtime.
Trace buffer full	Trace continue	This function always overwrites the oldest trace information to acquire the latest trace information.
	Trace stop	When the trace buffer becomes full, the trace information is no longer acquired. The user program is continuously executed.

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.

(2) Displaying the trace result

- Set the start and end addresses, which are in the memory range that outputs the trace result, in the [Start Address] and [End Address] 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.
- Run the program as shown in the example in section 6.17.1, PC Break Function. The trace results are displayed in the [Trace] window after program execution is completed.

The following is an example of the trace display.

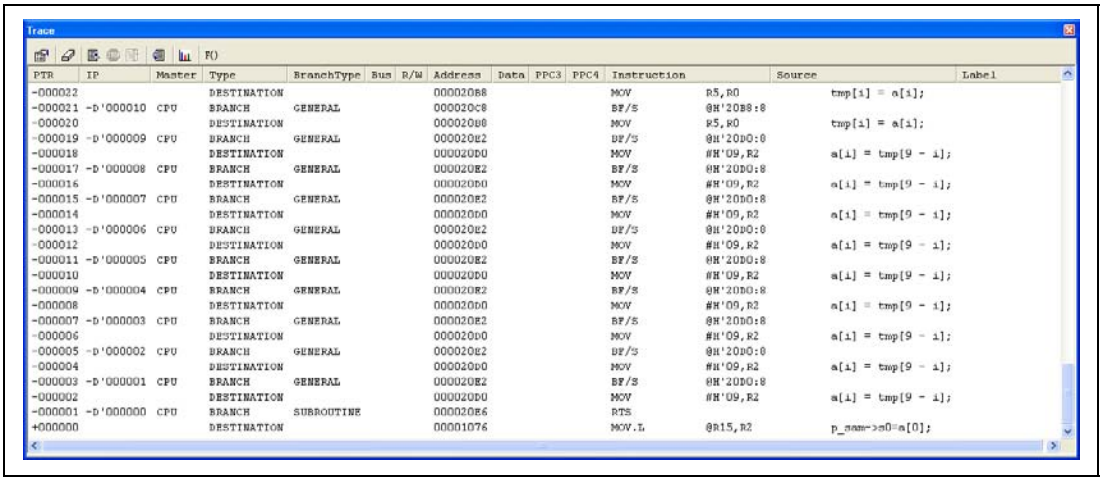


Figure 6.50 [Trace] Window (Example)

6.19.5 MMU Support

This function can be used when the supported MCU/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 MCU/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 MCU/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 6.51.

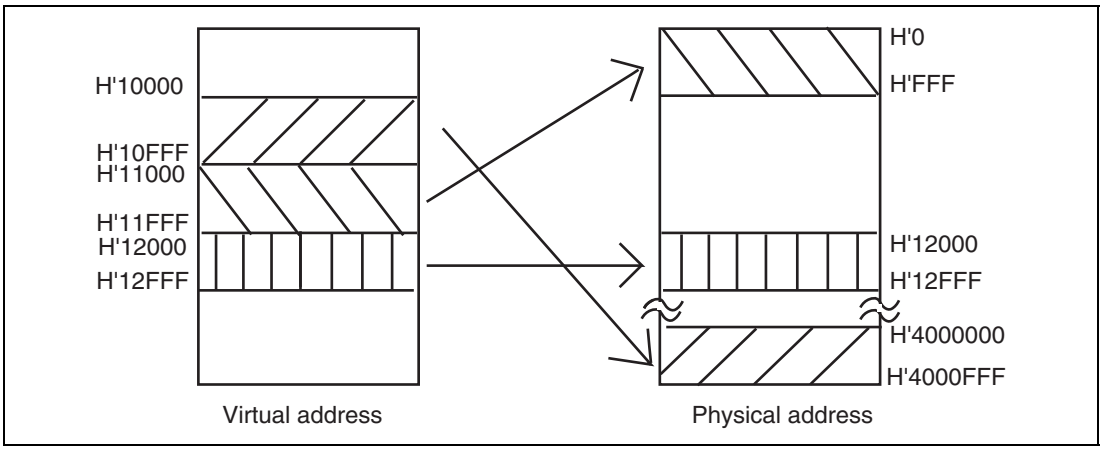


Figure 6.51 Address Translation according to VP_MAP Tables

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.

6.20 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 Elf/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 [S/W breakpoint] column in the `sort` function and set a PC breakpoint.

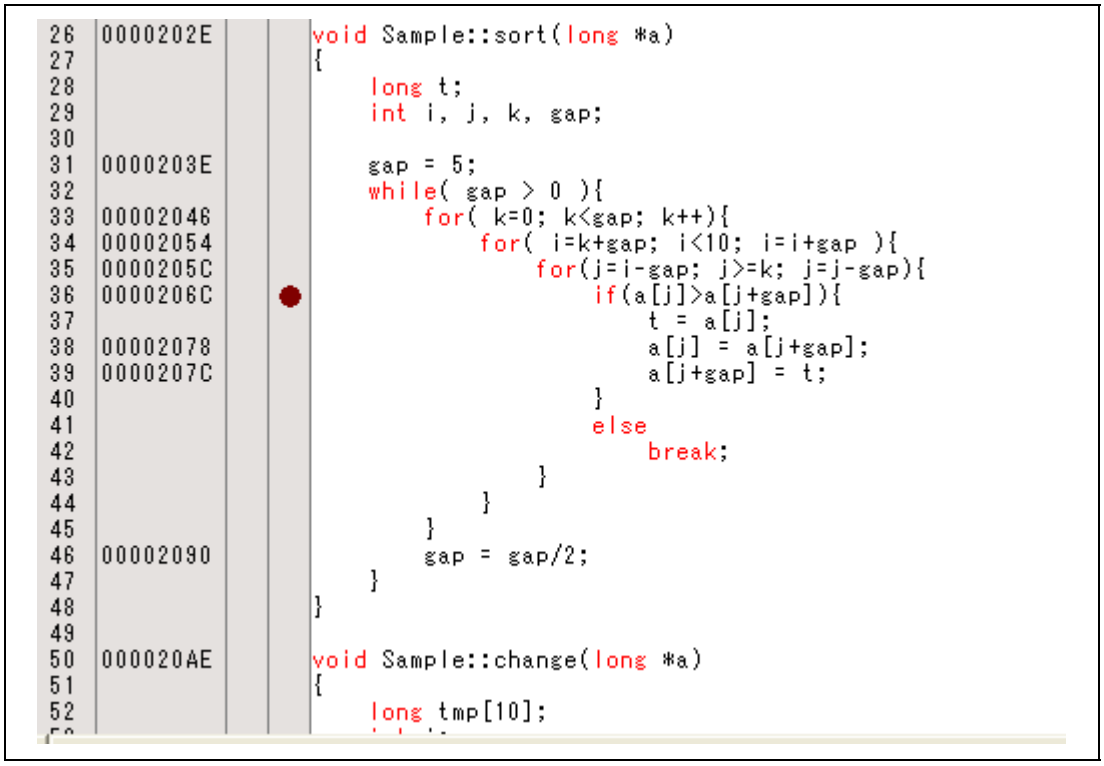


Figure 6.52 [Editor] Window (PC Breakpoint Setting)

- 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.
  When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
- 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.

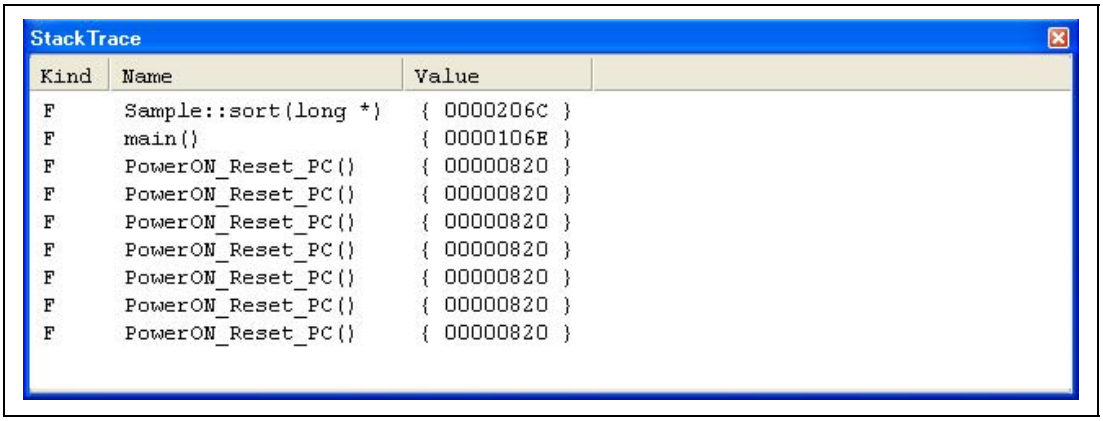


Figure 6.53 [Stack Trace] Window

Figure 6.53 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 `tutorial()` function.

To remove the PC breakpoint, double-click the [S/W breakpoint] column in the `sort` function again.

Note: For details on this function, refer to the online help.

6.21 Performance Measurement Function

The emulator has performance measurement functions.

- Performance measurement function

  This function applies a counter in the MCU/MPU to measure the number of times various events have occurred and cycle count. A start and end condition for counting can be set.

  Various items that can be measured differ according to the supported MCU/MPU.
- Profiling function

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

  When execution of the user program is ended, the number of times each function is called and the measured data are displayed. The measured items are the same as those for the number of times various events have occurred, and cycle count.

6.21.1 Performance Measurement Function

The following is an example of the use of a counter in the MCU/MPU to measure the number of times various events have occurred and cycle count.

This function cannot be used with the profiling function that is described later.

(1) Setting method

Select [Performance Analysis] from the [Performance] submenu of the [View] menu. When the [Select Performance Analysis Type] dialog box will open, click the [OK] button.

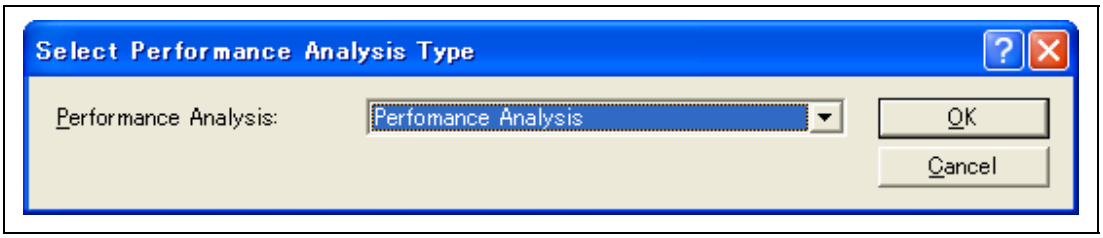


Figure 6.54 [Select Performance Analysis Type] Dialog Box

- The [Performance Analysis] window will be displayed.
- Place the mouse cursor on any event channel in this window, click the right-hand mouse button, and then select [Set] from the popup menu. The [Performance Analysis] dialog box will open. The events to be measured and measuring conditions can be set in this dialog box.

Note: The items that can be displayed in this dialog box differ according to the product. For details on the settings for each product, refer to the online help.

After the conditions have been set, clicking the [OK] button and executing the user program will display the result of measurement in the [Performance Analysis] window.

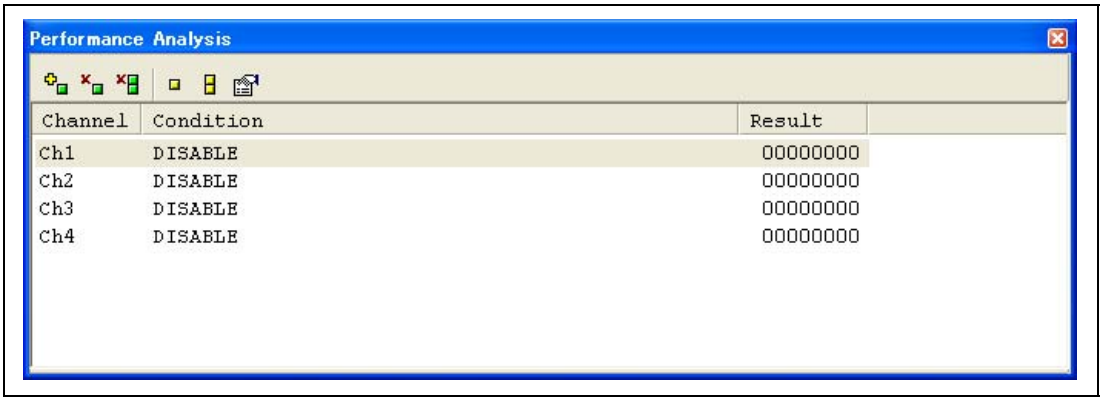


Figure 6.55 [Performance Analysis] Window

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

6.21.2 Profiling Function

The profiling function can measure performance for each function.

Notes: 1. Realtime operation is not possible while this function is in operation, since internal breaks are generated during program execution. Measuring the profile itself affects the measurements.

2. Performance profile measurement is not supported for all products. On those products for which it is supported, its characteristics differ according to the product. For details on the specifications of each product, refer to the additional document, Supplementary Information on Using the SHxxxx.

3. This function cannot be used with the performance measurement function that has been described before. Use one of these functions.

- First, download the tutorial program by referring to section 6.6, Downloading the Tutorial Program.
- Select [Profile] from the [Performance] submenu of the [View] menu. The [Profile] window will be displayed.

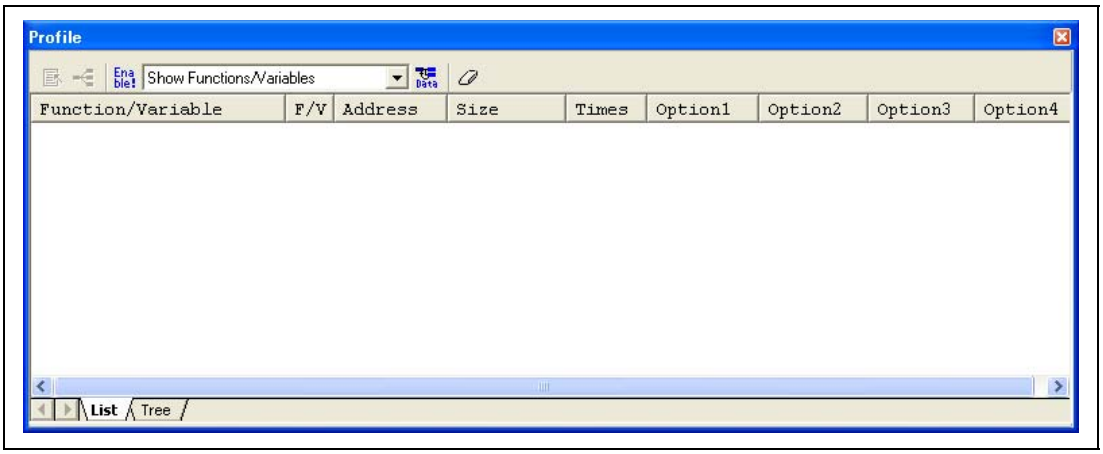


Figure 6.56 [Profile] Window

- To enable the profiling function, place the mouse cursor on an entry in the [List] sheet of the [Profile] window, click the right-hand mouse button, and then select [Enable Profiler] from the popup menu.

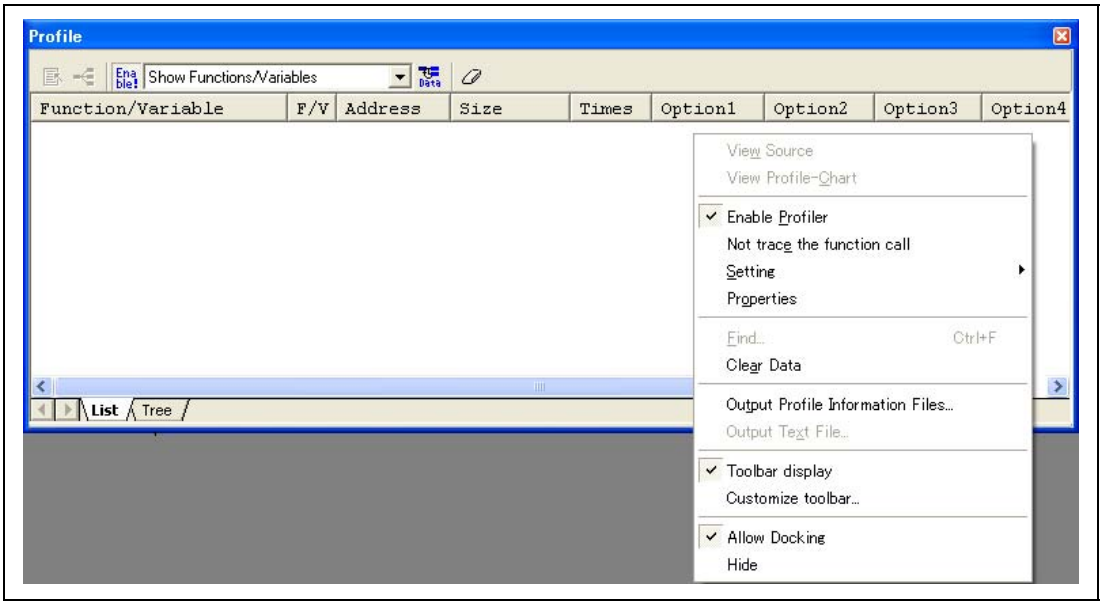


Figure 6.57 Selection of [Enable Profiler]

- To set the data to be measured for the selected function, place the mouse cursor on an entry in the [List] sheet of the [Profile] window, click the right-hand mouse button, and then select [Properties] from the popup menu. The [Select Data] dialog box is displayed.

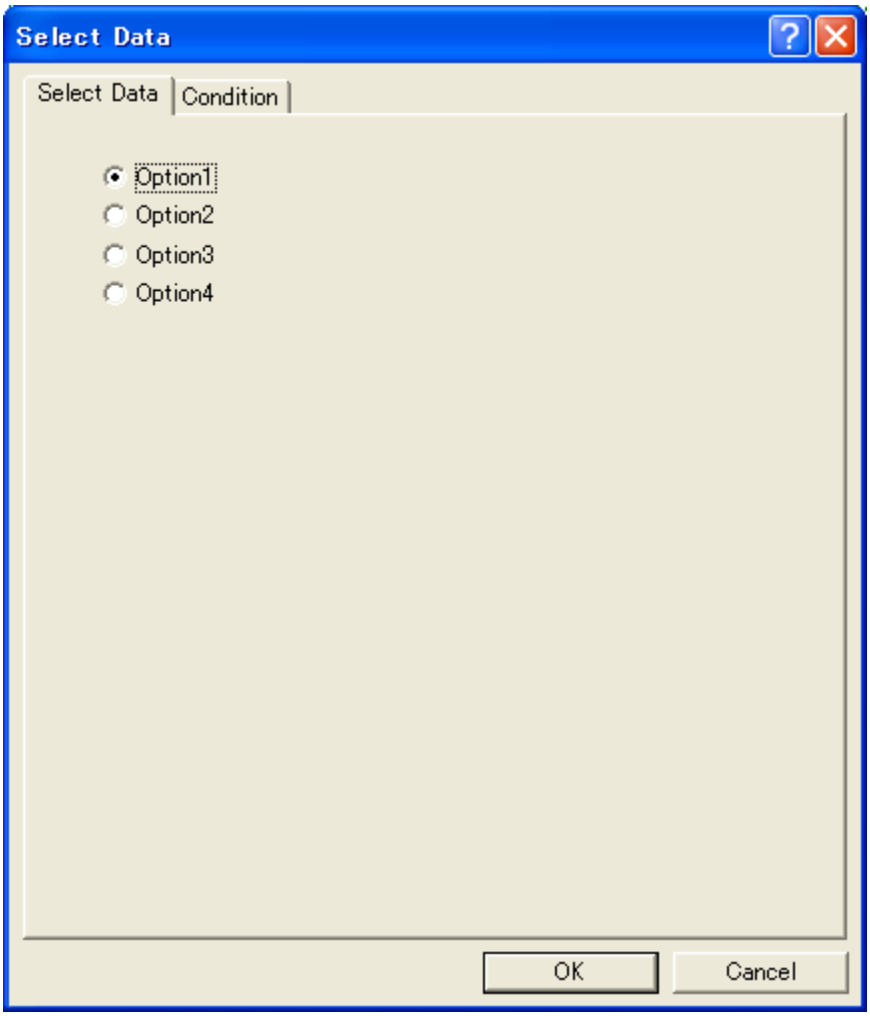


Figure 6.58 [Select Data] Page

- Open the [Select Data] page and select an option as the data to be measured. Select [Option 1] in this example.
- Open the [Condition] page and specify the type of data to be measured. To measure the number of elapsed cycles, deselect the [Disable] checkbox and select the [Cycle] checkbox on the [Condition] page. Then select the [Elapsed cycles] checkbox on the [Cycle] page.
- After the data has been selected, press the [OK] button.

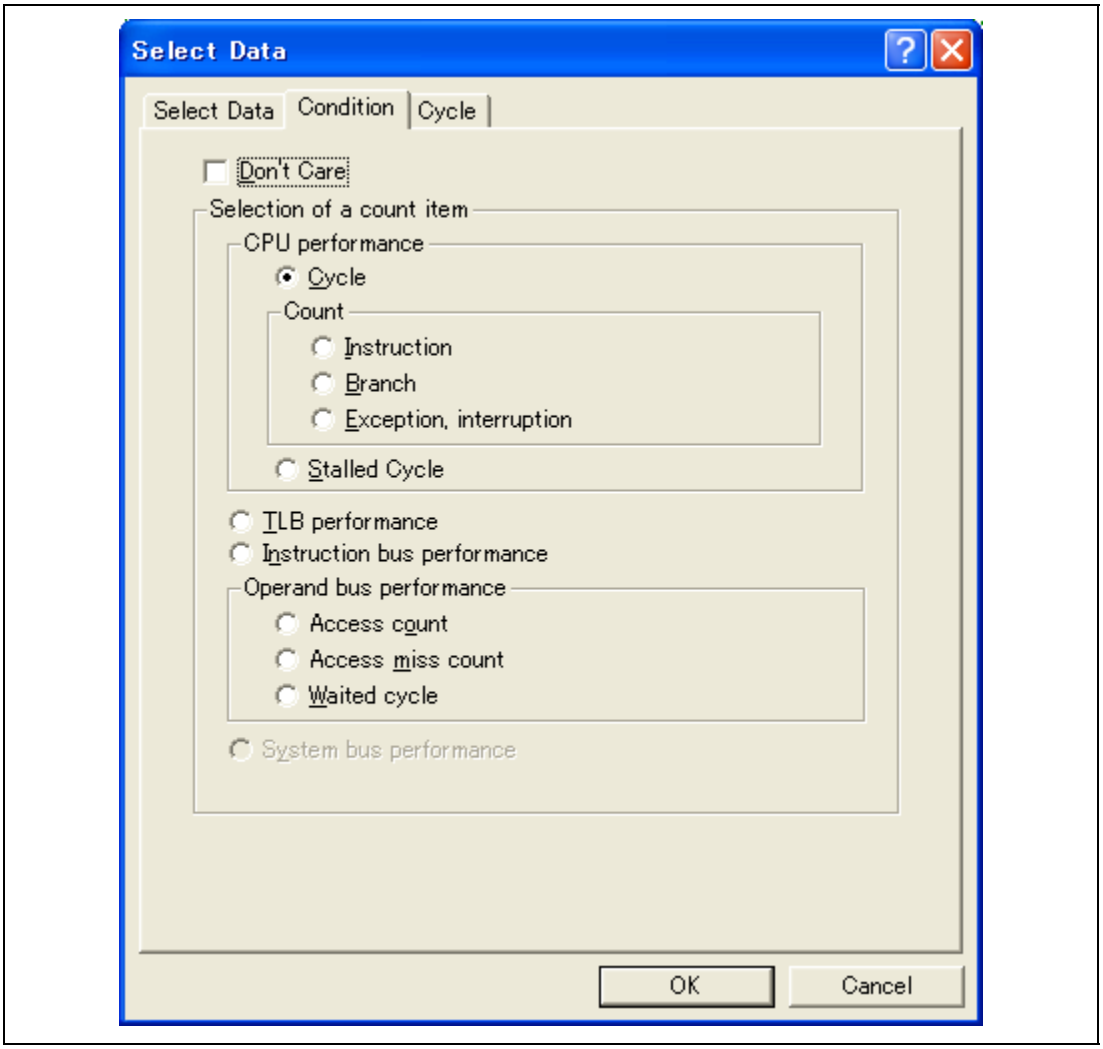


Figure 6.59 [Condition] Page

You may need to select the type of data from the [Option 1] drop-down list in the [Select Data] dialog box (figure 6.60) depending on the MCU/MPU in use.

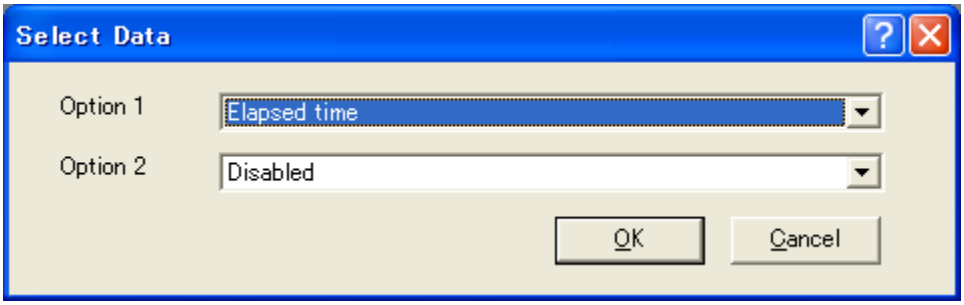


Figure 6.60 [Select Data] Dialog Box

Note: The contents of this dialog box vary with the product. For details on the specifications of each product, refer to the online help.

- Double-click the [S/W breakpoint] column for the `while` statement of the `main` function to set a PC breakpoint.

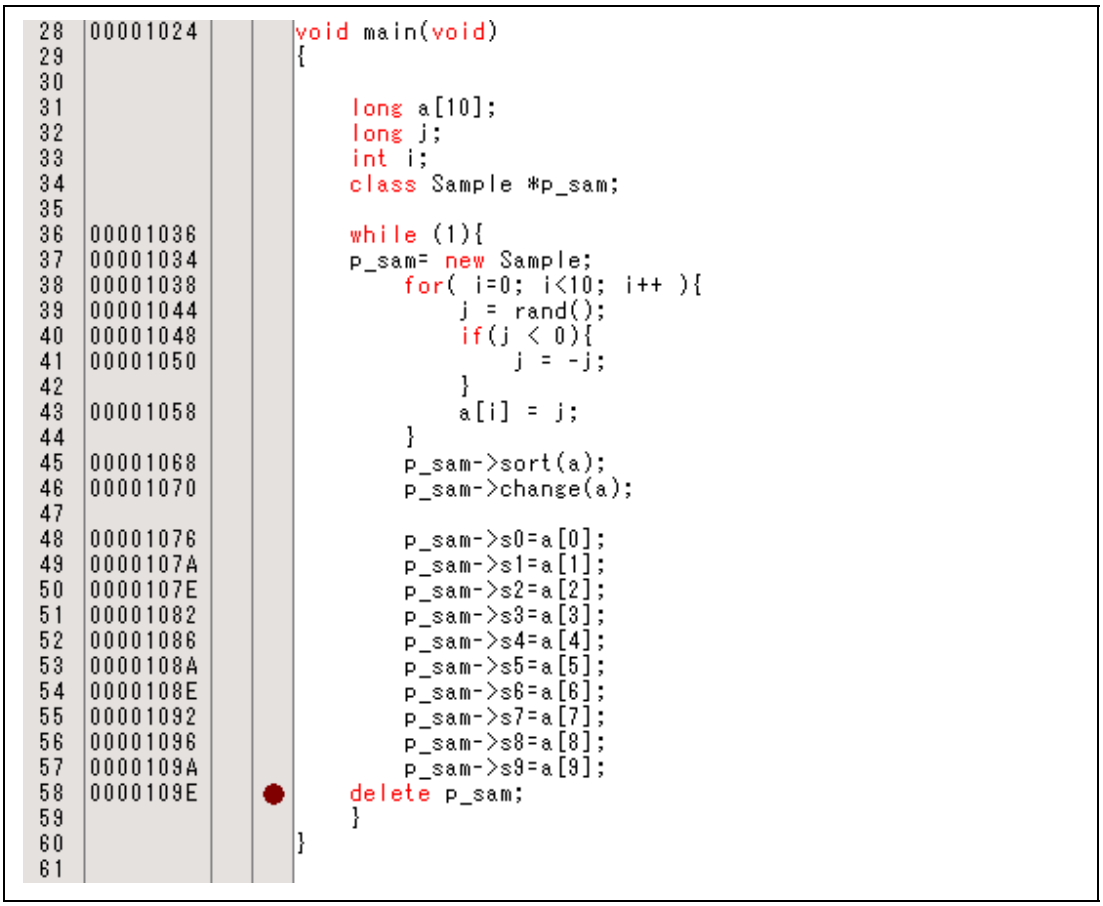
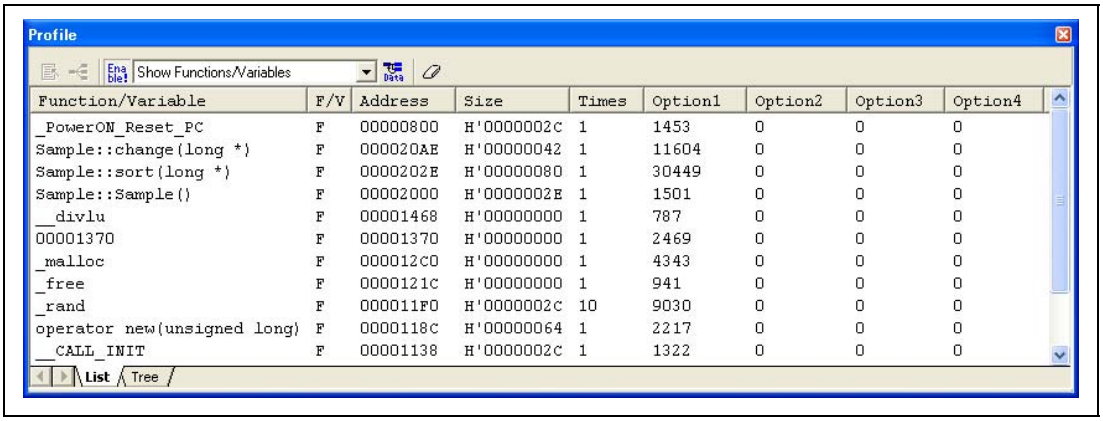


Figure 6.61 [Editor] Window (PC Breakpoint Setting)

- 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.
  When using the MCU with flash memory, specify the end address of the internal RAM for the stack pointer (SP). The internal RAM area differs depending on the MCU. Refer to the hardware manual of the MCU used.
- If program execution is failed, reset the device and execute again the procedures above.

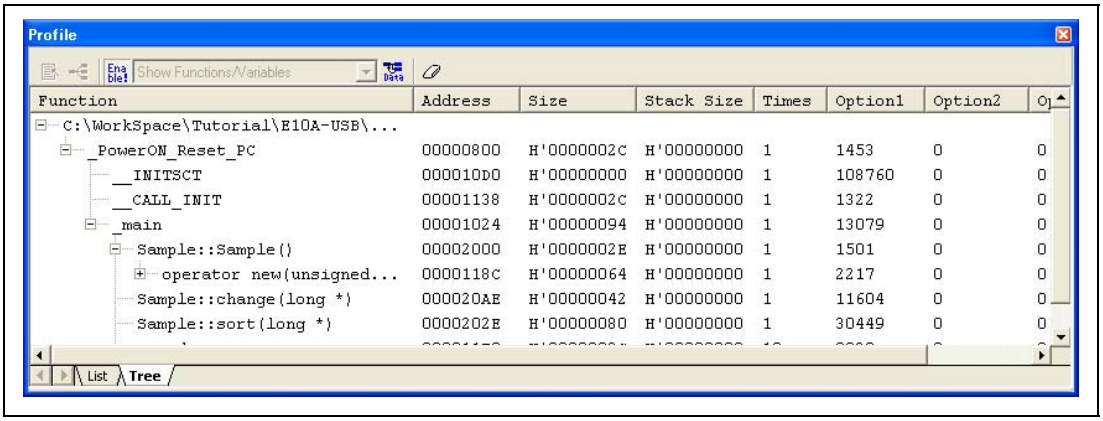
After the break in program execution, the results of the measurements are displayed in the [List] sheet of the [Profile] window.



Function/Variable	F/V	Address	Size	Times	Option1	Option2	Option3	Option4
_PowerON_Reset_PC	F	00000800	H'0000002C	1	1453	0	0	0
Sample::change(long *)	F	000020AE	H'00000042	1	11604	0	0	0
Sample::sort(long *)	F	0000202E	H'00000080	1	30449	0	0	0
Sample::Sample()	F	00002000	H'0000002E	1	1501	0	0	0
__divlu	F	00001468	H'00000000	1	787	0	0	0
00001370	F	00001370	H'00000000	1	2469	0	0	0
_malloc	F	000012C0	H'00000000	1	4343	0	0	0
_free	F	0000121C	H'00000000	1	941	0	0	0
_rand	F	000011F0	H'0000002C	10	9030	0	0	0
operator new(unsigned long)	F	0000118C	H'00000064	1	2217	0	0	0
__CALL_INIT	F	00001138	H'0000002C	1	1322	0	0	0

Figure 6.62 [List] Sheet of the [Profile] Window

Figure 6.63 shows the [Tree] sheet of the [Profile] window.



Function	Address	Size	Stack Size	Times	Option1	Option2	Option3
C:\WorkSpace\Tutorial\E10A-USB\...							
_PowerON_Reset_PC	00000800	H'0000002C	H'00000000	1	1453	0	0
__INITSCT	000010D0	H'00000000	H'00000000	1	108760	0	0
__CALL_INIT	00001138	H'0000002C	H'00000000	1	1322	0	0
_main	00001024	H'00000094	H'00000000	1	13079	0	0
Sample::Sample()	00002000	H'0000002E	H'00000000	1	1501	0	0
operator new(unsigned...	0000118C	H'00000064	H'00000000	1	2217	0	0
Sample::change(long *)	000020AE	H'00000042	H'00000000	1	11604	0	0
Sample::sort(long *)	0000202E	H'00000080	H'00000000	1	30449	0	0

Figure 6.63 [Tree] Sheet of the [Profile] Window

When [View Profile-Chart] is selected from the popup menu, a chart diagram is shown.

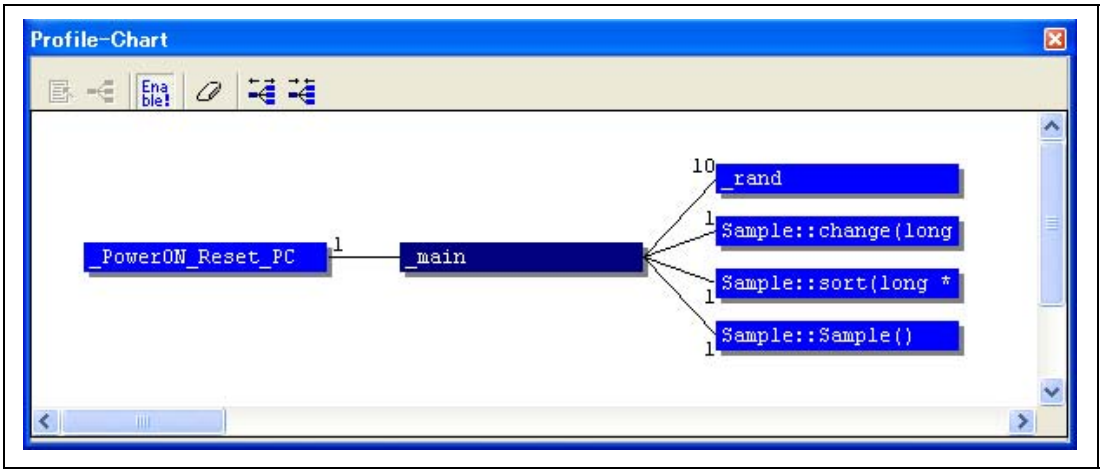


Figure 6.64 [Profile-Chart] Window

6.22 Downloading to the Flash Memory Area

The emulator enables downloading to the external flash memory area. This function requires a program for programming 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.

Notes: 1. The write and erase modules must be prepared by the user.
2. This function is not available when the SH7047F, SH7144F, or SH7145F is in use. For these MCUs, the [Loading flash memory] page shown in figure 6.65 will not be displayed.

- Interface of the write and erase modules with the emulator firmware

  The emulator firmware branches to the write and erase modules. The following conditions must be satisfied if branching from the emulator firmware to the write and erase modules and return from the write and erase modules to the emulator firmware are to be successful.

  — The write and erase modules must be entirely written in assembly language.
  — All values of general and control registers must be saved before the write or erase modules are called and restored on return from the modules.
  — Configure the write and erase modules so that execution returns to the origin of the call after processing.
  — The write and erase modules must be Motorola S-type files.
  — The write and erase modules must not contain SLEEP, DIVS, DIVU, or REBANK instructions.
  — FPU exceptions must not occur within the write and erase modules.

  The module interface must be as follows so that the information required for flash memory access is passed correctly.

Table 6.3 Module Interface

Module Name	Argument	Return Value
Write module	R4(L): Write address 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
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 [Setup] menu.

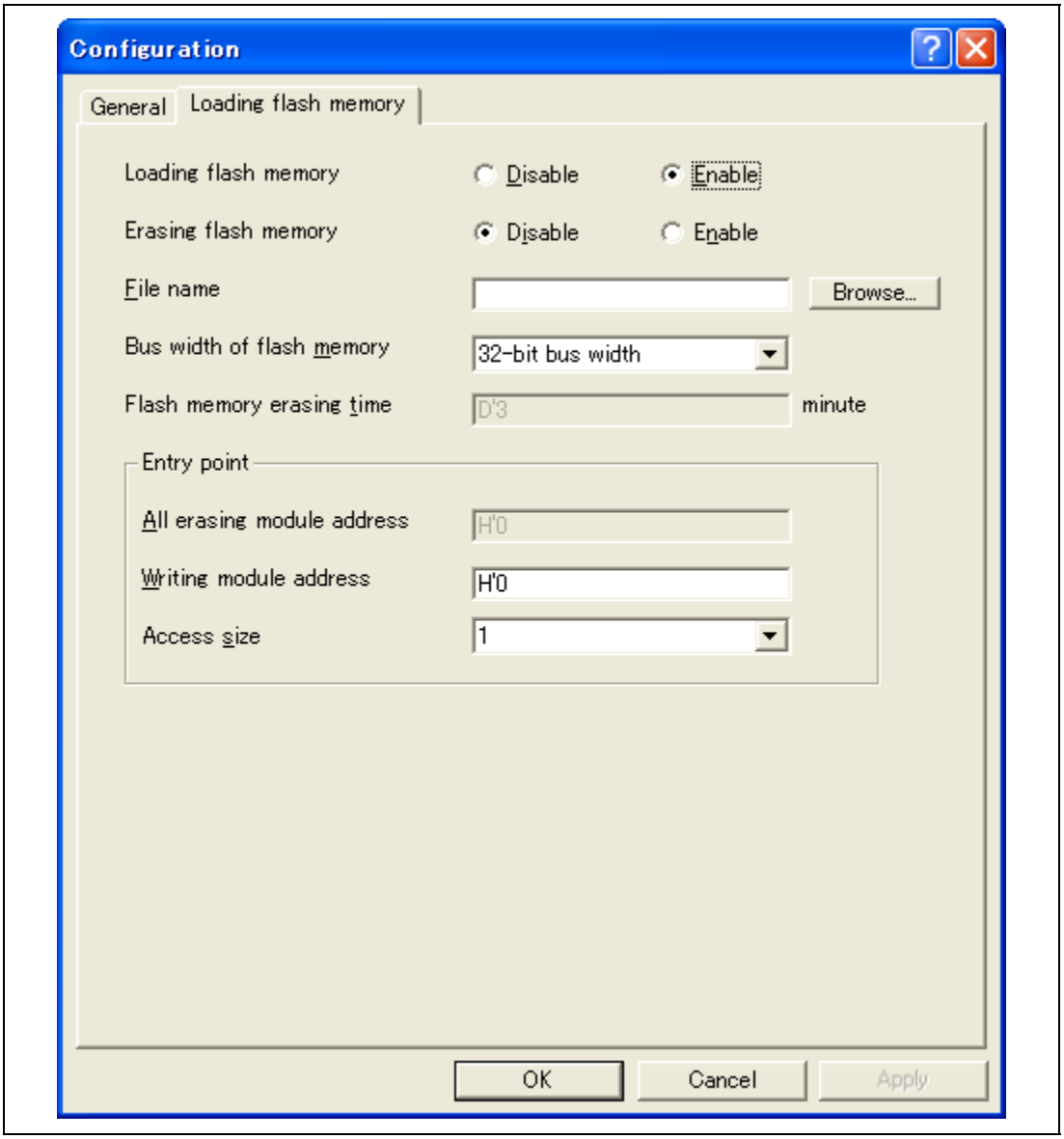


Figure 6.65 [Loading flash memory] Page

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 programmed. 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 or access size of the write and erase modules. [All erasing module address] edit box: Inputs the calling destination address of the erase module. [Writing module address] edit box: Inputs the calling destination address of the write module. [Access size] combo box: Selects the access size of the RAM area where the write/erase module is loaded.

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

  An example of downloading from the emulator to the flash memory is given below. This is given as an example because the actual values will vary with the target MCU and flash memory that are in use. 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.

  To use a sample workspace on Windows Vista® or Windows® 7, it must be copied into a folder that corresponds to the currently logged account.

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	CPU internal frequency	167 MHz
	Bus frequency	55.7 MHz
	CPU internal module frequency	27.84 MHz
	Endian	Big endian

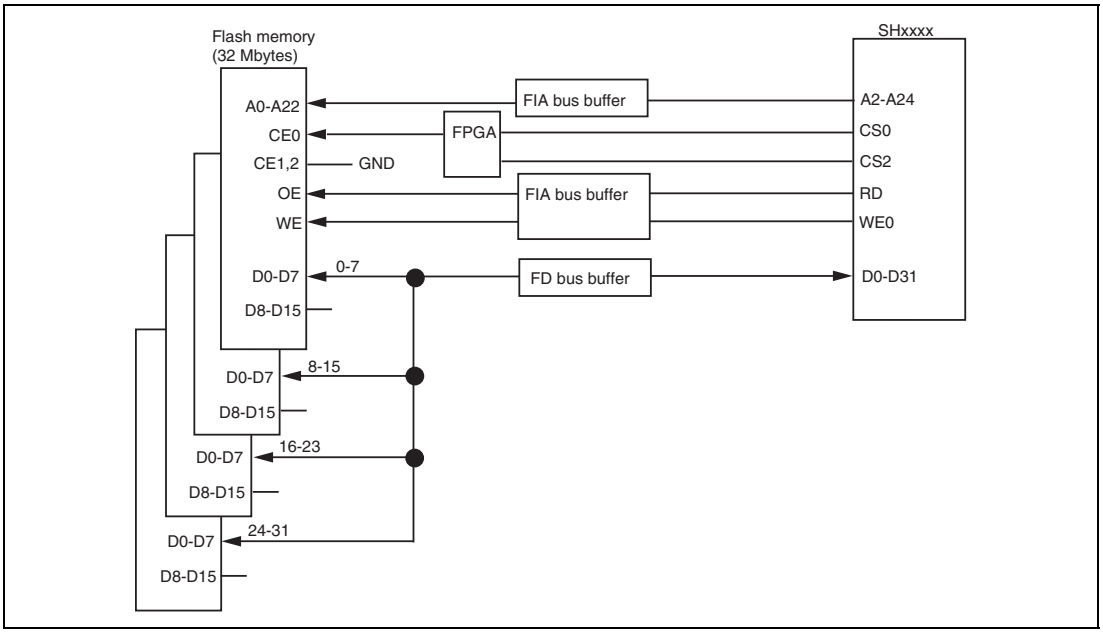


Figure 6.66 Flash Memory Wiring

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:

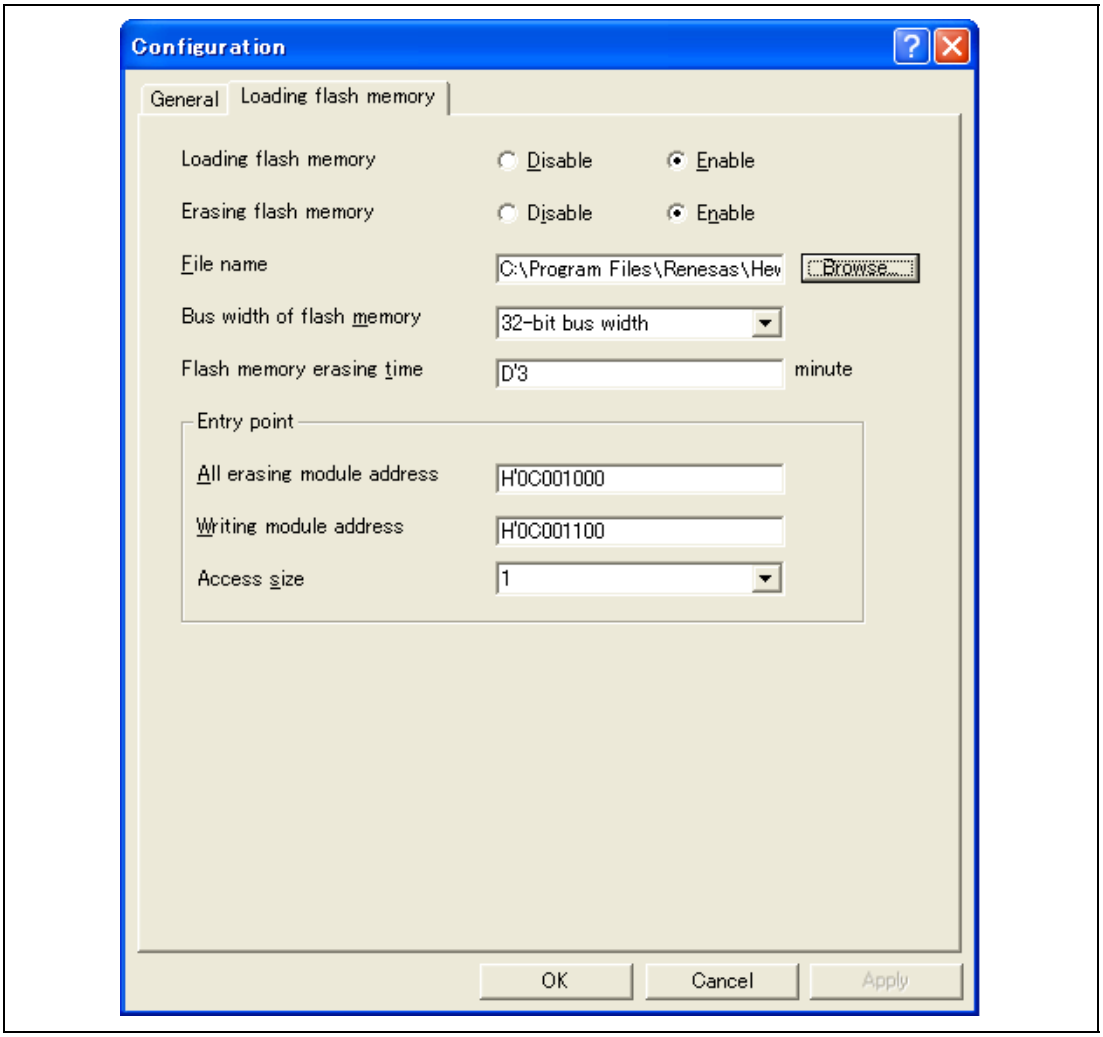


Figure 6.67 [Loading flash memory] Page

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 Maintenance and Guarantee

This section describes maintenance, guarantee, repair provisions, and how to request for repair of the emulator.

7.1 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.

7.2 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.

7.3 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.

7.4 Repair Provisions

7.4.1 Repair with Extra-Charge

The products elapsed more than one year after purchase can be repaired with extra-charge.

7.4.2 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

7.4.3 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.

7.4.4 Transportation Fees at Sending Your Product for Repair

Send your product to us for repair at your expense.

7.5 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 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.

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).

Appendix B Menus

Table B.1 shows GUI menus.

Table B.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.
	TCL toolkit		Shift + Ctrl + L		Opens the [Console] 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.
	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.

Table B.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
View (cont)	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.
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.
		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.

Table B.1 GUI Menus (cont)

Menu	Option	Shortcut	Toolbar Button	Remarks
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.
	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.

Table B.1 GUI Menus (cont)

Menu	Option		Shortcut	Toolbar Button	Remarks
Debug (cont)	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 C 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 D 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] 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 \E10A-USB\Tutorial\Tutorial\Debug_SHxxxx_E10A-USB_ 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 HEW 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 Functions

   — When the PC breakpoint is set in the internal flash memory area, the program is written to the internal flash memory each time the user program is executed. At this time, note that the number of rewritable times will be decreased.

   — 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 255. Therefore, when 255 BREAKPOINTs are set, specification by [Stop At] in the [Run...] menu becomes invalid. Use the BREAKPOINTs and [Stop At] in the [Run...] menu with 255 or less total settings.

12. Note on RUN-TIME Display

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

13. 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 HEW.

14. 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.

15. Memory Access during User Program Execution

When a memory is accessed from the memory window, etc. during user program execution, the user program is resumed after it has stopped in the emulator to access the memory. Therefore, realtime emulation cannot be performed.

The stopping time of the user program is as follows:

Environment:

Host computer: 3.00 GHz (Pentium® 4)
SH72633: 100 MHz (system clock frequency)
JTAG clock: 5 MHz

When a one-byte memory is read from the command-line window, the stopping time will be about 40 ms.

16. BREAKPOINT Setting for SLEEP Instruction

When a break is set for the SLEEP instruction, use the Event Condition instead of the BREAKPOINT.

17. 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

18. 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.

19. Memory Test Function

This product does not support the memory test function, which is used by selecting [Test…] from the [Memory] menu.

20. MCU for use in debugging

However, an actual MCU which has been used in connection with the E10A-USB Emulator for debugging will have been programmed at emulation and subjected to stress accordingly. Do not use an MCU that has been used for debugging in a mass-produced product.

21. Writing Flash Memory Mode

This mode is only intended for writing a user program to the internal flash memory. In this mode, do not attempt anything other than downloading. When microcomputers are to be continuously programmed, be sure to turn the target on or off.

22. Memory Access in the Writing Flash Memory Mode

Memory cannot be accessed in the Writing Flash Memory mode. In this mode, values displayed in the [Memory] or [IO] window are dummy.

23. Memory Access during Flash Memory Programming

During flash memory programming (e.g., user program execution), operation for memory accessing such as opening the [Memory] window is not allowed. Values displayed here are dummy. Access the memory again after flash memory programming has been completed.

24. Host Computer in the Sleep or Hibernating Mode

The host computer must not enter the sleep or hibernating mode while the emulator is in use: otherwise the emulator will not be operable. In such a case, re-connect the emulator after the host computer has left the sleep or hibernating mode.

25. Manual Navigator

Follow the procedure below to execute this program under Windows Vista® or Windows® 7.

Work-around:

(1) Log in with administrative rights.
(2) Open the properties window for file man_navi.exe in the Manuals folder under the installation folder for the High-performance Embedded Workshop.
(3) On the [Compatibility] tabbed page, check the [Run this program as an administrator] box.

Appendix E I/O File Format

The High-performance Embedded Workshop formats the [IO] window based on information it finds in an I/O Register definition file. When you select a debugging platform, the High-performance Embedded Workshop will look for a "<*device*>.IO" file corresponding to the selected device and load it if it exists. This file is a formatted text file that describes the I/O modules and the address and size of their registers. You can edit this file, with a text editor, to add support for memory mapped registers or peripherals you may have specific to your application (e.g. registers in an ASIC device mapped into the microcomputer's address space).

The following describes two formats of the "<*device*>.IO" file that supports or not the bit field.

E.1 File Format (Bit Field Not Supported)

Each module name must be defined in the [Modules] definition section and the numbering of each module must be sequential. Each module corresponds to a register definition section and within the section each entry defines an I/O register.

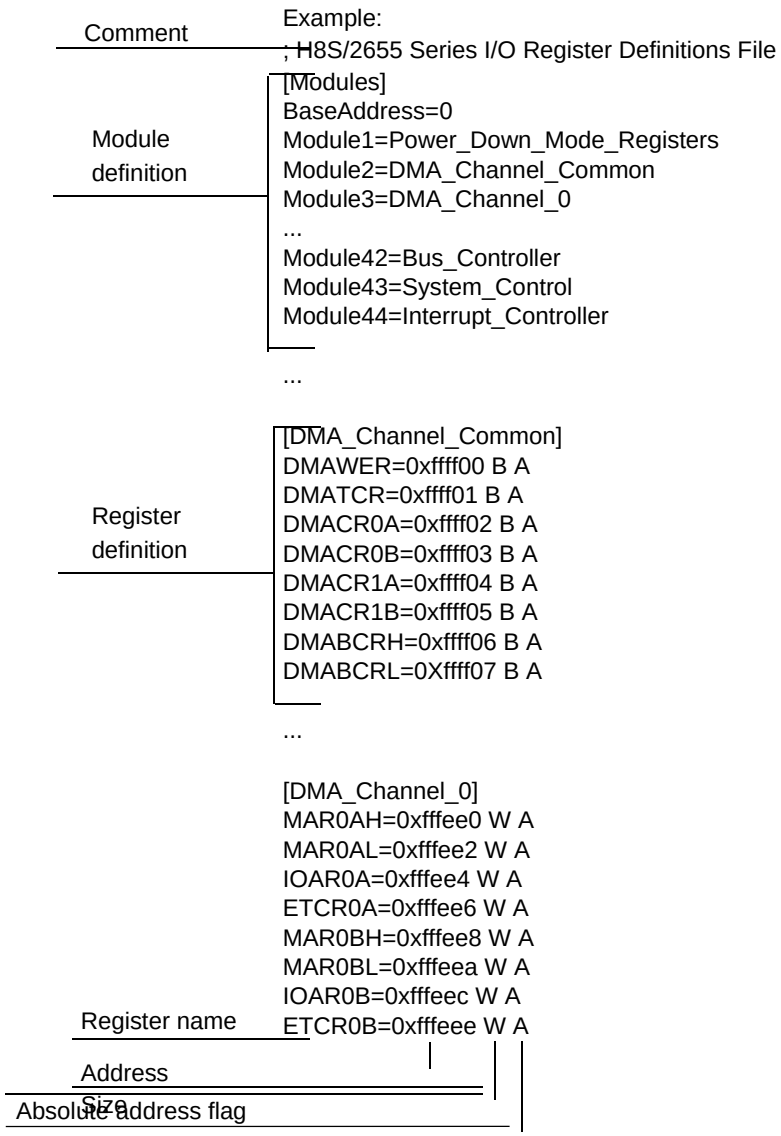
The [BaseAddress] definition is for devices where the location of I/O registers moves in the address space depending on the CPU mode. In this case, the [BaseAddress] value is the base address of the I/O registers in one specific mode and the addresses used in the register definitions are the address locations of the registers in the same mode. When the I/O register file is actually used, the [BaseAddress] value is subtracted from the defined register address and the resultant offset added to the relevant base address for the selected mode.

The [Register] definition entry is entered in the format <name> = <address> [<size> [<absolute>]].

1. <name> register name to be displayed.
2. <address> address of the register.
3. <size> which may be B, W, or L for byte, word, or longword (default is byte).
4. <absolute> which can be set to A if the register is at an absolute address. This is only relevant if the I/O area address range moves about on the CPU in different modes. In this case, if a register is defined as absolute the base address offset calculation is not performed and the specified address is used directly.

Comment lines are allowed and must start with a ";" character.

An example is shown below.



E.2 File Format (Bit Field Supported)

Each module name must be defined in the [Modules] definition section and the numbering of each module must be sequential. Each module corresponds to a register definition section and within the section each entry defines an I/O register.

The user must define “FileVersion=2” at the start of the section. It means that this I/O register file is described with the version that supports the bit field.

The [BaseAddress] definition is for devices where the location of I/O registers moves in the address space depending on the CPU mode. In this case, the [BaseAddress] value is the base address of the I/O registers in one specific mode and the addresses used in the register definitions are the address locations of the registers in the same mode. When the I/O register file is actually used, the [BaseAddress] value is subtracted from the defined register address and the resultant offset added to the relevant base address for the selected mode.

Each module has a section that defines the registers forming it along with an optional dependency. The dependency is checked to see if the module is enabled or not. Each register name must be defined in the section and the numbering of each register must be sequential. The dependency is entered in the section as dep=<reg> <bit> <value>.

1. <reg> is the register id of the dependency.
2. <bit> is the bit position within the register.
3. <value> is the value that the bit must be for the module to be enabled.

The [Register] definition entry is entered in the format id=<name> <address> [<size> [<absolute>[<format>[<bitfields>]]]].

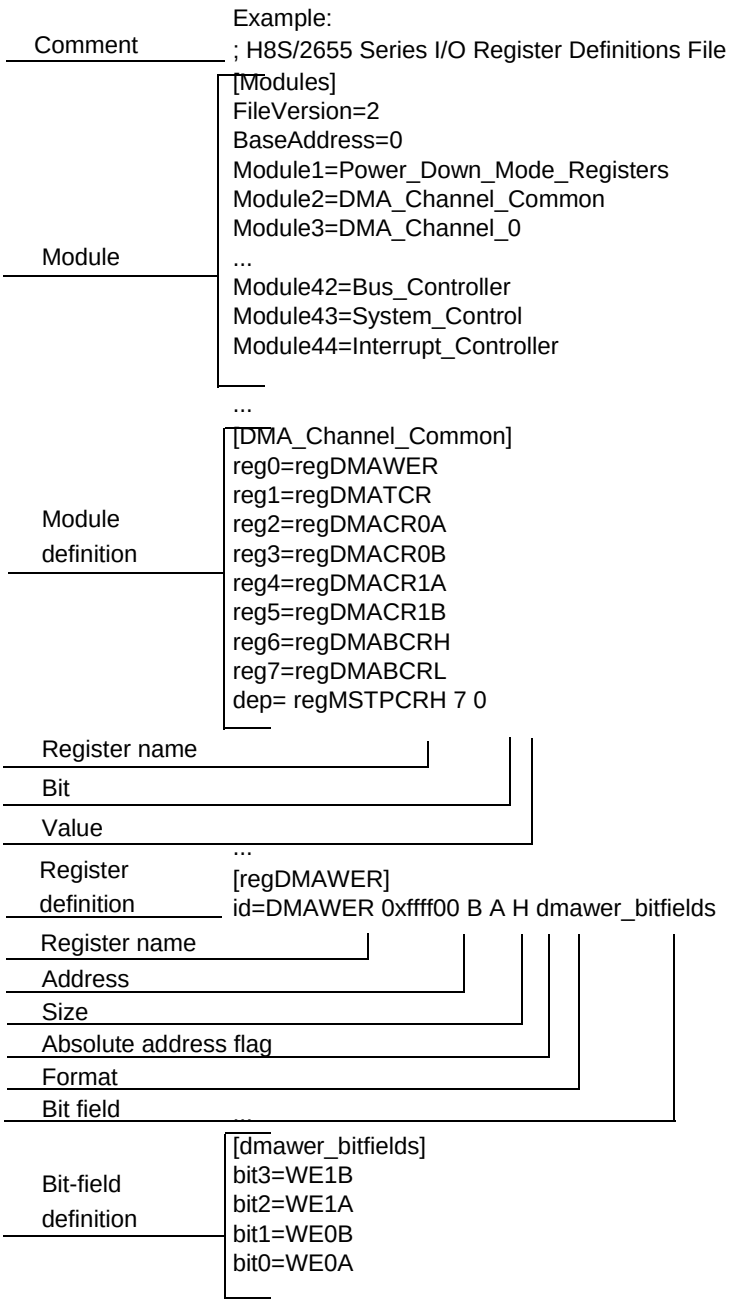
1. <name> register name to be displayed.
2. <address> address of the register.
3. <size> which may be B, W, or L for byte, word, or longword (default is byte).
4. <absolute> which can be set to A if the register is at an absolute address. This is only relevant if the I/O area address range moves about on the CPU in different modes. In this case, if a register is defined as absolute the base address offset calculation is not performed and the specified address is used directly.
5. <format> format for register output. Valid values are H for Hexadecimal, D for decimal, and B for binary.
6. <bitfields> section defining the bits within the register.

Bitfield sections define the bits within a register each entry is of the type bit<no>=<name>.

1. <no> is the bit number.
2. <name> is a symbolic name of the bit.

Comment lines are allowed and must start with a “;” character.

An example is shown below.



Appendix F Diagnostic Test Procedure

For the diagnostic test procedure using the emulator test program, refer to the test program manual for the emulator (file name: E10A-USBTME.PDF) in the CD-R.

Appendix G Repair Request Sheet

Thank you for purchasing the E10A-USB emulator (HS0005KCU01H or HS0005KCU02H).

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: • E10A-USB emulator (HS0005KCU01H or HS0005KCU02H): Serial No.: Revision: The above items are written on the label for product management at the bottom of the emulator unit; the serial no. is the five-digit number and the revision is the string of letters following the number. • Provided CD-R (HS0005KCU01SR): Version: V. Shown as 'V.x.xx release xx' on the CD-R (x: numeral). • Host computer in use: Manufacturer: Type number: OS: (Windows® XP, Windows Vista®, or Windows® 7)
4. Settings when the malfunction occurred	(1) MCU/MPU: Part number: (2) 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.

SuperH™ Family E10A-USB Emulator
User's Manual
(HS0005KCU01H, HS0005KCU02H)

Publication Date: Rev. 10.01 Mar. 25, 2016

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

SuperH™ Family E10A-USB Emulator
User's Manual

Renesas Electronics Corporation

R20UT0870EJ1001