To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

To all our customers

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

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

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

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

Cautions

Keep safety first in your circuit designs!

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

Notes regarding these materials

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

User's Manual

RENESAS

SH7604 E7000PC Graphical User Interface Software

User's Manual

Renesas Microcomputer Support Software

Renesas Electronics
www.renesas.com

2003.04

When using this document, keep the following in mind:

1. This document may, wholly or partially, be subject to change without notice.
2. All rights are reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without Hitachi's permission.
3. Hitachi will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's unit according to this document.
4. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.
5. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.
6. MEDICAL APPLICATIONS: Hitachi's products are not authorized for use in MEDICAL APPLICATIONS without the written consent of the appropriate officer of Hitachi's sales company. Such use includes, but is not limited to, use in life support systems. Buyers of Hitachi's products are requested to notify the relevant Hitachi sales offices when planning to use the products in MEDICAL APPLICATIONS.

Preface

The SH7604 E7000PC graphical user interface software allows Hitachi's in-circuit emulator E7000PC for SH7604 microcomputers to be used for source-level debugging on Windows*[1]. This software thus provides effective application program debugging for systems using the SH7604 MCU.

This manual gives an overview and describes the operating instructions of the SH7604 E7000PC graphical user interface software.

Please refer to the following manuals before using the SH7604 E7000PC graphical user interface software.

For details on the SH7604 hardware:
SH7604 Hardware Manual

For details on the SH7604 in-circuit emulator:
E7000 SH7604 Emulator User's Manual
Description Notes on Using the IBM PC*[2] Interface Board (HS7000EII01H) Used for the E7000PC Emulator or the Compact Evaluation Board

For details on software development support tools:
SH Series C Compiler User's Manual
SH Series Cross Assembler User's Manual
H Series Linkage Editor User's Manual
H Series Librarian User's Manual

Notes: 1. Windows is a trademark of Microsoft Corporation.
2. IBM PC is a registered trademark of International Business Machines Corporation.
3. All other products are trademarks or registered trademarks of corresponding companies.

Symbols Used in This Manual

The following symbols are used in this manual:

___ (underlining): Input

< >: Information within < > must be specified.

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

...: The entry specified just before this symbol can be repeated.

| : Select one of the parameters separated by this symbol or select both.

| | : Select only one of the parameters separated by this symbol.

: Indicates a space or a tab.

(key): Press the key on the keyboard.

(Enter): Press the ENTER key.

(Ctrl+C): Press the CONTROL and C keys simultaneously.

(Alt+B): Press the Alt and B keys simultaneously.

(Alt+Ctrl+A): Press the Alt, CONTROL, and A keys simultaneously.

[Command]: Indicates a menu command opening a window.

[Command...]: Indicates a menu command opening a dialog box.

<Button>: Indicates a button.

C>: Indicates the DOS* prompt.

Note: MS-DOS is a registered trademark of Microsoft Corporation.

Data values in this manual are expressed as follows:

Binary:	Prefixed by B'
Octal:	Prefixed by O'
Decimal:	Prefixed by D'
Hexadecimal:	Prefixed by H'

Data without a prefix is in hexadecimal unless otherwise specified.

The display examples shown in this manual are operating results under Microsoft Windows Version 3.1.

Contents

Section 1 Overview.. 1-1
1.1 Functions and Features 1-1
1.2 Configuration 1-2
1.3 Operating Environment.. 1-3
1.4 File Configuration.. 1-3
1.5 Debug Target Programs 1-4
1.6 Limitations on Debug Target Programs.. 1-5
1.7 Limitations on Functions 1-6

Section 2 Installation 2-1
2.1 Installation Procedure 2-1
2.2 Confirming Operating Environment.. 2-2
2.3 Initiating Windows.. 2-3
2.4 Executing Installer Program 2-4
2.5 Modifying System Files 2-9
2.6 Initiating GUI-SH 2-11

Section 3 Graphical Debugging Environment 3-1
3.1 Base Window 3-1
3.1.1 Base Window Configuration.. 3-1
3.1.2 Menu Bar.. 3-3
3.1.3 Source Information Area.. 3-4
3.1.4 Source Area.. 3-4
3.1.5 Tool Bar 3-6
3.1.6 Emulation Information Area 3-7
3.1.7 Optimization Status Area 3-8
3.1.8 Command Area 3-8
3.2 Debug Windows.. 3-10
3.3 Dialog Box 3-12
3.4 Message Box 3-14
3.5 Edit Function.. 3-15
3.5.1 Character String Search 3-15
3.5.2 Copy to Clipboard.. 3-16
3.6 Mouse 3-16
3.7 Key Input.. 3-17
3.8 Input Format.. 3-18
3.8.1 Expressions 3-18
3.8.2 File Names 3-20
3.9 Short-Cut Keys.. 3-21

Section 4 GUI-SH Operation 4-1
4.1 Emulator Power-On 4-1
4.2 Initiating GUI-SH 4-1
4.3 Initiating Emulator 4-3

4.4 Loading Load Module 4-5
4.4.1 Allocating Emulation Memory 4-5
4.4.2 Loading Program 4-5
4.5 Displaying the Source File 4-7
4.6 Setting and Cancelling Breakpoints 4-9
4.6.1 Setting Breakpoints 4-9
4.6.2 Cancelling Breakpoints 4-10
4.7 Setting Execution Start Address and Stack Pointer 4-11
4.8 Executing Programs 4-12
4.8.1 Executing Programs Using the Tool Bar 4-12
4.8.2 Executing Programs Using the Go Command 4-13
4.8.3 Terminating Programs 4-14
4.9 Displaying Symbol Contents 4-15
4.10 Inputting Emulator Commands 4-16
4.11 Displaying Help Information 4-17
4.11.1 GUI-SH Operation Help 4-17
4.11.2 Emulator Command Help 4-17
4.12 Calling DOS 4-18
4.13 Terminating GUI-SH 4-20

Section 5 Command Reference 5-1
5.1 Menu Command Functions 5-1
5.1.1 Filing Functions (File Menu) 5-1
5.1.2 Execution Functions (Execution Menu) 5-2
5.1.3 Trace Functions (Trace Menu) 5-3
5.1.4 Debugging Information Display Functions (View Menu) 5-4
5.1.5 Help Functions (Help Menu) 5-5
5.2 Menu Command Reference Format 5-6
5.3 Filing Functions 5-8
5.3.1 Loading Program 5-8
5.3.2 Loading Other Files 5-13
5.3.3 Loading Directly into the E7000 5-17
5.3.4 Inputting and Executing Command File 5-21
5.3.5 Storing Command Area Contents 5-23
5.3.6 Terminating GUI-SH 5-26
5.4 Execution Function 5-29
5.4.1 Specifying Emulation Execution Conditions 5-29
5.4.2 Executing Program Emulation 5-32
5.4.3 Setting and Cancelling Breakpoints 5-37
5.4.4 Specifying Hardware Break Condition1 5-40
5.4.5 Specifying Hardware Break Condition2 5-44
5.4.6 Specifying Hardware Break Condition3,4 5-47
5.4.7 Specifying Hardware Break Condition5 5-49
5.5 Trace Function 5-50
5.5.1 Specifying Memory Conditions for Acquiring Trace Information 5-50
5.5.2 Specifying Trace Acquisition Conditions 5-52
5.5.3 Specifying Trace Information Display Conditions 5-59

5.5.4 Displaying Trace Information in Text........ 5-66
5.5.5 Displaying Trace Information in Graph Form........ 5-73
5.6 Debug Information Display Functions........ 5-84
5.6.1 Specifying Memory Display Range........ 5-84
5.6.2 Displaying and Modifying Memory Contents........ 5-86
5.6.3 Displaying and Modifying Register Contents........ 5-91
5.6.4 Setting Watch Points........ 5-94
5.6.5 Displaying Watch Point Contents........ 5-97
5.6.6 Displaying Symbol Information........ 5-99
5.6.7 Displaying and Modifying Symbol Values........ 5-104
5.6.8 Setting Source Area........ 5-118
5.6.9 Displaying and Selecting a Source File........ 5-120
5.6.10 Disassembling and Displaying........ 5-122
5.6.11 Displaying Function Call Sequence........ 5-125
5.6.12 Controlling Command Area........ 5-127
5.7 Help Function........ 5-129
5.7.1 Displaying GUI Operating Help........ 5-129
5.7.2 Displaying Emulator Commands........ 5-130
5.7.3 Describing GUI-SH........ 5-132

Appendix A E7000PC Emulator Command List........ A-1

Appendix B Error Messages........ B-1
B.1 Error Message Levels........ B-1
B.2 List of Error Messages........ B-1
B.2.1 Operation Error Messages........ B-2
B.2.2 System Error Messages........ B-6

Appendix C Command File........ C-1
C.1 Command File Description........ C-1
C.2 Command File Execution at GUI-SH Initiation........ C-1

Appendix D Initial Setting File (guish.ini)........ D-1

Figures

1-1 Hardware Configuration 1-2
1-2 Debug Target Programs 1-4
2-1 Installation Procedure 2-1
2-2 Example of Windows Start-Up Screen 2-3
2-3 File Name Input (INSTALL) 2-4
2-4 Installation Directory Specification (INSTALL) 2-4
2-5 E7000PC System Disk Specification (INSTALL) 2-5
2-6 Addition Confirmation Dialog Box (INSTALL) 2-5
2-7 AUTOEXEC.BAT File Drive Specification (INSTALL) 2-6
2-8 Specification of Memory Address Range for IBM PC Interface Board (INSTALL) 2-7
2-9 Dialog Box for Confirmation of Memory Area Specification Range Rewrite (INSTALL) 2-8
2-10 Installer Termination Message (INSTALL) 2-8
2-11 Program Manager Window and GUI-SH Initiation 2-11
2-12 Start-up Display 2-12
2-13 Backup File Load Confirmation Message Box 2-12
2-14 Emulator Status Confirmation Message Box 2-13
3-1 Base Window Configuration 3-1
3-2 Command List 3-3
3-3 Source Information Area 3-4
3-4 Source Area 3-4
3-5 Tool Bar 3-6
3-6 Emulation Information Area 3-7
3-7 Optimization Status Area 3-8
3-8 Command Area 3-8
3-9 Example of a Debug Window 3-11
3-10 Dialog Box 3-12
3-11 Operation Confirmation Message Box 3-14
3-12 Error Message Box 3-14
3-13 Edit Menu 3-15
3-14 FIND Dialog Box 3-15
3-15 Mouse 3-16
3-16 Example of a Dialog Box for File Operation 3-20
4-1 Display at Windows Initiation 4-1
4-2 Display at GUI-SH Initiation 4-2
4-3 Emulator Monitor Message 4-3
4-4 Emulator System Message 4-4
4-5 Confirmation Message Box for Backup File Load 4-4
4-6 Example of Emulation Memory Allocation with MAP Command 4-5
4-7 LOAD PROGRAM FILE Dialog Box 4-6
4-8 Loading Message Box 4-6
4-9 SOURCE DISPLAY Dialog Box 4-7
4-10 SOURCE SETTING Dialog Box 4-8
4-11 Example of Setting Breakpoints 4-9
4-12 Example of Breakpoint Cancellation 4-10

4-13 Example of Setting Registers with [Register] Command 4-11
4-14 Program Execution Example Using <CONTINUE> Button 4-13
4-15 GO Dialog Box 4-14
4-16 Example of Program Execution Termination with the <STOP> Button 4-14
4-17 Example of Symbol Contents Display 4-15
4-18 Emulator Command Input Format 4-16
4-19 GUI-SH Operation Help Display 4-17
4-20 Emulator Command Help Display 4-18
4-21 DOS Call Procedure 4-18
4-22 DOS Application Execution in a Window 4-19
4-23 Confirmation Message for Backup File Storing 4-20
5-1 Menu for Filing Functions (File Menu) 5-1
5-2 Menu for Execution Functions (Execution Menu) 5-2
5-3 Menu for Trace Functions (Trace Menu) 5-3
5-4 Menu for Debugging Information Display Functions (View Menu) 5-4
5-5 Menu for Help Functions (Help Menu) 5-5
5-6 Command Description Format 5-6
5-7 LOAD PROGRAM FILE Dialog Box 5-8
5-8 LOAD INFORMATION Dialog Box (Load Module File) 5-11
5-9 VERIFY INFORMATION Dialog Box (Load Module File) 5-11
5-10 Verification Error Message 5-11
5-11 LOAD OTHER FILE Dialog Box 5-13
5-12 LOAD INFORMATION Dialog Box (Other File) 5-15
5-13 VERIFY INFORMATION Dialog Box (Other File) 5-15
5-14 Verification Error Message (Other File) 5-16
5-15 E7000 LOAD Dialog Box 5-17
5-16 LOAD INFORMATION Dialog Box (E7000 LOAD) 5-19
5-17 VERIFY INFORMATION Dialog Box (E7000 LOAD) 5-20
5-18 Verification Error Message 5-20
5-19 INPUT FROM Dialog Box 5-21
5-20 Continuation Confirmation Message Box (INPUT FROM) 5-22
5-21 OUTPUT TO Dialog Box 5-23
5-22 File Overwrite Confirmation Message (OUTPUT TO) 5-25
5-23 QUIT Dialog Box 5-26
5-24 BACKUP FILE Dialog Box 5-27
5-25 File Overwrite Confirmation Message (BACKUP FILE) 5-28
5-26 EXECUTION MODE Dialog Box 5-29
5-27 GO Dialog Box 5-32
5-28 BREAK Dialog Box 5-37
5-29 Display Format for Breakpoint Information (BREAK Dialog Box) 5-37
5-30 BREAK CONDITION1 Dialog Box 5-40
5-31 BREAK CONDITION2 Dialog Box 5-44
5-32 BREAK CONDITION3 Dialog Box 5-47
5-33 BREAK CONDITION5 Dialog Box 5-49
5-34 TRACE MEMORY Dialog Box 5-50
5-35 TRACE CONDITION Dialog Box 5-52
5-36 PIN/EXTERNAL PROBE CONDITION Dialog Box 5-55

5-37 TRACE DISPLAY SETTING Dialog Box 5-59
5-38 SEARCH CONDITION Dialog Box 5-61
5-39 TRACE DISPLAY TEXT Window 5-66
5-40 Display Header Format in Instruction Mnemonic Units (TRACE DISPLAY SETTING Window) 5-67
5-41 Display Header Format in Bus-Cycle Units (TRACE DISPLAY SETTING Window) 5-68
5-42 TRACE OUTPUT Dialog Box 5-70
5-43 File Overwrite Confirmation Message (TRACE OUTPUT Dialog Box) 5-71
5-44 TRACE DISPLAY GRAPH TARGET SELECTION Dialog Box 5-73
5-45 TRACE DISPLAY GRAPH Window 5-76
5-46 Trace Information Display for Address Bus Waveform (TRACE DISPLAY GRAPH Window) 5-78
5-47 Trace Information Display for Data Bus Waveform (TRACE DISPLAY GRAPH Window) 5-79
5-48 Trace Information Display for Signal Waveform (TRACE DISPLAY GRAPH Window) 5-80
5-49 Trace Information Display for Memory Change (TRACE DISPLAY GRAPH Window) 5-81
5-50 Trace Information Display for Execution Time (TRACE DISPLAY GRAPH Window) 5-82
5-51 Zooming Function Example (TRACE DISPLAY GRAPH Window) 5-83
5-52 MEMORY RANGE SETTING Dialog Box 5-84
5-53 MEMORY DUMP Window 5-86
5-54 Example of Modifying Memory Contents (MEMORY DUMP Window) 5-88
5-55 MEMORY OUTPUT Dialog Box 5-89
5-56 File Overwrite Confirmation Message (MEMORY OUTPUT Dialog Box) 5-90
5-57 REGISTER Window 5-91
5-58 Example of Modifying Register Contents (REGISTER Window) 5-93
5-59 WATCH SETTING Dialog Box 5-94
5-60 Watch Point Display Format (WATCH SETTING Dialog Box) 5-95
5-61 WATCH DISPLAY Window 5-97
5-62 Watch Point Contents Display Format (WATCH DISPLAY Window) 5-97
5-63 SYMBOL DISPLAY Window 5-99
5-64 Execution Stop Address Example 5-100
5-65 Display Format of Symbol Information Display Area (SYMBOL DISPLAY Window) 5-101
5-66 SYMBOL VALUE Window 5-104
5-67 Symbol Value Display Format (SYMBOL VALUE Window) 5-105
5-68 Symbol Value Modification Procedure (SYMBOL VALUE Window) 5-111
5-69 Modification Example of a char Pointer Variable (SYMBOL VALUE Window) 5-112
5-70 Modification Example of a long-Type Integer Variable (SYMBOL VALUE Window) 5-113
5-71 Modification Example of a double Floating-Point Variable (SYMBOL VALUE Window) 5-114
5-72 Modification Example of a short-Type Six-Dimensional Array Variable (SYMBOL VALUE Window) 5-115
5-73 SOURCE SETTING Dialog Box 5-118

5-74 SOURCE DISPLAY Dialog Box 5-120
5-75 DISASSEMBLE Window........ 5-122
5-76 ROUTE Dialog Box........ 5-125
5-77 Closed Command Area........ 5-127
5-78 GUI OPERATING HELP Window 5-129
5-79 EMULATOR COMMAND DISPLAY Window 5-130
5-80 EMULATOR COMMAND HELP Window 5-131
5-81 ABOUT Window 5-132
A-1 LOAD Command Syntax (Emulator Command) A-5
A-2 INPUT Command Syntax A-6
C-1 Command File Example C-1
C-2 Command File Execution at GUI-SH Initiation C-1
D-1 Initial Setting File (guish.ini) Contents........ D-1

Tables

1-1 GUI-SH Operating Environment..1-3
1-2 GUI-SH File Configuration..1-3
1-3 Limitations on Debug Target Programs..1-5
1-4 Limitations on Functions..1-6
3-1 Button Functions..3-6
3-2 Special Key Codes in Command Area..3-9
3-3 Debug Windows..3-10
3-4 Buttons in Dialog Box..3-13
3-5 Short-Cut Keys..3-21
4-1 Program Execution Buttons in Tool Bar..4-12
5-1 Output Status Messages (OUTPUT TO Dialog Box)..5-23
5-2 Output File Status Message and Buttons in OUTPUT TO Dialog Box..5-25
5-3 Information Stored in Backup File (QUIT Dialog Box)..5-28
5-4 Emulation Termination Causes (GO Dialog Box)..5-35
5-5 Hardware Break Condition Specifications According to Access Size (Break Condition1)..5-42
5-6 Trace Modes..5-56
5-7 Trace Acquisition Condition Settings According to Bus Size..5-57
5-8 Trace Search Condition Settings According to Bus Size..5-64
5-9 [Signal] List in [Trace display - Graph]..5-75
5-10 Types of Graph..5-77
5-11 Storage Class Display Format (SYMBOL DISPLAY Window)..5-101
5-12 Symbol Attribute Display Format (SYMBOL DISPLAY Window)..5-102
5-13 Symbol Type Display Format (SYMBOL VALUE Window)..5-107
5-14 Floating-Point Display Format (SYMBOL VALUE Window)..5-108
5-15 Symbol Value Display Examples (SYMBOL VALUE Window)..5-109
5-16 Symbol Value Display Field Characters (SYMBOL VALUE Window)..5-111
5-17 File Type and Disassembly Display (DISASSEMBLE Window)..5-124
A-1 E7000PC Emulator Commands..A-1
B-1 Operation Error Messages..B-2
B-2 System Error Messages..B-6

Section 1 Overview

1.1 Functions and Features

The SH7604 E7000PC graphical user interface software (GUI-SH) allows Hitachi's SH7604 E7000PC in-circuit emulator (E7000PC emulator) to be operated on Windows, thus enabling the E7000PC emulator to be used for efficient source-level debugging. The GUI-SH thus provides effective application program debugging for systems using the SH7604 MCU.

The GUI-SH provides the following functions and features:

- Source-level debugging
  - Displays the source program (assembly language or C) and execution points.
  - Sets and cancels breakpoints from the source program.
  - Displays the values of symbols selected from the source program in a specific window.
  - Executes program in units of one source program line displayed in a window (single-step execution).
  - Acquires trace information during program execution, and displays the source program contents corresponding to the trace information.
- Multi-windows
  - Provides multi-windows (displays multiple windows at the same time).
  - Provides frequently used commands in button form, enabling efficient operation by mouse.
- Powerful debugging functions
  - Displays and modifies memory, register, and symbol contents in windows.
  - Displays subroutine call sequence.
  - Enables easy program and signal analysis by displaying trace information in graphical form.
- Multi-load module file
  Loads multi-load module files and debugs all the integrated load modules.

1.2 Configuration

Figure 1-1 shows the hardware configuration required for GUI-SH operation.

The GUI-SH operates under the control of Windows on a personal computer. The personal computer must be connected to the E7000PC emulator station via an IBM PC interface board to allow the computer and emulator to interact. The E7000PC emulator station must be connected to the user system via the emulator pod.

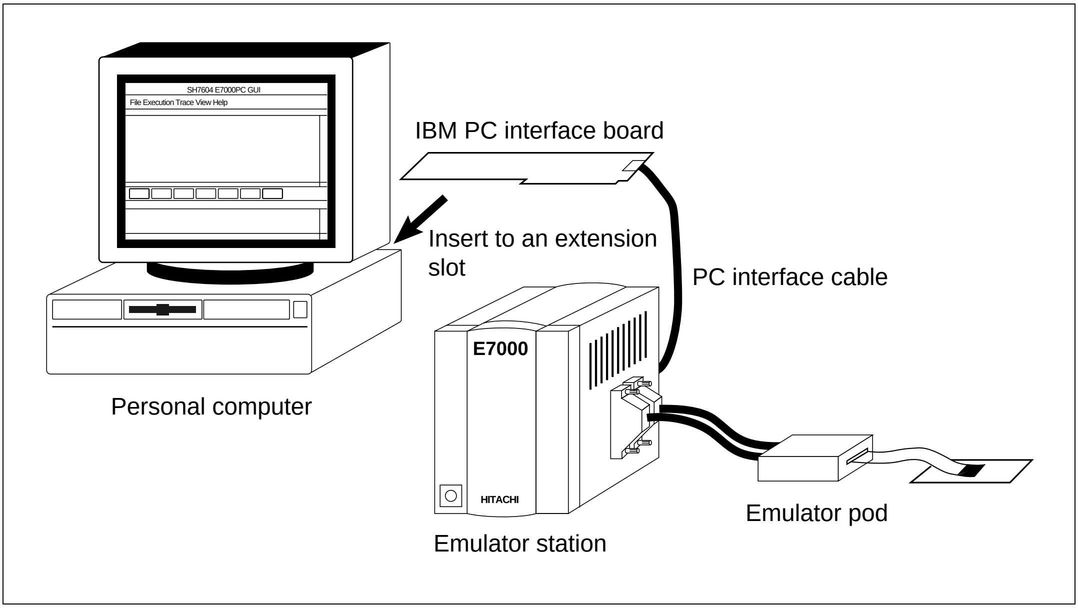


Figure 1-1 Hardware Configuration

1.3 Operating Environment

The GUI-SH operating environment is shown in table 1-1.

Table 1-1 GUI-SH Operating Environment

Item	Operating Environment
Personal computer	IBM PC or compatible machine having an 80386 or 80486 CPU, and an AT bus, on which Microsoft Windows Version 3.1 operates
OS	MS-DOS under which Microsoft Windows Version 3.1 operates
Windows	Microsoft Windows Version 3.1
Minimum operating memory	640-kbyte or larger main memory and 8-Mbyte or larger protect memory in 386 enhanced mode
Display	Color or monochrome display which can be connected to the personal computer and can run Windows (VGA or higher is recommended)
Hard disk	2-Mbyte or larger free space available (space for installation)
Mouse	Any mouse which can be connected to the personal computer and can operate under Microsoft Windows Version 3.1.

Note: The GUI-SH may output display contents to a file on the hard disk. If a large amount of data is output, an error message for insufficient memory may be displayed. In this case, make at least 3 Mbytes of free space available on the disk.

1.4 File Configuration

The GUI-SH software is provided on a 1.44-Mbyte floppy disk. The GUI-SH file configuration is shown in table 1-2.

Table 1-2 GUI-SH File Configuration

File Name	Contents
GUISH.EXE	GUI-SH execution file
GUISHOP.HLP	Help file explaining GUI-SH basic operations
GUISH.INI	File specifying initial values at GUI-SH initiation
INSTALL.EXE	Program file installing the GUI-SH

1.5 Debug Target Programs

Figure 1-2 shows programs that can be debugged using the GUI-SH. To perform source-level debugging using the GUI-SH, an absolute load module file must be created with specifying the debug option by the compiler, assembler, or linkage editor and then it must be loaded to the GUI-SH.

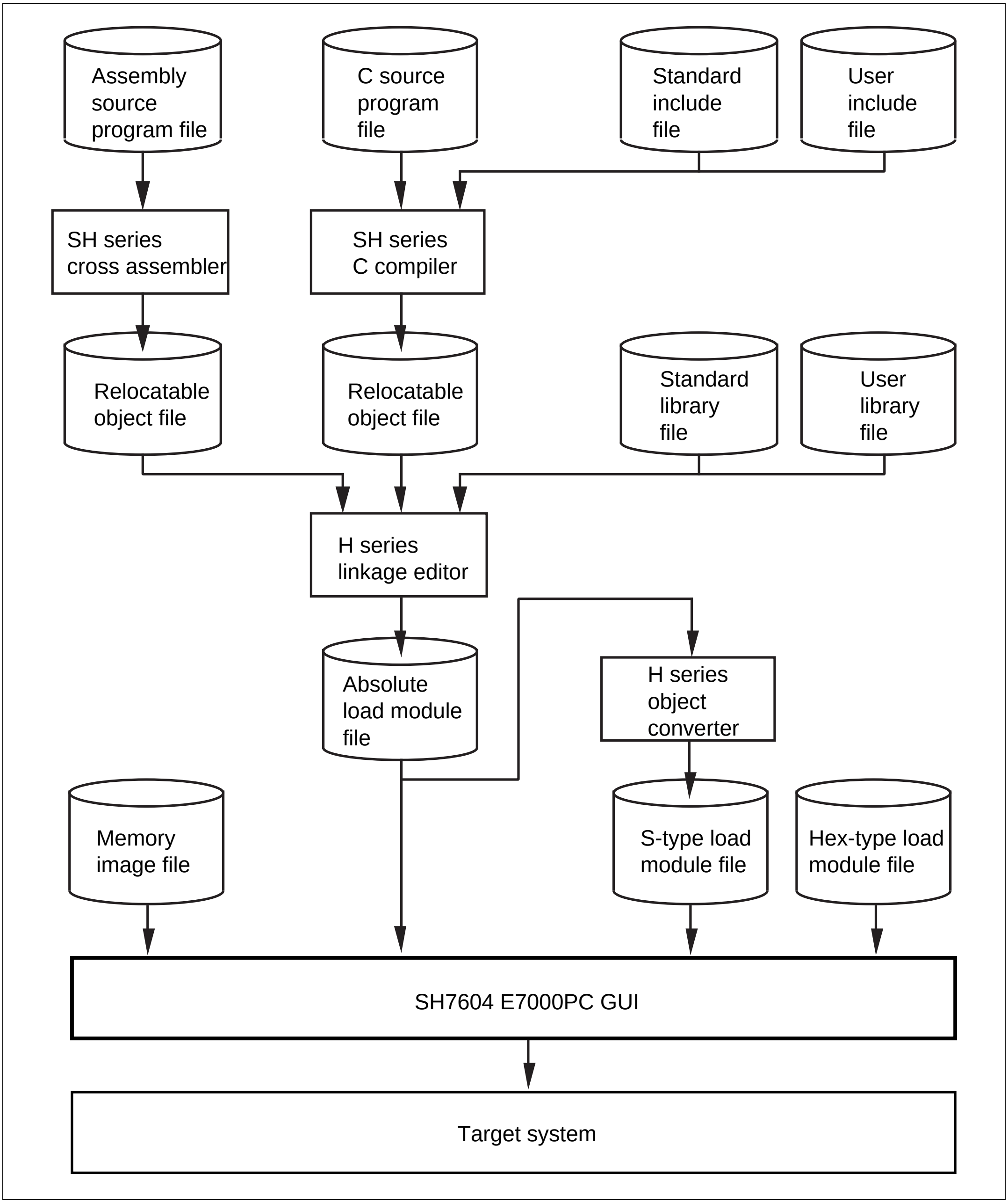


Figure 1-2 Debug Target Programs

1.6 Limitations on Debug Target Programs

The limitations on programs that can be debugged using the GUI-SH are listed in table 1-3.

Table 1-3 Limitations on Debug Target Programs

Item	Maximum	Remarks
Symbols	16,777,215 symbols	65,535 symbols/unit x 2,340 units (When not limited by memory capacity)
Symbol name length	32 characters	Up to 255 characters can be used for structure symbol names
Segments	2,729 segments	
Units in one segment	2,340 units	
Sections in one segment	4,094 sections	
Functions in one unit	1,364 functions	
Files in one unit	16,383 files	
Lines in one file	32,767 lines	

1.7 Limitations on Functions

The limitations on the GUI-SH functions are listed in table 1-4.

Table 1-4 Limitations on Functions

Item	Maximum	Remarks
File name length	115 characters	Limited by the DOS file system
Data length	255 characters	
Breakpoints set at one time	255 points	
Waveforms drawn at one time	32 waveforms	
Memory locations changed at one time	256 points	
Watch points	8 points	
Bytes of information displayed for one watch point	32 bytes	
Symbols displayed at one time	16 symbols	For SUMBOL VALUE command
Symbol contents locations changed at one time	256 points	
Bytes of information displayed for one symbol	20 kbytes	
Nest levels displayed at function call sequence display	64 nests	
Characters input in the command area	78 characters	Limited by the maximum number of command input characters in E7000PC

Section 2 Installation

2.1 Installation Procedure

Install the GUI-SH using the procedure shown in figure 2-1.

(1) Confirm operating environment.
Check the operating environment of the personal computer on which the GUI-SH is to operate. (For details, refer to section 2.2, Confirming Operating Environment.)

(2) Connect the E7000PC emulator to the personal computer.
Install the IBM PC* interface board in the personal computer, and connect the E7000PC interface cable. (For details, refer to the Description Notes on Using the IBM PC Interface Board Used for the E7000PC Emulator or the Compact Evaluation Board (HS7000EIL01H).)

(3) Initiate Windows.
Initiate the Windows program on which the GUI-SH is to be installed. (For details, refer to section 2.3, Initiating Windows.)

(4) Execute the installer program.
Create a directory to store the GUI-SH and copy the files from the provided disk and emulator system disk to the directory. Create a GUI-SH group and register the GUI-SH icon. (For details, refer to section 2.4, Executing Installer Program.)

(5) Modify system files.
Modify files such as CONFIG.SYS and AUTOEXEC.BAT (For details, refer to section 2.5, Modifying System Files.)

(6) Initiate the GUI-SH.
(For details, refer to section 2.6, Initiating GUI-SH.)

Figure 2-1 Installation Procedure

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

2.2 Confirming Operating Environment

Check that the personal computer and peripheral devices are correctly connected before installing the GUI-SH. Windows must be correctly installed before the GUI-SH is installed. Confirm the following before installing the GUI-SH:

- 2 Mbytes or more free space is available on the hard disk to which the GUI-SH is to be installed.

- Windows has been correctly installed and must be able to be initiated in 386 enhanced mode.

2.3 Initiating Windows

Initiate Windows using the following procedure:

(1) Initiate DOS from the hard disk.

(2) Initiate Windows from the hard disk. After Windows has been installed and placed in subdirectory \windows on hard disk drive C, enter as follows:

```
C>cd \windows(Enter)
C>win(Enter)
```

If Windows has been installed in another drive or subdirectory, change the current directory to that subdirectory, and then enter win (Enter).

If the subdirectory in which Windows has been installed is registered as a path name, the current directory change command (CD command) need not be executed.

Once Windows is initiated, the display shown in figure 2-2 appears. (The display may be different depending on the Windows environment.)

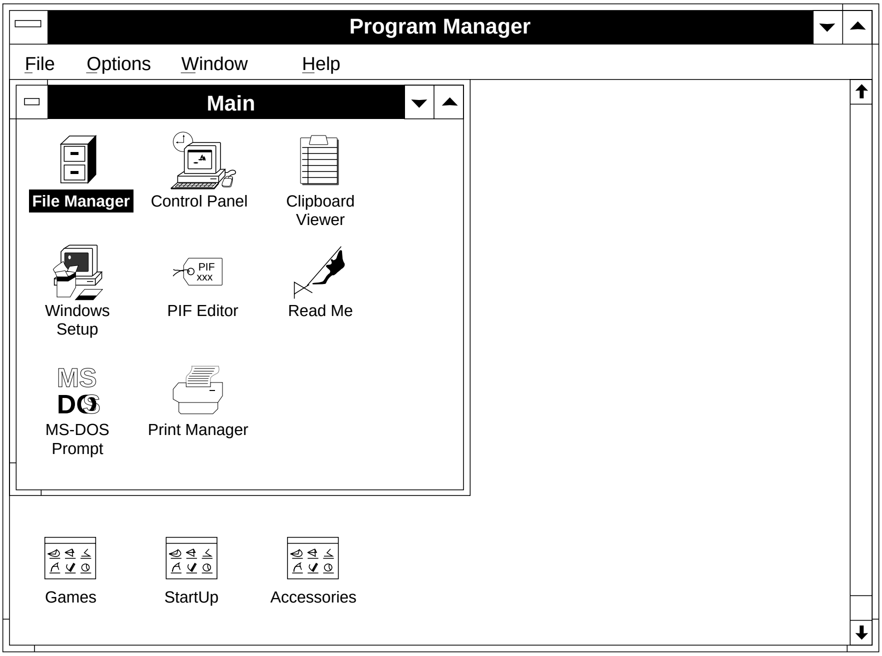


Figure 2-2 Example of Windows Start-Up Screen

2.4 Executing Installer Program

The GUI-SH is installed on Windows by initiating program INSTALL.EXE from the provided disk. Before initiating the INSTALL.EXE, terminate all Windows tasks other than the program manager.

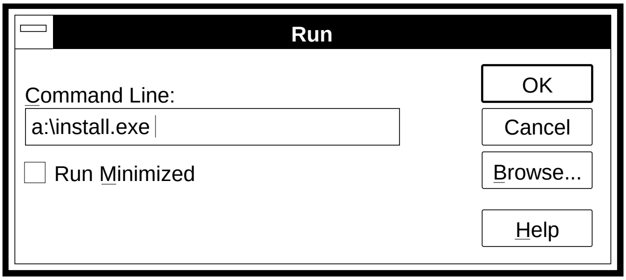
Initiate the INSTALL.EXE by the following procedure:

(1) Insert the provided disk into floppy disk drive A.

(2) Click the [File] menu on the Program Manager window.

(3) Select the [RUN...] command by clicking on it; the dialog box for the [File - RUN...] command will be displayed.

(4) Enter a:\install.exe in the file name text box as shown in figure 2-3.



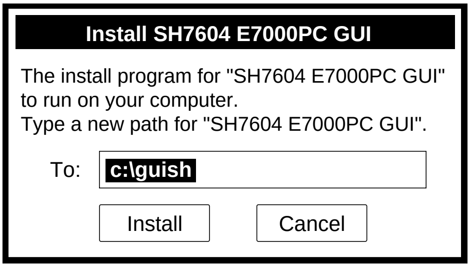
Command Line: a:\install.exe

□ Run Minimized

Figure 2-3 File Name Input (INSTALL)

(5) Click the <OK> button to initiate installer program.

(6) When the installer program is initiated, a window is displayed prompting the user to specify the directory in which the GUI-SH is to be installed as shown in figure 2-4. The installer program copies the files necessary for GUI-SH execution to the directory specified in this window. When the specified directory already exists, a message box is displayed to confirm whether the user wants to overwrite the directory.



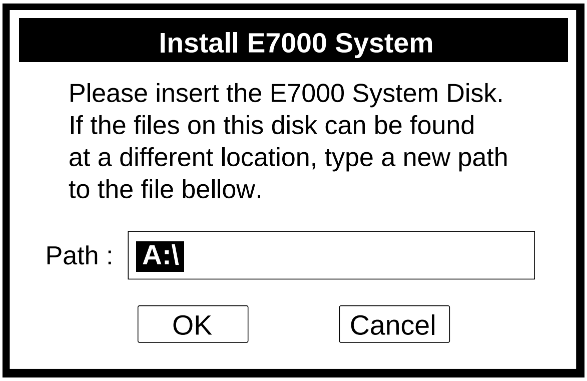
The install program for "SH7604 E7000PC GUI" to run on your computer.
Type a new path for "SH7604 E7000PC GUI".

To: c:\guish

Figure 2-4 Installation Directory Specification (INSTALL)

(7) Enter the directory name and click the <Install> button, which starts copying the necessary files.

(8) When the files necessary for GUI-SH execution have been copied, a dialog box prompting the user to locate the E7000PC system disk as shown in figure 2-5. Insert the E7000PC system disk into a floppy disk drive and click the <OK> button, which starts copying the E7000PC system files. Clicking the <Cancel> button terminates installation.



Install E7000 System

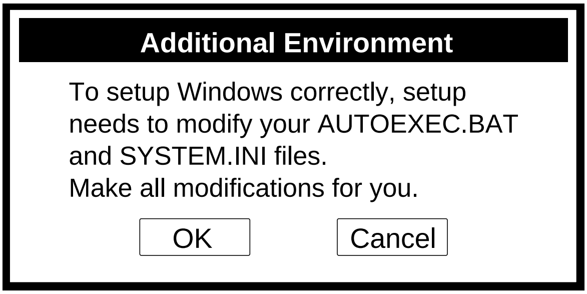
Please insert the E7000 System Disk.
If the files on this disk can be found
at a different location, type a new path
to the file bellow.

Path : A:\

Figure 2-5 E7000PC System Disk Specification (INSTALL)

(9) When the E7000PC system files have been copied, a dialog box is displayed to confirm whether the user wants to add to the AUTOEXEC.BAT the path name of the directory in which the GUI-SH has been installed and whether the user wants to modify the SYSTEM. INI, as shown in figure 2-6.

To add the path name to the AUTOEXEC.BAT or modify the SYSTEM.INI using the installer, click the <OK> button. To terminate the installer, click the <Cancel> button. In this case, change the required file using the procedure described in section 2.5, Modifying System Files.

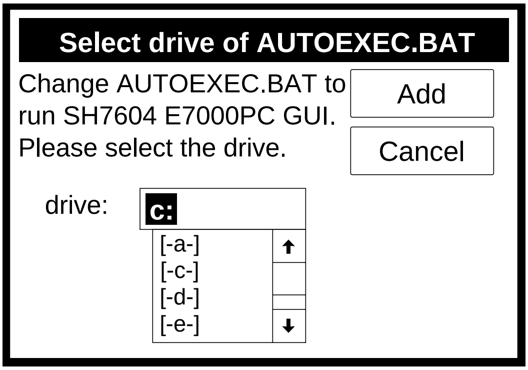


Additional Environment

To setup Windows correctly, setup
needs to modify your AUTOEXEC.BAT
and SYSTEM.INI files.
Make all modifications for you.

Figure 2-6 Addition Confirmation Dialog Box (INSTALL)

(10) Click the <OK> button; a dialog box is displayed prompting the user to specify the drive storing the AUTOEXEC.BAT file as shown in figure 2-7. Click or input from the keyboard the drive name storing AUTOEXEC.BAT.



Select drive of AUTOEXEC.BAT

Change AUTOEXEC.BAT to run SH7604 E7000PC GUI. Please select the drive.

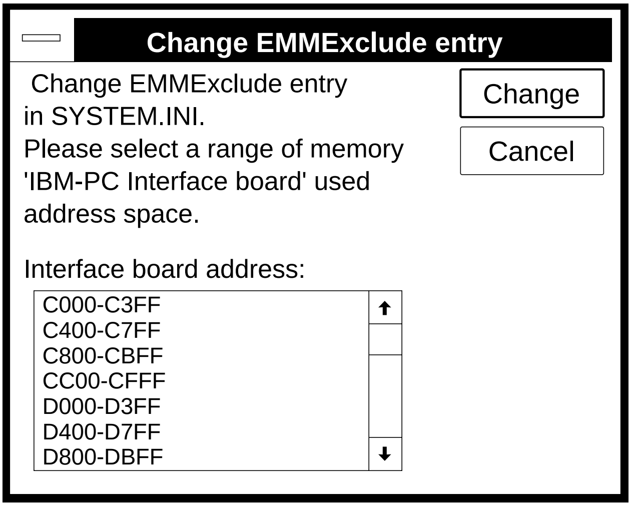
drive: c:

Figure 2-7 AUTOEXEC.BAT File Drive Specification (INSTALL)

(11) To add to the AUTOEXEC.BAT file, environment variable GUISHPATH and the directory path in which the GUI-SH has been installed, click the <Add> button. The file before adding the path name is saved as AUTOEXEC.GUI.

To terminate the program without changing the AUTOEXEC.BAT file, click the <Cancel> button. In this case, change the AUTOEXEC.BAT using the procedure described in section 2.5, Modifying System Files.

(12) A dialog box is displayed to select the memory address range for the IBM PC interface board, as shown in figure 2-8. Specify the memory address range. For example, select "D000-D3FF" to specify the range from D000:0000 - D3FF:000F.



Change EMMExclude entry

Change EMMExclude entry in SYSTEM.INI.
Please select a range of memory 'IBM-PC Interface board' used address space.

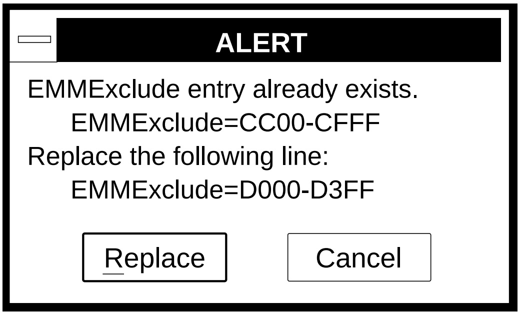
Interface board address:

C000-C3FF
C400-C7FF
C800-CBFF
CC00-CFFF
D000-D3FF
D400-D7FF
D800-DBFF

Figure 2-8 Specification of Memory Address Range for IBM PC Interface Board (INSTALL)

(13) To add to the Windows system file SYSTEM.INI the memory address range for the IBM-PC interface board, click the <Change> button. The file before adding the memory address is saved as SYSTEM.GUI. To terminate the installer without changing the SYSTEM.INI file, click the <Cancel> button. In this case, change the SYSTEM.INI using the procedure described in section 2.5, Modifying System Files.

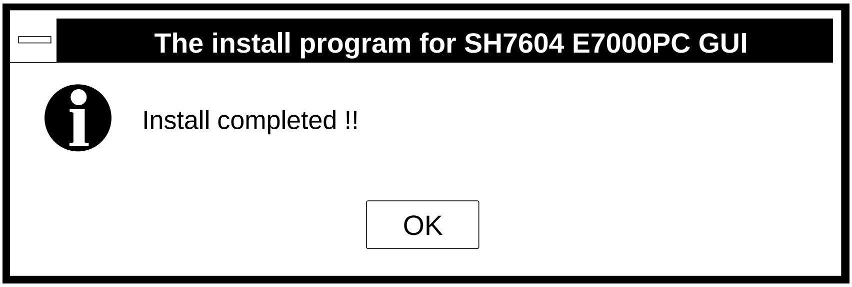
(14) To add the memory address range for the IBM PC interface board, EMMExclude parameter of the SYSTEM.INI file is used. If the memory address range has already been specified, a rewrite confirmation message is displayed as shown in figure 2-9. Select <Replace> or <Cancel> according to the operating personal computer.



EMMExclude entry already exists.
EMMExclude=CC00-CFFF
Replace the following line:
EMMExclude=D000-D3FF

Figure 2-9 Dialog Box for Confirmation of Memory Address Specification Range Rewrite (INSTALL)

(15) The installer program terminates.



Install completed !!

Figure 2-10 Installer Termination Message (INSTALL)

2.5 Modifying System Files

CAUTION!

Back up the CONFIG.SYS, AUTOEXEC.BAT, and SYSTEM.INI files before modifying the files.

(1) Terminate Windows.
Select [eXit] from the [File] menu of the program manager to terminate Windows.

(2) Modify CONFIG.SYS.
The GUI-SH accesses the IBM PC interface board. When a virtual EMS driver installation is specified in CONFIG.SYS, the memory addresses for the IBM PC interface board must be set outside the range managed by the virtual EMS driver.

An example of how to set the memory addresses for the IBM PC interface board to D000:0000–D3FF:000F, which is outside the range managed by the virtual EMS driver (EMM386) and how to specify the base address of the EMS page frame as E000:0000, is shown below (enter the underlined part).

```
DEVICE=C:\WINDOWS\EMM386.EXE 1024 RAM x=D000-D3FF frame=E000
```

Specify the memory address range selected in (12) of section 2.4, Executing Installer Program.

The memory addresses for the IBM PC interface board can be changed using the DIP switch on the board. For details, refer to the Description Notes on Using the IBM PC Interface Board Used for the E7000PC Emulator or the Compact Evaluation Board.

(3) Modify AUTOEXEC.BAT.
When AUTOEXEC.BAT has not been modified by the installer program, set the directory path and environment variable in AUTOEXEC.BAT. If the directory name for the GUI-SH described in section 2.4, Executing Installer Program is GUISH, modify as follows:

```
PATH C:\;C:WINDOWS;C:\GUISH
SET GUISHPATH=C:\GUISH
```

Note: Add the underlined parts.

(4) Modify SYSTEM.INI
When the memory address range for the IBM PC interface board has not been specified using the installer program, modify the SYSTEM.INI, which is a Windows initialization file, to specify the memory address range for the IBM PC interface board outside the Windows memory management area. The SYSTEM.INI file exists in the Windows directory. Specify the EMMExclude in the [386Enh] section in the SYSTEM.INI file. To specify the memory address range for the IBM PC interface board [D000:0000-D3FF:000F] outside the Windows memory management area, modify as follows:

```
[386Enh]
EMMExclude = D000–D3FF
```

Note: Add the underlined part.

(5) Initiate DOS.
When DOS file modification is completed, temporarily terminate DOS, then re-initiate it.

2.6 Initiating GUI-SH

Initiate the GUI-SH using the following procedure:

(1) Initiate Windows. Immediately after GUI-SH installation, temporarily terminate Windows and DOS, and then re-initiate them.

(2) Power on the emulator.

(3) Using the left button on the mouse, double-click the GUI-SH icon in the GUI-SH group window within the Program Manager window, as shown in figure 2-11.

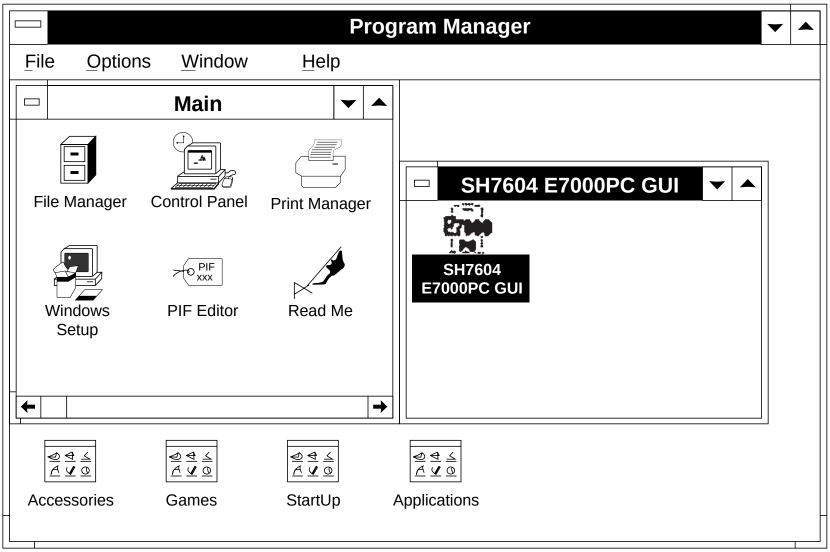
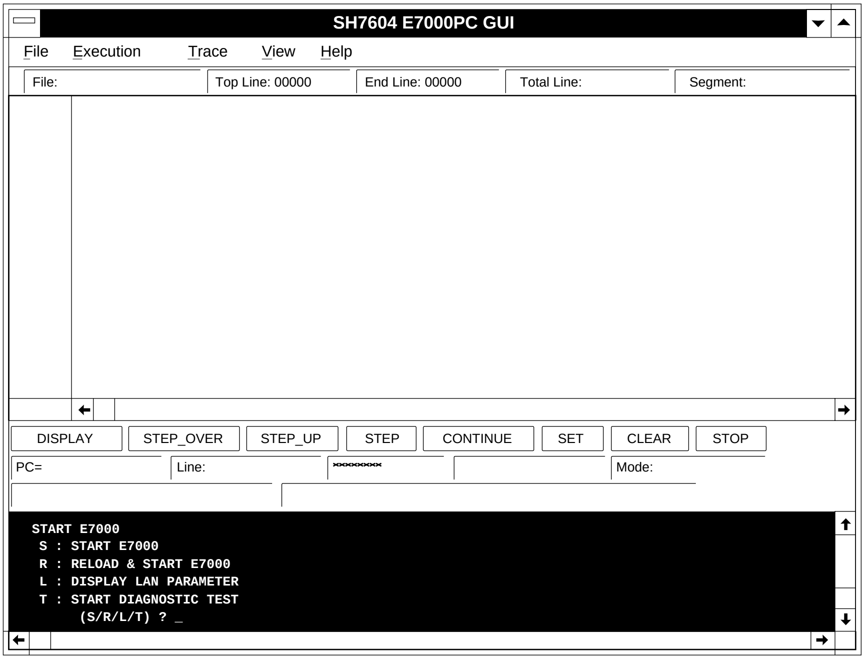


Figure 2-11 Program Manager Window and GUI-SH Initiation

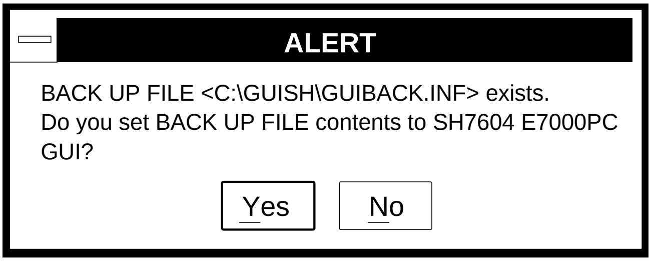
(4) When the GUI-SH is initiated, the display shown in figure 2-12 appears. The E7000PC emulator start-up message is displayed at the bottom, and the system waits for monitor command input. Enter the S command. For other monitor commands, refer to the E7000 SH7604 Emulator User's Manual. When initiation is completed, the WARM/COLD start selection request message is displayed. When the (Enter) key is pressed to select the COLD start, the GUI-SH prompts for emulator command input by displaying a colon (:).



```
  START E7000
   S : START E7000
   R : RELOAD & START E7000
   L : DISPLAY LAN PARAMETER
   T : START DIAGNOSTIC TEST
        (S/R/L/T) ? _
```

Figure 2-12 Start-up Display

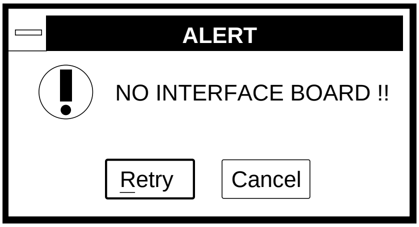
(5) When a backup file is found, the message box shown in figure 2-13 is displayed to confirm operating environment backup file load. To load the operating environment backup file previously saved, click the <Yes> button. To initiate the GUI-SH without loading the backup file, click the <No> button.



BACK UP FILE <C:\GUISH\GUIBACK.INF> exists.
Do you set BACK UP FILE contents to SH7604 E7000PC GUI?

Figure 2-13 Backup File Load Confirmation Message Box

(6) If the E7000PC emulator power is off, or if the IBM PC interface board or interface cable is not correctly connected, the message box shown in figure 2-14 is displayed. In this case, power on the E7000PC emulator and click the <OK> button, or terminate the GUI-SH by clicking the <Cancel> button and check the hardware settings.



ALERT

NO INTERFACE BOARD !!

Figure 2-14 Emulator Status Confirmation Message Box

Section 3 Graphical Debugging Environment

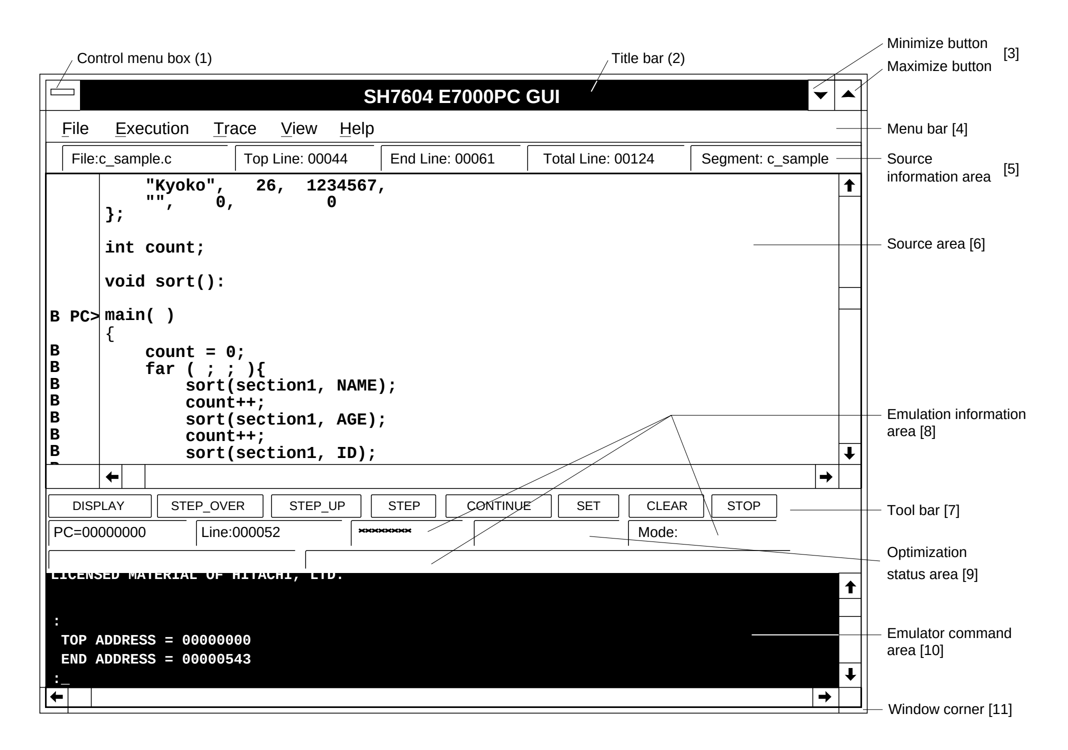
This section describes each window and operation of the GUI-SH.

3.1 Base Window

3.1.1 Base Window Configuration

After GUI-SH initiation, the base window shown in figure 3-1 opens.

The base window provides source-level debugging functions such as displaying the source file and symbol contents, executing programs, and setting and cancelling breakpoints.



File:c_sample.c | Top Line: 00044 | End Line: 00061 | Total Line: 00124 | Segment: c_sample

```
           "Kyoko",   26,  1234567,
           "",    0,         0
      };

      int count;

      void sort():

B PC> main( )
      {
B          count = 0;
B          far ( ; ; ){
B              sort(section1, NAME);
B              count++;
B              sort(section1, AGE);
B              count++;
B              sort(section1, ID);
```

PC=00000000 | Line:000052 | ×××××××× | Mode:

```
LICENSED MATERIAL OF HITACHI, LTD.

 :
  TOP ADDRESS = 00000000
  END ADDRESS = 00000543
 :_
```

Figure 3-1 Base Window Configuration

The base window consists of the following:

(1) Control menu box
Clicking this box opens the base window control menu.

(2) Title bar
The title of the GUI-SH program is displayed. Clicking the title bar activates the window. When active, the window can receive key entries and commands.

(3) Minimize button and maximize button
Clicking the minimize button reduces the base window to an icon. Clicking the maximize button enlarges the base window to its maximum display size.

(4) Menu bar
Displays command menus available for use. Clicking a command menu displays a list of commands included in the command menu.

(5) Source information area
Displays information on the source file displayed in the source area.

(6) Source area
Displays the source file being debugged.

(7) Tool bar
Displays buttons for frequently used commands. Clicking a button directly executes the command.

(8) Emulation information area
Displays emulation information during program execution or at termination.

(9) Optimization status area
Displays whether or not the source file displayed in the source area has been optimized.

(10) Emulator command area
Area for inputting emulator commands. The command area can be closed or opened by the [Command area] command in the View menu.

(11) Window corner
Dragging the corner changes the window size.
Note that tool bar buttons will be obstructed from view when the window size is reduced.

3.1.2 Menu Bar

The menu bar displays command menus available for use.

Place the mouse pointer on a menu name displayed on the menu bar and click the left button of the mouse; a command list as shown in figure 3-2 is displayed.

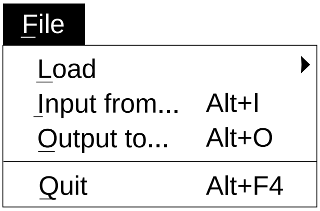


Figure 3-2 Command List

Commands in command lists can be executed by the following procedure:

Click the command to be executed using the left button of the mouse, or drag the mouse until the command name becomes highlighted and release the mouse button.

Symbols used in command lists indicate the following meaning:

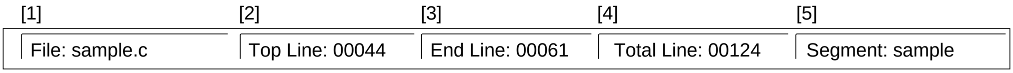
<command>...: Displays a dialog box.
<command>▶ : The command has a subcommand menu.
<command>: The command can also be executed by entering only the underlined part of the command name from the keyboard.

When the selected command has already been executed and the window is open, the window is brought to the top of the screen.

The key name, such as (Alt + F4), displayed on the right of each command name in the command list indicates the short-cut key for the command. Only inputting the keys executes the corresponding command.

3.1.3 Source Information Area

The source information area displays information on the source program displayed in the source area. Figure 3-3 shows the source information area. This area displays the following information:



File: sample.c | Top Line: 00044 | End Line: 00061 | Total Line: 00124 | Segment: sample

Figure 3-3 Source Information Area

(1) File: (source file name)
Source program file name

(2) Top Line: (first line number)
Line number of the first line displayed

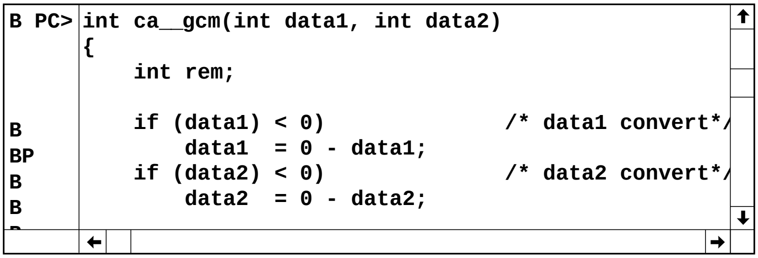
(3) End Line: (last line number)
Line number of the last line displayed

(4) Total Line: (total number of lines)
Total number of lines in the source program

(5) Segment: (segment name)
Segment name of the source program

3.1.4 Source Area

The source area displays a section of the source file corresponding to the current program counter (PC). Figure 3-4 shows the source area display.



```
B PC> int ca__gcm(int data1, int data2)
      {
          int rem;

B         if (data1) < 0)              /* data1 convert*/
BP            data1  = 0 - data1;
B         if (data2) < 0)              /* data2 convert*/
B             data2  = 0 - data2;
```

Figure 3-4 Source Area

When a program is loaded by the [Load-Load program file...] command in the File menu, a section of the source file corresponding to the execution start address of the program is displayed. When program execution stops, for example, at a breakpoint, the section of the source file corresponding to that address is displayed.

The displayed source file can be scrolled using the scroll bar in the source area. The source file can be changed using the [Source - Display...] command in the View menu. In addition, the font, font size, and tab width of the source file can be changed using the [Source - Setting...] command in the View menu.

The PC mark (PC>) indicating the current program counter location and B marks (B) indicating lines where breakpoints can be set are displayed at the left of the source file. After setting the breakpoint, BP marks (BP) are displayed to indicate lines to which breakpoints are set.

By clicking the source program in the source area with the left button of the mouse, a cursor (|) is displayed to indicate the clicked position. The <CONTINUE>, <SET>, and <CLEAR> buttons in the tool bar operate after the cursor is displayed.

Double-clicking a word in the source program highlights and selects the word. Dragging with the left button of the mouse highlights and selects text. The <DISPLAY> button in the tool bar, the character string search function, and function copying the text to the clipboard operate on the selected text.

3.1.5 Tool Bar

The tool bar consists of buttons corresponding to frequently used commands. Clicking a button executes the corresponding command. Figure 3-5 shows the buttons in the tool bar.

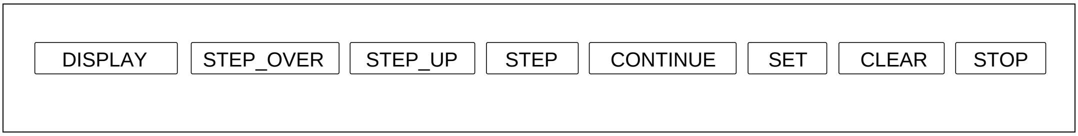


Figure 3-5 Tool Bar

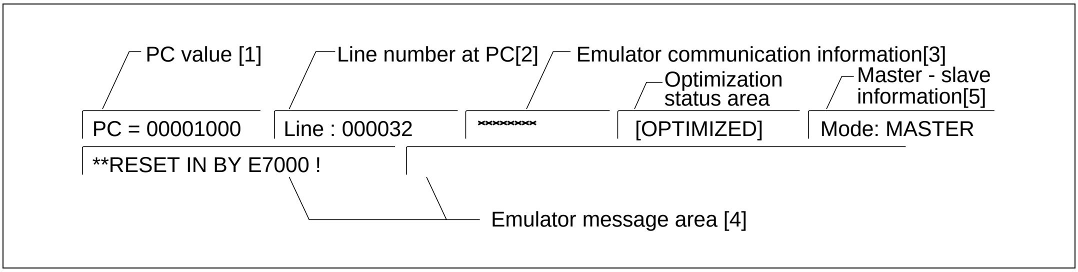
Button functions are listed in table 3-1.

Table 3-1 Button Functions

Button	Short-Cut Keys	Function
<DISPLAY>	(Alt + N)	Adds the symbol selected in the source area to the SYMBOL VALUE window.
<STEP_OVER>	(F7)	Executes one program line pointed to by the current program counter (PC) and then stops. When the line includes a function (subroutine) call, program execution stops after executing the function. This button cannot be used in the ROM area.
<STEP_UP>	(F8)	Executes the function including the current program counter (PC), and stops execution when returning to the upper function (calling function). This button cannot be used in the ROM area.
<STEP>	(F9)	Executes one program line pointed to by the current program counter (PC) and stops. When the line includes a function (subroutine) call, program execution stops at the first line of the function.
<CONTINUE>	(Alt + G)	Executes the program from the line pointed to by the current program counter (PC). Execution continues until a line including a cursor is reached. Note that execution stops only when the line marked with B in the source area are clicked.
<SET>	(Alt + B)	Sets a breakpoint to a line including a cursor in the source area. A breakpoint can be specified only to a line marked with a B in the source area.
<CLEAR>	(Alt + C)	Cancels the breakpoint set to a line including a cursor in the source area.
<STOP>	(Ctrl + C)	Forcibly stops program execution.

3.1.6 Emulation Information Area

The emulation information area displays emulation information during program execution or at execution termination. Figure 3-6 shows the emulation information area.



```
PC = 00001000    Line : 000032    ********    [OPTIMIZED]    Mode: MASTER
**RESET IN BY E7000 !
```

PC value [1]
Line number at PC[2]
Emulator communication information[3]
Optimization status area
Master - slave information[5]
Emulator message area [4]

Figure 3-6 Emulation Information Area

The emulation information area displays the following information:

(1) PC = xxxxxxxx

Displays the current program counter value. During program execution, the value is updated to the current program counter value. At execution termination, the program counter value indicates the next address to be executed.

(2) Line:

Displays the line number of the source file corresponding to the program counter value at execution termination.

(3) Emulator communication information

Communication status between the GUI-SH and the emulator is displayed as follows:

ACCESS: Sending a command to the emulator
PROCESS: Receiving a command from the emulator
*******: Not communicating with the emulator

(4) Emulator message

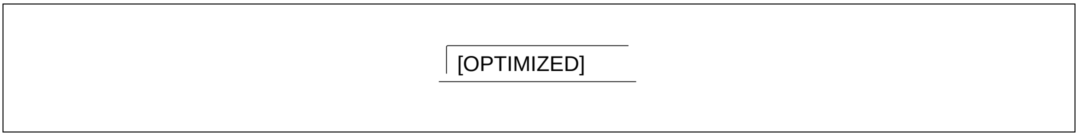
Displays the cause of program execution termination and execution time. When <Enable> is specified in [Display memory trace data] of the [Go...] command, the contents of the memory address specified in [Trace memory condition] of the [Trace memory...] command is displayed during emulation.

(5) Master-slave information

When simultaneously operating two or more emulators, operating status is displayed as <MASTER> and <SLAVE>.

3.1.7 Optimization Status Area

The optimization status area displays (OPTIMIZED) as shown in figure 3-7 if the program displayed in the source area has been optimized. If not, nothing is displayed.

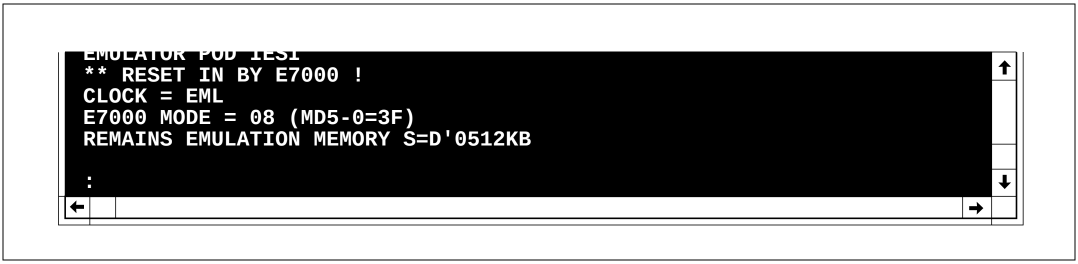


[OPTIMIZED]

Figure 3-7 Optimization Status Area

3.1.8 Command Area

The command area is used for inputting emulator commands. Figure 3-8 shows the command area.



```
EMULATOR POD TEST
** RESET IN BY E7000 !
CLOCK = EML
E7000 MODE = 08 (MD5-0=3F)
REMAINS EMULATION MEMORY S=D'0512KB

:
```

Figure 3-8 Command Area

Input emulator commands in the command area using the following procedure:

(1) Click the command area using the left button of the mouse to display the cursor (_).

(2) When the emulator command input prompt (: or #) is not displayed, press the (Enter) key.

(3) After the prompt, input emulator commands.

Command execution results are also displayed in the command area. Former execution results can be referenced by scrolling the display with the scroll bar in the command area.

The command area window size can be changed by the [Command area] command in the View menu.

Special key codes can be used in the command area as shown in table 3-2.

By using the history function ((Alt + ↑) keys or (Alt + ↓) keys), the previously input command can be displayed, edited, and executed.

For the emulator commands that can be used in the command area, refer to appendix A, E7000 Emulator Command List.

Table 3-2   Special Key Codes in Command Area

Buttons	Function
(Ctrl + C)	Stops command execution or program execution.
(Backspace)	Inserts a backspace.
(Delete)	Deletes the character at the cursor.
(Esc)	Cancels the characters on the input line.
(Home)	Moves the cursor to the beginning of the input line.
(End)	Moves the cursor to the last character on the input line.
(Page UP)	Scrolls up by one page.
(Page Down)	Scrolls down by one page.
(↑)	Scrolls up by one line.
(↓)	Scrolls down by one line.
(Alt + ↑)	Displays the previously input command (history function).
(Alt + ↓)	Displays the previously input command (history function).
(←) or (→)	Moves the cursor left or right.
(Space)	Enters parallel mode when pressed during GO command execution.
(Enter)	Executes the command. Enters parallel mode when pressed during GO command execution
(Ctrl + Tab)	Switches the cursor display to the source area (If this key code is input again, the cursor display returns to the command area.)

3.2 Debug Windows

The GUI-SH includes the debug windows shown in table 3-3 in addition to the base window, which can be opened at the same time during debugging.

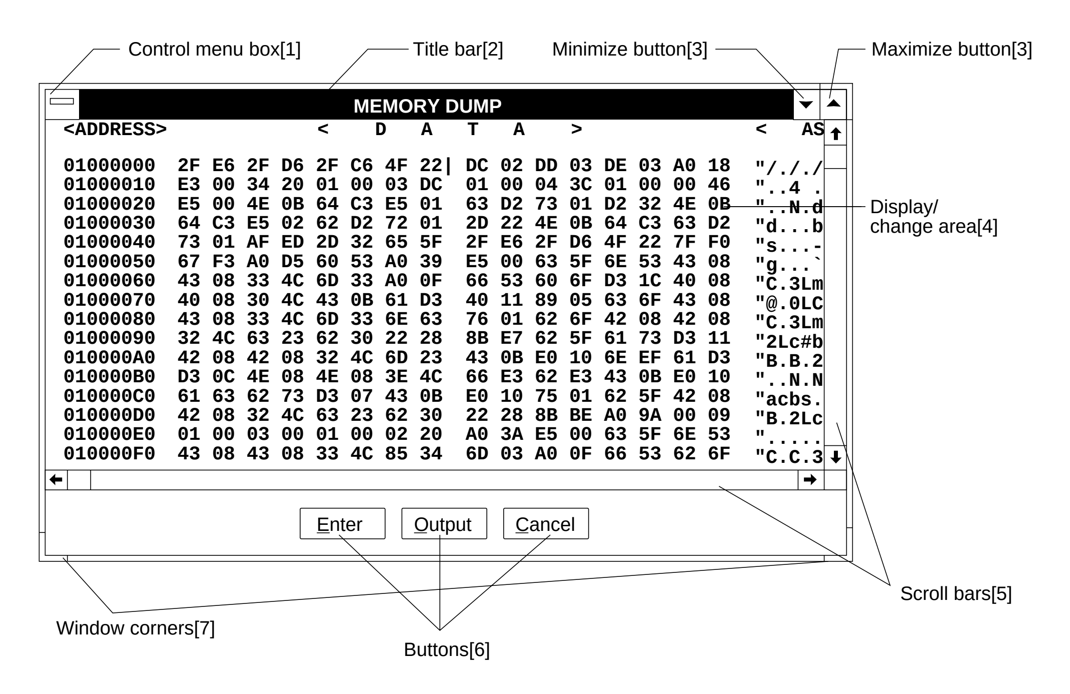
Each window opens by a corresponding command. To close a window, click the <Cancel> button in the window or select <Close> in the control menu box.

In some of the windows, the contents of the window can be output to a file or specific text in the windows can be searched for.

Table 3-3 Debug Windows

Window	Function	File Output	Text Search
Trace text	Displays trace information as text	Enabled	Enabled
Trace graph	Displays trace information as a graph	Disabled	Disabled
Memory dump	Displays memory contents in dump format and allows memory contents to be modified	Enabled	Enabled
Watch	Displays contents of the watch point	Disabled	Enabled
Register	Displays register contents	Disabled	Enabled
Symbol display	Displays symbol attributes	Disabled	Enabled
Symbol value display	Displays symbol values and allows values to be changed	Disabled	Enabled
Disassembly display	Disassembles and displays a program in the source area	Disabled	Enabled
Function call sequence display	Displays function call sequence up to the one including the current program counter value	Disabled	Enabled
Emulator help	Displays a list of emulator commands	Disabled	Enabled
Emulator command help	Displays how to use emulator commands	Disabled	Enabled
GUI operating help	Displays how to use graphical debugging environment	Disabled	Enabled

Figure 3-9 shows an example of a debug window configuration.



```
<ADDRESS>                <   D   A   T   A    >               <   AS

01000000  2F E6 2F D6 2F C6 4F 22| DC 02 DD 03 DE 03 A0 18  "/././
01000010  E3 00 34 20 01 00 03 DC  01 00 04 3C 01 00 00 46  "..4 .
01000020  E5 00 4E 0B 64 C3 E5 01  63 D2 73 01 D2 32 4E 0B  "..N.d
01000030  64 C3 E5 02 62 D2 72 01  2D 22 4E 0B 64 C3 63 D2  "d...b
01000040  73 01 AF ED 2D 32 65 5F  2F E6 2F D6 4F 22 7F F0  "s...-
01000050  67 F3 A0 D5 60 53 A0 39  E5 00 63 5F 6E 53 43 08  "g...`
01000060  43 08 33 4C 6D 33 A0 0F  66 53 60 6F D3 1C 40 08  "C.3Lm
01000070  40 08 30 4C 43 0B 61 D3  40 11 89 05 63 6F 43 08  "@.0LC
01000080  43 08 33 4C 6D 33 6E 63  76 01 62 6F 42 08 42 08  "C.3Lm
01000090  32 4C 63 23 62 30 22 28  8B E7 62 5F 61 73 D3 11  "2Lc#b
010000A0  42 08 42 08 32 4C 6D 23  43 0B E0 10 6E EF 61 D3  "B.B.2
010000B0  D3 0C 4E 08 4E 08 3E 4C  66 E3 62 E3 43 0B E0 10  "..N.N
010000C0  61 63 62 73 D3 07 43 0B  E0 10 75 01 62 5F 42 08  "acbs.
010000D0  42 08 32 4C 63 23 62 30  22 28 8B BE A0 9A 00 09  "B.2Lc
010000E0  01 00 03 00 01 00 02 20  A0 3A E5 00 63 5F 6E 53  ".....
010000F0  43 08 43 08 33 4C 85 34  6D 03 A0 0F 66 53 62 6F  "C.C.3
```

Figure 3-9 Example of a Debug Window

(1) Control menu box

(2) Title bar
The title of the debug window is displayed. Clicking the title bar activates the window. Once activated, the window can receive key entries and commands.

(3) Minimize button and maximize button

(4) Display/change area
Displays debugging information. The contents of some windows can be directly changed.

(5) Scroll bars
Scrolls the display/change area.

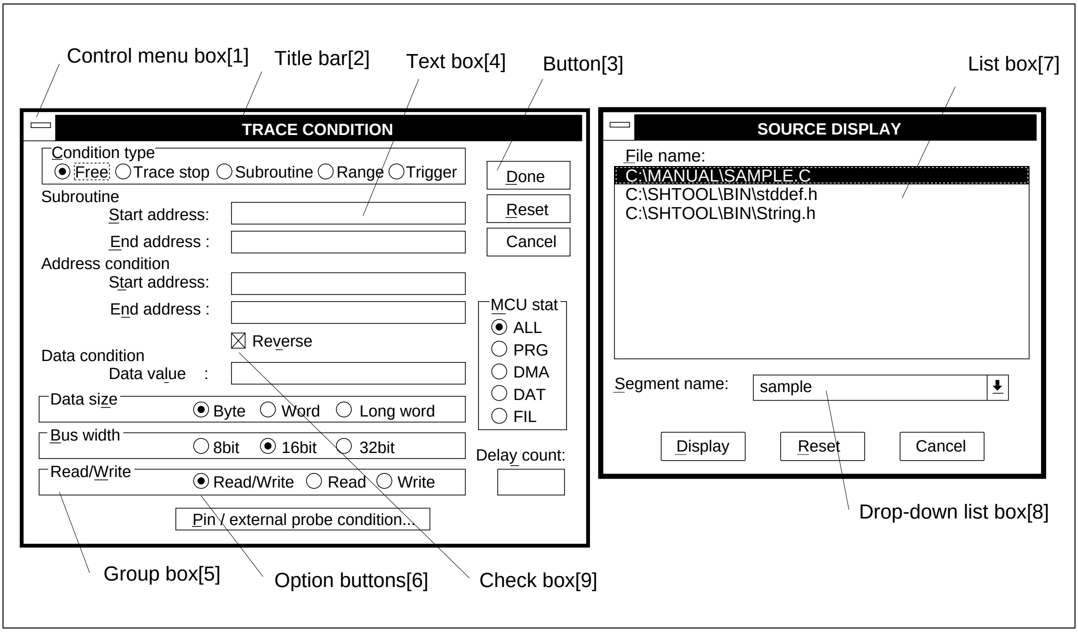
(6) Buttons
Include functions such as outputting window contents to a file or closing the window.

(7) Window corners
Dragging a corner changes the window size.

Note: When the window is reduced, the buttons may be obstructed from view.

3.3 Dialog Box

A dialog box is a window used for inputting parameters necessary for command execution. Figure 3-10 shows a dialog box.



TRACE CONDITION

Condition type: ● Free ○ Trace stop ○ Subroutine ○ Range ○ Trigger

Subroutine
Start address:
End address :

Address condition
Start address:
End address :

☒ Reverse

Data condition
Data value :

Data size: ● Byte ○ Word ○ Long word

Bus width: ○ 8bit ● 16bit ○ 32bit

Read/Write: ● Read/Write ○ Read ○ Write

MCU stat: ● ALL ○ PRG ○ DMA ○ DAT ○ FIL

Delay count:

SOURCE DISPLAY

File name:
C:\MANUAL\SAMPLE.C
C:\SHTOOL\BIN\stddef.h
C:\SHTOOL\BIN\String.h

Segment name: sample

Figure 3-10 Dialog Box

(1) Control menu box

(2) Title bar
The title of the dialog box is displayed. Clicking the title bar activates the dialog box. Once activated, the window can receive key entries and commands.

(3) Buttons
The buttons used in the dialog box and their functions are listed in table 3-4.

Table 3-4 Buttons in Dialog Box

Button	Function
<Done>	Activates the input parameters and executes the command
<Cancel>	Cancels the input parameters and closes the dialog box
<Reset>	Resets the parameters to the default values
<Command>	Executes the <Command> command using the input parameters
<Button...>	Opens another dialog box to input more detailed parameters
<Close>	Closes the dialog box

(4) Text box
Receives text from the keyboard such as addresses or symbols. Click the box with the mouse to display the input pointer (|), and then enter from the keyboard.

(5) Group box
Includes items related to each other.

(6) Option buttons
Exclusively selects one input parameter. Only one button can be selected.

(7) List box
Lists parameters from which required ones can be selected. When the box cannot display all parameters, a scroll bar is displayed. The list can be scrolled using the mouse or keyboard.

(8) Drop-down list box
A list box with an arrow is called a drop-down list box. Clicking the arrow displays the list box.

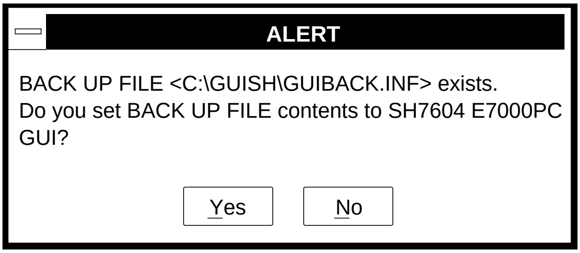
(9) Check box
Options which can be turned on and off. When selected, a x is displayed in the box. When not selected, nothing is displayed in the box.

(10) Tri-state buttons
The pin conditions and external probe signal levels can be selected from H (high), L (low), and * (high or low). Clicking the button changes the indication on the button and the setting.

3.4 Message Box

The message box is displayed when command operation needs to be confirmed or if an error occurs. The following two types of message boxes will be displayed:

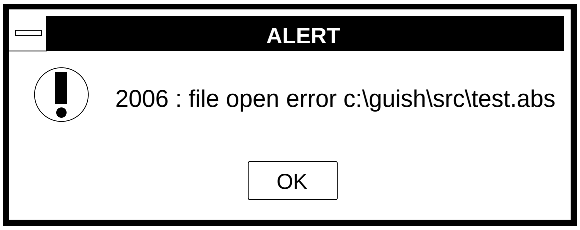
- Operation confirmation message box
  The message box shown in figure 3-11 will be displayed when GUI-SH operations must be selected at GUI-SH initiation and termination and during command execution. Click the <Yes> or <No> button with the mouse. The <Yes> button can also be selected by pressing the <Y> key or <Enter> key, and the <No> button by the <N> key or <Esc> key.



BACK UP FILE <C:\GUISH\GUIBACK.INF> exists.
Do you set BACK UP FILE contents to SH7604 E7000PC GUI?

Figure 3-11 Operation Confirmation Message Box

- Error message box
  If an error occurs, the error message box shown in figure 3-12 will be displayed. After checking the error contents, click the <OK> button or press the <Enter> key to continue.



2006 : file open error c:\guish\src\test.abs

Figure 3-12 Error Message Box

3.5 Edit Function

When a debug window is active, clicking the right button of the mouse displays the edit menu shown in figure 3-13, which enables character strings to be searched and copied to the clipboard.

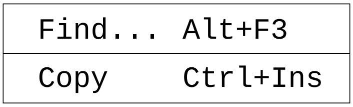


Figure 3-13 Edit Menu

3.5.1 Character String Search

Selecting the [Find...] command from the edit menu opens the dialog box for character string search (hereafter called the FIND dialog box) shown in figure 3-14.

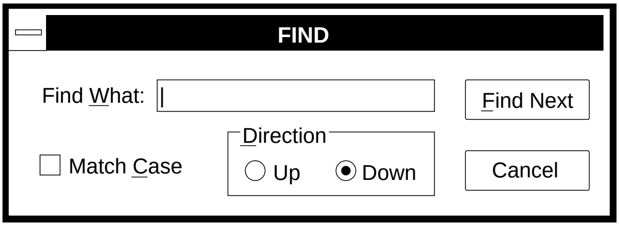


Figure 3-14 FIND Dialog Box

Enter in the [Find What:] text box the character string to be searched for, and click the <Find Next> button; the character string begins to be searched for from the current cursor position. When the character string is found, the found character string is highlighted in the window. To find the next occurrence of the character string, click the <Find Next> button. The search direction can be specified with the [Direction] group box. Selecting <Up> searches text from the cursor position to the top of the text, and <Down> searches text to the end of the text.

For the source area, the source file currently displayed in the area can be searched.

To terminate the search operation and close the FIND dialog box, click the <Cancel> button.

When a character string is selected in the active window, the character string automatically enters the [Find What:] text box by opening the FIND dialog box. To select a word in the window, double-click the word. To select a character string, drag the mouse to highlight the character string.

Click the <Match Case> check button to search for characters while distinguishing between upper-case and lower-case letters.

3.5.2 Copy to Clipboard

Selecting the [Copy] command from the edit menu copies the selected character string to the clipboard. To select a character string, drag or double-click with the left button of the mouse.

3.6 Mouse

Almost all operations on the GUI-SH can be performed by the mouse shown in figure 3-15. Values such as addresses, however, must be input from the keyboard.

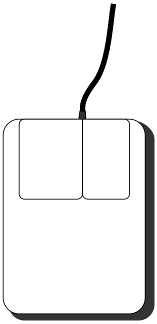


Figure 3-15 Mouse

The left button of the mouse is usually used, but for edit function, the right button is used. For text search, refer to section 3.5.1, Character String Search.

The mouse location is shown as a mouse pointer on the display. The mouse pointer can be moved to another point as needed by moving the mouse. The mouse pointer is an arrow (↑) or an I beam (|) depending on the window.

Basic mouse operations are as follows:

- Click
  Press the left button and quickly release it. Click is used to select a window, menu, list, and button.

- Double click
  Press the left button twice quickly. Double click is used to activate the item selected in a list box and to select a word in a window.

- Drag
  Press and hold the button while moving the mouse. Drag is used to select a character string in a window.

3.7 Key Input

All operations on the GUI-SH can be performed from the keyboard. To operate using the keyboard, move to the window to be operated on, and then move to the target operation (subwindow or button).

Operate the GUI-SH with the keyboard as follows:

(1) Activate the window to be operated on. To activate a window, first enter (Alt-Space) from the keyboard to open the control menu. Then, select <Switch to (w)> using the arrow keys and move to the desired window.

The window can also be switched using the (Ctrl + Esc) keys, (Alt + Tab) keys, or (Alt + Esc) keys.

(2) When the base window is active, use the following procedure:
Using the (Alt) key in combination with certain key(s), or short-cut keys will open the command menus. The tool bar button can be operated by the short-cut key. The tool bar function included in the Execution and the View menus can also be used as a menu command.

When the base window is active, the cursor is displayed in the command or the source area. This cursor can be switched between the command and the source area by the (Ctrl + Tab) keys.

The (Ctrl+C) key input in the command area stops emulator command execution. The (Ctrl+C) key input outside the command area is invalid. Use the <STOP> button instead.

(3) When a debug window is active, use the following procedure:
Press the (Tab) or (Shift+Tab) keys to move the object of operation among windows and buttons. Wherever the object of operation is located, the (Enter) key performs the same operation as the <Done> button, and the (Esc) key performs the same operation as the <Cancel> button. Pressing the key corresponding to the underlined character of the displayed button name in combination with the (Alt) key moves to the button.

(4) When a dialog box is active, use the following procedure:
Press the (Tab) or (Shift+Tab) keys to move the object of operation among subwindows and buttons. Wherever the objet of operation is located, the (Enter) key performs the same operation as the <Done> button, and the (Esc) key performs the same operation as the <Cancel> button. Pressing the key corresponding to the underlined character of the displayed button name in combination with the (Alt) key moves to the button.

3.8 Input Format

This section describes the input format for windows other than the command area. For input to the command area, refer to the emulator user's manual.

Expressions and file names can be input.

3.8.1 Expressions

Expressions can include numerical constants, symbols, and operators. Operators are + (addition) and – (subtraction). Expressions must not start with an operator.

- Numerical constants
  Numerical constants must be in the following format.

  S'nnn...nn
  S: Radix
  B: Binary
  Q: Octal
  D: Decimal
  H: Hexadecimal
  When no radix is specified, the value is assumed to be in hexadecimal.

  n: Number
  Binary: Numerical characters 0 and 1
  Octal: Numerical characters 0 to 7
  Decimal: Numerical characters 0 to 9
  Hexadecimal: Numerical characters 0 to 9 and alphabetical characters A to F and a to f

- Symbols
  Symbols in assembly language or C source programs can be specified in place of addresses. Symbols are classified into two types: ordinary symbols including label names, variable names, and function names, and line number symbols indicating the line number of the assembler or C compiler listing file.

  — Ordinary symbol specification
  (a) Load module file
  ! <unit name>/<symbol name>
  (b) Multi-load module
  !%<segment name>/<unit name>/<symbol name>

  — Line number symbol specification
  (a) Load module file
  &<unit name>/<line number>
  (b) Multi-load module
  &%<segment name>/<unit name>/<line number>

<segment name>: Name of the multi-load module file output by the linkage editor before linking. This must be specified when using multi-load module files, but must not be specified when using normal load modules.

<unit name>: Character string excluding an extension name from the name of the object file output by the assembler or C compiler. The unit name is generally the source file name.

<line number>: Decimal number without radix

<symbol name>: The following items included in the unit:
Variable names (array variable name when an array-type symbol is used)
Function name
Pointer name
Structure name [. member name]
Union name [. member name]

— When specifying an external symbol for <symbol name>, specify the unit name whose external symbol definition has been declared for <unit name>. If the symbol names differ in the external definition and in the external reference, specify the name in the external definition.

— When symbols are nested, specify symbol names after a slash (/).

— To specify a local variable in a function, specify <function name> then /<symbol name>.

— Variable names, structure names, union names, and pointer names can be specified as member names.

— To specify an array-type symbol, specify an array variable name. The address value corresponding to the symbol points to the beginning of the array.

— Symbol names must be expressed by the characters _ and $, the alphabetical characters A to Z and a to z, and numerical characters 0 to 9.

— Symbol names must start with either _ or $ or one of the alphabetical characters A to Z and a to z.

— Line numbers must be decimal numbers starting with any number from 1 to 9. They must not start with a 0.

— Uppercase and lowercase letters are distinguished.

3.8.2 File Names

File names can be specified up to 127 bytes and must be in the following format:

<drive name>: \<directory path>\<basic file name>. <file extension>

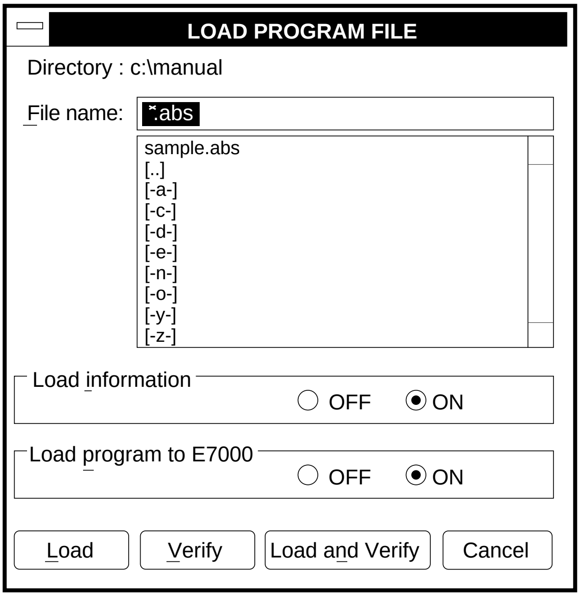
<drive name>: Drive name (first name is A)

<directory path>: Name of the path to the directory where the file is stored or to be stored

<basic file name>: File name expressed by one to eight characters

<file extension>: File type
Example: .C (C source file)
.SRC (assembly language source file)
.ABS (absolute object file)
.DBG (debugging information file)
.BIN (memory image file)

When a file name is input from a dialog box, it can be specified with a relative path name, as shown in figure 3-16, by selecting, from the list box, the current directory from the drive name to the directory path. When a file is to be read, the file name can also be selected from the list box. Wild card characters can be used for file names.



Directory : c:\manual

File name: *.abs

sample.abs
[..]
[-a-]
[-c-]
[-d-]
[-e-]
[-n-]
[-o-]
[-y-]
[-z-]

Load information ○ OFF ● ON

Load program to E7000 ○ OFF ● ON

Figure 3-16 Example of a Dialog Box for File Operation

3.9 Short-Cut Keys

The buttons of the commands or on the base window can be operated by short cut keys. Table 3-5 lists the short cut keys.

Table 3-5 Short-Cut Keys

Classification	Button/Command Names	Short Cut Keys
Buttons	<DISPLAY>	(Alt + N)
	<STEP_OVER>	(F7)
	<STEP_UP>	(F8)
	<STEP>	(F9)
	<CONTINUE>	(Alt + G)
	<SET>	(Alt + B)
	<CLEAR>	(Alt + C)
	<STOP>	(Ctrl + C)
FILE menu	Load program file...	(Alt + L)
	Load other file...	(Alt + K)
	E7000 load...	(Alt + J)
	Input from...	(Alt + I)
	Output to...	(Alt + O)
	Quit	(Alt + F4)
EXECUTION menu	Execution mode...	(Ctrl + E)
	Go...	(Alt + Ctrl + G)
	Break...	(Alt + Ctrl + B)
	Break condition1...	(Ctrl + 1)
	Break condition2...	(Ctrl + 2)
	Break condition3...	(Ctrl + 3)
	Break condition4...	(Ctrl + 4)
	Break condition5...	(Ctrl + 5)
TRACE menu	Trace memory...	(Alt + Ctrl + T)
	Trace condition...	(Ctrl + T)
	Trace display setting...	(Alt + Ctrl + X)
	Trace display text	(Alt + X)
	Trace display graph	(Alt + P)

Table 3-5 Short-Cut Keys (cont)

Classification	Button/Command Names	Short Cut Keys
VIEW menu	Memory setting...	(Alt + Ctrl + M)
	Memory dump	(Alt + M)
	Watch setting...	(Alt + Ctrl + W)
	Watch display	(Alt + W)
	Register	(Alt + R)
	Symbol display	(Alt + S)
	Symbol value	(Alt + Ctrl + N)
	Source setting...	(Alt + Ctrl + A)
	Source display...	(Alt + A)
	Disassemble	(Alt + D)
	Route	(Alt + U)
	Command area	(Ctrl + F6)
HELP menu	GUI operating help	(F1)
	EMULATOR command help	(Alt + F1)
	About...	None

Section 4 GUI-SH Operation

This section describes GUI-SH initiation, basic operations, and termination, assuming that the GUI-SH has been installed. If it has not been installed yet, do so according to the procedure given in section 2, Installation.

This section mainly describes operations using the mouse. For operations using only the keyboard, refer to section 5, Command Reference.

4.1 Emulator Power-On

Power on the emulator station before initiating the GUI-SH.

4.2 Initiating GUI-SH

After Windows has been initiated, the program manager window shown in figure 4-1 is displayed. The group window of the SH7604 E7000PC GUI is in the program manager window. (When the group window is not open, place the mouse pointer on the SH7604 E7000PC GUI group icon and double-click.)

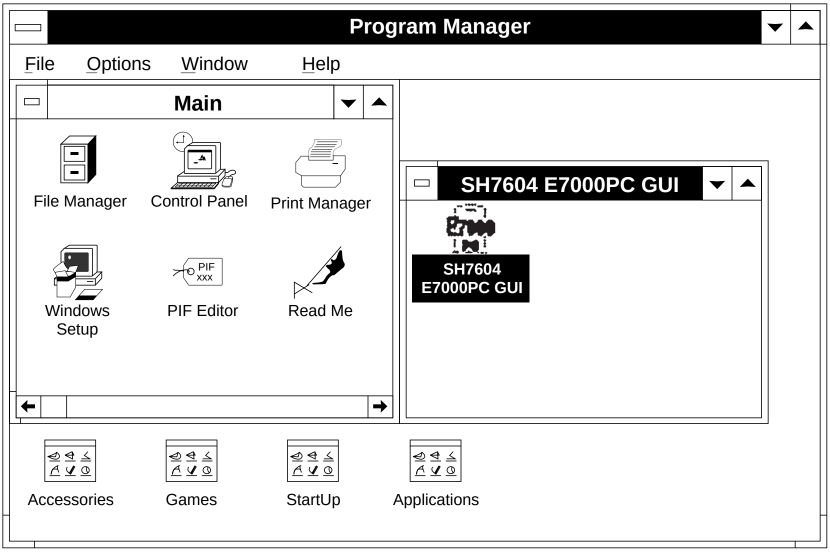


Figure 4-1 Display at Windows Initiation

The icon named SH7604 E7000PC GUI is in the GUI-SH window, which is used to initiate the GUI-SH. The GUI-SH can be initiated using the mouse or the keyboard.

- Using the mouse

(1) Place the mouse pointer on the GUI-SH icon.

(2) Double-click the left button of the mouse.

(3) The GUI-SH base window shown in figure 4-2 appears.

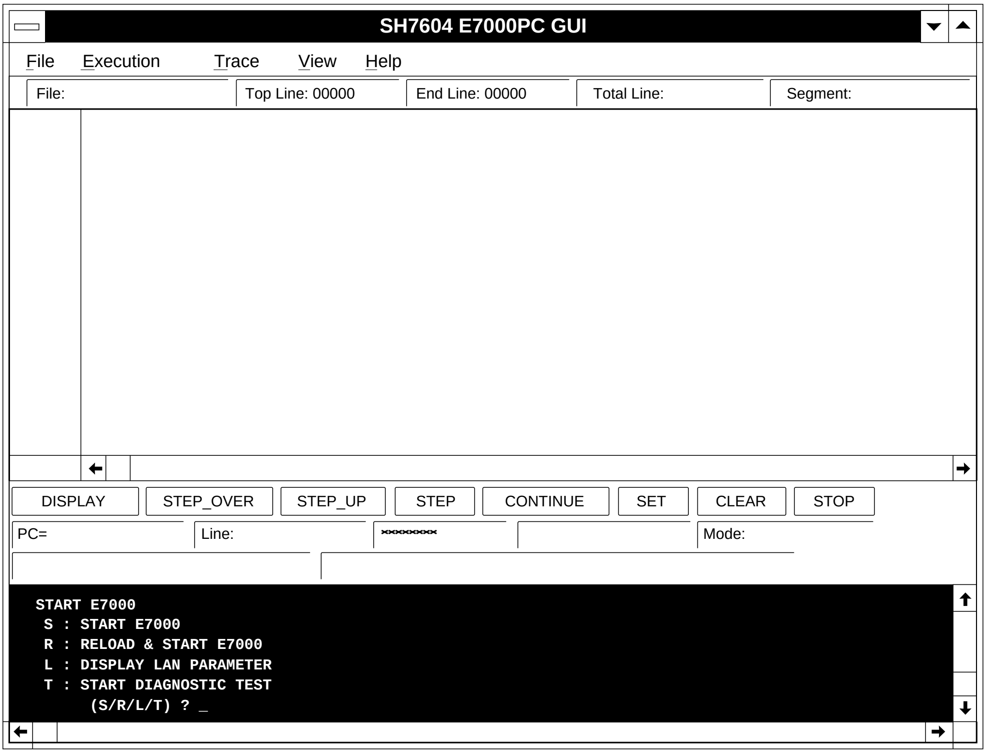
- Using the keyboard

(1) Press the (Ctrl + Tab) keys to select the SH7604 E7000PC GUI group window. When selected, the window title is highlighted.

(2) Press the (←) or (→) key to select the GUI-SH icon. When selected, the icon title is highlighted.

(3) Press the (Enter) key.

(4) The GUI-SH base window shown in figure 4-2 appears.



```
START E7000
 S : START E7000
 R : RELOAD & START E7000
 L : DISPLAY LAN PARAMETER
 T : START DIAGNOSTIC TEST
      (S/R/L/T) ? _
```

Figure 4-2 Display at GUI-SH Initiation

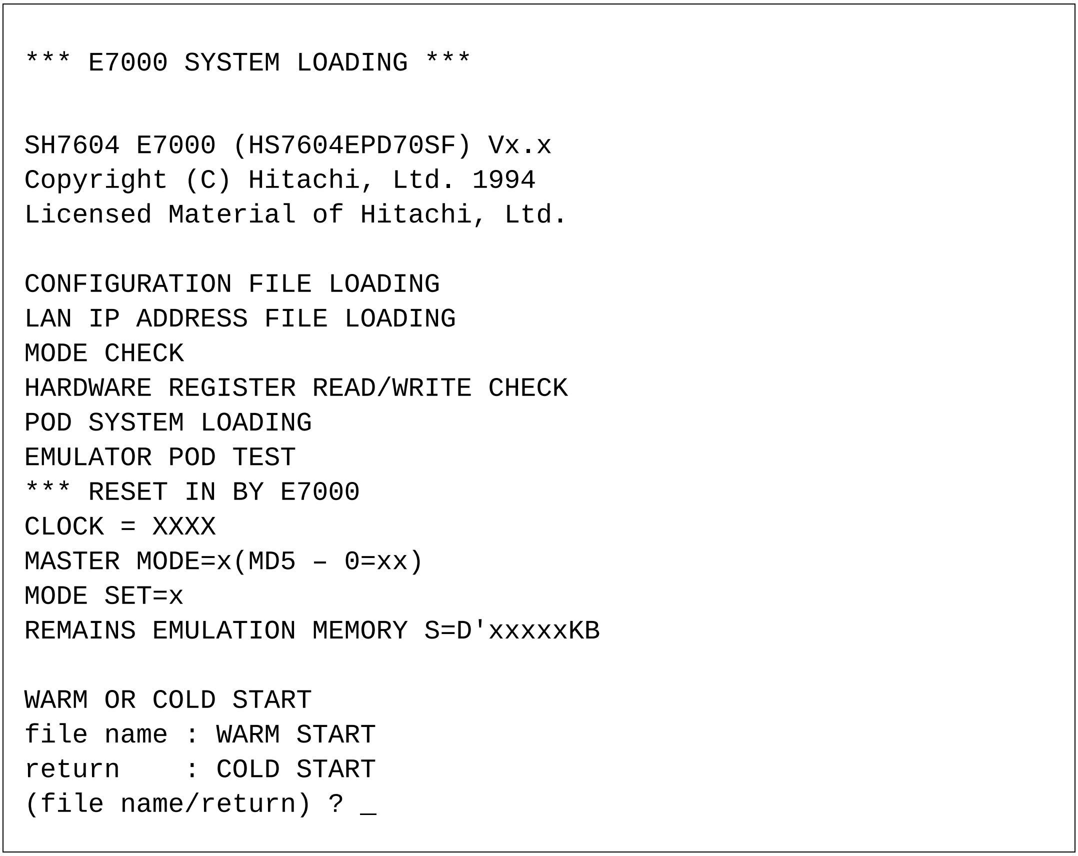
4.3 Initiating Emulator

After the GUI-SH has been initiated, the emulator message shown in figure 4-3 is displayed in the command area, and the GUI-SH waits for an emulator monitor command.

```
SH7604 E7000PC GUI (HS7604G7IW1SF) Vx.y
Copyright (c) Hitachi, Ltd. 1994
Licensed Material of Hitachi, Ltd.


E7000 MONITOR Vx.x
Copyright(C) 1993 Hitachi, Ltd.
Licensed Material of Hitachi Ltd


TESTING
  RAM 0123


START E7000
  S:  START E7000
  R:  RELOAD & START E7000
  L:  DISPLAY LAN PARAMETER
  T:  START DIAGNOSTIC TEST
       (S/R/L/T)_
```

Figure 4-3 Emulator Monitor Message

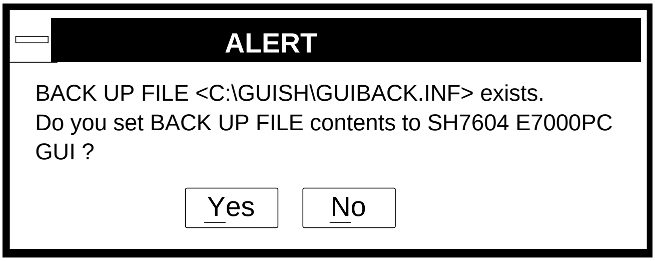
Enter the (S) key (initiation command) followed by the (Enter) key to display the emulator message shown in figure 4-4 and initiate the emulator.



```
*** E7000 SYSTEM LOADING ***


SH7604 E7000 (HS7604EPD70SF) Vx.x
Copyright (C) Hitachi, Ltd. 1994
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
LAN IP ADDRESS FILE LOADING
MODE CHECK
HARDWARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TEST
*** RESET IN BY E7000
CLOCK = XXXX
MASTER MODE=x(MD5 – 0=xx)
MODE SET=x
REMAINS EMULATION MEMORY S=D'xxxxxKB

WARM OR COLD START
file name : WARM START
return    : COLD START
(file name/return) ? _
```

Figure 4-4   Emulator System Message

When a backup file is found, the confirmation message box for backup file load shown in figure 4-5 is displayed. Click the <Yes> button to read the backup file and restore the previous execution information in the GUI-SH. To initiate the GUI-SH using default settings, click the <No> button. When no backup file is found, this message box is not displayed and the GUI-SH is initiated using default settings.



BACK UP FILE <C:\GUISH\GUIBACK.INF> exists.
Do you set BACK UP FILE contents to SH7604 E7000PC GUI ?

Figure 4-5 Confirmation Message Box for Backup File Load

4.4 Loading Load Module

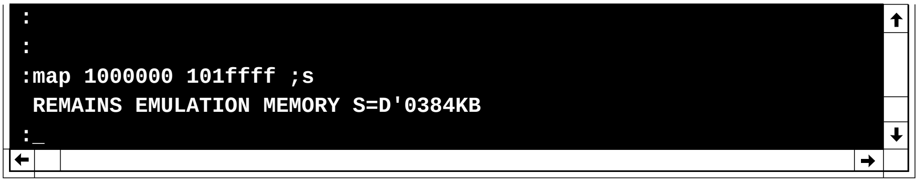
4.4.1 Allocating Emulation Memory

When the user system does not have memory to load programs, the emulation memory must be allocated with the MAP command of the emulator. The MAP command is entered in the command area. An example of emulation memory allocation with the MAP command is shown in figure 4-6.

(1) Place the mouse pointer in the command area and click to display the cursor (_).

(2) If the emulator prompt (:) is not displayed, press the (Enter) key to display it.

(3) Enter the MAP command.



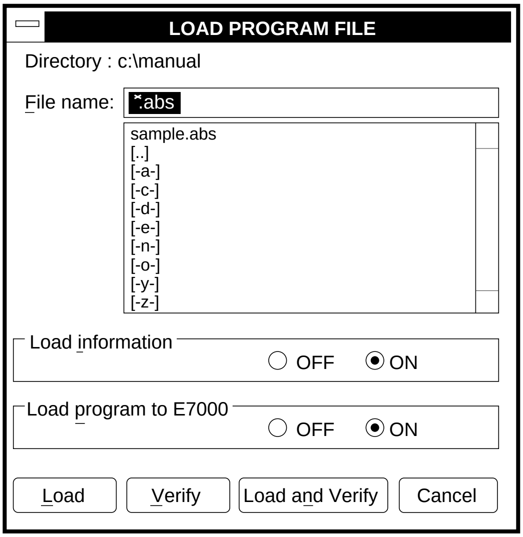
```
:
:
:map 1000000 101ffff ;s
 REMAINS EMULATION MEMORY S=D'0384KB
:_
```

Figure 4-6 Example of Emulation Memory Allocation with MAP Command

4.4.2 Loading Program

Load programs using the following procedure:

(1) Open LOAD dialog box
Select the [Load-Load program file...] command from the File menu. The LOAD PROGRAM FILE dialog box shown in figure 4-7 is displayed.



LOAD PROGRAM FILE

Directory : c:\manual

File name: *.abs

sample.abs
[..]
[-a-]
[-c-]
[-d-]
[-e-]
[-n-]
[-o-]
[-y-]
[-z-]

Load information ○ OFF ● ON

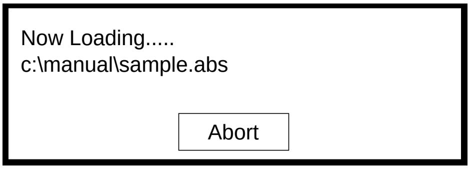
Load program to E7000 ○ OFF ● ON

Figure 4-7 LOAD PROGRAM FILE Dialog Box

(2) Set current directory
Select from the list box the current directory where the program is to be loaded. The drive name and directory name must be specified in that order.
Place the mouse pointer on the target drive name or the target directory name in the list box and double-click to display the selected current directory in the directory display box.

(3) Set file name
After the current directory has been selected, file names in the current directory having a file extension of .ABS are displayed in the list box. Place the mouse pointer on the file to be loaded and click to select it.

(4) Start load operation
Click the <Load> button to start loading. The message box shown in figure 4-8 is displayed during program load.



Now Loading.....
c:\manual\sample.abs

Figure 4-8 Loading Message Box

(5) Load completed
When the load operation is completed, the message box closes, and the section of the source file including the program start address is displayed in the source area. The line corresponding to the program start address is marked with the PC mark (PC>) indicating the location pointed to by the program counter.

Start and end addresses in the load memory are displayed in the command area.

4.5 Displaying the Source File

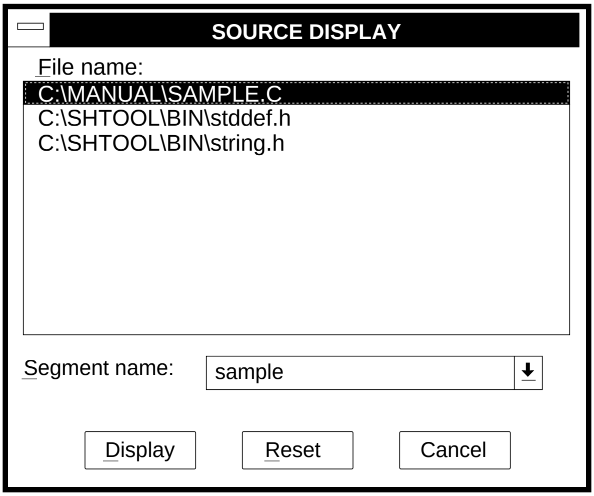
Source files can be displayed in the source area with the [Source - Display...] command in the View menu.

(1) Select the [Source - Display...] command from the View menu; the SOURCE DISPLAY dialog box shown in figure 4-9 appears.

(2) For a multi-load module, specify the target segment in [Segment name :].

(3) Select from [File name :] the file name to be displayed in the source area by placing the mouse pointer on the file name and clicking it.

(4) Click the <Display> button to display the source file in the source area.



SOURCE DISPLAY

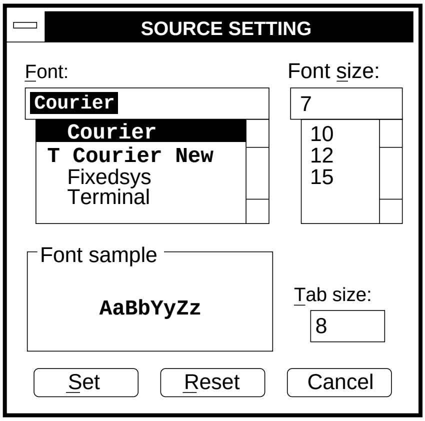
File name:

C:\MANUAL\SAMPLE.C
C:\SHTOOL\BIN\stddef.h
C:\SHTOOL\BIN\string.h

Segment name: sample

Figure 4-9 SOURCE DISPLAY Dialog Box

In addition, a font and a tab size of the source file displayed in the source area can be specified using the [Source -Setting...] command.



SOURCE SETTING

Font: Courier

Font size: 7

Font sample AaBbYyZz

Tab size: 8

Figure 4-10 SOURCE SETTING Dialog Box

4.6 Setting and Cancelling Breakpoints

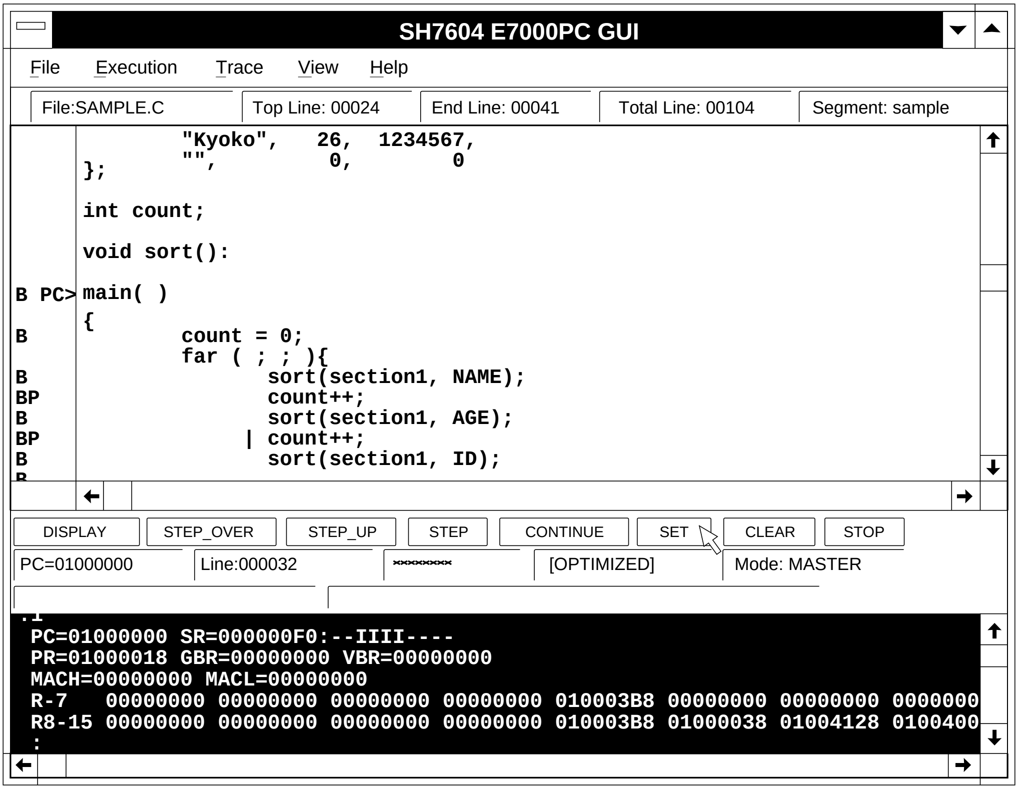
4.6.1 Setting Breakpoints

Breakpoints can be set within the source area. They can also be set with the [Break...] command in the Execution menu. An example of breakpoint setting is shown in figure 4-11.

(1) Scroll the source area to display the line where a breakpoint is to be set, and click the line with the mouse pointer. The cursor (|) will appear on the clicked line. Note that breakpoints can be set only on lines marked with B.

(2) Click the <SET> button on the tool bar.

(3) After a breakpoint is set for a selected line, the BP mark appears on the line.



File:SAMPLE.C | Top Line: 00024 | End Line: 00041 | Total Line: 00104 | Segment: sample

```
              "Kyoko",   26,  1234567,
              "",         0,        0
      };

      int count;

      void sort():

B PC> main( )
      {
B             count = 0;
              far ( ; ; ){
B                    sort(section1, NAME);
BP                   count++;
B                    sort(section1, AGE);
BP                 | count++;
B                    sort(section1, ID);
B
```

PC=01000000 | Line:000032 | xxxxxxxx | [OPTIMIZED] | Mode: MASTER

```
.1
 PC=01000000 SR=000000F0:--IIII----
 PR=01000018 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R-7    00000000 00000000 00000000 00000000 010003B8 00000000 00000000 0000000
 R8-15  00000000 00000000 00000000 00000000 010003B8 01000038 01004128 0100400
 :
```

Figure 4-11 Example of Setting Breakpoints

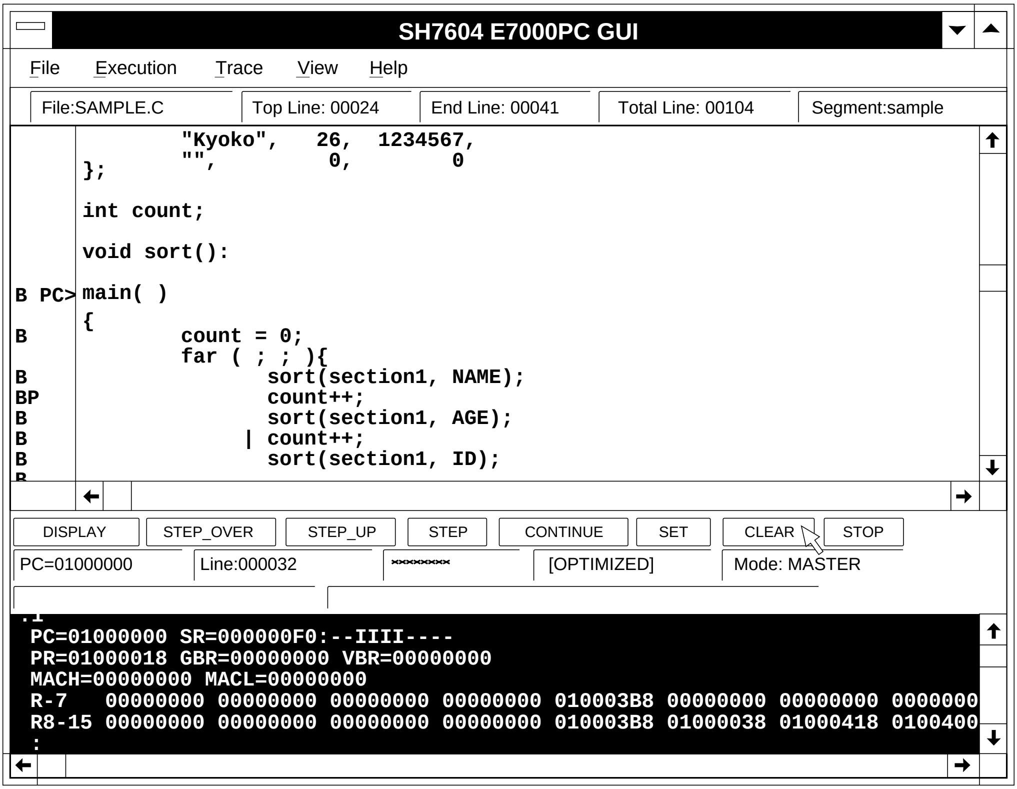
4.6.2 Cancelling Breakpoints

Breakpoints can be cancelled within the source area. They can also be cancelled with the [Break...] command in the Execution menu. An example of breakpoint cancellation is shown in figure 4-12.

(1) Scroll the source area to display the line for which the breakpoint is to be cancelled, and click the line with the mouse pointer. The cursor (|) appears on the clicked line. Note that breakpoints can be cancelled only on lines marked with BP.

(2) Click the <CLEAR> button on the tool bar.

(3) After the breakpoint on the line is cancelled, the BP mark on the line returns to a B mark.



File:SAMPLE.C   Top Line: 00024   End Line: 00041   Total Line: 00104   Segment:sample

```
                "Kyoko",   26,  1234567,
                "",         0,        0
        };

        int count;

        void sort():

B PC>   main( )
        {
B               count = 0;
                far ( ; ; ){
B                       sort(section1, NAME);
BP                      count++;
B                       sort(section1, AGE);
B                     | count++;
B                       sort(section1, ID);
```

PC=01000000   Line:000032   ********   [OPTIMIZED]   Mode: MASTER

```
 PC=01000000 SR=000000F0:--IIII----
 PR=01000018 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R-7   00000000 00000000 00000000 00000000 010003B8 00000000 00000000 0000000
 R8-15 00000000 00000000 00000000 00000000 010003B8 01000038 01000418 0100400
 :
```

Figure 4-12 Example of Breakpoint Cancellation

4.7 Setting Execution Start Address and Stack Pointer

Before executing the user program, the program counter (PC) and stack pointer (SP) must be set. The PC and SP can be set with the [Register] command in the View menu. They can also be set or modified by entering an emulator command in the command area. An example of register modification is shown in figure 4-13.

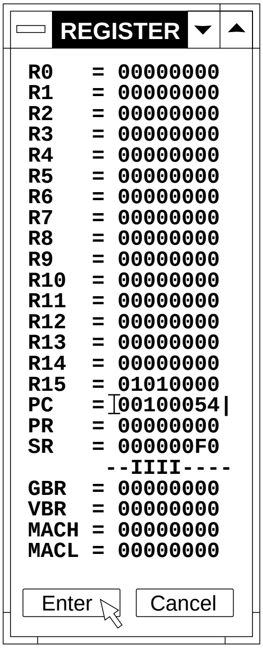
(1) Select the [Register] command in the View menu. The register window as shown in figure 4-13 will appear.

(2) Place the mouse pointer over the hexadecimal data for the PC and click to display the character-insertion cursor (|).

(3) Enter data from the keyboard.

(4) Enter data for the SP (R15) in the same way if necessary.

(5) Click the <Enter> button to set the modified values in the registers.



```
R0   = 00000000
R1   = 00000000
R2   = 00000000
R3   = 00000000
R4   = 00000000
R5   = 00000000
R6   = 00000000
R7   = 00000000
R8   = 00000000
R9   = 00000000
R10  = 00000000
R11  = 00000000
R12  = 00000000
R13  = 00000000
R14  = 00000000
R15  = 01010000
PC   = 00100054|
PR   = 00000000
SR   = 000000F0
      --IIII----
GBR  = 00000000
VBR  = 00000000
MACH = 00000000
MACL = 00000000
```

Figure 4-13 Example of Setting Registers with [Register] Command

4.8 Executing Programs

Program execution can be started with several procedures. This section describes the procedure using the tool bar and that using the [GO...] command in the Execution menu.

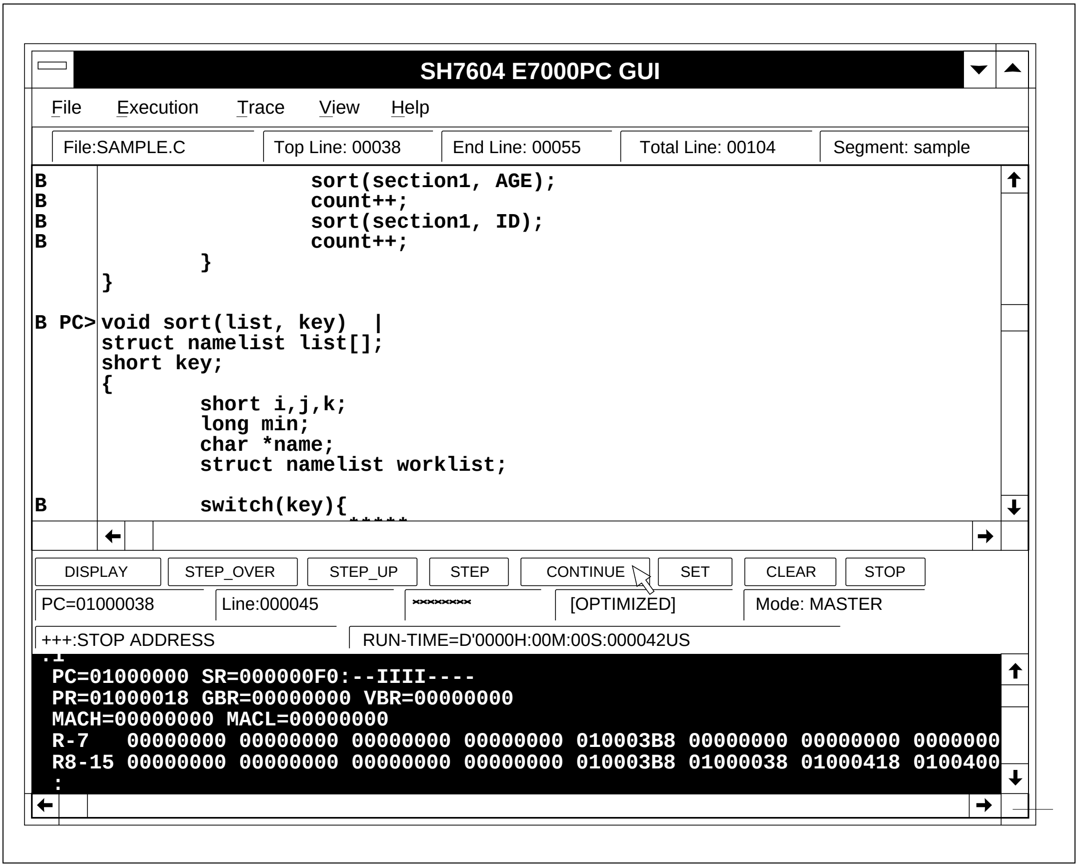
4.8.1 Executing Programs Using the Tool Bar

The buttons for program execution on the tool bar are listed in table 4-1.

Table 4-1 Program Execution Buttons in Tool Bar

Button	Short-Cut Keys	Function
STEP_OVER	(F7)	Executes one program line pointed to by the current program counter (PC) and then stops. When the line includes a function (subroutine) call, program execution stops after executing the function. Note that this button cannot be used to execute programs in the ROM area.
STEP_UP	(F8)	Executes the function including the current program counter (PC), and stops execution when returning to the upper function (calling function). This button cannot be used to execute programs in the ROM area.
STEP	(F9)	Executes one program line pointed to by the current program counter (PC) and then stops. When the line includes a function (subroutine) call, program execution stops at the first line of the function.
CONTINUE	(Alt+G)	Executes the program from the line pointed to by the current program counter (PC). Execution continues until a line including the cursor in the source area is reached. Execution stops when a line marked with B in the source area is clicked.

An example of program execution using the <CONTINUE> button is shown in figure 4-14. When program execution stops at the cursor location in the source area, the emulation information area displays the execution stop address, execution stop line number, cause of termination, and execution time.



File:SAMPLE.C | Top Line: 00038 | End Line: 00055 | Total Line: 00104 | Segment: sample

```
B                    sort(section1, AGE);
B                    count++;
B                    sort(section1, ID);
B                    count++;
               }
      }

B PC> void sort(list, key)  |
      struct namelist list[];
      short key;
      {
               short i,j,k;
               long min;
               char *name;
               struct namelist worklist;

B              switch(key){
                     *****
```

PC=01000038 | Line:000045 | ******** | [OPTIMIZED] | Mode: MASTER

+++:STOP ADDRESS | RUN-TIME=D'0000H:00M:00S:000042US

```
.I
 PC=01000000 SR=000000F0:--IIII----
 PR=01000018 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R-7    00000000 00000000 00000000 00000000 010003B8 00000000 00000000 0000000
 R8-15  00000000 00000000 00000000 00000000 010003B8 01000038 01000418 0100400
 :
```

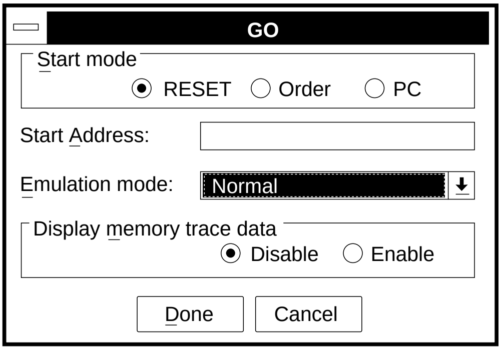
Figure 4-14   Program Execution Example Using <CONTINUE> Button

4.8.2 Executing Programs Using the Go Command

Program execution can be started from a specified address with the [Go...] command in the Execution menu. When the [Go...] command is executed, the GO dialog box shown in figure 4-15 opens.

- By selecting <RESET> in the [Start mode] group box and clicking the <Done> button, the emulator outputs a RESET signal to the user system and program execution starts from the reset vector.

- By selecting <Order> in the [Start mode] group box, specifying an address in the [Start address:] text box, and then clicking the <Done> button, program execution starts from the specified address.



Start mode: ● RESET ○ Order ○ PC

Start Address:

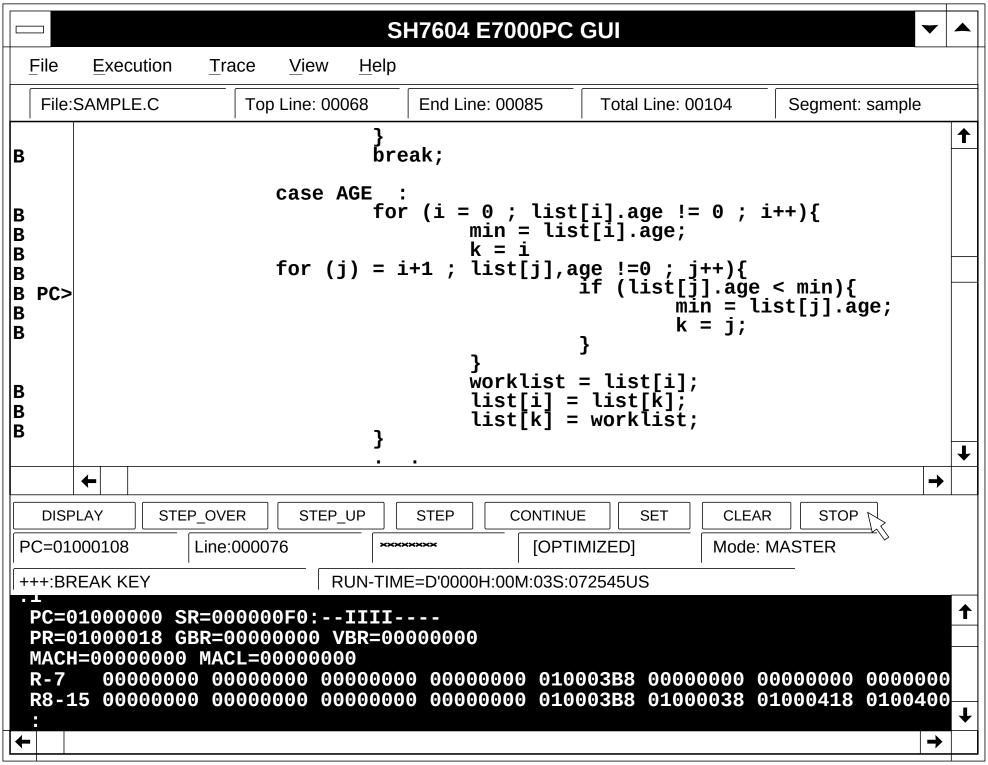
Emulation mode: Normal

Display memory trace data: ● Disable ○ Enable

Figure 4-15 GO Dialog Box

4.8.3 Terminating Programs

To forcibly terminate program execution, click the <STOP> button on the tool bar as shown in figure 4-16. After program execution stops, the emulation information area displays the execution stop address, execution stop line number, cause of termination, and execution time. The section of the source file corresponding to the stop location is displayed in the source area.



File:SAMPLE.C | Top Line: 00068 | End Line: 00085 | Total Line: 00104 | Segment: sample

```
                                 }
B                                break;

                         case AGE   :
B                                for (i = 0 ; list[i].age != 0 ; i++){
B                                        min = list[i].age;
B                                        k = i
B                        for (j) = i+1 ; list[j],age !=0 ; j++){
B  PC>                                            if (list[j].age < min){
B                                                         min = list[j].age;
B                                                         k = j;
                                                  }
                                         }
B                                        worklist = list[i];
B                                        list[i] = list[k];
B                                        list[k] = worklist;
                                 }
                                 .   .
```

PC=01000108 | Line:000076 | ×××××××× | [OPTIMIZED] | Mode: MASTER

+++:BREAK KEY | RUN-TIME=D'0000H:00M:03S:072545US

```
:I
 PC=01000000 SR=000000F0:--IIII----
 PR=01000018 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R-7   00000000 00000000 00000000 00000000 010003B8 00000000 00000000 0000000
 R8-15 00000000 00000000 00000000 00000000 010003B8 01000038 01000418 0100400
 :
```

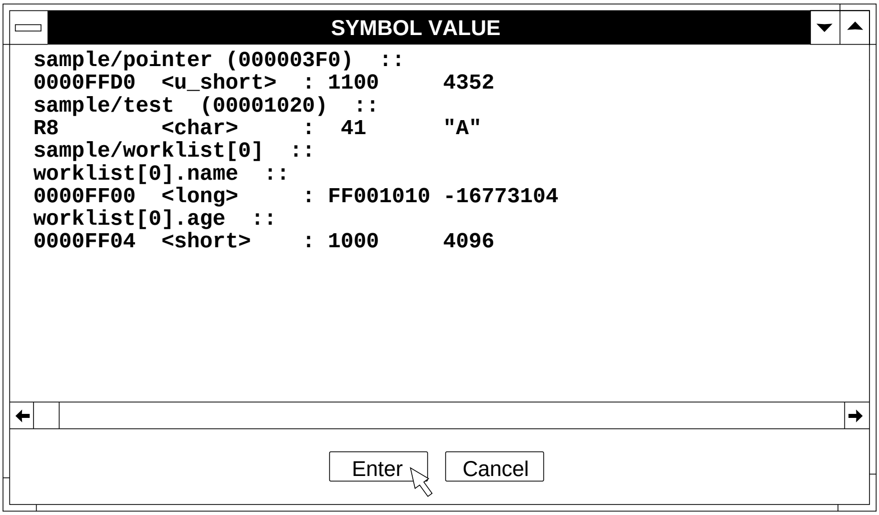
Figure 4-16 Example of Program Execution Termination with the <STOP> Button

4.9 Displaying Symbol Contents

Symbol contents can be displayed by selecting symbols in the source area using the procedure shown in figure 4-17.

(1) Select symbol
Select a symbol in the source area by placing the mouse pointer on the symbol and double-clicking on it. When selected, the symbol is highlighted.

(2) Display symbol contents
Click the <DISPLAY> button on the tool bar to open the SYMBOL VALUE window and display the contents of the selected symbol.



```
sample/pointer (000003F0)  ::
0000FFD0  <u_short>  : 1100     4352
sample/test  (00001020)  ::
R8        <char>     :  41      "A"
sample/worklist[0]  ::
worklist[0].name  ::
0000FF00  <long>     : FF001010 -16773104
worklist[0].age  ::
0000FF04  <short>    : 1000     4096
```

Figure 4-17 Example of Symbol Contents Display

4.10 Inputting Emulator Commands

Emulator commands can be input and executed in the command area of the base window. Some emulator commands, however, cannot be executed using the GUI-SH. For available commands, refer to appendix A, E7000 Emulator Command List.

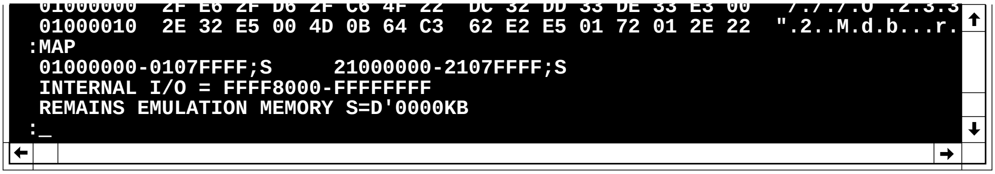
Emulator commands can be executed using the procedure shown in figure 4-18.

(1) Place the mouse pointer in the command area and click the left button of the mouse to display the cursor (_).

(2) If the emulator command prompt (: or #) is not displayed, press the (Enter) key to display it.

(3) Enter an emulator command and press the (Enter) key.

(4) The command execution results are displayed in the command area.



```
 01000000  2F E6 2F D6 2F C6 4F 22  DC 32 DD 33 DE 33 E3 00  /./.O .2.3.3
 01000010  2E 32 E5 00 4D 0B 64 C3  62 E2 E5 01 72 01 2E 22  ".2..M.d.b...r.
:MAP
 01000000-0107FFFF;S     21000000-2107FFFF;S
 INTERNAL I/O = FFFF8000-FFFFFFFF
 REMAINS EMULATION MEMORY S=D'0000KB
:_
```

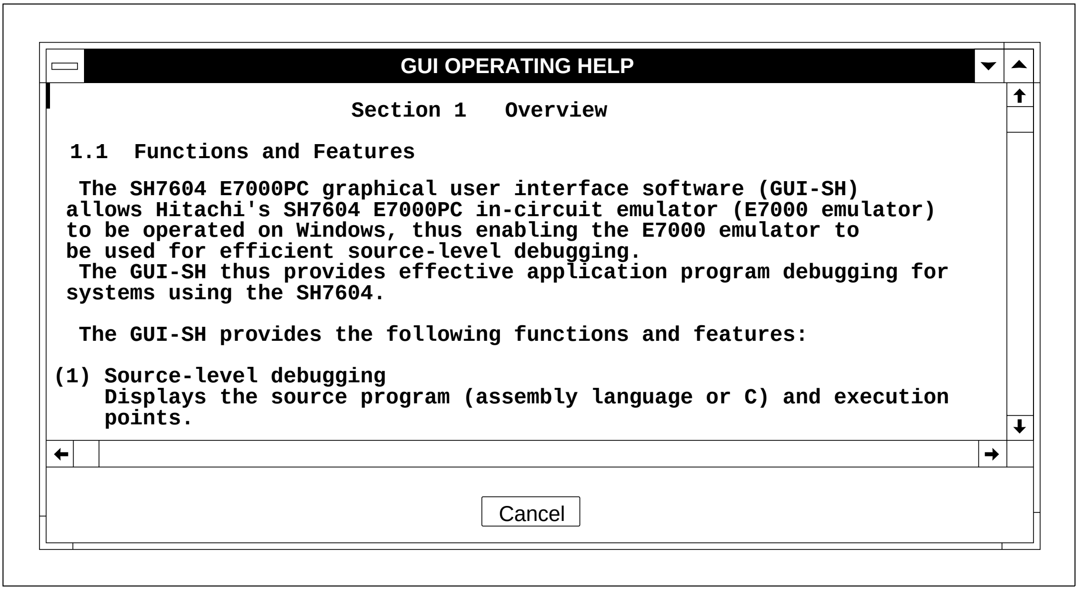
Figure 4-18 Emulator Command Input Format

4.11 Displaying Help Information

The help function provides information on GUI-SH operation and emulator commands.

4.11.1 GUI-SH Operation Help

The GUI OPERATING HELP window shown in figure 4-19 appears by selecting the [GUI operating help] command from the Help menu. Click <Cancel> to close this window.



```
                    Section 1   Overview

 1.1  Functions and Features

  The SH7604 E7000PC graphical user interface software (GUI-SH)
 allows Hitachi's SH7604 E7000PC in-circuit emulator (E7000 emulator)
 to be operated on Windows, thus enabling the E7000 emulator to
 be used for efficient source-level debugging.
  The GUI-SH thus provides effective application program debugging for
 systems using the SH7604.

  The GUI-SH provides the following functions and features:

(1) Source-level debugging
    Displays the source program (assembly language or C) and execution
    points.
```

Figure 4-19 GUI-SH Operation Help Display

4.11.2 Emulator Command Help

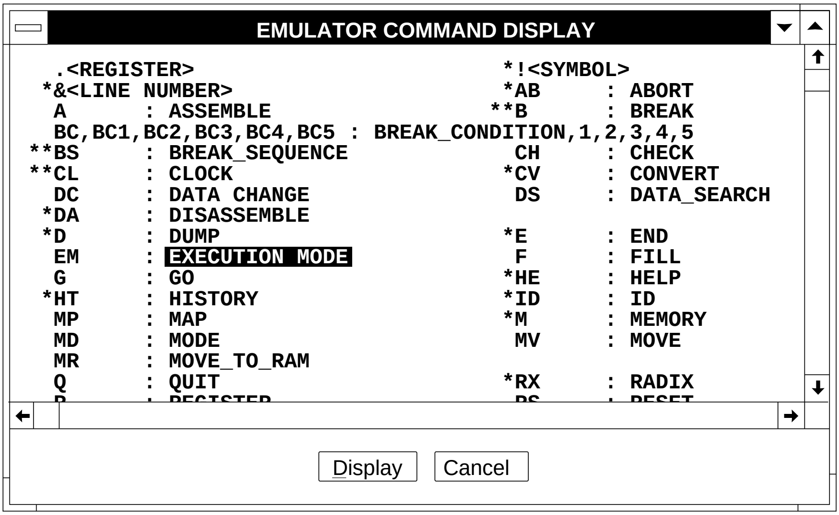
The EMULATOR COMMAND DISPLAY window shown in figure 4-20 appears by selecting the [EMULATOR command display] command from the Help menu. The emulator commands are listed in this window. Detailed help information on each command can be displayed using the following procedure:

(1) Select a command by placing the mouse pointer on the command name displayed in the EMULATOR COMMAND DISPLAY window and double-clicking. When selected, the command name is highlighted.

(2) Click the <Display> button.

(3) The EMULATOR COMMAND HELP window opens and detailed help information on the command is displayed.

(4) Click <Cancel> to close the window.



EMULATOR COMMAND DISPLAY

```
  .<REGISTER>                            *!<SYMBOL>
 *&<LINE NUMBER>                         *AB     : ABORT
  A       : ASSEMBLE                    **B      : BREAK
  BC,BC1,BC2,BC3,BC4,BC5 : BREAK_CONDITION,1,2,3,4,5
**BS      : BREAK_SEQUENCE                CH     : CHECK
**CL      : CLOCK                        *CV     : CONVERT
  DC      : DATA CHANGE                   DS     : DATA_SEARCH
 *DA      : DISASSEMBLE
 *D       : DUMP                         *E      : END
  EM      : EXECUTION MODE                F      : FILL
  G       : GO                           *HE     : HELP
 *HT      : HISTORY                      *ID     : ID
  MP      : MAP                          *M      : MEMORY
  MD      : MODE                          MV     : MOVE
  MR      : MOVE_TO_RAM
  Q       : QUIT                         *RX     : RADIX
  R       : REGISTER                      RS     : RESET
```

Figure 4-20 Emulator Command Help Display

4.12 Calling DOS

DOS can be called with the DOS prompting program in the program manager when source files need to be edited, compiled, or linked during debugging. Figure 4-21 shows the procedure to call DOS.

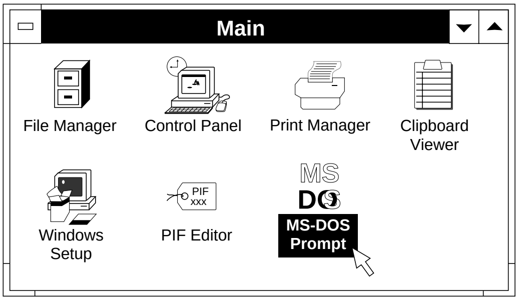


Figure 4-21 DOS Call Procedure

(1) Execute the DOS prompting program in the program manager to switch the display from Windows to DOS full-screen display; the DOS prompt appears.

(2) Execute a DOS application. Some DOS applications cannot run in this state. For details, refer to the documents related to Windows.

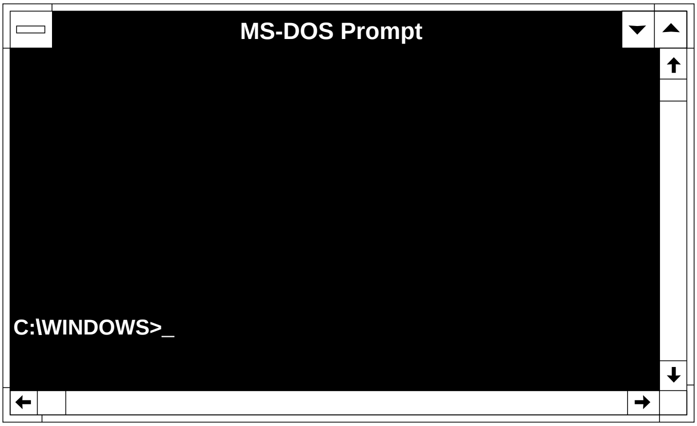
(3) While Windows is operating in 386 enhanced mode, almost all DOS applications can run in a window using the following procedure.

   (a) Press the (Alt + Enter) keys during DOS application execution.

   (b) The contents of the full-screen display appears in a window. In this state, another Windows application can be executed.

   (c) Press the (Alt + Enter) keys again to switch the window to full-screen display.

(4) Enter exit and press the (Enter) key to return to Windows.



```
C:\WINDOWS>_
```

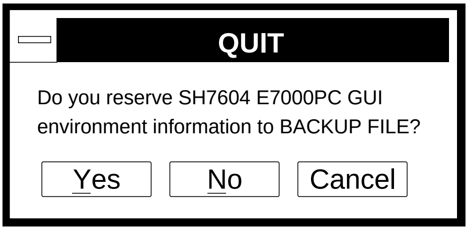
Figure 4-22 DOS Application Execution in a Window

4.13 Terminating GUI-SH

The GUI-SH is terminated with the [Quit...] command in the File menu. If the program debugged is being executed, execution continues even after the GUI-SH is terminated.

At termination, the confirmation message for storing a backup file shown in figure 4-23 appears. Click <Yes> to store current settings during program execution of the GUI-SH. The backup file can be loaded at GUI-SH initiation to restore the settings in the GUI-SH. After <Yes> is clicked, a dialog box is displayed to enter the backup file name.

When the <No> button is clicked, the GUI-SH terminates without saving settings.



Do you reserve SH7604 E7000PC GUI environment information to BACKUP FILE?

Figure 4-23   Confirmation Message for Backup File Storing

Section 5 Command Reference

This section describes the GUI-SH menu commands in detail.

5.1 Menu Command Functions

5.1.1 Filing Functions (File Menu)

The menu for filing functions is shown in figure 5-1.

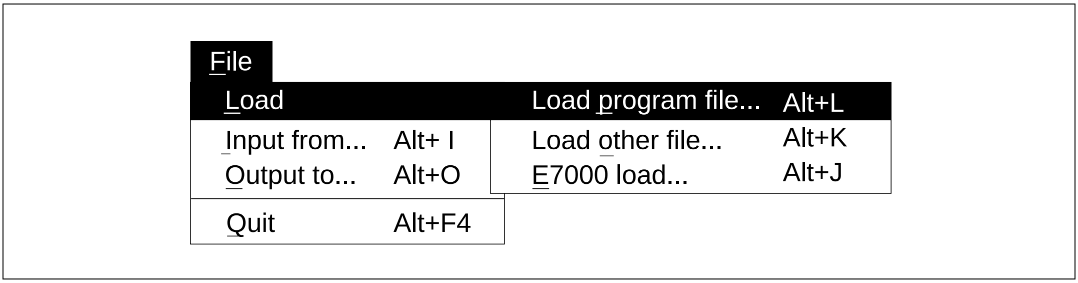


Figure 5-1 Menu for Filing Functions (File Menu)

- Loading program file ([Load - Load program file...])
  Loads the load module file to be debugged into the GUI-SH and the E700PC emulator. Memory contents can be compared with the load module file for verification purposes.

- Loading other files ([Load - Load other file...])
  Loads a memory image file, an S-type load module file, and a Hex-type load module file into the E7000PC emulator. Memory contents can be compared with the file for verification purposes.

- Loading directly into the E7000 ([Load - E7000 load...])
  Loads the load module file to be debugged only into the E7000PC emulator. Symbol information can be loaded into the E7000PC emulator in load module file units. Memory contents can be compared with the load module file for verification.

- Inputting and executing emulator commands ([Input from...])
  Inputs a command file and executes emulator commands.

- Storing the contents displayed in the command area ([Output to...])
  Stores the contents displayed in the command area.

- Terminating GUI-SH ([Quit...])
  Terminates the GUI-SH.

5.1.2 Execution Functions (Execution Menu)

The menu for execution functions is shown in figure 5-2.

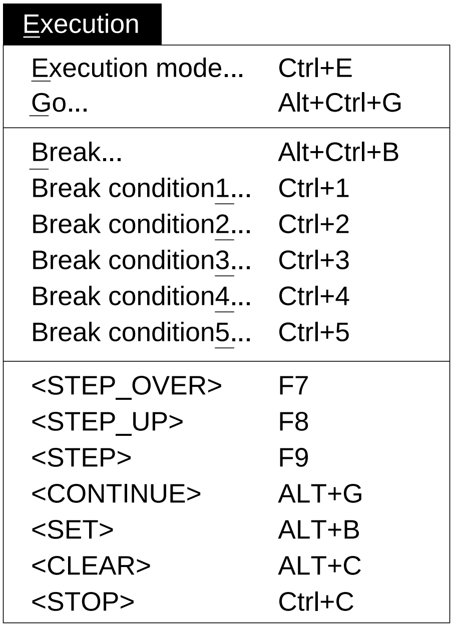


Figure 5-2 Menu for Execution Functions (Execution Menu)

- Setting program execution conditions ([Execution mode...])
  Sets conditions under which program emulation is executed.

- Executing program ([Go...])
  Starts program emulation.

- Setting and cancelling breakpoints ([Break...])
  Sets and cancels breakpoints.

- Setting hardware break conditions ([Break condition 1,2,3,4,5...])
  Sets hardware break conditions. A maximum of five conditions (1 to 5) can be set.

- Execution button
  The buttons related to program execution that are located on the tool bar (<STEP_OVER>, <STEP_UP>, <STEP>, <CONTINUE>, <SET>, <CLEAR>, and <STOP>) can be operated from the menu.

5.1.3 Trace Functions (Trace Menu)

The menu for trace functions is shown in figure 5-3.

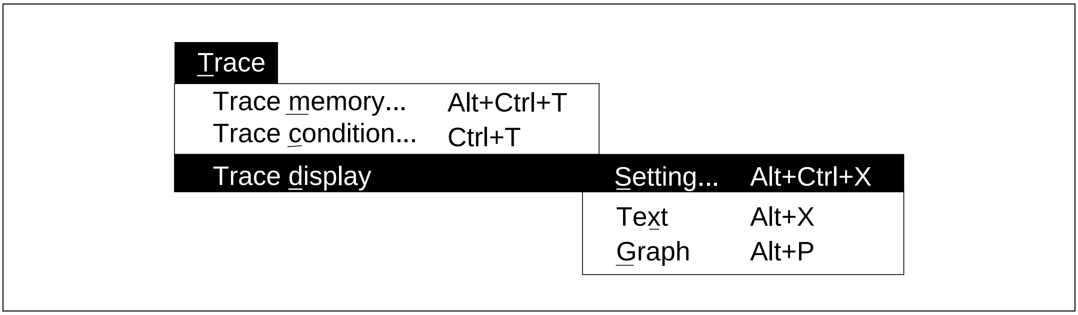


Figure 5-3 Menu for Trace Functions (Trace Menu)

- Setting memory trace conditions ([Trace memory...])
  Sets conditions for acquiring trace information on memory. During program emulation, trace information is acquired according to these settings.

- Setting trace information acquisition conditions ([Trace condition...])
  Sets conditions for acquiring trace information. During program emulation, trace information is acquired according to these settings.

- Setting trace information display conditions ([Trace display - Setting...])
  Sets conditions for text display of trace information ([Trace display - Text]) and graphic display of trace information ([Trace display - Graph]). With this function, only necessary trace information can be displayed.

- Displaying trace information in text ([Trace display - Text])
  Displays trace information in the TRACE DISPLAY TEXT window in text form according to the conditions set with the [Trace display - Setting...] command. The display contents can be output to a file.

- Displaying trace information in graph form ([Trace display - Graph])
  Displays trace information in the TRACE DISPLAY GRAPH window in graph form according to the conditions set with the [Trace display - Setting...] command. The graph can be magnified with the zooming function.

5.1.4 Debugging Information Display Functions (View Menu)

The menu for debugging information display functions is shown in figure 5-4.

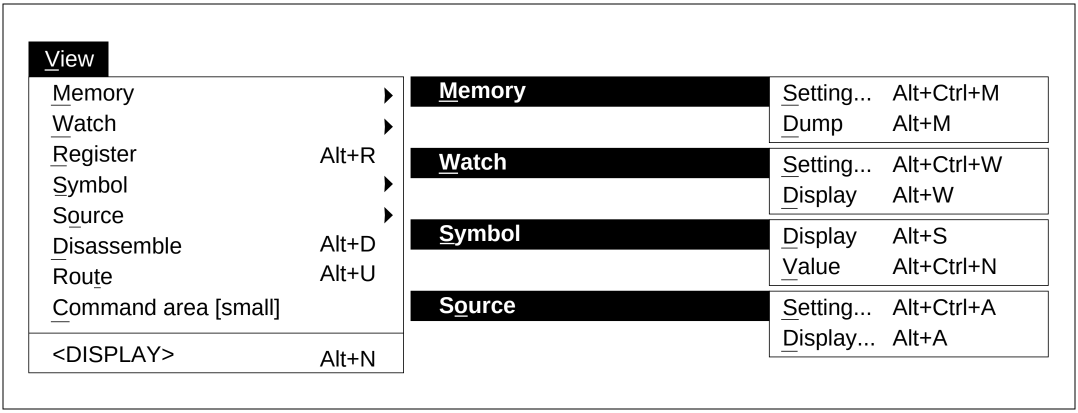


Figure 5-4 Menu for Debugging Information Display Functions (View Menu)

- Setting memory dump display conditions ([Memory - Setting...])
  Sets conditions for dump display of memory contents ([Memory - Dump]).

- Displaying in dump format and modifying memory contents ([Memory - Dump])
  Displays memory contents in the MEMORY DUMP window according to the conditions set with the [Memory - Setting...] command. The memory contents can be changed in the MEMORY DUMP window. The display contents can be output to a file.

- Setting watch points ([Watch - Setting...])
  Sets watch points to display memory contents during emulation.

- Displaying watch point contents ([Watch - Display])
  Displays the contents of the specified watch point in the WATCH DISPLAY window during emulation.

- Displaying and modifying register contents ([Register])
  Displays register contents in the REGISTER window. The register contents can be changed in the REGISTER window.

- Displaying symbol information ([Symbol Display])
  Displays in the SYMBOL DISPLAY window the information on all symbols within the symbol scope determined by the current program counter (PC) value.

- Displaying and changing symbol contents ([Symbol Value])
  Displays in the SYMBOL VALUE window the contents of the symbols specified with the [Symbol Display] command and with the <DISPLAY> button on the tool bar. The symbol contents can be changed in the SYMBOL VALUE window.

- Setting source area ([Source - Setting...])
  Sets the font, the font size, and the tab width of the source program displayed in the source area.

- Listing source file names ([Source - Display])
  Displays a list of the source files making up the loaded load module. A source file can be selected from the list for display in the source area of the base window.

- Displaying disassembly list ([Disassemble])
  Disassembles and displays in the DISASSEMBLE window the source file displayed in the source area of the base window. Breakpoint setting and step execution can also be performed at an assembly language level.

- Displaying function call sequence ([Route])
  Displays the function names called up by the program to the function currently pointed to by the program counter (PC) in the order in which they were called.

- Controlling command area ([Command area])
  Changes the size of the command area of the base window to three sizes.

- Symbol display button
  Executes the same functions as the <DISPLAY> button on the tool bar from the menu.

5.1.5 Help Functions (Help Menu)

The menu for help functions is shown in figure 5-5.

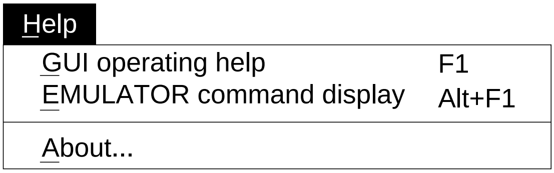


Figure 5-5 Menu for Help Functions (Help Menu)

- Displaying help information on GUI-SH operations ([GUI operating help])
  Displays GUI-SH operation instructions in the GUI OPERATING HELP window.

- Displaying help information on emulator commands ([EMULATOR command display])
  Displays the emulator command list in the EMULATOR COMMAND HELP window.

- Displaying GUI-SH information ([About...])
  Displays information on the GUI-SH, such as the version No.

5.2 Menu Command Reference Format

This section describes menu commands in the format shown in figure 5-6.

(1) (2) (4)

5.3.1 Loading Program **(Alt + L)**

[Load - Load program file...] LOAD PROGRAM FILE dialog box

(3) (5)

Overview (6)

:

:

Window (7)

:

:

Operation (8)

:

:

Function (9)

:

:

Notes (10)

:

:

Related Functions (11)

:

:

Figure 5-6 Command Description Format

The above item numbers indicate the following:

(1) Section number
(2) Command name
(3) Command name in the menu
(4) Short-cut keys for command input
(5) Window name
(6) Overview of command functions
(7) Window display
(8) Operations on the items in the window
(9) Function of the command
(10) Notes on use (omitted if nothing is to be noted)
(11) Related functions or commands

5.3 Filing Functions

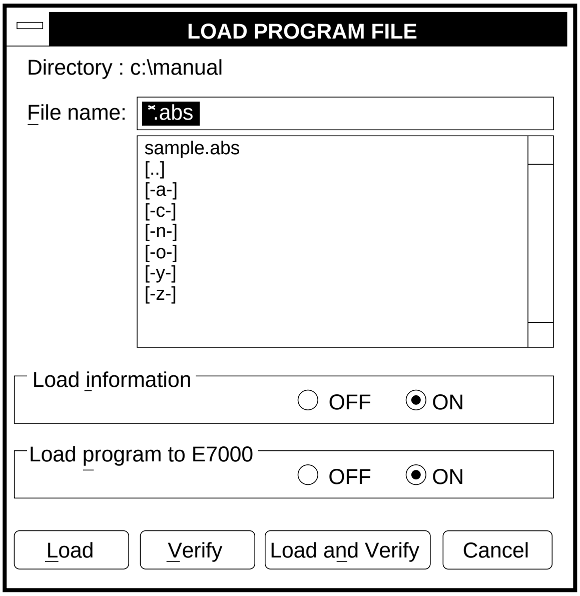
5.3.1 Loading Program (Alt + L)

[Load - Load program file...] LOAD PROGRAM FILE dialog box

Overview

Loads a load module file into the GUI-SH and the E7000PC emulator and compares memory contents with the load module file for verification purposes.

Window



LOAD PROGRAM FILE

Directory : c:\manual

File name: *.abs

sample.abs
[..]
[-a-]
[-c-]
[-n-]
[-o-]
[-y-]
[-z-]

Load information ○ OFF ● ON

Load program to E7000 ○ OFF ● ON

Figure 5-7 LOAD PROGRAM FILE Dialog Box

Operation

- [File name:] text box
  Specifies the name of the file to be loaded. The file name can be selected from the list box; the selected file name is displayed in the text box.

  Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list bos starts loading.

  The following two types of files can be input:
  — Absolute load module file or multi-load module file
  — Debugging information file

When the LOAD PROGRAM FILE dialog box opens, the text box displays *.abs, and therefore the list box displays a list of file names having extension .abs in the current directory. A file having extension .abs is an absolute load module file or a multi-load module file. Debugging information files have extension .dbg.

When the file to be loaded is not in the current directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

— Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

— Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

— Continue selecting subdirectories until the directory including the target file is reached.

- [Directory:]
  Displays the current directory. When the drive or subdirectory is changed in [File name:], this display also changes.

- [Load information] group box
  Specifies whether load module debugging information is to be loaded into the GUI-SH, as follows:
  <ON>: Loads debugging information (default at system initiation)
  <OFF>: Does not load debugging information

- [Load program to E7000] group box
  Specifies whether or not the load module file is to be loaded into the E7000PC emulator. When debugging information is to be loaded, the load module is not loaded into the E7000PC emulator.
  <ON>: Loads load module file (default at system initiation)
  <OFF>: Does not load load module file

- <Load> button
  Clicking the <Load> button starts loading a load module file, closes the LOAD PROGRAM FILE dialog box, and stores the dialog box settings.

- <Verify> button
  Clicking the <Verify> button starts verifying memory contents against the load module file, closes the LOAD PROGRAM FILE dialog box, and stores the dialog box settings.

- <Load and Verify> button
  Clicking the <Load and Verify> button starts loading the load module file. After loading is completed, the load module file and memory contents are automatically verified. The LOAD PROGRAM FILE dialog box is closed and the dialog box settings are stored.

- <Cancel> button
  Clicking the <Cancel> button closes the LOAD PROGRAM FILE dialog box without storing the dialog box settings. The dialog box settings return to those when the LOAD PROGRAM FILE dialog box was opened.

Function

- Program load
  The following two types of files can be input:
  — Absolute or multi-load module files
  — Debugging information files

  If the specified file is a multi-load module file, the GUI-SH automatically loads all the integrated load modules.

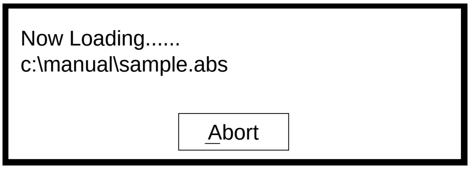
  With this command, debugging information for the load module previously loaded will be lost.

  When an entry address is set in the file to be loaded, the source file corresponding with the entry address is displayed in the source area after loading. The entry address can be set using the H-series linkage editor. When a multi-load module file is specified, the source file corresponding with the specified entry address of the start segment is displayed.

  Whether or not to load debugging information can be specified. [Load information] is used to specify loading of debugging information controlled by the GUI-SH. <ON> should usually be selected. If <OFF> is selected, operations using symbols, such as source-level program debugging or symbol contents display, cannot be performed.

  [Load program to E7000] is used to specify whether the load module is to be loaded to the E7000PC emulator. <ON> should usually be selected. Select <OFF> when debugging a ROM program.

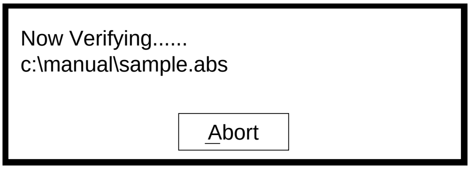
  When debugging information is to be loaded, the load module is not loaded into the emulator regardless of the [Load program to E7000] group box setting.
  During the load, the LOAD INFORMATION dialog box (figure 5-8) automatically opens. When the load is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort the load.



Now Loading......
c:\manual\sample.abs

Figure 5-8 LOAD INFORMATION Dialog Box (Load Module File)

- Program verification
  Clicking the <Verify> button verifies the memory contents against the file specified in [File name:]. Debugging information for the load module previously loaded is not lost.
  During verification the VERIFY INFORMATION dialog box (figure 5-9) automatically opens.

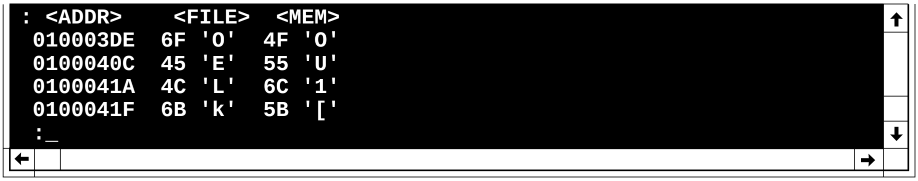


Now Verifying......
c:\manual\sample.abs

Figure 5-9 VERIFY INFORMATION Dialog Box (Load Module File)

If a verification error occurs, the verification results are displayed in the command area. When verification is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort verification.

The verification error message displayed in the command area is shown in figure 5-10.



```
: <ADDR>    <FILE>  <MEM>
 010003DE  6F 'O'  4F 'O'
 0100040C  45 'E'  55 'U'
 0100041A  4C 'L'  6C '1'
 0100041F  6B 'k'  5B '['
 :_
```

① <ADDR>: Address where a verification error occurs
② <FILE>: Data in the load module file in hexadecimal and ASCII characters
③ <MEM>: Data in memory in hexadecimal and ASCII characters

Figure 5-10 Verification Error Message

- Automatic verification after loading
  Clicking the <Load and Verify> button starts loading the file specified in [File name:]. When file loading is completed normally, the loaded file and the memory contents are automatically verified.

  At this time, if <OFF> is selected in the [Load program to E7000] group box, verification is not performed.

- Debugging ROM programs
  To perform source-level debugging of a program stored in ROM on the user system, load the file using the following procedure:

  (1) Create a load module file (or multi-load module file) with debugging information corresponding to the program on the user system.

  (2) Input the name of this file in [File name:].

  (3) Select <ON> in [Load information].

  (4) Select <OFF> in [Load program to E7000].

  (5) When the load module is loaded, the debugging information is loaded only to the GUI-SH and not to the E7000PC emulator and the user system. Accordingly, the program can be debugged at the source level on the GUI-SH.

Related Function

Source area, command area
GUI commands: [Load-Load other file...] and [Load-E7000 load...]

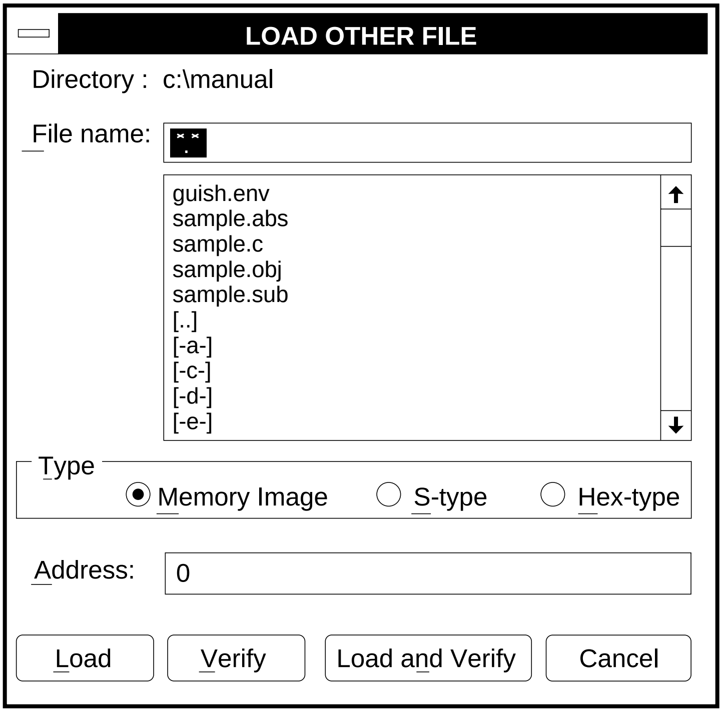
5.3.2 Loading Other Files (Alt + K)

[Load - Load other file...] LOAD OTHER FILE dialog box

Overview

Loads a memory image file, an S-type load module file, and a Hex-type load module file into the E7000PC emulator and compares memory contents with the file for verification purposes.

Window



Directory : c:\manual

File name: *.*

guish.env
sample.abs
sample.c
sample.obj
sample.sub
[..]
[-a-]
[-c-]
[-d-]
[-e-]

Type
● Memory Image ○ S-type ○ Hex-type

Address: 0

Figure 5-11 LOAD OTHER FILE Dialog Box

Operation

- [File name:] text box
  Specifies the name of the file to be loaded. The file name can be selected from the list box; the selected file name is displayed in the text box.

  Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list box starts loading.

  When the LOAD OTHER FILE dialog box opens, the text box displays *.*, and therefore, the list box displays a list of file names in the current directory.

When the file to be loaded is not in the current directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

(1) Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

(2) Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory including the target file is reached.

- [Directory:]
  Displays the current directory. When the drive or subdirectory is changed in [File name:], this display also changes.

- [Type] group box
  Selects the format of the load module file to be loaded into the E7000PC emulator.

  <Memory Image>: Loads a memory image file (default at system initiation).
  <S-type>: Loads an S-type load module file.
  <Hex-type>: Loads a Hex-type load module file.

- [Address:] text box
  Specifies the memory start address when loading a memory image file. The offset value is input when loading an S-type load module or a Hex-type load module file.

- <Load> button
  Clicking the <Load> button starts loading the file specified in [File name:], closes the LOAD OTHER FILE dialog box, and stores the dialog box settings.

- <Verify> button
  Clicking the <Verify> button starts verifying memory contents against the file specified in [File name:], closes the LOAD OTHER FILE dialog box, and stores the dialog box settings.

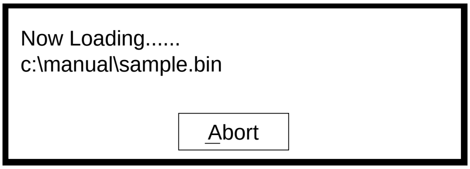
- <Load and Verify> button
  Clicking the <Load and Verify> button starts loading the file specified in [File name:]. After loading is completed, the load module file and memory contents are automatically verified. The LOAD OTHER FILE dialog box is closed and the dialog box settings are stored.

- <Cancel> button
  Clicking the <Cancel> button closes the LOAD OTHER FILE dialog box without storing the dialog box settings. The settings return to those when the LOAD OTHER FILE dialog box was opened.

Function

- Other file load
  Clicking the <Load> button loads the file specified in [File name:] into the E7000PC emulator in the format selected in the [Type] group box. The load address is input to the [Address:] text box when loading a memory image file. The offset value from the start address is input when loading an S-type load module file or a Hex-type load module file. With this command, debugging information previously loaded by the load program file will not be lost.

  During the load, the LOAD INFORMATION dialog box (figure 5-12) automatically opens. When the load is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort the load.

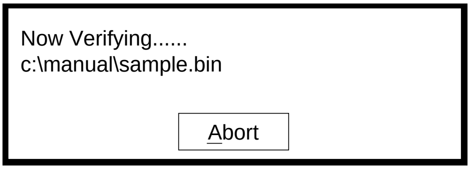


Now Loading......
c:\manual\sample.bin

Figure 5-12 LOAD INFORMATION Dialog Box (Other File)

- Other file verification
  Clicking the <Verify> button verifies the memory contents against the file specified in [File name:] in the format selected in the [Type] group box. Debugging information for the load module previously loaded is not lost.

  During verification, the VERIFY INFORMATION dialog box (figure 5-13) automatically opens.

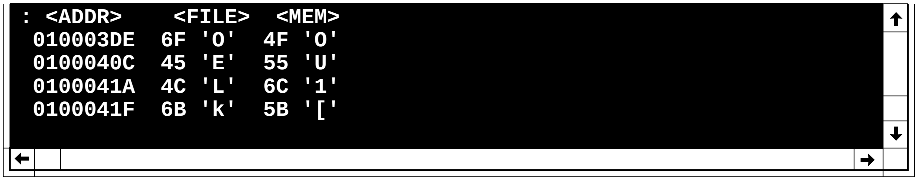


Now Verifying......
c:\manual\sample.bin

Figure 5-13 VERIFY INFORMATION Dialog Box (Other File)

If a verification error occurs, the verification results are displayed in the command area. When verification is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort verification.

The verification error message displayed in the command area is shown in figure 5-14.



```
: <ADDR>    <FILE>  <MEM>
 010003DE  6F 'O'  4F 'O'
 0100040C  45 'E'  55 'U'
 0100041A  4C 'L'  6C '1'
 0100041F  6B 'k'  5B '['
```

① <ADDR>: Address where a verification error occurs
② <FILE>: Data in the file specified in [File name:] in hexadecimal and ASCII characters
③ <MEM>: Data in memory in hexadecimal and ASCII characters

Figure 5-14 Verification Error Message (Other File)

- Automatic verification after loading
  Clicking the <Load and Verify> button starts loading the file specified in [File name:] in the format selected in the [Type] group box. After completing loading normally, the loaded file and the memory contents are automatically verified.

Related Function

Command area, [Load - Load program file...]

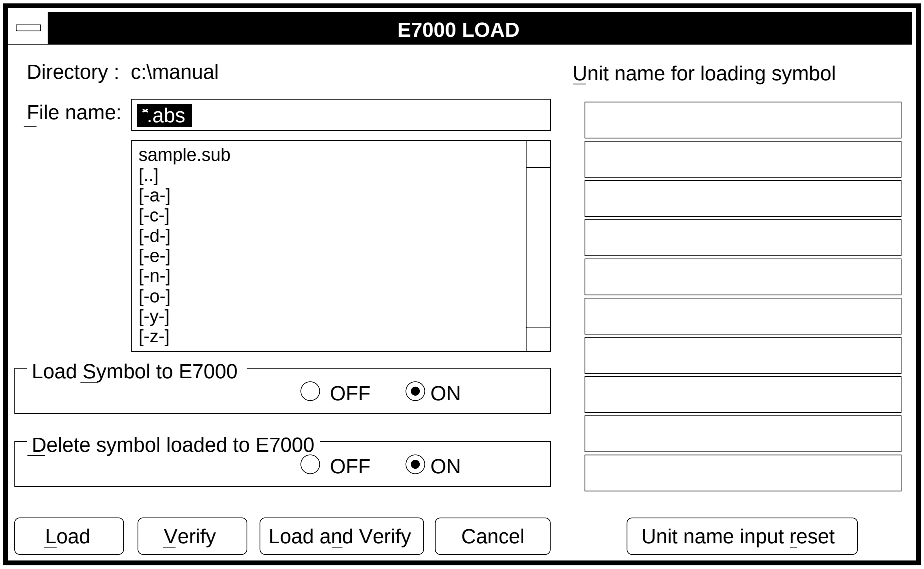
5.3.3 Loading Directly into the E7000 (Alt + J)

[Load-E7000 load...] E7000 LOAD dialog box

Overview

Loads a load module file to be debugged only into the E7000PC emulator. Symbol information can be registered in the E7000PC emulator in load module file units. Memory contents and the load module file contents can be verified.

Window



E7000 LOAD

Directory : c:\manual

File name: *.abs

sample.sub
[..]
[-a-]
[-c-]
[-d-]
[-e-]
[-n-]
[-o-]
[-y-]
[-z-]

Load Symbol to E7000 ○ OFF ● ON

Delete symbol loaded to E7000 ○ OFF ● ON

Unit name for loading symbol

Figure 5-15 E7000 LOAD Dialog Box

Operation

- [File name:] text box
  Specifies the name of the file to be loaded. The file name can be selected from the list box; the selected file name is displayed in the text box.
  The file that can be input is an absolute load module file or a multi-load module file.

  Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list box starts loading.

  When the E7000 LOAD dialog box opens, the text box displays *.abs, and therefore the list box displays a list of file names having extension .abs in the current directory. A file having extension .abs is an absolute load module file or an multi-load module file.

When the file to be loaded is not in the current directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

(1) Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and the current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

(2) Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After it is selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory including the target file is reached.

- [Directory:]
  Displays the current directory. When the name or subdirectory is changed in [File name:], this display also changes.

- [Load Symbol to E7000] group box
  Specifies whether the symbol information is registered in one unit or all units.
  <ON>: Registers all units of symbol information (default at system initiation)
  <OFF>: Registers one unit of symbol information

- [Delete symbol loaded to E7000] group box
  Selects whether or not the registered symbol information is deleted before loading the load module file into the E7000PC emulator.
  <ON>: Deletes the registered symbol information (default at system initiation)
  <OFF>: Does not delete the registered symbol information.

- [Unit name for loading symbol] text box
  Specifies the unit name of the symbol information to be registered.

- <Load> button
  Clicking the <Load> button starts loading a load module file, closes the E7000 LOAD dialog box, and stores the dialog box settings.

- <Verify> button
  Clicking the <Verify> button starts verifying memory contents against the load module file, closes the E7000 LOAD dialog box, and stores the dialog box settings.

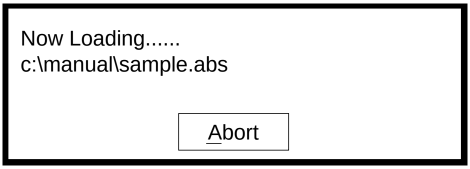
- <Load and Verify> button
  Clicking the <Load and Verify> button starts loading the load module file. After loading is completed, the load module file and the memory contents are automatically verified. The E7000 LOAD dialog box is closed and the dialog box settings are stored.

- <Cancel> button
  Clicking the <Cancel> button closes the E7000 LOAD dialog box without storing the dialog box settings. The dialog box settings return to those when the E7000 LOAD dialog box was opened.

- <Unit name input reset> button
  Clicking the <Unit name input reset> button deletes the contents of the [Unit name for loading symbol] text box.

Function

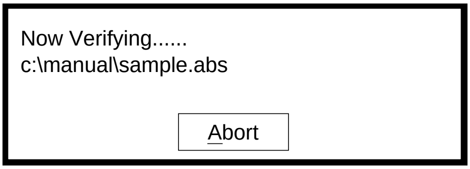
- Program load
  An absolute load module file (or a multi-load module file) program can be loaded into the E7000PC emulator.
  With this command, debugging information for the load module previously loaded will not be lost.
  When loading a load module file, symbol information registration and registered symbol information deletion can be specified.
  The [Load Symbol to E7000] group box selects symbol information registration. Selecting <ON> registers all symbol information of the load module into the E7000PC emulator; selecting <OFF> registers only the symbol information in the unit specified in the [Unit name for loading symbol] text box to the E7000PC emulator.
  The [Delete symbol loaded to E7000] group box deletes the symbol information registered in the E7000PC emulator. Selecting <ON> deletes symbol information, while selecting <OFF> does not delete.
  During the load, the LOAD INFORMATION dialog box (figure 5-16) automatically opens.
  When the load is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort the load.



Now Loading......
c:\manual\sample.abs

Figure 5-16 LOAD INFORMATION Dialog Box (E7000 LOAD)

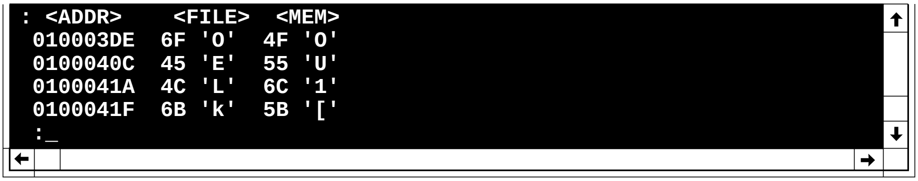
- Program verification
  Clicking the <Verify> button verifies the memory contents against the file specified in [File name:]. Debugging information for the load module previously loaded is not lost.
  During verification, the VERIFY INFORMATION dialog box (figure 5-17) automatically opens.



Now Verifying......
c:\manual\sample.abs

Figure 5-17 VERIFY INFORMATION Dialog Box (E7000 LOAD)

If a verification error occurs, the verification results are displayed in the command area. When verification is completed, the dialog box automatically closes. Click the <Abort> button in the dialog box to abort verification.
The verification error message displayed in the command area is shown in figure 5-18.



```
: <ADDR>    <FILE>  <MEM>
 010003DE  6F 'O'  4F 'O'
 0100040C  45 'E'  55 'U'
 0100041A  4C 'L'  6C '1'
 0100041F  6B 'k'  5B '['
 :_
```

① <ADDR>: Address where a verification error occurs
② <FILE>: Data in the load module file in hexadecimal and ASCII characters
③ <MEM>: Data in memory in hexadecimal and ASCII characters

Figure 5-18 Verification Error Message

- Automatic verification after loading
  Clicking the <Load and Verify> button starts loading the file specified in [File name:]. When file load is completed normally, the loaded file and the memory contents are automatically verified.

Related Function

Command area, [Load - Load program file...]

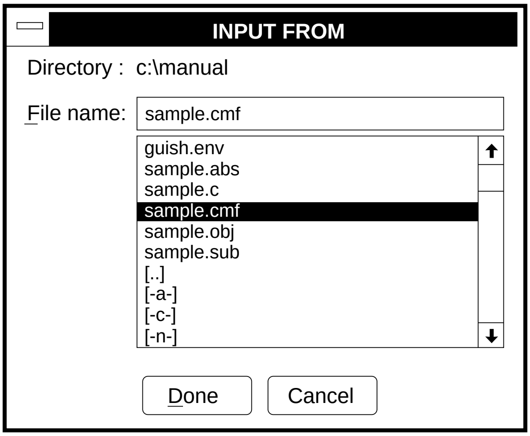
5.3.4 Inputting and Executing Command File (Alt + I)

[Input from...] INPUT FROM dialog box

Overview

Inputs a command file and automatically executes emulator commands.

Window



INPUT FROM

Directory : c:\manual

File name: sample.cmf

guish.env
sample.abs
sample.c
sample.cmf
sample.obj
sample.sub
[..]
[-a-]
[-c-]
[-n-]

Figure 5-19 INPUT FROM Dialog Box

Operation

- [File name:] text box
  Specifies the name of the command file to be input. The file name can be selected from the list box; the selected file name is displayed in the text box.

  Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list box starts automatic execution.

  When the INPUT FROM dialog box opens, the text box displays *.*, and therefore the list box displays all file names in the current directory. When the file to be input is not in the current directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

  (1) Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

(2) Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory including the target file is reached.

- [Directory:]
  Displays the current directory. When the drive or subdirectory is changed in [File name:], this display also changes.

- <Done> button
  Clicking the <Done> button stores the INPUT FROM dialog box settings in the GUI-SH and closes the dialog box. Then the command file specified in [File name:] is automatically executed. The command execution results are displayed in the command area.

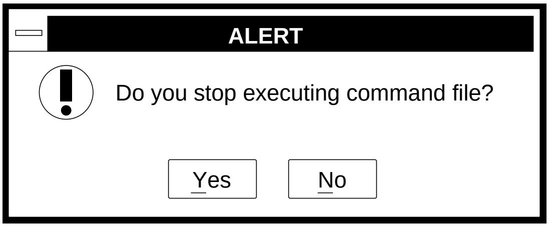
- <Cancel> button
  Clicking the <Cancel> button closes the INPUT FROM dialog box without storing the INPUT FROM dialog box settings. The settings return to those when the INPUT FROM dialog box was opened.

Function

The emulator commands included in the command file specified in [File name:] are automatically executed. For command file creation, refer to appendix C, Command File Creation.

Clicking the <Done> button starts emulator command execution. The command execution results are displayed in the command area.

Click the <STOP> button on the tool bar to suspend command execution, and display stop confirmation message box shown in figure 5-20 is displayed. Click the <No> button to continue command execution. When the <Yes> button is clicked, emulator command execution terminates.



ALERT

Do you stop executing command file?

Figure 5-20 Stop Confirmation Message Box (INPUT FROM)

Related Function

Command area, <STOP> button

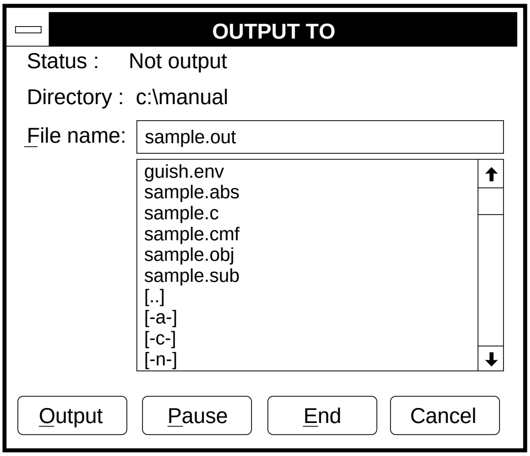
5.3.5 Storing Command Area Contents (Alt + O)

[Output to...] OUTPUT TO dialog box

Overview

Outputs the contents input to and displayed in the command area to a file.

Window



Status : Not output

Directory : c:\manual

File name: sample.out

```
guish.env
sample.abs
sample.c
sample.cmf
sample.obj
sample.sub
[..]
[-a-]
[-c-]
[-n-]
```

Figure 5-21 OUTPUT TO Dialog Box

Operation

- [Status:] text box
  Displays file output status as shown in table 5-1.

Table 5-1 Output Status Messages (OUTPUT TO Dialog Box)

Message	Description
Not output	No file output
Output	File being output
Pause	File output suspended

- [File name:] text box
  Specifies the name of the file for storing the command area contents. The specified file is saved in the current directory, which is displayed in [Directory:].

  Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list box starts output to the specified file.

  To save a file in a different directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

  (1) Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

  (2) Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

  (3) Continue selecting subdirectories until the directory including the target file is reached.

- [Directory:]
  Displays the current directory. When the drive or subdirectory is changed in [File name:], this display also changes.

- <Output> button
  Clicking the <Output> button starts outputting command area information to the file specified in [File name:]. When Pause is displayed at [Status:], that is, file output has been temporarily suspended, clicking the <Output> button restarts the output. After this button is clicked, Output is displayed at [Status:].

- <Pause> button
  When Output is displayed at [Status:], clicking the <Pause> button changes display at [Status:] to Pause and suspends outputting to a file until the <Output> button is clicked.

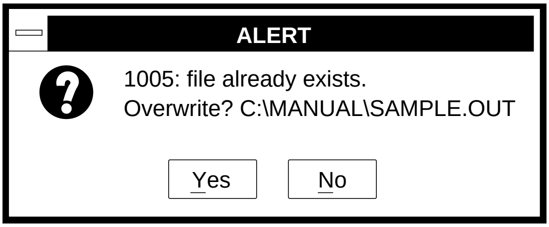
- <End> button
  When Output or Pause is displayed at [Status:], clicking the <End> button changes file output status to Not output at [Status:] and terminates outputting to a file.

- <Cancel> button
  Clicking the <Cancel> button closes the OUTPUT TO dialog box without storing the OUTPUT TO dialog box settings. The settings return to those when the OUTPUT TO dialog box was opened.

Function

- Output to a file
  Clicking the <Output> button outputs the contents input to and displayed in the command area to the file specified in [File name:]. Even when the command area of the base window is closed, the display contents are stored in the file.
  Outputting to a file depends on the file output status displayed at [Status:]

  — Not output is displayed
  Clicking the <Output> button checks whether or not a file having the same name as that specified in [File name:] exists. When such a file is found, the message box shown in figure 5-22 appears. Click <Yes> to change file output status to Output and start overwriting the file. When <No> is clicked, file output status does not change and display returns to the OUTPUT TO dialog box.



ALERT

1005: file already exists.
Overwrite? C:\MANUAL\SAMPLE.OUT

Figure 5-22 File Overwrite Confirmation Message (OUTPUT TO)

  — Pause is displayed
  Clicking the <Output> button restarts suspended file output.

- Relation between file output status and buttons
  Clicking buttons in the OUTPUT TO dialog box changes the file output status displayed at [Status:] as shown in table 5-2.

Table 5-2 Output File Status Message and Buttons in OUTPUT TO Dialog Box

Message	Description	Related Button
Not output	No file output	After initiation or <End> button clicked
Output	File output started or restarted	<Output> button clicked
Pause	File output suspended	<Pause> button clicked

Related Function

Command area

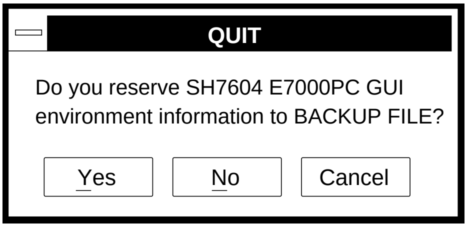
5.3.6 Terminating GUI-SH (Alt + F4)

[Quit...] QUIT dialog box

Overview

Terminates the GUI-SH with a prompt asking whether to store debugging settings in a backup file.

Window



Do you reserve SH7604 E7000PC GUI environment information to BACKUP FILE?

Figure 5-23 QUIT Dialog Box

Operation

- <Yes> button
  Clicking the <Yes> button stores debugging settings in a backup file and then terminates the GUI-SH.

- <No> button
  Clicking the <No> button terminates the GUI-SH without storing the settings.

- <Cancel> button
  Clicking the <Cancel> button returns to the previous state without terminating the GUI-SH.

Function

- Output to a backup file
  Clicking the <Yes> button in the QUIT dialog box displays the BACKUP FILE dialog box for storing a backup file, as shown in figure 5-24.

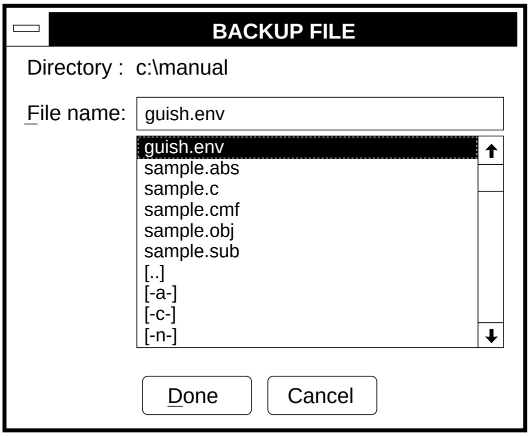


Figure 5-24 BACKUP FILE Dialog Box

— [File name:] text box
Specifies the name of the backup file for storing debugging settings. The specified file is output to the current directory, which is displayed in [Directory:]. Pressing the (Enter) key after specifying a file name or double-clicking the file name selected in the list box starts storing the backup file.

To output a file to a different directory, select the drive and the directory from the list box to change the current directory, using the following procedure:

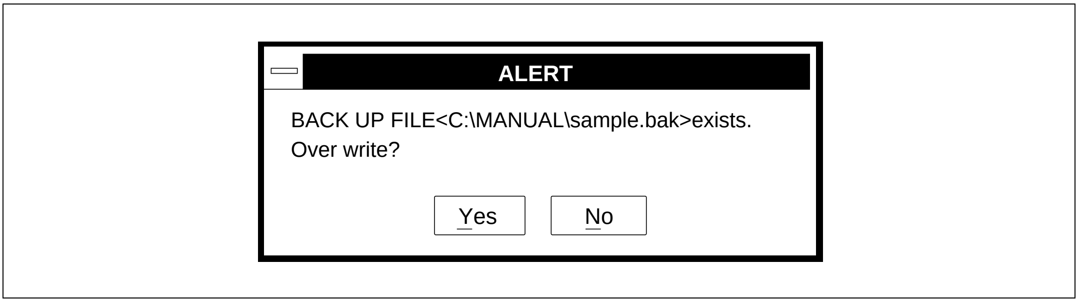
(1) Select the target drive from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory of the selected drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

(2) Select a subdirectory from the list box by double-clicking. Subdirectory names are displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:] and the subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory including the target file is reached.

— [Directory:]
Displays the current directory. When the drive or subdirectory is changed in [File name:], this display also changes. If a backup file is found when the BACKUP FILE dialog box is open, the subdirectory of the backup file is displayed.

— <Done> button
Clicking the <Done> button starts storing debugging settings to the backup file. When a file having the same name as that specified in [File name:] already exists, the message box shown in figure 5-25 appears. Click <Yes> to start overwriting the file. When <No> is clicked, display returns to the QUIT dialog box.



BACK UP FILE<C:\MANUAL\sample.bak>exists.
Over write?

Figure 5-25 File Overwrite Confirmation Message (BACKUP FILE)

— <Cancel> button
Clicking the <Cancel> button closes the BACKUP FILE dialog box and returns to the QUIT dialog box without storing the settings. The settings return to those when the BACKUP FILE dialog box was opened.

- Information stored in the backup file
  The information to be stored in the backup file is listed in table 5-3.

Table 5-3 Information Stored in Backup File (QUIT Dialog Box)

Information	Command
Directory and file extension	[Load - Load program file...], [Load - Load other file...], [Load - E7000 load...], [Input from...], [Output to...], [Quit...] (file extensions are not stored), [Trace display - Text], [Memory - Dump]
Settings in dialog window	[Load - Load program file...], [Load - Load other file...], [Load - E7000 load...], [Input from...], [Output to...], [Execution mode...]*, [Go...], [Break condition1,2,3,4,5...]*, [Trace memory...], [Trace condition...]*, [Trace display - Setting...], [Memory - Setting...], [Source - Setting...]
Window location and size	All debug windows (return to the stored size and location when reopened). Note, however, that the dialog window position cannot be stored.

Note: The settings of the commands marked with * must be set to the E7000PC at E7000PC re-initialization using the <Done> button of each command because they are stored only in the GUI-SH.

Related Function

Loading backup file

5.4 Execution Function

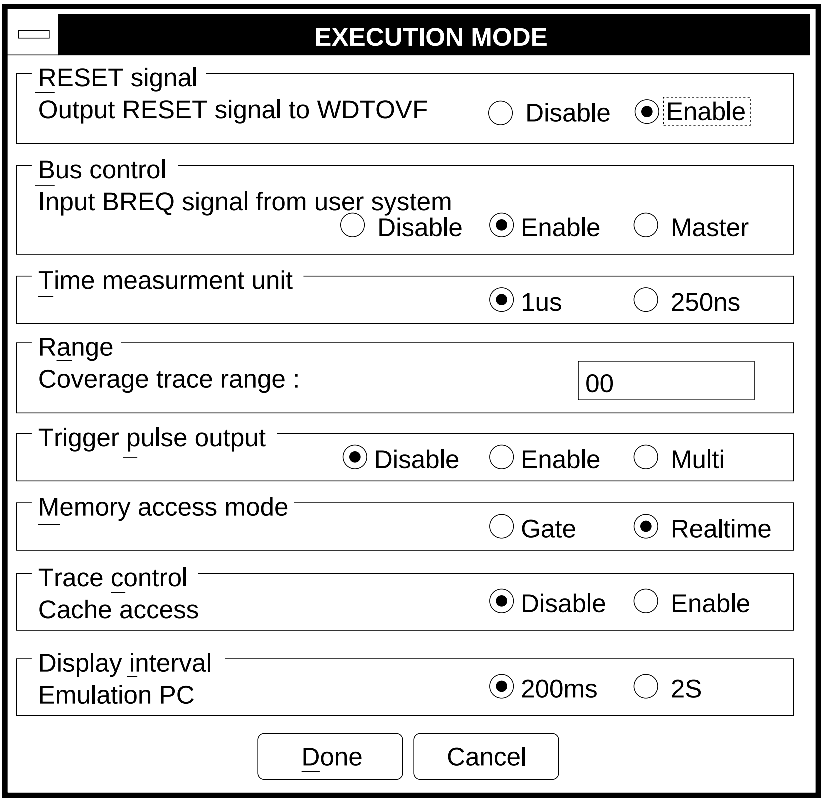
5.4.1 Specifying Emulation Execution Conditions (Ctrl + E)

[Execution mode...] EXECUTION MODE dialog box

Overview

Specifies emulation execution conditions.

Window



EXECUTION MODE

RESET signal
Output RESET signal to WDTOVF ○ Disable ● Enable

Bus control
Input BREQ signal from user system
○ Disable ● Enable ○ Master

Time measurment unit ● 1us ○ 250ns

Range
Coverage trace range : 00

Trigger pulse output ● Disable ○ Enable ○ Multi

Memory access mode ○ Gate ● Realtime

Trace control
Cache access ● Disable ○ Enable

Display interval
Emulation PC ● 200ms ○ 2S

Figure 5-26 EXECUTION MODE Dialog Box

Operation

- [RESET signal]
  [Output RESET signal to WDTOVF] group box
  Specifies if the system reset signal is to be output to the WDTOVF pin of the SH7604 during a clock-switch reset, a cycle reset (if [Emulation mode:] of [Execution - Go...] command is specified), or the execution of a RESET command (emulator command) as follows:

<Disable>:	Disables system reset signal output
<Enable>:	Enables system reset signal output (default at system initiation)

- [Bus control]
  [Input BREQ signal from user system] group box
  Specifies if the bus request signal input (BREQ) is enabled in the user system during emulation as follows:

<Disable>:	Disables BREQ signal input
<Enable>:	Enables BREQ signal input (default at system initiation)
<Master>:	Disables BREQ signal input during E7000PC operation, and enables it during emulation.

- [Time measurement unit] group box
  Specifies the time unit to be used when measuring execution time with the [Go...] command as follows:

<1us>:	Measured in 1-µs units (default at system initiation)
<250ns>:	Measured in 250-ns units

- [Range]
  [Coverage trace range:] text box
  Must not be used because this is reserved for future extension.

- [Trigger pulse output] group box
  Specifies whether to continue program execution and output a pulse from the trigger output pin in the pod when the hardware break conditions (set by the [Break condition1,2,3,4,5...] command) have been satisfied as follows:

<Disable>:	Terminates program execution instead of outputting a pulse when the break conditions have been satisfied (default at system initiation)
<Enable>:	Continues program execution and outputs a pulse when the break conditions have been satisfied
<Multi>:	Terminates program execution and outputs a pulse when the break conditions have been satisfied

- [Memory access mode] group box
  Specifies the memory access mode during emulation execution to be set to the realtime mode or gate insertion mode as follows:

<Gate>:	Sets memory access mode to the gate insertion mode (default at system initiation)
<Realtime>:	Sets memory access mode to the realtime mode

- [Trace control]
  [Cache access] group box
  Selects whether or not trace information is acquired at cache access. An operation speed is reduced when trace information is acquired at cache access because the external bus is used for trace information acquisition.

<Disable>:	Disables trace information acquisition at cache access
<Enable>:	Enables trace information acquisition at cache access (default at system initiation)

- [Display interval]
  [Emulation PC] group box
  Selects the display interval of the program counter to be displayed in the emulator message area during emulation.

<200ms>:	Displays the program counter every 200 ms (default at system initiation)
<2s>:	Displays the program counter every 2 s.

- <Done> button
  Clicking the <Done> button stores the EXECUTION MODE dialog box settings in the GUI-SH, closes the EXECUTION MODE dialog box, and sends the above settings to the E7000PC emulator.

- <Cancel> button
  Clicking the <Cancel> button closes the EXECUTION MODE dialog box and returns the system to its state before the EXECUTION MODE dialog box was opened without storing the above settings.

Function

Specifies emulation execution conditions.

Related Functions

GUI commands: [Go...], [Break...], and [Break condition1,2,3,4,5...]
Emulator commands: RESET and BREAK_SEQUENCE

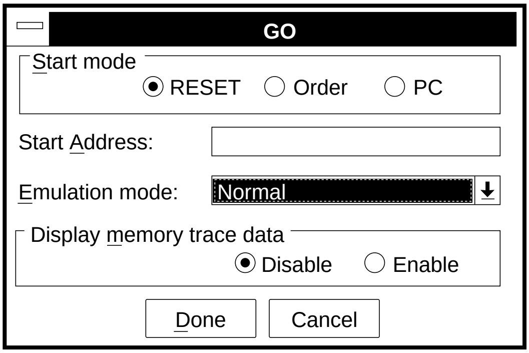
5.4.2 Executing Program Emulation (ALt + Ctrl + G)

[Go...] GO dialog box

Overview

Performs emulation.

Window



GO

Start mode

● RESET ○ Order ○ PC

Start Address:

Emulation mode: Normal

Display memory trace data

● Disable ○ Enable

Figure 5-27 GO Dialog Box

Operation

- [Start mode] group box
  Specifies the emulation start method as follows:
  <RESET>: Initiates emulation from the reset vector (default at system initiation)
  <Order>: Initiates emulation from the address specified in [Start Address:]
  <PC>: Initiates emulation from the current PC

- [Start Address:] text box
  When <Order> is specified in [Start mode], specifies the emulation start address in hexadecimal or as a symbol. When there is no specification, an error occurs.

- [Emulation mode:] drop-down list box
  Specifies the emulation method.

  — <Normal>: Normal mode (default at system initiation)

  — <Sequential break 2-1>: Sequential break mode
  The program stops only when hardware break conditions set by [Break condition2...] and [Break condition1...] commands are satisfied in the sequence of <condition 2> followed by <condition 1>.

— <Cycle reset 32us>: Cycle reset mode 1
  Continues execution while forcibly inputting the RES signal to the SH7604 every specified time and simultaneously outputting the oscilloscope trigger signal from the trigger output probe pin. In this mode, conditions specified with the [Break...], [Break condition1,2,3,4,5...], [Trace condition...], and BREAK_SEQUENCE (emulator) commands are ignored. Also, parallel mode cannot be entered from this mode.
  Note that the [Display memory trace data] specification is automatically set to <Disable>. (These conditions are the same in the following cycle reset modes.)

— <Cycle reset 96us>: Cycle reset mode 2

— <Cycle reset 512us>: Cycle reset mode 3

— <Cycle reset 1.024ms>: Cycle reset mode 4

— <Cycle reset 5.12ms>: Cycle reset mode 5

— <Cycle reset 10.24ms>: Cycle reset mode 6

— <Cycle reset 51.2ms>: Cycle reset mode 7

— <Cycle reset 102.4ms>: Cycle reset mode 8

— <Cycle reset 512ms>: Cycle reset mode 9

— <Ignore break>: Break conditions are temporarily invalidated

- [Display memory trace data] group box
  Specifies if the memory contents at the address specified in [Trace Memory Condition] of the [Trace memory...] command is displayed in the emulation information area during emulation execution, as follows:
  <Disable>: Disables memory content display (default at system initiation)
  <Enable>: Enables memory content display

- <Done> button
  Clicking the <Done> button stores the GO dialog box settings, closes the GO dialog box, and starts emulation execution.

- <Cancel> button
  Clicking the <Cancel> button closes the GO dialog box and returns the system to its state before the GO dialog box was opened without storing the above settings.

Function

- Emulation mode
  Emulation execution is initiated in the mode specified in [Start mode:]. One of the following emulation modes can be selected.

  — Sequential break mode
  The program stops only when hardware break conditions set by [Break condition2...] and [Break condition1...] commands are satisfied in the sequence of <condition 2> followed by <condition 1>.

  — Cycle reset modes
  A RES signal is forcibly output from the E7000PC emulator to the user system at specified intervals. At the same time, while continuing emulation execution, a trigger signal for an oscilloscope is output from the trigger output probe pin. In this mode, all break conditions and trigger conditions are invalidated.

  — Break condition temporary invalidation
  Temporarily invalidates the settings of the [Break...] and the [break condition1,2,3,4,5...] commands and executes the program. An invalidation interval is during program execution with one [Go...] command.

- Parallel mode
  Enters parallel mode when (Space) or (Enter) is input from the command area during emulation. For details, refer to the E7000PC user's manual.

- Forcible emulation termination
  To forcibly terminate emulation, click the <STOP> button or input (Ctrl + C). In parallel mode, execute the ABORT command in the command area.

- Causes of GO command emulation termination
  Table 5-4 lists the messages displayed in the emulation information area when emulation is terminated.

Table 5-4 Emulation Termination Causes (GO Dialog Box)

Display	Termination Cause
BREAK KEY	Forcibly terminated by pressing the (Ctrl + C) keys or the <STOP> button
BREAK POINT	Emulation stops at a breakpoint specified with the [Break...] command
STOP ADDRESS	Emulation initiated by the <CONTINUE> button stops at where the cursor indicates
BREAK SEQUENCE	PC break condition specified with the BREAK_SEQUENCE command (emulator command) was satisfied
BREAK CONDITION1	A break condition specified with the [Break condition1...] command was satisfied
BREAK CONDITION2	A break condition specified with the [Break condition2...] command was satisfied
BREAK CONDITION3	A break condition specified with the [Break condition3...] command was satisfied
BREAK CONDITION4	A break condition specified with the [Break condition4...] command was satisfied
BREAK CONDITION5	A break condition specified with the [Break condition5...] command was satisfied
BREAK CONDITION1,2,3,4,5	A break condition specified with a [Break condition1,2,3,4,5...] command was satisfied
BREAK CONDITION SB	Sequential break conditions specified with [Break condition1,2...] commands were satisfied
GUARDED AREA ACCESSED	A guarded area was accessed
WRITE PROTECT	A write-protected area was written to
ILLEGAL INSTRUCTION	A break instruction was executed
NO EXECUTION	The user program was not executed.
RESET IN BY E7000	Forcibly terminated with the RES signal output from the E7000PC because an error has occurred in the user system
DMA GUARDED OR WRITE PROTECT	A write-protected area is written to or a guarded memory area is accessed by DMA during [Break...] command processing

Note

If the condition of the [Break condition1,2,3,4,5...] command is satisfied during program execution, program execution may not stop immediately; it may stop after executing multiple instructions.

Related Functions

GUI commands: [Execution mode...], [Break...], [Break condition1,2,3,4,5...], [Trace memory...], and [Trace condition...]
Emulator command: BREAK_SEQUENCE

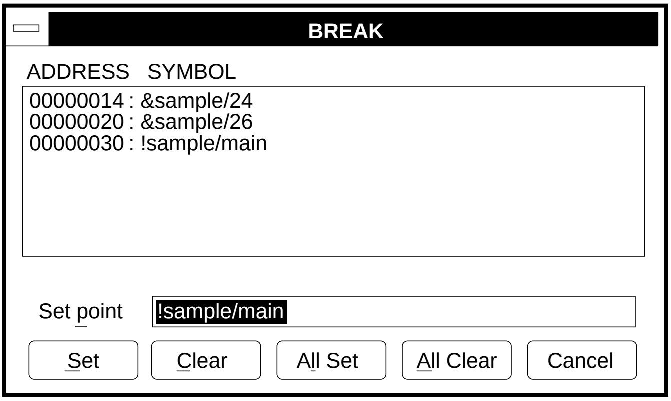
5.4.3 Setting and Cancelling Breakpoints **(Alt + Ctrl + B)**

[Break...] BREAK dialog box

Overview

Sets and cancels breakpoints.

Window



BREAK

ADDRESS SYMBOL

```
00000014 : &sample/24
00000020 : &sample/26
00000030 : !sample/main
```

Set point: !sample/main

Figure 5-28 BREAK Dialog Box

Operation

- Breakpoint display area
  Displays breakpoint settings in the format shown in figure 5-29.

```
ADDRESS      SYMBOL

XXXXXXXX     XXXXXXXX...............XXX
    ①            ②
```

Figure 5-29 Display Format for Breakpoint Information (BREAK Dialog Box)

① ADDRESS: Displays the address where a breakpoint is set

② SYMBOL: Displays a symbol name, a line number symbol, or an address (when there is no corresponding symbol information)

Displayed breakpoints are set by the [Break...] command or the<SET> button on the tool bar within the base window. The breakpoints set by the BREAK command (emulator command) in the command area are not displayed.
By selecting with the mouse pointer and double-clicking a line including a breakpoint, the information displayed under SYMBOL is automatically set in [Set point:].

- [Set point:] text box
  Accepts input of an address in which a breakpoint is to be set or cancelled in hexadecimal or as a symbol.

- <Set> button
  Clicking the <Set> button after specifying an address in [Set point:] specifies a breakpoint at that address, and displays the address in the breakpoint display area.

- <Clear> button
  Clicking the <Clear> button cancels the breakpoint set at the address which is input in [Set point:], or the breakpoint at the address selected in the breakpoint display area. The cancelled breakpoint disappears from the breakpoint display area.

- <All Set> button
  Clicking the <All Set> button clears all breakpoints set in the E7000PC emulator at once and then sets breakpoints to the addresses shown in the breakpoint display area. Therefore, all breakpoints set by the BREAK command (emulator command) in the command area are cleared.

- <All Clear> button
  Clicking the <All Clear> button clears all the breakpoints set in the E7000PC emulator, including those shown in the breakpoint display area.

- <Cancel> button
  Clicking the <Cancel> button closes the BREAK dialog box after storing all of the above settings.

Function

- Setting
  A breakpoint is set at the address input in [Set point:]. A maximum of 255 breakpoints can be set in total. A BP mark is displayed on the corresponding source line.
  A breakpoint can be set by clicking the <SET> button on the tool bar or by entering the BREAK command (emulator command) in the command area.
  When setting a breakpoint with the <SET> button, select a source line with the mouse pointer and click the <SET> button. Breakpoints set in this way are displayed in the breakpoint display area.
  However, breakpoints set with the BREAK command (emulator command) are not displayed in the breakpoint display area.

- Cancellation
  Breakpoints set at addresses specified in [Set point:] or those selected in the breakpoint display area are cancelled. The BP mark displayed on the source line corresponding to the cancelled breakpoint disappears.
  A breakpoint can be cancelled by clicking the <CLEAR> button on the tool bar or by executing the BREAK command (emulator command) in the command area.
  When cancelling a breakpoint with the <CLEAR> button, click the source line on which the BP mark is displayed to display the cursor, and then click the <CLEAR> button. The cancelled breakpoint disappears from the breakpoint display area.

Notes

(1) Breakpoints set by the [Break...] command become invalid during step execution using the <STEP> or the <STEP_OVER> button.

(2) If the breakpoint is set to the address specified by [Break condition 2...] by the [Break...], command, breakpoints set by [Break condition 2...] becomes invalid.

(3) Do not set breakpoints to the slot instruction of the delayed branch instruction using the [Break...] command; it causes a slot illegal interrupt.

Related Functions

Source area, and <STEP>, <STEP_OVER>, <SET>, and <CLEAR> buttons
GUI command: [Break condition1,2,3,4,5...]
Emulator command: BREAK

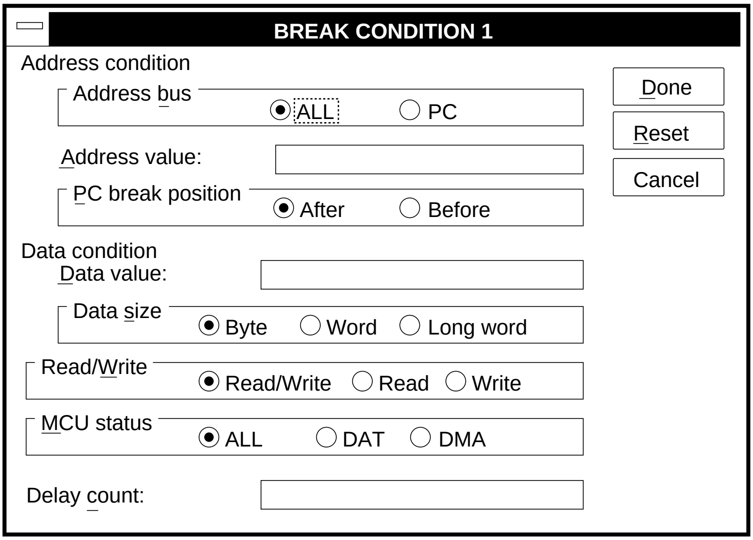
5.4.4 Specifying Hardware Break Condition1 (Ctrl+ 1)

[Break Condition1...] BREAK CONDITION1 dialog box

Overview

Specifies a hardware break condition (BREAK CONDITION1). When all of the specified conditions are satisfied during emulation execution with the [GO...] command, program execution terminates.

Window



BREAK CONDITION 1

Address condition

Address bus ● ALL ○ PC

Address value:

PC break position ● After ○ Before

Data condition

Data value:

Data size ● Byte ○ Word ○ Long word

Read/Write ● Read/Write ○ Read ○ Write

MCU status ● ALL ○ DAT ○ DMA

Delay count:

Figure 5-30 BREAK CONDITION1 Dialog Box

Operation

- [Address condition]

— [Address bus] group box
Specifies an address bus type as a break condition.
<ALL>: All address bus cycles (default at system initiation)
<PC>: Only program fetch cycles

— [Address value:] text box
Specifies an address bus value where a hardware break is to occur.

— [PC break position] group box
  Specifies whether the break is to occur after or before execution of the instruction when <PC> is specified in [Address bus], as follows:

  <After>: The break occurs after instruction execution (default at system initiation)
  <Before>: The break occurs before instruction execution

- [Data condition]

— [Data value:] text box
  Specifies a data bus value where a hardware break is to occur. If this data is specified, no break occurs during program fetch cycles. A data bus value can be masked.

— [Data size] group box
  Specifies the size of the data bus value specified in [Data value:] as follows:

<Byte>:	1-byte units (default at system initiation)
<Word>:	2-byte units
<Long word>:	4-byte units

- [Read/Write] group box
  Specifies a read or write cycle condition as a hardware break condition as follows:
  <Read/Write>: Specifies both read and write cycles as hardware break conditions (default at system initiation)
  <Read>: Specifies a read cycle as a hardware break condition
  <Write>: Specifies a write cycle as a hardware break condition

- [MCU status] group box
  Specifies an SH7604-bus cycle condition as a hardware break condition as follows. For memory access cycle conditions in PC relative instructions such as MOV.W @(10, PC), R0, data access cycle <DAT> must be specified.
  <ALL>: Specifies all bus cycles as hardware break conditions (default at system initiation)
  <DAT>: Specifies a data access cycle as a hardware break condition
  <DMA>: Specifies a DMA cycle as a hardware break condition

- [Delay count:] text box
  Specifies a delay count starting from the time a hardware break condition is satisfied. When conditions (1) to (4) have been satisfied, the program will be terminated after the number of bus cycles specified in [Delay count:] has been executed. A delay count ranging from H'001 to H'FFF can be specified. If nothing is specified, 0 is set.

- <Done> button
  Clicking the <Done> button stores the BREAK CONDITION1 dialog box settings, closes the BREAK CONDITION1 dialog box, and sends the above settings to the E7000PC emulator.

- <Reset> button
  Clicking the <Reset> button cancels the settings in the BREAK CONDITION1 dialog box and the E7000PC emulator, returns the system to a state wherein no hardware break conditions are specified, and closes the BREAK CONDITION1 dialog box.

- <Cancel> button
  Clicking the <Cancel> button closes the BREAK CONDITION1 dialog box and returns the system to its state before the BREAK CONDITION1 dialog box was opened without storing the above settings.

Function

- Setting
  The following conditions can be specified as hardware break conditions: address ([Address condition]), data ([Data condition]), read/write ([Read/Write]), and bus cycle condition ([MCU status]).
  By combining the above conditions with each [Emulation mode:] of the [Go...] command, program emulation can be executed in each emulation mode.

- Specifying conditions for hardware breaks according to access size
  When a data condition is specified, the hardware break condition can only be satisfied when it matches the 32-bit data bus value. Table 5-5 shows the hardware break conditions for each access size.

Table 5-5 Hardware Break Condition Specifications According to Access Size (Break Condition1)

Access Size	Hardware Break Condition Specification Method
Long access	Long data is accessed in one bus cycle. [Address value:] must be a multiple of four and [Data size] must be <Long word> (long data). Other specifications are invalid.
Word access	Word data is accessed in one bus cycle. [Address value:] must be a multiple of two and [Data size] must be <Word> (word data). Other specifications are invalid.
Byte access	Byte data is accessed in one bus cycle. [Data size] must be <Byte> (byte data). Other specifications are invalid. [Address value:] can be both even and odd addresses.

- Mask specifications
  A mask in 1-bit or 4-bit units can be specified for the settings in [Address condition] and [Data condition]. The meaning of a masked bit is that the corresponding bit value is arbitrary when determining whether the condition is satisfied.

— Mask specification for [Address condition]
In [Address value:], mask is performed from the low-order bit. To implement the mask, specify each digit to be masked at input as an asterisk (*). Note that masking cannot be performed when specifying an address range, and that arbitrary bit positions cannot be masked.
For example, if an address condition is to be satisfied when A16 to A31 bits are 0, specify the mask as H'0000****.

— Mask specification for [Data condition]
To implement a mask for [Data condition], specify each digit of the value specified in [Data value:] to be masked at input as an asterisk (*).
For example, if a byte data condition is to be satisfied when the D0 bit is 0, specify the mask as B'*******0.

Notes

(1) During address specification, all address bits are checked. Accordingly, if the break condition is specified in the cache area, no break occurs even if the cache through area in the same memory space is accessed. To set break conditions in both cache area and cache through area, use mask specification.

(2) In a slot instruction of a delayed branch instruction, <Before> cannot be specified in the [PC break position] group box.

(3) The BREAK CONDITION1 function is achieved by the SH7604 user break controller according to the E7000PC emulator specifications. Accordingly, the user cannot use the user break controller for other purposes.

Related Function

[Go...] command

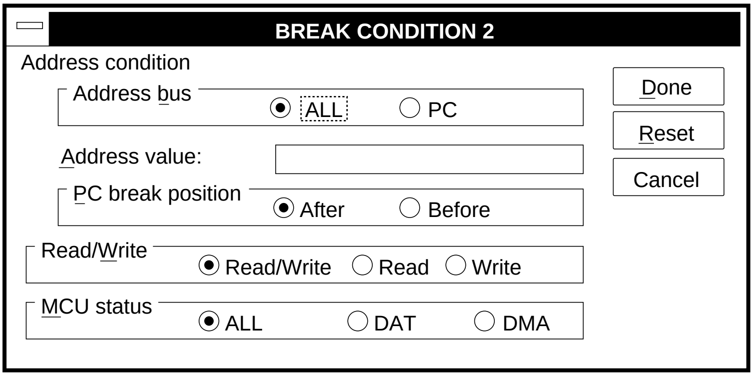
5.4.5 Specifying Hardware Break Condition2 (Ctrl + 2)

[Break Condition2...] BREAK CONDITION2 dialog box

Overview

Specifies a hardware break condition (BREAK CONDITION2). When all of the specified conditions are satisfied during emulation execution with the [Go...] command, program execution terminates.

Window



BREAK CONDITION 2

Address condition

Address bus: ● ALL ○ PC

Address value:

PC break position: ● After ○ Before

Read/Write: ● Read/Write ○ Read ○ Write

MCU status: ● ALL ○ DAT ○ DMA

Figure 5-31 BREAK CONDITION2 Dialog Box

Operation

- [Address condition]

— [Address bus] group box
Specifies an address bus type as a hardware break condition, as follows:
<ALL>: Specifies all address bus cycles as hardware break conditions (default at system initiation)
<PC>: Specifies a program fetch cycle as a hardware break condition

— [Address value:] text box
Specifies an address bus value where a hardware break is to occur.

— [PC break position] group box
Specifies if a break is to occur after or before instructions are fetched when <PC> is specified in [Address bus], as follows:
<After>: A break occurs after instruction execution (default at system initiation)
<Before>: A break occurs before instruction execution

- [Read/Write] group box
  Specifies a read or write cycle condition as a hardware break condition as follows:
  <Read/Write>: Specifies both read and write cycles as hardware break conditions (default at system initiation)
  <Read>: Specifies a read cycle as a hardware break condition
  <Write>: Specifies a write cycle as a hardware break condition

- [MCU status] group box
  Specifies an SH7604-bus cycle condition as a hardware break condition as follows. For memory access cycle conditions in PC relative instructions such as MOV.W @(10, PC), R0, data access cycle <DAT> must be specified.
  <ALL>: Specifies all bus cycles as hardware break conditions (default at system initiation)
  <DAT>: Specifies a data access cycle as a hardware break condition
  <DMA>: Specifies a DMA cycle as a hardware break condition

- <Done> button
  Clicking the <Done> button stores the BREAK CONDITION2 dialog box settings, closes the BREAK CONDITION2 dialog box, and sends the above settings to the E7000PC emulator.

- <Reset> button
  Clicking the <Reset> button cancels the settings in the BREAK CONDITION2 dialog box and the E7000PC emulator, returns the system to a state wherein no hardware break conditions are specified, and closes the BREAK CONDITION2 dialog box.

- <Cancel> button
  Clicking the <Cancel> button closes the BREAK CONDITION2 dialog box and returns the system to its state before the BREAK CONDITION2 dialog box was opened without storing the above settings.

Function

- Setting
  The following conditions can be specified as hardware break conditions: address ([Address condition]), read/write ([Read/Write]), and bus cycle condition ([MCU status]).
  By combining the above conditions with each [Emulation mode:] of the [Go...] command, program emulation can be executed in each emulation mode.

- Mask specifications
  A mask in 1-bit or 4-bit units can be specified for the [Address condition] settings. The meaning of a masked bit is that the corresponding bit value is arbitrary when determining whether the condition is satisfied. Mask is performed from the low-order bit in [Address value:]. To implement the mask, specify each digit to be masked at input as an asterisk (*). Note that masking cannot be performed when specifying an address range, and that arbitrary bit positions cannot be masked.
  For example, if an address condition is to be satisfied when A16 to A31 bits are 0, specify the mask as H'0000****.

Notes

(1) The BREAK CONDITION2 function is invalid if it is executed by pressing the <CONTINUE>, <STEP>, <STEP_OVER> or <STEP_UP> button according to the E7000PC specifications. The BREAK CONDITION2 function must be executed by the [Go...] command.

(2) During address specification, all address bits are checked. Accordingly, if the break condition is specified in the cache area, no break occurs even if the cache through area in the same memory space is accessed. To set break conditions in both cache area and cache through area, use mask specification.

(3) The BREAK CONDITION2 is invalid at the instruction where the [Break...] command or the <SET> button is specified according to the E7000PC specifications.

(4) In a slot instruction of a delayed branch instruction, <Before> cannot be specified in the [PC break position] group box.

(5) The BREAK CONDITION2 function is achieved by the SH7604 user break controller (UBC) according to the E7000PC specifications. Accordingly, the user cannot use the UBC for other purposes.

Related Function

[Go...] command

5.4.6 Specifying Hardware Break Condition3,4 (Ctrl + 3) (Ctrl + 4)

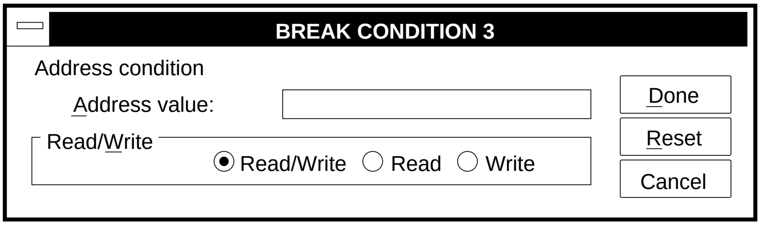
[Break Condition3...] BREAK CONDITION3 dialog box
[Break Condition4...] BREAK CONDITION4 dialog box

Overview

Specifies a hardware break condition (BREAK CONDITION3). When all of the specified conditions are satisfied during emulation execution with the [GO...] command, program execution terminates.

BREAK CONDITION4 is specified in the same way.

Window



BREAK CONDITION 3

Address condition

Address value:

Read/Write

● Read/Write ○ Read ○ Write

Figure 5-32 BREAK CONDITION3 Dialog Box

Operation

- [Address condition]

  — [Address value:] text box
  Specifies an address bus value where a hardware break is to occur.

- [Read/Write] group box
  Specifies a read or write cycle condition as a hardware break condition as follows:
  <Read/Write>: Specifies both read and write cycles as hardware break conditions (default at system initiation)
  <Read>: Specifies a read cycle as a hardware break condition
  <Write>: Specifies a write cycle as a hardware break condition

- <Done> button
  Clicking the <Done> button stores the BREAK CONDITION3 dialog box settings, closes the BREAK CONDITION3 dialog box, and sends the above settings to the E7000PC emulator.

- <Reset> button
  Clicking the <Reset> button cancels the settings in the BREAK CONDITION3 dialog box and the E7000PC emulator, returns the system to a state wherein no hardware break conditions are specified, and closes the BREAK CONDITION3 dialog box.

- <Cancel> button
  Clicking the <Cancel> button closes the BREAK CONDITION3 dialog box and returns the system to its state before the BREAK CONDITION3 dialog box was opened without storing the above settings.

Function

Address ([Address condition]) and read/write ([Read/Write]) conditions can be specified as hardware break conditions.

Note

During address specification, all address bits are checked. Accordingly, if the break condition is specified in the cache area, no break occurs even if the cache through area in the same memory space is accessed.

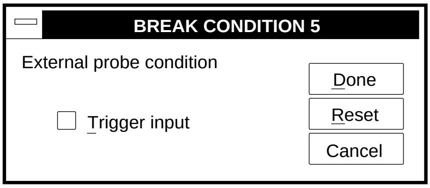
5.4.7 Specifying Hardware Break Condition5 (Ctrl + 5)

[Break Condition5...] BREAK CONDITION5 dialog box

Overview

Specifies a hardware break condition (BREAK CONDITION5). When a trigger signal is input from the probe pin during emulation execution with the [GO...] command, the condition is satisfied and program execution terminates.

Window



BREAK CONDITION 5

External probe condition

□ Trigger input

Figure 5-33 BREAK CONDITION5 Dialog Box

Operation

- [External probe condition]
  [Trigger input] check box
  Specifies a trigger signal input from the probe pin as a hardware break condition, as follows:
  <No check>: The trigger input is not specified as a hardware break condition (default at system initiation)
  <Check>: The trigger input is specified as a hardware break condition

- <Done> button
  Clicking the <Done> button stores the BREAK CONDITION5 dialog box settings, closes the BREAK CONDITION5 dialog box, and sends the above settings to the E7000PC emulator.

- <Reset> button
  Clicking the <Reset> button cancels the settings in the BREAK CONDITION5 dialog box and the E7000PC emulator, returns the system to a state wherein no hardware break conditions are specified, and closes the BREAK CONDITION5 dialog box.

- <Cancel> button
  Clicking the <Cancel> button closes the BREAK CONDITION5 dialog box and returns the system to its state before the BREAK CONDITION5 dialog box was opened without storing the above settings.

Function

Probe input conditions can be specified as hardware break conditions.

5.5 Trace Function

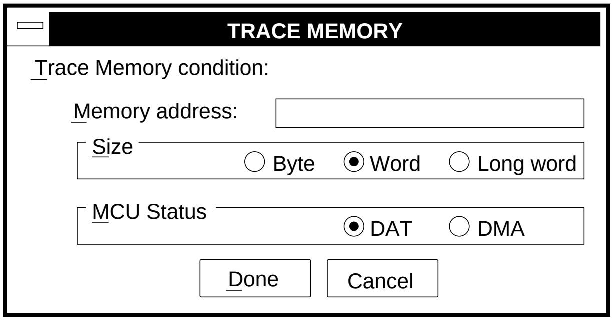
5.5.1 Specifying Memory Conditions for Acquiring Trace Information (Alt + Ctrl + T)

[Trace memory...] TRACE MEMORY dialog box

Overview

Specifies memory conditions for acquiring trace information.

Window



TRACE MEMORY

Trace Memory condition:

Memory address:

Size ○ Byte ● Word ○ Long word

MCU Status ● DAT ○ DMA

Figure 5-34 TRACE MEMORY Dialog Box

Operation

- [Trace Memory condition]

— [Memory address:] text box
Specifies a memory address whose contents are to be traced.
Memory contents during emulation execution can be displayed in the trace information display by the [Trace display - Text] command.
The memory contents of the specified address can be displayed in the emulation information area of the base window during emulation execution, depending on the [Display memory trace data] specification of the [Go...] command.

— [Size] group box
Specifies the size of the memory data to be traced as follows:

<Byte>:	1-byte data
<Word>:	2-byte data (default at system initiation)
<Long word>:	4-byte data

— [MCU status] group box
  Specifies an instruction execution cycle or a DMA cycle as a trace acquisition cycle.

  <DAT>: Trace acquisition is performed during instruction execution cycles (default at system initiation)

  <DMA>: Trace acquisition is performed during DMA cycles

- <Done> button
  Clicking the <Done> button stores the TRACE MEMORY dialog box settings, closes the TRACE MEMORY dialog box, and sends the above settings to the E7000PC emulator.

- <Cancel> button
  Clicking the <Cancel> button closes the TRACE MEMORY dialog box and returns the system to its state before the TRACE MEMORY dialog box was opened without storing the above settings. In addition, these settings are not specified in the E7000PC emulator.

Function

Sets memory conditions whose contents are to be traced. If trace information display in bus cycle units is specified by the [Trace display -Text] command, the traced memory contents are displayed in the TM column during emulation.

Memory contents being traced can be displayed in real time in the emulation information area by selecting <Enable> in the [Display memory trace data] group box in the [Go...] command.

Notes

(1) This command cannot be set during emulation.

(2) For the memory to be modified within the CPU such as timer counters of the internal I/O modules, the trace memory contents will not be modified until it is accessed by an instruction or the DMA.

(3) The trace memory contents are modified when the specified address matches the data on the address bus. Note, however, that the trace memory contsnts will not be modified if the specified address does not match the A1 bit of the address bus as shown below.

Example: If address H'F000002 (32-bit bus area) is specified while address H'F000000 is accessed in long-word units by the program.

(4) The trace memory contents will not be modified during memory access by a PC relative instruction such as MOV.W @(10, PC), R0.

Related Functions

[Go...] command

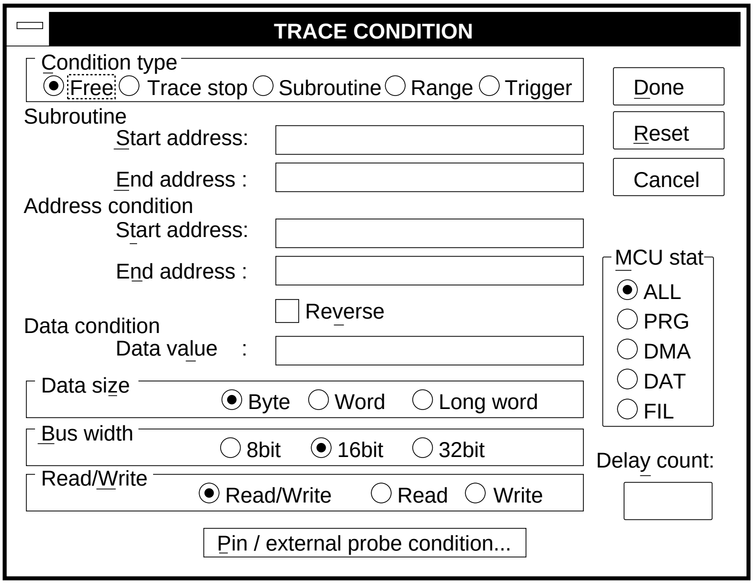
5.5.2 Specifying Trace Acquisition Conditions (Ctrl + T)

[Trace condition...] TRACE CONDITION dialog box

Overview

Specifies trace acquisition conditions. Trace information is acquired during emulation execution according to specifications.

Window



TRACE CONDITION

Condition type
● Free ○ Trace stop ○ Subroutine ○ Range ○ Trigger

Subroutine
Start address:
End address :

Address condition
Start address:
End address :

□ Reverse

Data condition
Data value :

Data size
● Byte ○ Word ○ Long word

Bus width
○ 8bit ● 16bit ○ 32bit

Read/Write
● Read/Write ○ Read ○ Write

MCU stat
● ALL
○ PRG
○ DMA
○ DAT
○ FIL

Delay count:

Figure 5-35 TRACE CONDITION Dialog Box

Operation

- [Condition type] group box
  Specifies how to acquire trace information during emulation.
  - <Free>: Acquires trace information during all bus cycles. Trace acquisition conditions are ignored. (Default at system initiation)
  - <Trace stop>: Acquires trace information during all bus cycles until the specified trace stop condition is satisfied; program execution, however, continues. A trace stop condition is determined by ANDing the following: [Address condition], [Data condition], [Read/Write], [MCU stat], [Delay count:], [Pin condition], and [External probe condition].
  - <Subroutine>: Acquires trace information for instructions and operand accesses in the range (subroutine) specified by a start address and an end address. A trace acquisition condition is specified by [Subroutine].

<Range>: Acquires trace information during a bus cycle corresponding with the trace acquisition condition. A trace acquisition condition is determined by ANDing [Address condition] and [Read/Write].

<Trigger>: Outputs a low-level pulse from the trigger output pin of the emulator pod during a bus cycle corresponding with the trace acquisition condition. Trace information is acquired during all bus cycles. A trace acquisition condition is determined by ANDing the following: [Address condition], [Data condition], [Read/Write], [MCU stat], [Delay:], [Pin condition], and [External probe condition].

- [Subroutine]
  Specifies an address range determining the subroutine in which trace information is to be acquired, for selecting <Subroutine> in [Condition type].

  — [Start address:] text box
  Specifies a number or symbol as the start address of the subroutine in which trace information is to be acquired.

  — [End address:] text box
  Specifies a number or symbol as the end address of the subroutine in which trace information is to be acquired.
  If the source program is written in C, the end address not to be specified because the end address of the function is automatically set.

- [Address condition]
  Specifies an address bus condition as a trace acquisition condition, for selecting <Trace stop>, <Range>, or <Trigger> in [Condition type]. An address bus value can be masked.

  — [Start address:] text box
  Specifies a number or a symbol as an address bus value for trace information acquisition. For acquiring trace information in an address range, specify the start address.

  — [End address:] text box
  Specifies a number or symbol as the end address of the range in which trace information is to be acquired.

  — [Reverse] check box
  Specifies whether the address bus condition is within the address range determined by [Start address:] and [End address:], or is outside this range, as follows:

  <No check>: Addresses in the address range are trace acquisition conditions (default at system initiation)

  <Check>: Addresses outside the address range are trace acquisition conditions

- [Data condition]
  Specifies a data bus condition as a trace acquisition condition, for selecting <Trace stop> or <Trigger> in [Condition type].

  — [Data value:] text box
  Specifies a number or a symbol as a data bus value for trace information acquisition. A mask can be specified for the data bus value.

  — [Data size] group box
  Specifies the size of the data bus value set in [Data value:] as follows:

<Byte>:	1-byte data (default at system initiation)
<Word>:	2-byte data
<Long word>:	4-byte data

  — [Bus width] group box
  Specifies the size of the memory bus accessed by the SH7604 MCU as shown below. When <Long word> is specified in [Data size], the bus width automatically becomes 32 bits.

<8bit>:	8-bit bus
<16bit>:	16-bit bus (default at system initiation)
<32bit>:	32-bit bus

- [Read/Write] group box
  Specifies a read or write cycle condition as a trace acquisition condition, for selecting <Trace stop>, <Range>, or <Trigger> in [Condition type] as follows:

<Read/Write>:	Specifies both read and write cycles as trace acquisition conditions (default at system initiation)
<Read>:	Specifies a read cycle as a trace acquisition condition
<Write>:	Specifies a write cycle as a trace acquisition condition

- [MCU stat] group box
  Specifies an SH7604-bus cycle condition as a trace acquisition condition, for selecting <Trace stop> or <Trigger> in [Condition type] as follows:

<ALL>:	Specifies all bus cycles as bus cycle conditions (default at system initiation)
<PRG>:	Specifies a program fetch cycle as a bus cycle condition
<DMA>:	Specifies a DMA cycle as a bus cycle condition
<DAT>:	Specifies an execution cycle as a bus cycle condition
<FIL>:	Specifies a cache fill cycles as a bus cycle condition

- [Delay count:] text box
  For selecting <Trace stop> or <Trigger> in [Condition type], specifies a delay count starting from the time a trace acquisition condition determined by ANDing [Address condition], [Data condition], [Read/Write], [MCU stat], [Pin condition], and [External probe condition] is satisfied. A delay count ranging from H'1 to H'7FFF can be specified. If omitted, 0 is assumed.

- <Done> button
  Clicking the <Done> button stores the TRACE CONDITION and PIN/EXTERNAL PROBE CONDITION dialog box settings, closes the TRACE CONDITION dialog box, and sends the

above settings to the E7000PC emulator.

- <Reset> button
  Clicking the <Reset> button cancels the settings in the TRACE CONDITION and PIN/EXTERNAL PROBE CONDITION dialog boxes, and the E7000PC emulator settings, returns the system to a state wherein no trace acquisition conditions are specified, and closes the TRACE CONDITION dialog box.

- <Cancel> button (in TRACE CONDITION dialog box)
  Clicking the <Cancel> button closes the TRACE CONDITION dialog box and returns the system to its state when the TRACE CONDITION dialog box was opened without storing the settings in the TRACE CONDITION and PIN/EXTERNAL PROBE CONDITION dialog boxes.

- <Pin/external probe condition...> button
  Clicking the <Pin/external probe condition...> button opens the dialog box in which SH7604-control signal states and external probe (in the emulator pod) conditions are specified as trace acquisition conditions.
  The specifications are valid only when <Trace stop> or <Trigger> is set in [Condition type].
  The dialog box contents are shown in figure 5-36.

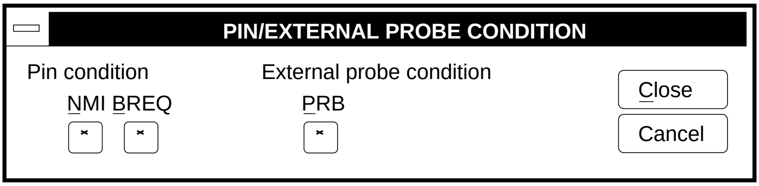


Figure 5-36 PIN/EXTERNAL PROBE CONDITION Dialog Box

- [Pin condition]
  Specifies an SH7604-control signal state as a trace acquisition condition. A condition is satisfied when the specified value and the signal are the same.
  Values can be specified for each tri-state button by clicking on the button desired.

  — [BREQ condition] state button
  <*>: Specifies no condition (default at system initiation)
  <L>: Specifies BREQ signal low level as a trace acquisition condition

  — [NMI condition] tri-state button
  <*>: Specifies no condition (default at system initiation)
  <H>: Specifies NMI signal high level as a trace acquisition condition
  <L>: Specifies NMI signal low level as a trace acquisition condition

- [External probe condition]
  Specifies external probe (in the emulator pod) conditions as a trace acquisition condition. A condition is satisfied when the specified value and the external probe signal are the same. Values can be specified for each tri-state button by clicking on the button desired.

  [PRB] tri-state button

<*>:	Specifies no condition (default at system initiation)
<H>:	Specifies external probe signal high level as a trace acquisition condition
<L>:	Specifies external probe signal low level as a trace acquisition condition

- <Close> button (in PIN/EXTERNAL PROBE CONDITION dialog box)
  Clicking the <Close> button closes the PIN/EXTERNAL PROBE CONDITION dialog box. In this case, the settings in the PIN/EXTERNAL PROBE CONDITION dialog box is stored.

- <Cancel> button
  Clicking the <Cancel> button closes the PIN/EXTERNAL PROBE CONDITION dialog box and returns the system to its state when the PIN/EXTERNAL PROBE CONDITION dialog box was opened without storing the settings in the PIN/EXTERNAL PROBE CONDITION dialog box.

Function

- Trace acquisition condition setting
  Sets the conditions for acquiring trace information during program execution initiated by the [Go...] command or the <CONTINUE> button.
  The trace acquisition conditions depend on the trace modes listed in table 5-6.

Table 5-6 Trace Modes

Trace Mode	Description
<Free> Free trace	Acquires trace information during all bus cycles with no specified conditions.
<Trace stop> Trace stop	Stops trace acquisition when the specified condition is satisfied, enters command wait state in parallel mode, and displays a prompt (#) in the command area.
<Subroutine> Subroutine trace	Acquires trace information for instructions and operand accesses in the range (subroutine) determined by a start address and an end address. If the specified subroutine calls other subroutine, does not acquire trace information during the called subroutine.
<Range> Range trace	Acquires trace information during a bus cycle corresponding with the specified address range and condition.
<Trigger> Trigger	Outputs a low-level pulse from the trigger output pin in the emulator pod during a bus cycle corresponding with the specified condition. Acquires trace information during all bus cycles.

- Specifying trace acquisition condition according to memory bus size
  Trace acquisition conditions for <Trace stop> and <Trigger> are specified according to the size of the memory bus accessed by the SH7604. Table 5-7 shows the trace acquisition conditions for each bus size.

Table 5-7 Trace Acquisition Condition Settings According to Bus Size

Bus Size	Access Size	Trace Acquisition Condition Setting Method
32-bit bus area	Long access	Long data is accessed in one bus cycle. [Address condition] must be a multiple of four and [Data size] must be <Long word> (long data). Other specifications are invalid.
	Word access	Word data is accessed in one bus cycle. [Address condition] must be a multiple of two, [Data size] must be <Word> (word data), and [Bus width] must be <32 bit>. Other specifications are invalid.
	Byte access	Byte data is accessed in one bus cycle. [Data size] must be <Byte> (byte data) and [Bus width] must be <32 bit>. Other specifications are invalid. [Start address:] can be both even and odd addresses.
16-bit bus area	Long access	Long data is accessed in two bus cycles in word units. [Address condition] must be a multiple of two and [Data size] must be <Word>. Other specifications are invalid.
	Word access	Word data is accessed in one bus cycle. [Address condition] must be a multiple of two, [Data size] must be <Word>, and [Bus width] must be <16 bit>. Other specifications are invalid.
	Byte access	Byte data is accessed in one bus cycle. [Data size] must be <Byte> and [Bus width] must be <16 bit>. Other specifications are invalid. [Start address:] can be both even and odd addresses.
8-bit bus area	Long, word, or byte access	All addresses are accessed in byte units. Long data, word data, and byte data are accessed in four bus cycles, two bus cycles, and one bus cycle, respectively. [Start address:] can be both even and odd addresses, [Data size] must be <Byte>, and [Bus width] must be <8 bit>. Other specifications are invalid.

- Mask specifications
  A mask in 1-bit units can be specified for the settings in [Address condition], [Data condition], [Pin condition], and [External probe condition]. The meaning of a masked bit is that the corresponding bit value is arbitrary when determining whether the condition is satisfied.

  — The [Address condition] setting can be masked when <Trace stop> or <Trigger> is specified in [Condition type]. To implement the mask, specify each digit in [Start address:] to be masked at input as an asterisk (*).

    When <Range> is specified in [Condition type] and [End address:] specification is omitted, mask is performed from the low-order bit. For example, if an address range condition is to be satisfied when the A16 to A31 bits are 0s, specify the mask as H'0000****.

— The [Data condition] setting can be masked when <Trace stop> or <Trigger> is specified in [Condition type]. To implement a mask, specify each digit of the value specified in [Data value:] to be masked at input as an asterisk (*).
For example, if a byte data condition is to be satisfied when the D0 bit is 0, specify the mask as B'*******0.

— To implement a mask for [Pin condition] or [External probe condition], click the tri-state button and set it to <*>.

Notes

(1) If an address condition is specified in the cache area, no break occurs even if the cache through area in the same memory space is accessed. To set break conditions in both cache area and cache through area, use mask specification.

(2) If the parallel mode is entered in the command area during emulation, trace information is acquired in the following condition.

(a) If the parallel mode is entered by pressing the (Enter) key or by the satisfaction of trace information acquisition condition (<trace stop>), trace information will not be acquired in the parallel mode. When the E7000PC exits parallel mode, the acquired trace information is cleared and trace acquisition restarts under the conditions specified with this command.

(b) If the parallel mode is entered by pressing the (Space) key, trace information is acquired in the parallel mode.

(3) To specify memory access condition in a PC relative instruction such as MOV.W @(10, PC), R0, <PRG> must be selected in the [MCU stat] group box.

Related Function

[Trace display - Setting...] command

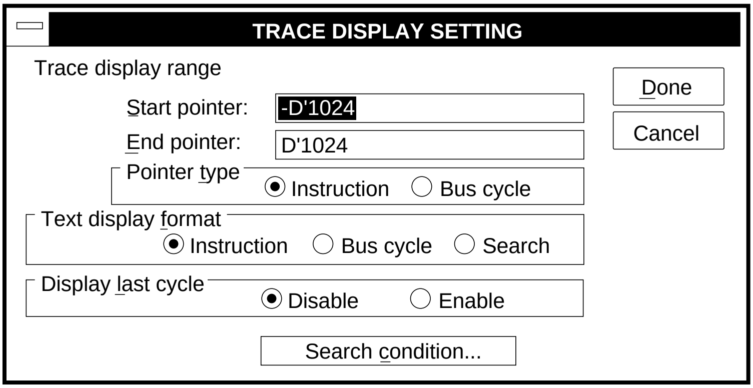
5.5.3 Specifying Trace Information Display Conditions (Alt + Ctrl + X)

[Trace display - Setting...] TRACE DISPLAY SETTING dialog box

Overview

Specifies trace information display conditions. Trace information displayed by the [Trace display - Text] or [Trace display - Graph] command follows these command specifications.

Window



TRACE DISPLAY SETTING

Trace display range

Start pointer: -D'1024

End pointer: D'1024

Pointer type ● Instruction ○ Bus cycle

Text display format ● Instruction ○ Bus cycle ○ Search

Display last cycle ● Disable ○ Enable

Search condition...

Done

Cancel

Figure 5-37 TRACE DISPLAY SETTING Dialog Box

Operation

- [Trace display range]

  — [Start pointer:] text box
  Specifies a numerical value as the start pointer for the range for which trace information is shown in text form or graph form. When omitted, the value
  –D'65535 is assumed; the initial value is –D'1024.

  — [End pointer:] text box
  Specifies a numerical value as the end pointer for the range for which trace information is shown in text display or is graphically drawn. When omitted, the value D'65535 is assumed; the initial value is D'1024.

  — [Pointer type] group box
  Specifies the pointer format in which trace information is displayed in the TRACE DISPLAY TEXT window as follows:

<Instruction>: Displays trace information in instruction pointer format
(default at system initiation)
<Bus cycle>: Displays trace information in bus cycle pointer format

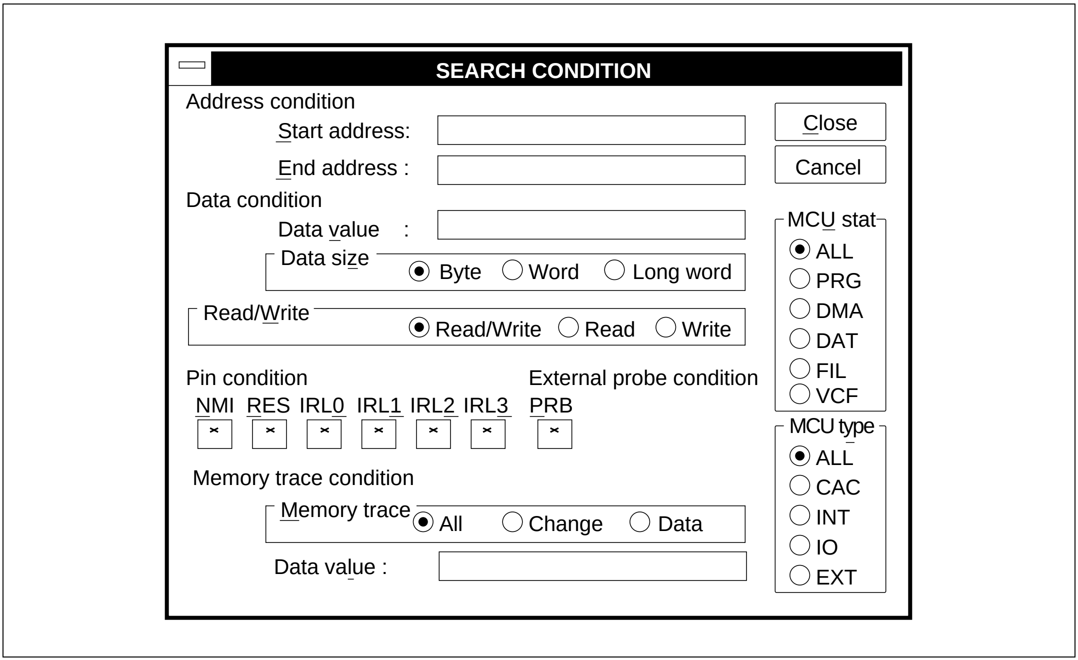
- [Text display format] group box
  Specifies the format in which trace information is displayed in the TRACE DISPLAY TEXT window. When <Search> is specified <Bus cycle> in [Trace display range], <Bus cycle> is automatically selected in [Pointer type].
  <Instruction>: Displays trace information in instruction mnemonic units
  (default at system initiation)
  <Bus cycle>: Displays trace information in bus cycle units
  <Search>: Searches for and displays trace information in bus cycle units

- [Display last cycle] group box
  Specifies whether trace information is to be displayed for only the last bus cycle or for all the bus cycles, when <Search> is selected in [Text display format], as follows:
  <Disable>: Displays trace information for the whole range set in [Trace display range]
  <Enable>: Displays trace information only during the last bus cycle

- <Search condition...> button
  Clicking the <Search condition...> button opens the SEARCH CONDITION dialog box in which trace information search conditions are set. These dialog box settings are valid when <Search> is specified in [Trace display format]. The SEARCH CONDITION dialog box is shown in figure 5-38.

- <Done> button
  Clicking the <Done> button stores the TRACE DISPLAY SETTING and SEARCH CONDITION dialog box settings before closing the TRACE DISPLAY SETTING dialog box.

- <Cancel> button (in TRACE DISPLAY SETTING dialog box)
  Clicking the <Cancel> button closes the TRACE DISPLAY SETTING dialog box and returns the system to its state when the TRACE DISPLAY SETTING dialog box was opened without storing the settings in the TRACE DISPLAY SETTING and SEARCH CONDITION dialog boxes.



Address condition

Start address:

End address :

Data condition

Data value :

Data size ● Byte ○ Word ○ Long word

Read/Write ● Read/Write ○ Read ○ Write

Pin condition

NMI ☒ RES ☒ IRL0 ☒ IRL1 ☒ IRL2 ☒ IRL3 ☒

External probe condition

PRB ☒

Memory trace condition

Memory trace ● All ○ Change ○ Data

Data value :

MCU stat ● ALL ○ PRG ○ DMA ○ DAT ○ FIL ○ VCF

MCU type ● ALL ○ CAC ○ INT ○ IO ○ EXT

Figure 5-38 SEARCH CONDITION Dialog Box

- [Address condition]
  Specifies an address bus condition as a trace search condition.

  — [Start address:] text box
  Specifies a number or a symbol as an address bus value for a trace search condition. For searching trace information in an address range, specify the start address. A mask can be specified for the address bus value.

  — [End address:] text box
  Specifies a number or symbol as the end address of the range in which trace information is to be searched.

- [Data condition]
  Specifies a data bus condition as a trace search condition.

  — [Data value:] text box
  Specifies a number or a symbol as a data bus value for a trace information search. A mask can be specified for the data bus value.

  — [Data size] group box
  Specifies the size of the data bus value set in [Data value:] as follows:

<Byte>:	1-byte data (default at system initiation)
<Word>:	2-byte data
<Long word>:	4-byte data

- [Read/Write] group box
  Specifies a read or write cycle condition as a trace search condition as follows:
  <Read/Write>: Specifies both read and write cycles as trace search conditions (default at system initiation)
  <Read>: Specifies a read cycle as a trace search condition
  <Write>: Specifies a write cycle as a trace search condition

- [MCU stat] group box
  Specifies an SH7604-bus cycle condition as a trace search condition as follows:
  <ALL>: Specifies all bus cycles as trace search conditions (default at system initiation)
  <PRG>: Specifies a program fetch cycle as a trace search condition
  <DMA>: Specifies a DMA cycle as a trace search condition
  <DAT>: Specifies an execution cycle as a trace search condition
  <FIL>: Specifies a cache fill cycle as a trace search condition
  <VFC>: Specifies a vector fetch cycle as a trace search condition

- [Mem type] group box
  Specifies an SH7604-memory type condition as a trace search condition as follows:
  <ALL>: Specifies all memory types as trace search conditions (default at system initiation)
  <CAC>: Specifies a cache hit as a trace search condition
  <INT>: Specifies an internal area access as a trace search condition
  <IO>: Specifies an internal I/O area access as a trace search condition
  <EXT>: Specifies an external area (CS0 to CS3) access as a trace search condition

- [Pin condition]
  Specifies an SH7604-control signal state as a trace search condition. When the specified value and the signal are the same, the condition is satisfied.
  Values can be specified for each tri-state button by clicking on the button desired.

  — [NMI] tri-state button
  <*>: Specifies no condition (default at system initiation)
  <H>: Specifies NMI signal high level as a trace search condition
  <L>: Specifies NMI signal low level as a trace search condition

  — [RES] state button
  <*>: Specifies no condition (default at system initiation)
  <L>: Specifies RES signal low level as a trace search condition

  — [IRL0], [IRL1], [IRL2], [IRL3] tri-state buttons
  <*>: Specifies no condition (default at system initiation)
  <H>: Specifies IRL signal high level as a trace search condition
  <L>: Specifies IRL signal low level as a trace search condition

- [External probe condition]
  Specifies external probe (in the emulator pod) conditions as trace search conditions. When the specified value and the external probe signal are the same, the condition is satisfied. Values can be specified for each tri-state button by clicking on the button desired.

  [PRB] tri-state button
  <*>: Specifies no condition (default at system initiation)
  <H>: Specifies external probe signal high level as a trace search condition
  <L>: Specifies external probe signal low level as a trace search condition

- [Memory trace condition]
  Specifies a trace search condition with regards to traced memory set by [Trace Memory condition] of the [Trace memory...] command.

  — [Memory trace] group box
  <All>: Searches for all trace information for a bus cycle where specified memory address is accessed (default at system initiation)
  <Change>: Searches for a bus cycle where traced memory has been modified
  <Data>: Searches for a bus cycle where data specified by [Data value:] is accessed

  — [Data value:] text box
  Specifies data for a trace search condition when <Data> is specified in the [Memory trace] group box. Immediately after system initiation, 0 is specified.

- <Close> button
  Clicking the <Close> button stores the above settings and closes the SEARCH CONDITION dialog box.

- <Cancel> button (SEARCH CONDITION dialog box)
  Clicking the <Cancel> button closes the SEARCH CONDITION dialog box and returns the system to its state when the SEARCH CONDITION dialog box was opened without storing the settings in the SEARCH CONDITION dialog boxs.

Function

- Trace display condition setting
  Sets the conditions for displaying trace information. The [Trace display - Text] or [Trace display - Graph] command displays trace information according to these command settings.

- Trace search condition specifications according to memory bus size
  Trace search conditions in SEARCH CONDITION dialog box are specified according to the size of the memory bus accessed by the SH7604. Table 5-8 shows the trace search conditions for each bus size.

Table 5-8 Trace Search Condition Settings According to Bus Size

Bus Size	Access Size	Trace Search Condition Setting Method
32-bit bus area	Long access	Long data is accessed in one bus cycle. [Address condition] must be a multiple of four and [Data size] must be <Long word> (long data). Other specifications are invalid.
	Word access	Word data is accessed in one bus cycle. [Address condition] must be a multiple of two and [Data size] must be <Word> (word data). Other specifications are invalid.
	Byte access	Byte data is accessed in one bus cycle. [Data size] must be <Byte> (byte data). Other specifications are invalid. [Address condition] can be both even and odd addresses.
16-bit bus area	Long access	Long data is accessed in two bus cycles in word units. [Address condition] must be a multiple of two and [Data size] must be <Word>. Other specifications are invalid.
	Word access	Word data is accessed in one bus cycle. [Address condition] must be a multiple of two and [Data size] must be <Word>. Other specifications are invalid.
	Byte access	Byte data is accessed in one bus cycle. [Data size] must be <Byte>. Other specifications are invalid. [Address condition] can be both even and odd addresses.
8-bit bus area	Long, word, or byte access	All addresses are accessed in byte units. Long data, word data, and byte data are accessed in four bus cycles, two bus cycles, and one bus cycle, respectively. [Address condition] can be both even and odd addresses and [Data size] must be <Byte>. Other specifications are invalid.

- Mask specifications
  A mask in 1-bit units can be specified for the [Data condition], [Pin condition], and [External probe condition] settings in the SEARCH CONDITION dialog box. The meaning of a masked bit is that the corresponding bit value is arbitrary when determining whether the condition is satisfied. Note, however, that a mask cannot be specified for the [Address condition] setting when an address range ([End address:]) is specified.

  — The [Data condition] setting can be masked. To implement a mask, specify each digit of the value specified in [Data value:] to be masked at input as *.
  For example, if a byte data condition is to be satisfied when the D0 bit is 0, specify the mask as B'*******0.

  — To implement a mask for [Pin condition] or [External probe condition], click the tri-state button and set it to <*>.

Notes

(1) To specify memory access condition in a PC relative instruction such as MOV.W @(10, PC), R0, <PRG> must be selected in the [MCU stat] group box.

(2) Conditions specified in the [Data condition] are invalid if vector fetch cycle <VCF> is selected in the [MCU stat] group box.

Related Functions

[Trace memory...], [Trace display - Text], and [Trace display - Graph] commands

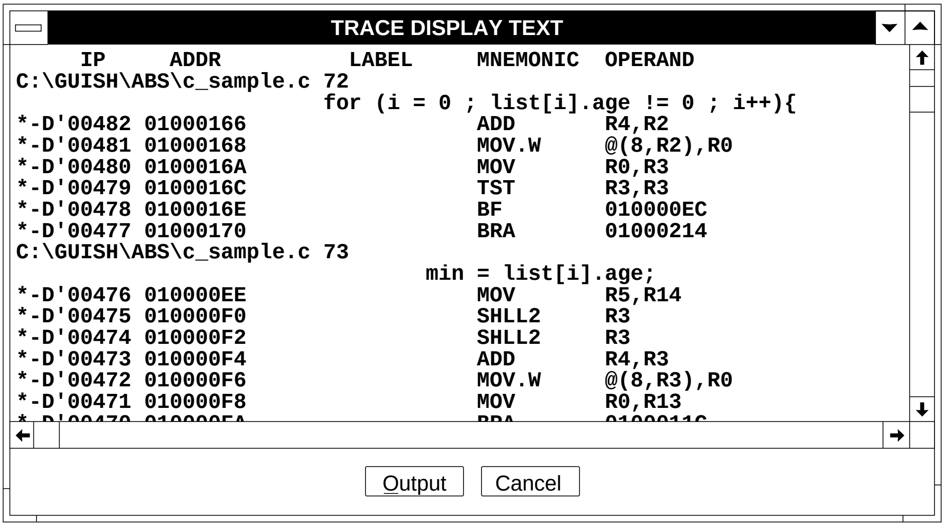
5.5.4 Displaying Trace Information in Text (Alt + X)

[Trace display - Text] TRACE DISPLAY TEXT window

Overview

Displays trace information in text according to the specifications in the [Trace display - Setting...] command. The displayed contents can be output to a file.

Window



```
     IP     ADDR          LABEL     MNEMONIC  OPERAND
C:\GUISH\ABS\c_sample.c 72
                        for (i = 0 ; list[i].age != 0 ; i++){
*-D'00482 01000166                  ADD       R4,R2
*-D'00481 01000168                  MOV.W     @(8,R2),R0
*-D'00480 0100016A                  MOV       R0,R3
*-D'00479 0100016C                  TST       R3,R3
*-D'00478 0100016E                  BF        010000EC
*-D'00477 01000170                  BRA       01000214
C:\GUISH\ABS\c_sample.c 73
                                min = list[i].age;
*-D'00476 010000EE                  MOV       R5,R14
*-D'00475 010000F0                  SHLL2     R3
*-D'00474 010000F2                  SHLL2     R3
*-D'00473 010000F4                  ADD       R4,R3
*-D'00472 010000F6                  MOV.W     @(8,R3),R0
*-D'00471 010000F8                  MOV       R0,R13
```

Figure 5-39 TRACE DISPLAY TEXT Window

Operation

- Display header
  Indicates the trace information display format with a header.

- Trace display area
  In the trace display area, trace information acquired during emulation is displayed. The contents of the source file of the corresponding program is included in the displayed trace information.

- <Output> button
  Clicking the <Output> button opens the TRACE OUTPUT dialog box. The contents currently displayed in the trace display area can be output to a file.

- <Cancel> button
  Clicking the <Cancel> button closes the TRACE DISPLAY TEXT window.

Description

- Display header format

  The display header format depends on the [Text display format] group box specification in the [Trace display - Setting...] command.

  — Trace information is displayed in instruction mnemonic units (<Instruction> must be specified in [Text display format] group box)

IP	ADDR	LABEL	MNEMONIC	OPERAND
*[-]D'xxxxx	00000000	!llll – llll	mmmm – mmmm	oooo – oooo
(a)	(b)	(c)	(d)	(e)

Figure 5-40 Display Header Format in Instruction Mnemonic Units (TRACE DISPLAY TEXT Window)

(a) Instruction pointer (IP)

Relative instruction location (instruction pointer) based on the instruction where a delay condition is satisfied. Although the instruction pointer usually has a negative value (*–D'xxxxx), it will be positive if it is acquired during delay cycles while the delay condition is specified.

Note: The point where a delay condition is satisfied means the starting point of a delay bus cycle which has been specified with the [Break condition1...] command or [Delay count:] in the [Trace condition...] command. When zero is specified in [Delay count:] or when program execution terminates due to a different cause, the latest trace information will be shown from D'00000.

(b) Instruction address (ADDR)

(c) Label name (LABEL)

(d) Instruction mnemonic (MNEMONIC)

(e) Instruction operand (OPERAND)

— Trace information is displayed in bus-cycle units (<Bus cycle> or <Search> must be specified in [Text display format])

```
    BP          AB         DB      MA  R/W  ST   IRL   NMI  RES  BRQ  PRB  VCC  CLK     TM
[-]D'xxxxx   xxxxxxxx   xxxxxxxx   xxx   x   xxx  xxxx   x    x    x    x    x    xx     **
   (a)         (b)        (c)      (d)  (e)  (f)  (g)   (h)  (i)  (j)  (k)  (l)  (m)    (n)

------------------------------------------------------------------------------------------
                                                     TOTAL CLOCK NUMBER = xxxxx
                                                                           (o)
```

Figure 5-41 Display Header Format in Bus-Cycle Units (TRACE DISPLAY TEXT Window)

(a) Bus cycle pointer (BP)
Relative bus cycle location (bus cycle pointer) based on the bus cycle where a delay condition is satisfied. In bus cycles which prefetch instructions, the instruction mnemonics and instruction addresses are displayed together. When two instructions are executed in one bus cycle, both mnemonics are displayed along with the address of the first instruction. The bus cycle pointer is normally negative (–D'xxxxx). However, it will be positive if it is acquired during delay cycles while the delay condition is specified.

(b) Address bus value (AB)

(c) Data bus value (DB)
Long-word, word, and byte values are displayed depending on the length of the data accessed by the MCU. When no data is on the data bus, ** is displayed.

(d) Memory area type (MA)
Accessed memory areas are displayed in one of the following types.

CAC:	Cache-hit access
INT:	Internal area access
IO:	Internal I/O area access
EXT:	External area (CS0 to CS1)access

(e) Read/Write signal type (R/W)

R:	Read cycle
W:	Write cycle

(f) MCU status (ST)

PRG:	Program fetch cycle
DAT:	Memory or I/O access cycle
FIL:	Cache-fill cycle
DMA:	DMA cycle
VCF:	Vector fetch cycle

(g) IRL0-IRL3 signal levels (IRL)

The signal levels of IRL0 to IRL3 are displayed as follows.

IRL
x3 x2 x1 x0 (n = 3 - 0; IRL3 - IRL0 level)
xn = 1: High level
= 0: Low level

(h) NMI signal level (NMI)

NMI = 1: High level
= 0: Low level

(i) RES signal level (RES)

RES = 1: High level
= 0: Low level

(j) BREQ signal level (BRQ)

BRQ = 1: High level
= 0: Low level

(k) External probe signal levels (PRB)

The signal levels of the external probes are displayed as follows.

PRB = 1: High level
= 0: Low level

(l) V_{CC} voltage (VCC)

Vcc = 1: V_{CC}voltage 4 V or more
= 0: V_{CC}voltage is less than 4 V; the MCU is not operating correctly

(m) Clock cycle (CLK)

The number of clock cycles required from the end of the previous bus cycle to the end of this bus cycle. If more than 128 clocks are counted, ** is displayed. Consider the trace acquisition timing of the E7000PC emulator station.

(n) Traced memory contents (TM)

Displays the address specified with the [Trace memory...] command and its traced contents.

(o) Total clock cycle (TOTAL CLOCK NUMBER)

The total number of clock cycles described in item (m). If the number 128 is exceeded in even one display line, ***** is displayed. The total clock cycle is displayed only when <Bus cycle> is selected in [Trace display format] of [Trace display -Setting].

- Source file display
  Displays in the source display area the source file contents corresponding to the address bus value in trace information. The display format depends on which language the corresponding program has been written in.

  — C language program

```
  <source file name> <line number>
  <C source file contents>
```

  — Assembly program

```
  <source file name >
```

- Trace information output
  The contents currently displayed in the trace display area can be output to a file. Clicking the <Output> button opens the TRACE OUTPUT dialog box shown in figure 5-42.

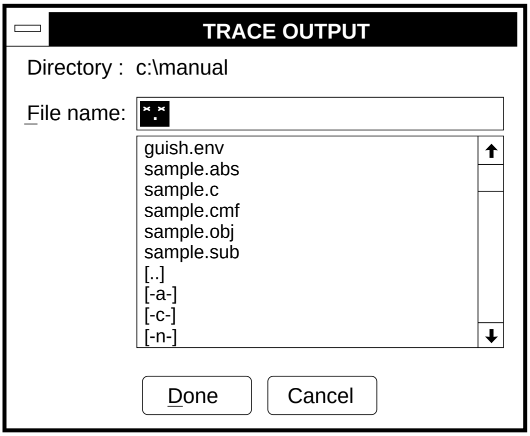

TRACE OUTPUT
Directory : c:\manual
File name: *.*
guish.env
sample.abs
sample.c
sample.cmf
sample.obj
sample.sub
[..]
[-a-]
[-c-]
[-n-]

Figure 5-42 TRACE OUTPUT Dialog Box

  — [File name:]
  Specifies the name of the file where the trace information displayed in the trace display area is to be stored. A file with the specified file name is output to the current directory displayed in [Directory:].

  Pressing the (Enter) key after inputting or selecting a fie name or double-clicking the file name specified in the list box outputs the contents of the MEMORY DUMP window to a file.

  The destination directory can be changed by specifying a new current directory in the sequence of drive name and directory name.

To change the current directory, use the following procedure.

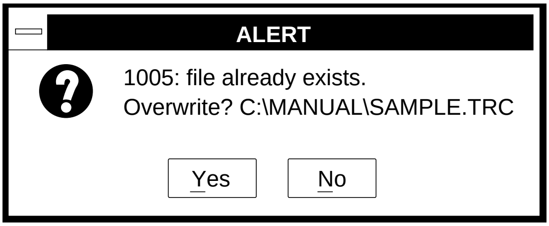
(1) Select the target drive name from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and the current directory of the targed drive are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

(2) Select a subdirectory from the list box by double-clicking. The subdirectory name is displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:]. The subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory where the trace information area is output is reached.

— [Directory:]
Displays the current directory. When the drive or subdirectory is modified in [File name:], this display will be modified too.

— <Done> button
Clicking the <Done> button checks whether there is another file with the same name as that specified in [File name:]. If there already is, the message box shown in figure 5-43 is displayed.
In this state, clicking the <Yes> button overwrites the existing file with the file specified in [File name:], and closes the TRACE OUTPUT dialog box after storing the settings in the TRACE OUTPUT dialog box. On the other hand, when clicking the <No> button, the system returns to the TRACE OUTPUT dialog box.



1005: file already exists.
Overwrite? C:\MANUAL\SAMPLE.TRC

Figure 5-43 File Overwrite Confirmation Message (TRACE OUTPUT Dialog Box)

— <Cancel> button
Clicking the <Cancel> button closes the TRACE OUTPUT dialog box without outputting the settings to a file. It also returns the system to its state when the TRACE OUTPUT dialog box was opened without storing the settings.

- Additional information in bus-cycle unit display
  When trace information is displayed in bus cycle units by selecting <Bus cycle> in [Text display format] of [Trace display - Setting], the message

  *** E7000 ***

  is added to the display to indicate the E7000PC emulator cycle in the last bus cycle wherein the user program was terminated. It is also displayed when emulation is temporarily halted by the break conditions set by the BREAK_SEQUENCE command (emulator command) being satisfied or by the number of breaks set by the BREAK command (emulator command) being encountered.

Notes:

(1) During memory access by a PC relative instruction such as MOV.W @ (PC, 10), R0, PRG is displayed in the MCU status (ST).

(2) If <Range> is specified in [Condition type] in the [Trace condition...] command, assembly codes cannot be displayed correctly. They can be used for only reference.

Related Functions

GUI commands: [Trace mode...], [Trace condition...], and [Trace display - Setting...]
Emulator commands: BREAK and BREAK_SEQUENCE

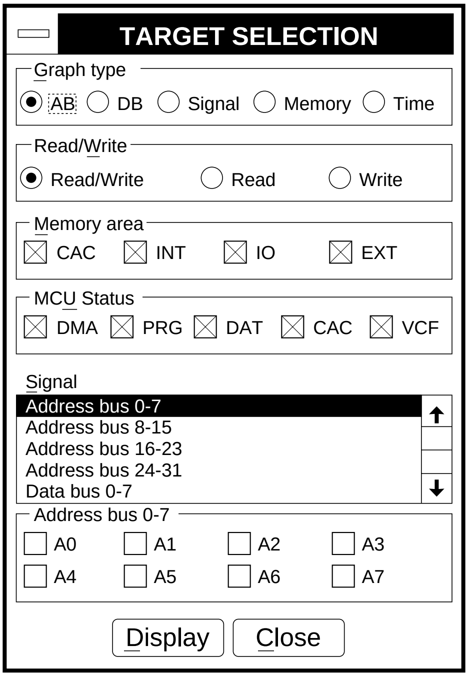
5.5.5 Displaying Trace Information in Graph Form (Alt + P)

[Trace display - Graph] TRACE DISPLAY GRAPH TARGET SELECTION dialog box

Overview

Displays trace information as a graph according to the specifications in the [Trace display - Setting...] command.

Window



TARGET SELECTION

Graph type
● AB ○ DB ○ Signal ○ Memory ○ Time

Read/Write
● Read/Write ○ Read ○ Write

Memory area
☒ CAC ☒ INT ☒ IO ☒ EXT

MCU Status
☒ DMA ☒ PRG ☒ DAT ☒ CAC ☒ VCF

Signal
Address bus 0-7
Address bus 8-15
Address bus 16-23
Address bus 24-31
Data bus 0-7

Address bus 0-7
□ A0 □ A1 □ A2 □ A3
□ A4 □ A5 □ A6 □ A7

Figure 5-44 TRACE DISPLAY GRAPH TARGET SELECTION Dialog Box

Operation

- [TARGET SELECTION]
  Specifies drawing conditions for the trace information displayed in the TRACE DISPLAY GRAPH window.

— [Graph type] group box

Specifies the type of graph to be used in displaying trace information as follows:

<AB>: Change in address bus value in bus cycle units (default at system initiation)

<DB>: Change in data bus value in bus cycle units

<Signal>: Changes in SH7604-control signals and external probe signals in bus cycle units

<Memory>: Change in memory data specified by [Trace Memory condition] in the [Trace memory...] command in bus cycle units

<Time>: Execution time (access time) as total number of clock cycles in address bus cycle units in histogram form

— [Read/Write] group box

Specifies a read or write cycle condition as a trace information display condition as shown below. This specification is valid only when <AB> or <DB> is selected in [Graph type].

<Read/Write>: Displays trace information acquired in read or write cycles as a graph (default at system initiation)

<Read>: Displays trace information acquired in read cycles as a graph

<Write>: Displays trace information acquired in write cycles as a graph

— [Memory area] check box

Specifies memory access conditions as a trace information display condition as shown below. This specification is valid only when <AB> or <DB> is selected in [Graph type].

<CAC>: Displays trace information acquired in cache access (cache-hit) cycles as a graph

<INT>: Displays trace information acquired in internal area access cycles as a graph

<IO>: Displays trace information acquired in internal I/O access cycles as a graph

<EXT>: Displays trace information acquired in external area access cycles as a graph

The check buttons work as follows:

<Check>: The access condition is specified as a trace information display condition

<No check>: The access condition is not specified as a trace information display condition

Immediately after system initiation, all check buttons are specified as trace information display conditions.

— [MCU status] check box

Specifies SH7604-MCU bus cycle condition as trace information display conditions. This specification is valid only when <AB> or <DB> is selected in [Graph type].

<DMA>: Displays trace information acquired in DMA cycles as a graph

<PRG>: Displays trace information acquired in program fetch cycles as a graph

<DAT>: Displays trace information acquired in I/O or memory access cycles as a graph

<CAC>: Displays trace information acquired in cache-fill access cycles as a graph

<VCF>: Displays trace information acquired in vector fetch cycles as a graph

The check button works as follows.

<Check>: The SH7604 bus cycle condition is specified as a trace information display condition

<No check>: The SH7604 bus cycle condition is not specified as a trace information display condition

Immediately after system initiation, all check buttons are specified as trace information display conditions.

— [Signal]

Specifies SH7604 control signal and external probe signal conditions as trace information display conditions. A maximum of 32 graphs can be selected and displayed in the graphic display area. This specification is valid only when <Signal> is specified in [Graph type]. Use the following procedure to perform the specification.

(1) Select from the list box a signal group including the required signal and double-click it. Selectable signal groups in the list box are shown in table 5-9.

Table 5-9 [Signal] List in [Trace display - Graph]

Signal Group	**Signals**
Address bus 0-7	Changes in address bus bits 0-7 waveforms in bus cycle units
Address bus 8-15	Changes in address bus bits 8-15 waveforms in bus cycle units
Address bus 16-23	Changes in address bus bits 16-23 waveforms in bus cycle units
Address bus 24-31	Changes in address bus bits 24-31 waveforms in bus cycle units
Data bus 0-7	Changes in data bus bits 0-7 waveforms in bus cycle units
Data bus 8-15	Changes in data bus bits 8-15 waveforms in bus cycle units
Data bus 16-23	Changes in data bus bits 16-23 waveforms in bus cycle units
Data bus 24-31	Changes in data bus bits 24-31 waveforms in bus cycle units
Non maskable interrupt	Change in the NMI signal waveform in bus cycle units
Reset	Change in the RESET signal waveforms in bus cycle units
Bus request	Change in the BREQ signal waveform in bus cycle units
Bus acknowledge	Change in the BACK signal waveform in bus cycle units
Interrupt 0-3	Changes in the IRL0 to IRL3 signal waveformss in bus cycle units
Trigger input	Changes in the trigger input pin signal waveform in bus cycl units

(2) The signals selected in the list box are displayed as check buttons. By checking displayed check buttons, the control signals and external probe signals for drawing trace information as a graph are specified.

<Check>: The control signal and external probe signal are specified as trace information display conditions

<No check>: The control signal and external probe signal are not specified as trace information display conditions

- <Display> button
  Clicking the <Display> button opens the TRACE DISPLAY GRAPH window and displays trace information as a graph depending on the [TARGET SELECTION] settings. Figure 5-45 shows the TRACE DISPLAY GRAPH window.

- <Close> button
  Clicking the <Close> button stores the settings and closes only the TRACE DISPLAY GRAPH TARGET SELECTION dialog box.

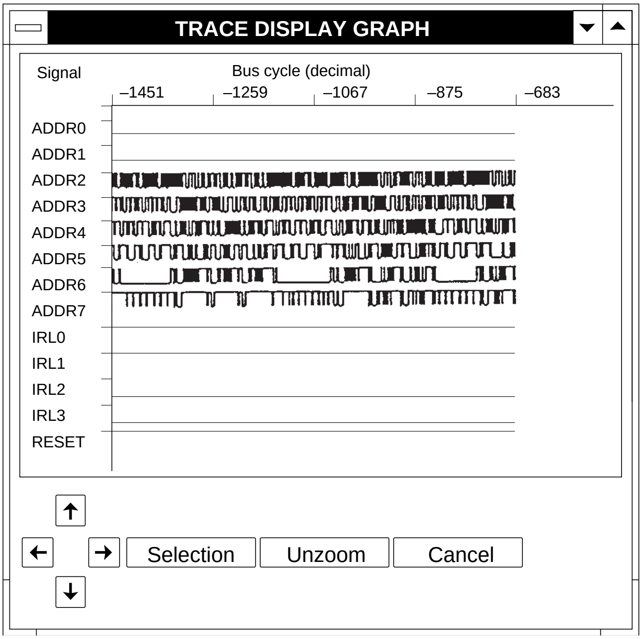


Figure 5-45 TRACE DISPLAY GRAPH Window

- Graphic display area
  Displays trace information as a graph in the graphic display area. Zooming can be performed on a rectangular area specified by dragging the mouse.

- Scroll buttons
  Clicking these scroll buttons enables scrolling of the graphic display area in the TRACE DISPLAY GRAPH window.

  — <↑> button
    Scrolls the graph display up by 1/4 display.

  — <←> button
    Scrolls the graph display left by 1/4 display.

  — <→> button
    Scrolls the graph display right by 1/4 display.

  — <↓> button
    Scrolls the graph display down by 1/4 display.

- <Selection> button
  Clicking the <Selection> button opens TRACE DISPLAY GRAPH TARGET SELECTION dialog box.

- <Unzoom> button
  Clicking the <Unzoom> button returns the display to the display before it was zoomed in by the mouse.

- <Cancel> button
  Clicking the <Cancel> button closes the TRACE DISPLAY GRAPH window.

Function

- Trace information graph display
  The acquired trace information is displayed as a graph in the TRACE DISPLAY GRAPH window. One of the five types of graph can be displayed according to the [Graph type] group box setting in the TRACE DISPLAY GRAPH TARGET SELECTION dialog box.

Table 5-10 Types of Graph

[Graph type]	Vertical Direction	Horizontal Direction
AB	Address bus value	Bus cycle count
DB	Data bus value	Bus cycle count
Signal	Specified signal	Bus cycle count
Memory	Data value	Bus cycle count
Time	Address bus value	Clock count

- Trace information display for an address bus waveform
  When <AB> is specified in [Graph type], trace information acquired for changes in an address bus value which satisfies the settings in [Read/Write], [Memory area], and [MCU stat] is displayed in bus cycle units as a graph. An example of trace information display for an address bus waveform is shown in figure 5-46.
  In an address bus graph, the vertical axis indicates the address bus value and the horizontal axis indicates the bus cycle count. Clicking the scroll buttons allows the user to scroll the graph display vertically and horizontally.

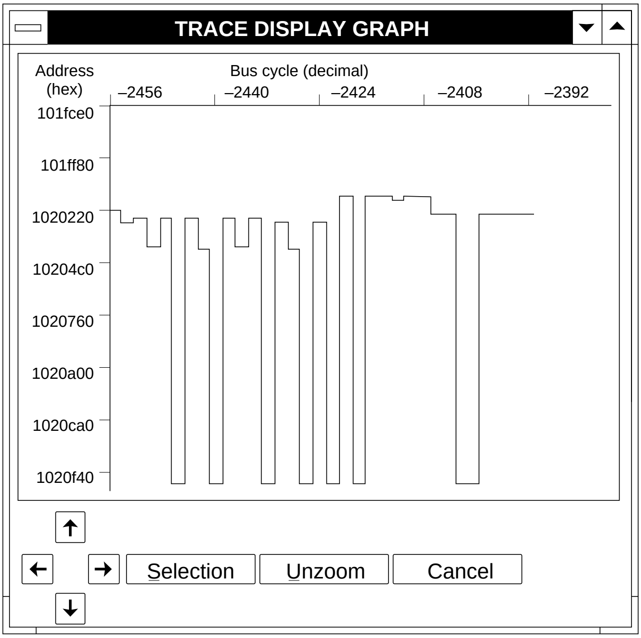


Figure 5-46 Trace Information Display for Address Bus Waveform (TRACE DISPLAY GRAPH Window)

- Trace information display for a data bus waveform
  When <DB> is specified in [Graph type], trace information acquired for changes in a data bus value which satisfies the settings in [Read/Write], [Memory area], and [MCU stat] is displayed in bus cycle units as a graph. An example of trace information display for a data bus waveform is shown in figure 5-47.
  In a data bus graph, the vertical axis indicates the data bus value and the horizontal axis indicates the bus cycle count. Clicking the scroll buttons allows the user to scroll the graph display vertically and horizontally.

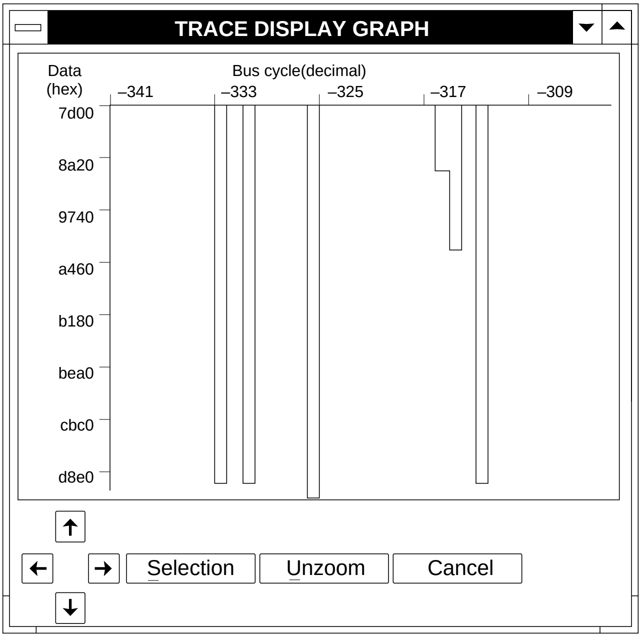


Figure 5-47 Trace Information Display for Data Bus Waveform (TRACE DISPLAY GRAPH Window)

- Trace information display for signal waveforms

  When <Signal> is specified in [Graph type], trace information acquired for changes in the SH7604-control signals and external probe signals is displayed in bus cycle units as a graph. An example of trace information display for signal waveforms is shown in figure 5-48.

  In a signal graph, the vertical axis indicates the specified signals and the horizontal axis indicates the bus cycle count. Clicking the scroll buttons allows the user to scroll the graph display vertically and horizontally.

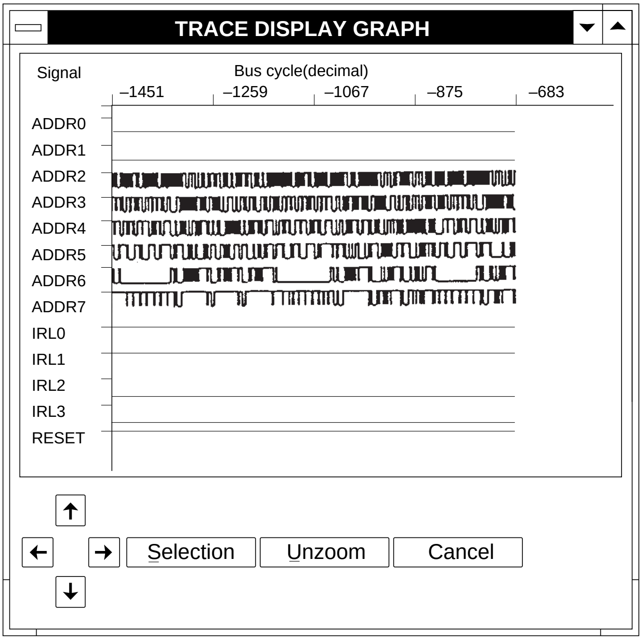


Figure 5-48 Trace Information Display for Signal Waveform (TRACE DISPLAY GRAPH Window)

- Trace information display for memory contents change

  When <Memory> is specified in [Graph type], trace information acquired for data changes within the address set in [Trace memory condition] of the [Trace memory...] command is displayed in bus cycle units as a graph. An example of trace information display for changes in memory is shown in figure 5-49.

  In a memory change graph, the vertical axis indicates the data bus value and the horizontal axis indicates the bus cycle count. Clicking the scroll buttons allows the user to scroll the graph display vertically and horizontally.

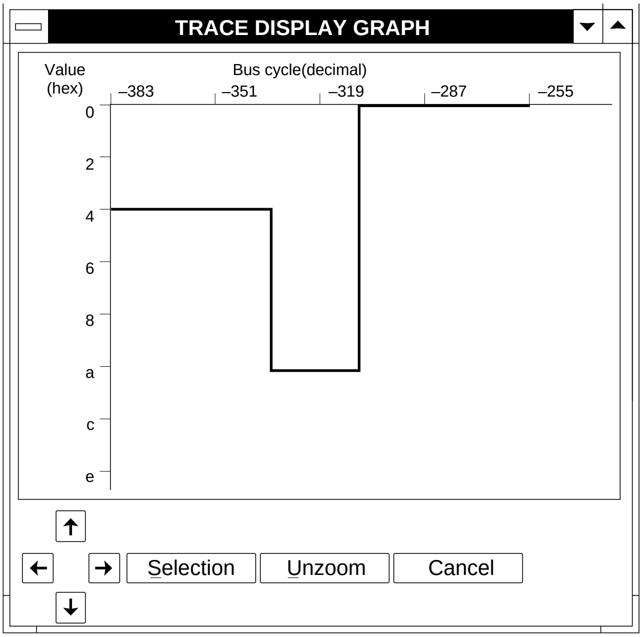


Figure 5-49 Trace Information Display for Memory Change (TRACE DISPLAY GRAPH Window)

- Trace information display for execution time (execution-time histogram)

  When <Time> is specified in [Graph type], total execution time (access time) as number of clock cycles is shown in a histogram in address bus cycle units. An example of trace information display for execution time is shown in figure 5-50 as a histogram.

  In an execution-time histogram, the vertical axis indicates the address bus value and the horizontal axis indicates the clock. Clicking the scroll buttons allows the user to scroll the graph display vertically and horizontally.

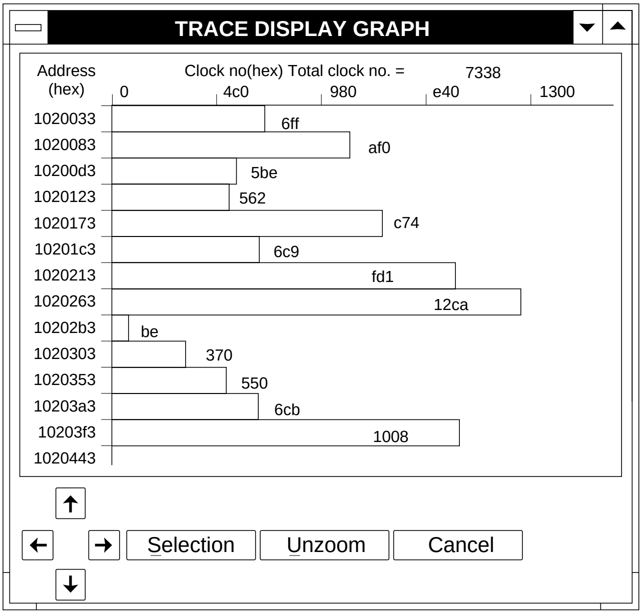


Figure 5-50   Trace Information Display for Execution Time (TRACE DISPLAY GRAPH Window)

- Zooming function

  The graphic display area in the TRACE DISPLAY GRAPH window can be enlarged and reduced by dragging a window corner.

  In order to enlarge or reduce the graph itself, use the zooming function. This function is useful for checking trace information displays in detail, since an area on the graph specified by the mouse can be enlarged.

  An example using the zooming function is shown in figure 5-51. Zooming is performed as follows.

  (1) Move the mouse pointer to the graphic display area and press the left button to set the start point.

  (2) Drag the mouse with the left button pressed so that a zoom display frame (dotted line) appears in the graphic display area.

  (3) Release the left button at the desired end point.

  (4) The part enclosed in the zoom display frame is enlarged.

  The specified area can be zoomed up to eight times continuously. Clicking the <Unzoom> button returns the display magnification to its previous value.

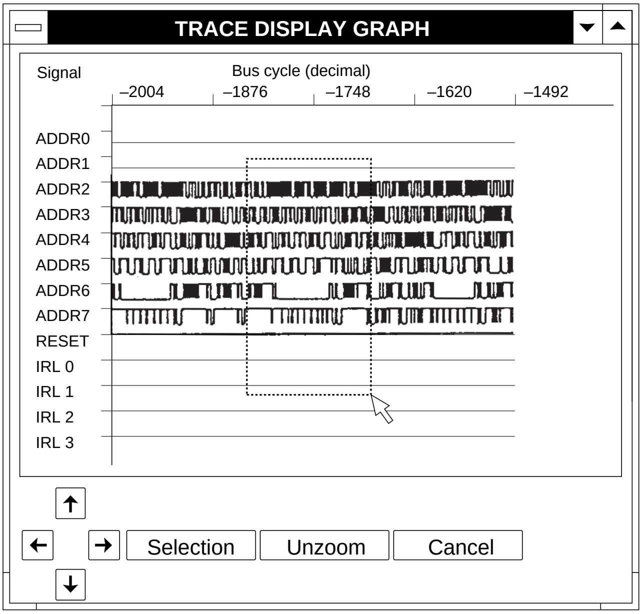


Figure 5-51 Zooming Function Example (TRACE DISPLAY GRAPH Window)

Related Functions

[Trace memory...], [Trace condition...], and [Trace display - Setting...] commands

5.6 Debug Information Display Functions

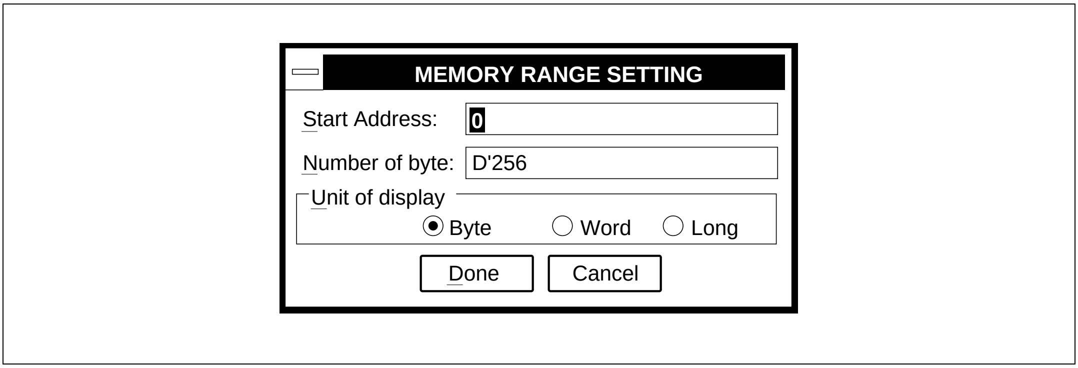
5.6.1 Specifying Memory Display Range (Alt + Ctrl + M)

[Memory - Setting...] MEMORY RANGE SETTING dialog box

Overview

Specifies the memory range whose contents are displayed with the [Memory - Dump] command.

Window



MEMORY RANGE SETTING

Start Address: 0

Number of byte: D'256

Unit of display

● Byte ○ Word ○ Long

Figure 5-52 MEMORY RANGE SETTING Dialog Box

Operation

- [Start address:] text box
  Specifies a number or a symbol for the memory address where display is started. Immediately after system initiation, no data is specified.

- [Number of byte:] text box
  Specifies the size of data to be displayed. Immediately after system initiation, 256 decimal is specified.

- [Unit of display] group box
  Specifies the display unit for memory contents as follows:

<Byte>:	1-byte units (default at system initiation)
<Word>:	2-byte units
<Long word>:	4-byte units

- <Done> button
  Clicking the <Done> button stores the MEMORY RANGE SETTING dialog box settings and closes the MEMORY RANGE SETTING dialog box. The MEMORY DUMP window then opens automatically and the contents of the specified memory range are displayed.

- <Cancel> button
  Clicking the <Cancel> button closes the MEMORY RANGE SETTING dialog box and returns the system to its state when the MEMORY RANGE SETTING dialog box was opened without storing the above settings.

Function

Specifies the memory range whose contents are displayed with the [Memory - Dump] command.

Clicking the <Done> button opens the MEMORY DUMP window in which memory contents are displayed according to the specifications in [Start address:], [Number of byte:], and [Unit of display].

Related Function

[Memory - Dump] command

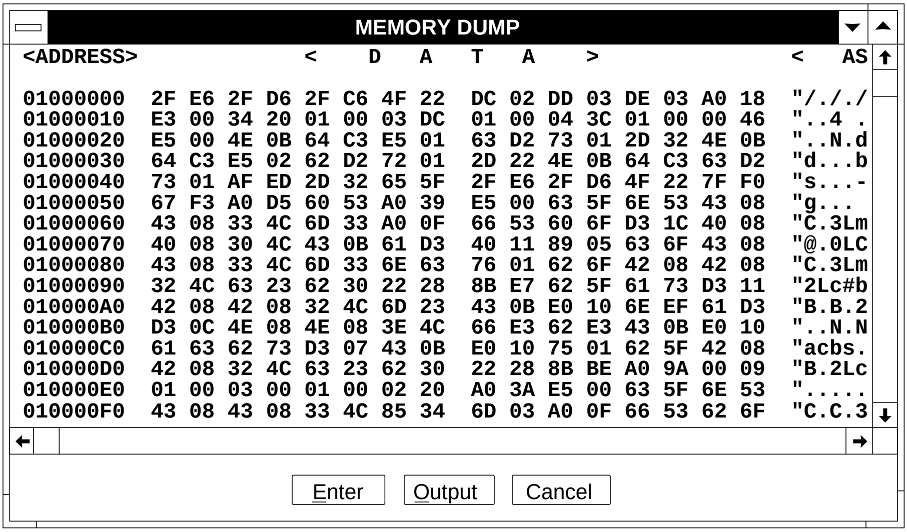
5.6.2 Displaying and Modifying Memory Contents (Alt + M)

[Memory - Dump] MEMORY DUMP window

Overview

Displays the memory contents specified by the [Memory - Setting...] command. The memory contents displayed in the window can be modified directly. The displayed memory contents can also be stored in a file.

Window



```
<ADDRESS>            <   D   A   T   A   >           <   AS

01000000  2F E6 2F D6 2F C6 4F 22  DC 02 DD 03 DE 03 A0 18  "/././
01000010  E3 00 34 20 01 00 03 DC  01 00 04 3C 01 00 00 46  "..4 .
01000020  E5 00 4E 0B 64 C3 E5 01  63 D2 73 01 2D 32 4E 0B  "..N.d
01000030  64 C3 E5 02 62 D2 72 01  2D 22 4E 0B 64 C3 63 D2  "d...b
01000040  73 01 AF ED 2D 32 65 5F  2F E6 2F D6 4F 22 7F F0  "s...-
01000050  67 F3 A0 D5 60 53 A0 39  E5 00 63 5F 6E 53 43 08  "g...
01000060  43 08 33 4C 6D 33 A0 0F  66 53 60 6F D3 1C 40 08  "C.3Lm
01000070  40 08 30 4C 43 0B 61 D3  40 11 89 05 63 6F 43 08  "@.0LC
01000080  43 08 33 4C 6D 33 6E 63  76 01 62 6F 42 08 42 08  "C.3Lm
01000090  32 4C 63 23 62 30 22 28  8B E7 62 5F 61 73 D3 11  "2Lc#b
010000A0  42 08 42 08 32 4C 6D 23  43 0B E0 10 6E EF 61 D3  "B.B.2
010000B0  D3 0C 4E 08 4E 08 3E 4C  66 E3 62 E3 43 0B E0 10  "..N.N
010000C0  61 63 62 73 D3 07 43 0B  E0 10 75 01 62 5F 42 08  "acbs.
010000D0  42 08 32 4C 63 23 62 30  22 28 8B BE A0 9A 00 09  "B.2Lc
010000E0  01 00 03 00 01 00 02 20  A0 3A E5 00 63 5F 6E 53  ".....
010000F0  43 08 43 08 33 4C 85 34  6D 03 A0 0F 66 53 62 6F  "C.C.3
```

Figure 5-53 MEMORY DUMP Window

Operation

- Memory contents display area
  Displays the addresses, data, and corresponding ASCII code of the memory range specified by the [Memory - Setting...] command. The memory contents displayed in the memory contents display area can be modified directly.

- <Enter> button
  Clicking the <Enter> button after changing memory data in the memory contents display area actually rewrites data to the E7000PC emulator memory. Note that memory contents are rewritten only after the <Enter> button has been clicked.

- <Output> button
  Clicking the <Output> button opens the MEMORY OUTPUT dialog box. The data currently displayed in the memory contents display area can be output to a file.

- <Cancel> button
  Clicking the <Cancel> button closes the MEMORY DUMP window.

Function

- Memory contents display
  Displays contents of the memory range specified by the [Memory - Setting...] command in the memory contents display area. Consequently, when changing the memory contents display area with the [Memory - Setting...] command while the MEMORY DUMP window is opened, the newly specified area will be displayed replacing the previous one.

  During emulation execution, the contents displayed in the memory contents display area are automatically modified when emulation is terminated in one of the following ways.

  —When the emulation initiated by clicking the <STEP>, <STEP_OVER>, <STEP_UP> or <CONTINUE> button in the base window is terminated

  — When the emulation initiated by clicking the <Step>, <Step_over>, or <Continue> button in the DISASSEMBLE window is terminated

  — When the emulation initiated by the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

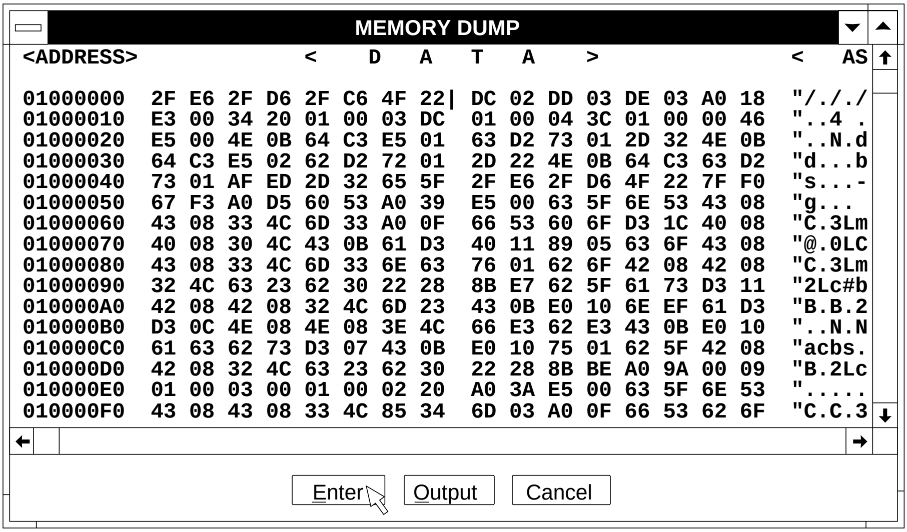
- Memory contents modification
  Memory contents displayed in the memory contents display area can be modified directly, as shown in figure 5-54.

  — After moving the mouse pointer to the address whose data is to be modified, click the left button and produce a cursor (|).
  When performing modification through only the keyboard, press the (Tab) key and produce a cursor in the memory contents display area. Then move the cursor to the data to be modified using the arrow keys.

— Enter new data from the keyboard. The cursor will move right by one digit for each new digit entered. The cursor will not move the basic display unit of memory data.
Data can be modified only in the area titled <DATA>.

— To move the cursor to the next data item to be modified, use the mouse or enter the arrow keys.

— Clicking the <Enter> button writes the modified data into the specified memory address on the E7000PC emulator. A maximum of 256 data items can be modified at one time.



```
<ADDRESS>             <    D   A   T   A    >             <   AS

01000000  2F E6 2F D6 2F C6 4F 22| DC 02 DD 03 DE 03 A0 18  "/././
01000010  E3 00 34 20 01 00 03 DC  01 00 04 3C 01 00 00 46  "..4 .
01000020  E5 00 4E 0B 64 C3 E5 01  63 D2 73 01 2D 32 4E 0B  "..N.d
01000030  64 C3 E5 02 62 D2 72 01  2D 22 4E 0B 64 C3 63 D2  "d...b
01000040  73 01 AF ED 2D 32 65 5F  2F E6 2F D6 4F 22 7F F0  "s...-
01000050  67 F3 A0 D5 60 53 A0 39  E5 00 63 5F 6E 53 43 08  "g...
01000060  43 08 33 4C 6D 33 A0 0F  66 53 60 6F D3 1C 40 08  "C.3Lm
01000070  40 08 30 4C 43 0B 61 D3  40 11 89 05 63 6F 43 08  "@.0LC
01000080  43 08 33 4C 6D 33 6E 63  76 01 62 6F 42 08 42 08  "C.3Lm
01000090  32 4C 63 23 62 30 22 28  8B E7 62 5F 61 73 D3 11  "2Lc#b
010000A0  42 08 42 08 32 4C 6D 23  43 0B E0 10 6E EF 61 D3  "B.B.2
010000B0  D3 0C 4E 08 4E 08 3E 4C  66 E3 62 E3 43 0B E0 10  "..N.N
010000C0  61 63 62 73 D3 07 43 0B  E0 10 75 01 62 5F 42 08  "acbs.
010000D0  42 08 32 4C 63 23 62 30  22 28 8B BE A0 9A 00 09  "B.2Lc
010000E0  01 00 03 00 01 00 02 20  A0 3A E5 00 63 5F 6E 53  ".....
010000F0  43 08 43 08 33 4C 85 34  6D 03 A0 0F 66 53 62 6F  "C.C.3
```

Figure 5-54 Example of Modifying Memory Contents (MEMORY DUMP Window)

- Memory contents display area output
  The memory contents displayed in the memory contents display area can be output to a file. Clicking the <Output> button displays the MEMORY OUTPUT dialog box shown in figure 5-55.

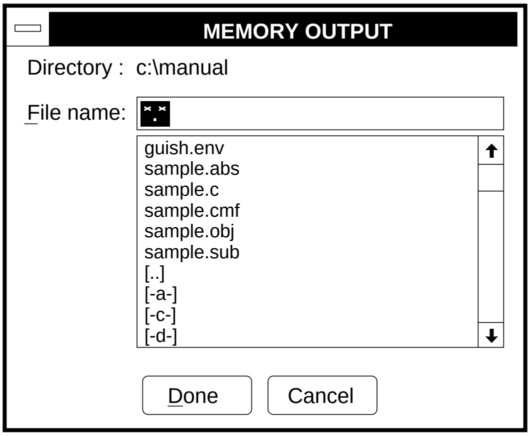


Figure 5-55 MEMORY OUTPUT Dialog Box

— [File name:]
Specifies the name of the file where the memory contents displayed in the memory contents display area are to be stored. A file with the specified file name is output to the current directory displayed in [Directory:].

Pressing the (Enter) key after inputting or selecting a file name or double-clicking the file name specified in the list box outputs the contents of the MEMORY DUMP window to a file.

The destination directory can be changed by specifying a new current directory in the sequence of drive name and directory name.

To change the current directory, use the following procedure.

(1) Select the target drive name from the list box by double-clicking. Drive names are displayed in the format [-a-]. After selected, the drive name and current directory name are displayed in [Directory:]. The current directory contents of the selected drive are displayed in the list box.

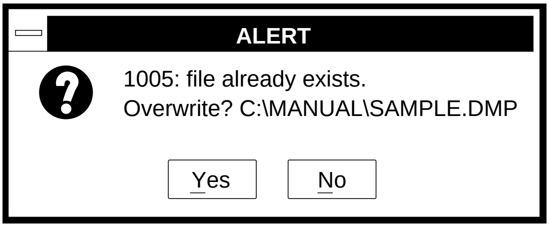
(2) Select a subdirectory name from the list box by double-clicking. The subdirectory name is displayed in the format [directory]. After selected, the subdirectory name is added to [Directory:]. The subdirectory contents are displayed in the list box.

(3) Continue selecting subdirectories until the directory where the target file is to be output is reached.

— [Directory:]
Displays the current directory. When the drive or subdirectory is modified in [File name:], this display will be modified too.

— <Done> button
Clicking the <Done> button checks whether there is another file with the same name as that specified in [File name:]. If there already is, the message box shown in figure 5-56 is displayed.

In this state, clicking the <Yes> button overwrites the existing file with the file specified in [File name:], and closes the MEMORY OUTPUT dialog box after storing the settings in the MEMORY OUTPUT dialog box. On the other hand, when clicking the <No> button, the system returns to the MEMORY OUTPUT dialog box.



1005: file already exists.
Overwrite? C:\MANUAL\SAMPLE.DMP

Figure 5-56 File Overwrite Confirmation Message (MEMORY OUTPUT Dialog Box)

— <Cancel> button
Clicking the <Cancel> button closes the MEMORY OUTPUT dialog box without outputting the settings to a file. The settings of the MEMORY OUTPUT dialog box are not stored.

Related Functions

<STEP>, <STEP_OVER>, <STEP_UP> <CONTINUE>, and <STOP> buttons
GUI commands: [Memory - Setting...], [Disassemble], and [Go...]
Emulator commands: STEP, STEP_OVER, and GO

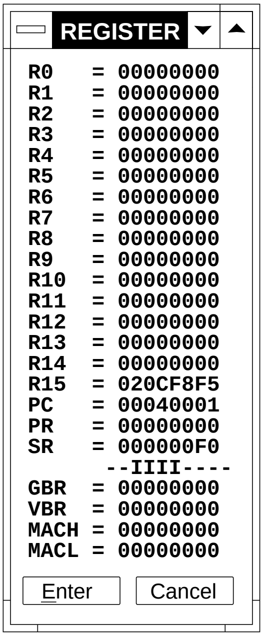
5.6.3 Displaying and Modifying Register Contents (Alt + R)

[Register] REGISTER window

Overview

Displays the contents of the SH7604 registers. The registers displayed in the window can be directly modified.

Window



```
R0   = 00000000
R1   = 00000000
R2   = 00000000
R3   = 00000000
R4   = 00000000
R5   = 00000000
R6   = 00000000
R7   = 00000000
R8   = 00000000
R9   = 00000000
R10  = 00000000
R11  = 00000000
R12  = 00000000
R13  = 00000000
R14  = 00000000
R15  = 020CF8F5
PC   = 00040001
PR   = 00000000
SR   = 000000F0
          --IIII----
GBR  = 00000000
VBR  = 00000000
MACH = 00000000
MACL = 00000000
```

Figure 5-57 REGISTER Window

Operation

- Register display area
  Displays the contents of the registers.

- <Enter> button
  Clicking the <Enter> button after changing register data in the register display area actually rewrites data to the registers. Note that register data is rewritten only after the <Enter> button has been clicked.

- <Cancel> button
  Clicking the <Cancel> button closes the REGISTER window.

Function

- Register data display
  Displays register data in the register display area when emulation execution has terminated. During emulation execution, the register contents displayed in the register display area are automatically read and re-displayed when emulation has terminated in one of the following ways.

  — When the emulation initiated by clicking the <STEP>, <STEP_OVER>, <STEP_UP> or <CONTINUE> button in the base window is terminated

  — When the emulation initiated by clicking the <Step> button, <Step_over> button, or <Continue> button in the DISASSEMBLE window is terminated

  — When the emulation initiated by the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

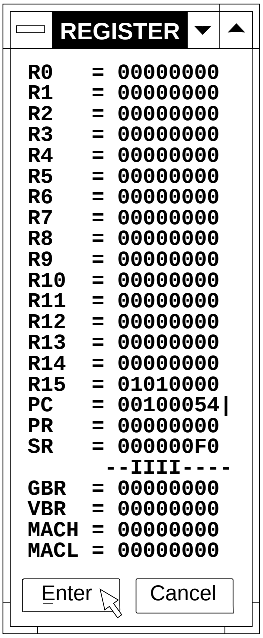
- Register data modification
  Register data displayed in the register display area can be modified directly, as shown in figure 5-58.

  — After moving the mouse pointer to the register whose data is to be modified, click the left button and produce a cursor (|).
  When performing modification through only the keyboard, press the (Tab) key and produce a cursor in the register display area. Then move the cursor to the data to be modified with the arrow keys.

  — Enter new data from the keyboard. The cursor will move right by one digit for each new digit entered. The cursor will not move the basic display unit of register data.

  — To move the cursor to the next data item to be modified, use the mouse or enter the arrow keys.

  — Clicking the <Enter> button writes the modified data into the specified register.



```
R0   = 00000000
R1   = 00000000
R2   = 00000000
R3   = 00000000
R4   = 00000000
R5   = 00000000
R6   = 00000000
R7   = 00000000
R8   = 00000000
R9   = 00000000
R10  = 00000000
R11  = 00000000
R12  = 00000000
R13  = 00000000
R14  = 00000000
R15  = 01010000
PC   = 00100054|
PR   = 00000000
SR   = 000000F0
         --IIII----
GBR  = 00000000
VBR  = 00000000
MACH = 00000000
MACL = 00000000
```

Figure 5-58 Example of Modifying Register Contents (REGISTER Window)

Related Functions

<STEP>, <STEP_OVER>, <STEP_UP>, <CONTINUE>, and <STOP> buttons
GUI commands: [Go...] and [Disassemble]
Emulator commands: STEP, STEP_OVER, and GO

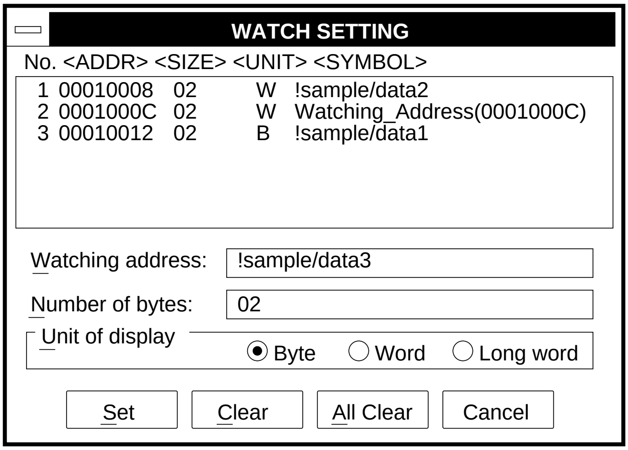
5.6.4 Setting Watch Points (Alt + Ctrl + W)

[Watch - Setting...] WATCH SETTING dialog box

Overview

Sets watch points (addresses whose memory contents are displayed in real time during emulation execution).

Window



```
No. <ADDR> <SIZE> <UNIT> <SYMBOL>
  1 00010008   02        W   !sample/data2
  2 0001000C   02        W   Watching_Address(0001000C)
  3 00010012   02        B   !sample/data1
```

Watching address: !sample/data3

Number of bytes: 02

Unit of display: ● Byte ○ Word ○ Long word

Figure 5-59 WATCH SETTING Dialog Box

Operation

- Watch point display area
  Displays watch point settings in the format shown in figure 5-60.

```
No.      <ADDR>        <SIZE>     <UNIT>          <SYMBOL>
 1     020CF8E5          02          B       !sample/ca__gcm/rem
 ①         ②             ③           ④                ⑤
```

Figure 5-60 Watch Point Display Format (WATCH SETTING Dialog Box)

① Watch point number

② Address at which a watch point is set

③ Display size of watch point (byte-size in hexadecimal)

④ Display unit of watch point
B: 1-byte units
W: 2-byte units
L: 4-byte units

⑤ Symbol name (If no symbol name is defined for the watch point address, then Watching_address (address value) is shown.)

When a displayed watch point is selected by the mouse pointer and double-clicked, the contents of the line are set in [Watching address:], [Number of bytes:], and [Unit of display].

- [Watching address:] text box
Specifies a numerical value or a symbol for an address where a watch point is set.

- [Number of bytes:] text box
Specifies the size of the watch point to be displayed. The display size can be specified from 1 to 32 bytes in 1-byte units. The default is 2 bytes.

- [Unit of display] group box
Specifies the unit for displaying the contents of the watch point.

<Byte>:	1-byte units (default at system initiation)
<Word>:	2-byte units
<Long word>:	4-byte units

- <Set> button
Clicking the <Set> button after setting values in [Watching address:], [Number of bytes], and [Unit of display] sets a watch point and additionally displays it in the watch point display area. A maximum of eight watch points can be specified.

- <Clear> button
Clicking the <Clear> button cancels a watch point specified by setting an address in [Watching address:] or by selecting an address in the watch point display area. The cancelled watch point disappears from the watch point display area.

- <All Clear> button
  Clicking the <All Clear> button clears all watch points. Consequently, all watch points disappear from the watch point display area.

- <Cancel> button
  Clicking the <Cancel> button closes the WATCH SETTING dialog box after storing all of the above settings.

Function

- Watch function
  The watch function provided for the E7000PC displays the contents of the watch point address executed during emulation in the WATCH DISPLAY window. The watch point display is updated in 1-second units.

- Watch point address
  An address in the emulation memory must be specified in [Watching address:]. Otherwise, realtime emulation cannot be performed.

- Watch function termination
  When a menu command or an emulator command is executed during emulation, the E7000PC emulator enters parallel mode, thus causing the watch function to be terminated.

Related Function

[Watch - Display] command

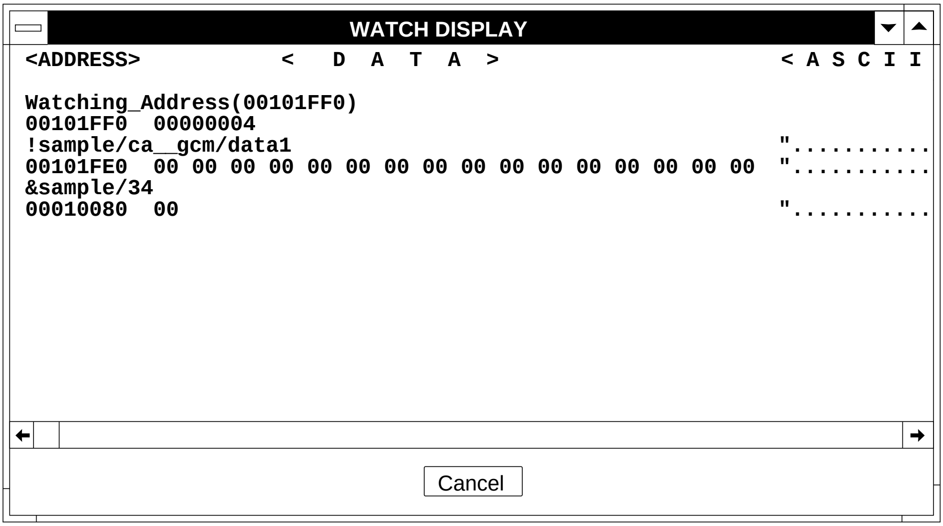
5.6.5 Displaying Watch Point Contents (Alt + W)

[Watch - Display] WATCH DISPLAY window

Overview

Displays the contents of the watch points (addresses whose memory contents are displayed in real time during emulation execution).

Window



```
<ADDRESS>           <   D  A  T  A  >                      < A S C I I
 
Watching_Address(00101FF0)
00101FF0  00000004
!sample/ca__gcm/data1                                          "..........
00101FE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  "..........
&sample/34
00010080  00                                                   "..........
```

Figure 5-61 WATCH DISPLAY Window

Operation

- Watch point contents display area
  According to the [Watch - Setting...] command specifications, displays watch point contents in the format shown in figure 5-62.

```
xxxxxxxx........xxxx
         ①

xxxxxxxx  xx......................xxx "xxxxxxxx........xxxx"
   ②                  ③                          ④
```

Figure 5-62 Watch Point Contents Display Format (WATCH DISPLAY Window)

① Symbol name
Symbol name in the same format as displayed in the watch point display area of the [Watch - Setting...] command

② Watch point address

③ Contents of the memory address where the watch point is set (in dump image)

④ Contents of the memory address where the watch point is set (in ASCII code)

- <Cancel> button
  Clicking the <Cancel> button closes the WATCH DISPLAY window.

Function

- Watch function
  The watch function provided for the E7000PC displays the contents of the watch point address executed during emulation in real time. The watch point display is updated in 1-second units.

- Watch function termination
  When the (Space) key or (Enter) key is pressed in the command area during emulation and the E7000PC enters parallel mode, the watch function terminates.

Related Function

[Watch - Setting...] command

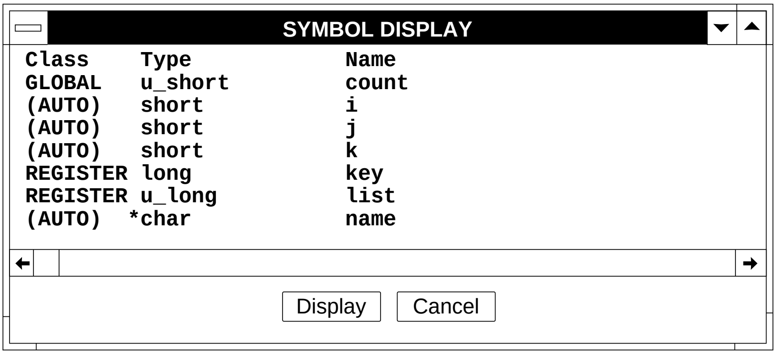
5.6.6 Displaying Symbol Information (Alt + S)

[Symbol - Display] SYMBOL DISPLAY window

Overview

Displays symbol information within the scope of the execution stop address.

Window



```
Class    Type            Name
GLOBAL   u_short         count
(AUTO)   short           i
(AUTO)   short           j
(AUTO)   short           k
REGISTER long            key
REGISTER u_long          list
(AUTO)  *char            name
```

Figure 5-63 SYMBOL DISPLAY Window

Operation

- Symbol information display area
  All symbols declared and defined within the scope of the execution stop address are displayed.

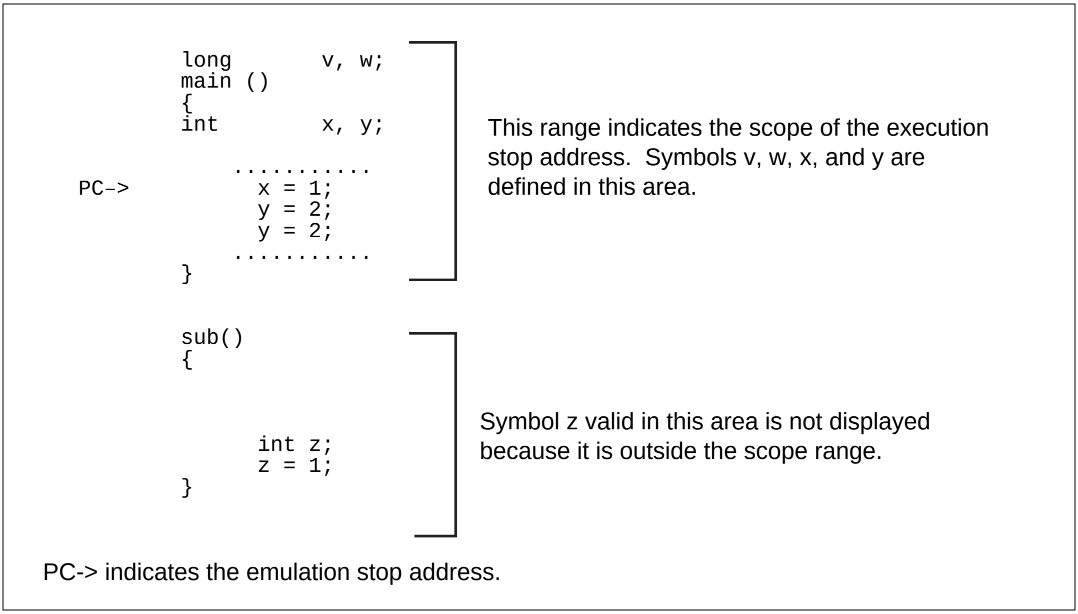
  The contents of the specified symbol can be displayed by selecting a symbol to be displayed and by clicking the <Display> button.

- <Display> button
  Clicking the <Display> button opens the SYMBOL VALUE window and displays that symbol's value in the SYMBOL VALUE window. For details on the SYMBOL VALUE window, refer to section 5.6.7, Displaying and Modifying Symbol Values.

- <Cancel> button
  Clicking the <Cancel> button closes the SYMBOL DISPLAY window.

Function

- Scope of the execution stop address
  In the symbol information display area, all symbols declared and defined within the scope of the execution stop address are displayed.
  The scope of the execution stop address means the range accessible from the function containing the PC when execution stops.
  Figure 5-64 shows an example of the execution stop address. In this case, the scope covers the area within the main function, and the area outside of functions in the file (unit) containing the main function.



```
          long       v, w;
          main ()
          {
          int        x, y;

              ...........
PC->            x = 1;
                y = 2;
                y = 2;
              ...........
          }

          sub()
          {


                int z;
                z = 1;
          }
```

This range indicates the scope of the execution stop address. Symbols v, w, x, and y are defined in this area.

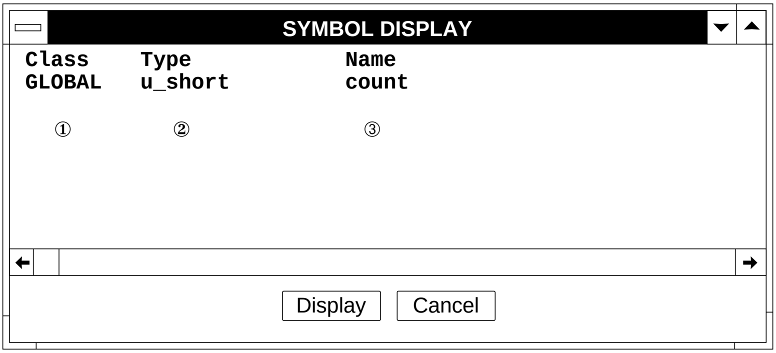
Symbol z valid in this area is not displayed because it is outside the scope range.

PC-> indicates the emulation stop address.

Figure 5-64 Execution Stop Address Example

When the program is optimized by the C compiler and there are symbols declared with the same name within and outside a function, information on multiple symbols may be displayed depending on the emulation stop address.
In a multi-load module, the scope is limited within the segment addressed by the PC when the program execution stops.

- Symbol information display area contents
  Displays information of symbols in the format shown in figure 5-65.



```
Class     Type              Name
GLOBAL    u_short           count
```

Figure 5-65 Display Format of Symbol Information Display Area (SYMBOL DISPLAY Window)

① Class: Storage class to which the symbol is assigned
② Type: Symbol attribute
③ Name: Symbol name

— Class
  Displays the storage class to which the symbol is assigned as listed in table 5-11.

Table 5-11 Storage Class Display Format (SYMBOL DISPLAY Window)

Display Format	**Storage Class**
REGISTER	Register variable
GLOBAL	Externally defined variable
EXTERN	External reference variable
STATIC	Static variable
AUTO	Local variable
(AUTO)	No assignment*1
NONE*2	EQUATE name

Notes: 1. When the program is optimized by the C compiler, it may not be possible to assign local variables and register variables declared within the scope to any storage class. In such a case, the storage class is displayed as (AUTO).

2. No data is displayed as Class for EQUATE name.

— Type
Displays the symbol attribute in the format shown in table 5-12.

Table 5-12 Symbol Attribute Display Format (SYMBOL DISPLAY Window)

Symbol Attribute	Display Format	Description
Integer	char	Signed 1-byte integer variables
	u_char	Unsigned 1-byte integer variables
	short	Signed 2-byte integer variables
	u_short	Unsigned 2-byte integer variables
	long	Signed 4-byte integer variables (including int)
	u_long	Unsigned 4-byte integer variables (including unsigned int)
Floating-point	float	Single-precision floating-point variables
	double	Double-precision floating-point variables (including long double)
Structure	structure	Structure
Union	union	Union
Array	xxxx[ ]	Array variables are indicated by [ ] following an integer, floating-point, structure, or union variable
		Example: struct [ ]
Pointer	*xxxx	Pointer variables are indicated by an asterisk (*) placed in front of an integer, floating-point, structure, or union variable
		Example: *float
Label	label	Labels in the source code
Equate	equate	Symbols assigned by the .equ divective in the assembly program
Enumeration	long	Enumeration type

— Name
Displays the symbol name.

- Symbol value display
The value of a symbol in the symbol information display area can be displayed by selecting it by the mouse and by clicking the <Display> button to open the SYMBOL VALUE window.

- Symbol information automatic modification
  The contents of the symbol information display area are automatically modified and displayed when emulation is terminated in one of the following ways. In these cases, the symbol information of all symbols within the scope of the execution stop address is displayed.

  — When the emulation initiated by clicking the <STEP>, <STEP_OVER>, <STEP_UP> or <CONTINUE> button in the base window is terminated

  — When the emulation initiated by clicking the <Step>, <Step_over>, or <Continue> button of the DISASSEMBLE window is terminated

  — When the emulation initiated by the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

Notes

(1) In regard to symbols declared outside the function, the symbol information is always displayed regardless of its declaration position.

(2) If there are symbols declared with the same name within a function, multiple symbol names may be displayed (only when optimized).

(3) Labels within a function are not displayed.

Related Functions

<STEP>, <STEP_OVER>, <CONTINUE>, and <STOP> buttons
GUI commands: [Symbol - Value], [Go...], and [Disassemble]
Emulator commands: STEP, STEP_OVER, and GO

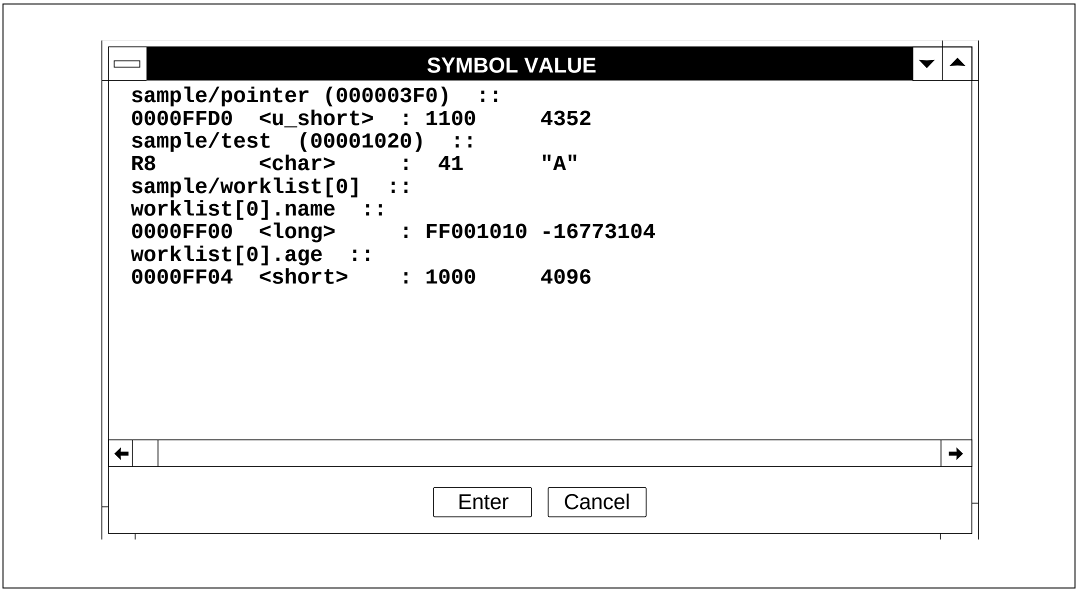
5.6.7 Displaying and Modifying Symbol Values (Alt + Ctrl + N)

[Symbol - Value] SYMBOL VALUE window

Overview

Displays and directly modifies the symbol values specified by the [Symbol - Display] command or by clicking the <DISPLAY> button on the tool bar.

Window



```
sample/pointer (000003F0)  ::
0000FFD0  <u_short>  : 1100     4352
sample/test  (00001020)  ::
R8        <char>     :  41      "A"
sample/worklist[0]  ::
worklist[0].name  ::
0000FF00  <long>     : FF001010 -16773104
worklist[0].age  ::
0000FF04  <short>    : 1000     4096
```

Figure 5-66 SYMBOL VALUE Window

Operation

- Symbol value display area
  Displays the symbol names and values or directly modifies the displayed symbol values.

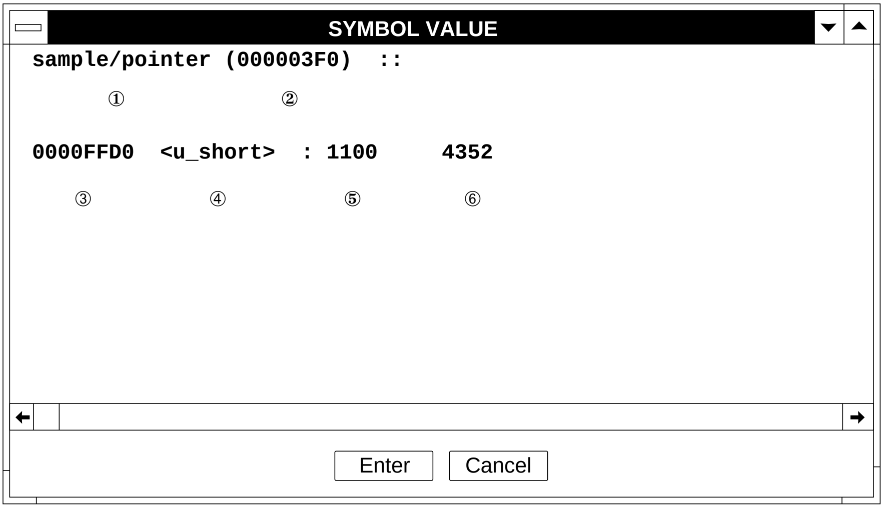
- <Enter> button
  Clicking the <Enter> button after modifying symbol values in the symbol value display area actually rewrites data to the E7000PC emulator. Note that symbol values are rewritten only after the <Enter> button has been clicked.

- <Cancel> button
  Clicking the <Cancel> button closes the SYMBOL VALUE window.

Function

- Symbol value display
  Displays a maximum of 16 symbol values specified with the [Symbol - Display] command or the <DISPLAY> button on the tool bar.

  — Display format
  Figure 5-67 shows the display format of the symbol value display area.



```
sample/pointer (000003F0)  ::

0000FFD0  <u_short>  : 1100     4352
```

Figure 5-67 Symbol Value Display Format (SYMBOL VALUE Window)

① Symbol name
Displays the symbol name specified with the [Symbol - Display] command or the <DISPLAY> button on the tool bar.

② Pointer address
Displays the address to which a pointer symbol is assigned. This address is displayed within parentheses, only when the symbol is a pointer-type.

③ Symbol address
Displays the address to which a symbol is assigned. For a pointer symbol, the symbol address indicates the address specified by the pointer, and for a register symbol (when the symbol class is register variable), it indicates the register name.

④ Type
Displays the symbol type enclosed in < >.

⑤ Hexadecimal display field
Displays the symbol value in hexadecimal.

⑥ Symbol value display field
Displays the actual symbol value according to the symbol type.

— Symbol name display format
The symbol name display format is determined depending on the area where the symbol is declared.

(a) All symbols declared in the assembly source program
[%<segment name>/]<unit name>/<variable name>
<variable name>

(b) An exernally defined symbol, external reference symbol, or static symbol (declared outside the function) declared in the C source program
[%<segment name>/]<unit name>/<variable name>[[<suffix>]...]
<variable name>[[<suffix>]...][.<member name>][[<suffix>]...]

(c) A local symbol, register symbol, or static symbol (declared within the function) declared in the C source program
[%<segment name>/]<unit name>/<function name>/<variable name>[[<suffix>]...]
<variable name>[[<suffix>]...][.<member name>][[<suffix>]...]

The items are described as follows:

<segment name>: Segment name (the name of a single absolute load module file output by the H-series linkage editor as part of a multi-load module file)
<unit name>: Unit name (object file name in compile and assembly units)
<function name>: Function name (only in the C source program)
<variable name>: Variable name (simple, pointer, array), label name, structure name, union name, or equation name (only a simple variable name or label name is available in an assembly source program)
<member name>: Member name of structure or union (only in the C source program)
<suffix>: Array elements (only in the C source program)

— Symbol type display format
Symbol types are displayed in the symbol value display area in the format shown in table 5-13.

Table 5-13 Symbol Type Display Format (SYMBOL VALUE Window)

Symbol Type	Type Name	Display Size
<char>	char	1 byte
<u_char>	unsigned char	1 byte
<short>	short	2 bytes
<u_short>	unsigned short	2 bytes
<long>	int or long	4 bytes
<u_long>	unsigned int or unsigned long	4 bytes
<float>	float	4 bytes
<double>	double or long double	8 bytes
<char_bit x:y>*	x-bit char	1-byte bit field
<u_char_bit x:y>	x-bit unsigned char	1-byte bit field
<short_bit x:y>	x-bit short	2-byte bit field
<u_short_bit x:y>	x-bit unsigned short	2-byte bit field
<long_bit x:y>	x-bit long	4-byte bit field
<u_long_bit x:y>	x-bit unsigned long	4-byte bit field
<label>	label	4 bytes
<equate>	EQUATE name	4 bytes

Note: The parameters x and y indicate the bit field size and offset from the address where the symbol is assigned, respectively.

Example: <u_long_bit 4:16> indicates unsigned long bit-field data whose bit field size and bit offset are 4 bits and 16 bits, respectively.

— Symbol value display format

The symbol value display format is determined depending on the symbol type.

(a) <char>

A <char> symbol is displayed in hexadecimal and ASCII code. If the character cannot be displayed, a period (.) is displayed instead.

For a char pointer variable or char array variable, the first 16 bytes are displayed in hexadecimal while the rest up to the null code are displayed in ASCII characters. ASCII characters can display a maximum of 256 bytes.

Example:
```
sample/main/a       : :
00001200         <char>    :     41          'A'
```

```
sample/main/p_ascii(00002000)   : :
00001080          <char>    :     4142434400          "ABCDEFG."
```

(b) Integer type other than <char>
Integer symbols other than <char> are displayed in hexadecimal and decimal.

Example:
```
sample/main/b       : :
00004020          <long>    :     0000FFFF            65535
```

(c) <float> or <double>
Floating-point data is displayed as shown in table 5-14.

Table 5-14 Floating-Point Display Format (SYMBOL VALUE Window)

Floating-Point Type	Display Format
Real number (excluding zero)	[–] x.xxxxxxe ± xxx (Signal) (Mantissa) (Exponent) x is a decimal
Zero	0.000000e + 000 or –0.000000e + 000
Positive infinity	Inf
Negative infinity	–Inf
Not-a-number (a value not corresponding to a real number or infinity)	NaN (MSB = 0), –NaN (MSB = 1)

Example:
```
sample/main/c       : :
00005000        <double>   :       C05ED9999999999A  –1.234500e+002
```

— Symbol value display examples
Symbol value display examples are summarized in table 5-15.

Table 5-15 Symbol Value Display Examples (SYMBOL VALUE Window)

Symbol Attribute	Declaration Example	Symbol Value Display Example	
Integer	short a;	sample/main/a ::	
		00002024 <short> : FFFF	-1
	int b;	sample/main/b ::	
		R7 <long> : 00000050	80
Floating-point	long double c;	sample/main/c ::	
		00020044 <double> : C05ED9999999999A -1.234500e + 002	
Array	char ddd[2];	sample/main/ddd ::	
		00050002 <char> : 4142	"AB"
	unsigned char	sample/main/abc ::	
	abc[2];	abc[0] ::	
		00030020 <u_char> : 01	1
		abc[1] ::	
		00030021 <u_char> : 11	17
Pointer	long*p_abcde;	sample/main/p_abcde (00030300) ::	
		00040080 <long> : FFFFFFFE	-2
Structure	struct {	sample/main/def ::	
	short a :2;	def.a ::	
	double b[2];	00207000 <short_bit 2 : 0>: 0001	1
	int *c;	def.b[0] ::	
	}def;	00207004 <double> : C05ED9999999999A -1.234500e + 002	
		def.b[1] ::	
		0020700C <double> : C05ED9999999999A -1.234500e + 002	
		def.c (00030404)::	
		00207014 <long> : 00000100	256
Union	union {	sample/main/kkk ::	
	int a;	kkk.a ::	
	char b[2];	00001008 <long> : 00000100	256
	}kkk;	kkk.b ::	
		00001008 <char> : 4749	"GI"
Label	exit:	asample/asm_exit ::	
		00010468 <label> : 00001000	4096
EQUATE	prg_end .equ	asample/prg_end ::	
	H'00001004	<equate> : 00001004	4100

- Symbol value automatic modification
  The symbol values displayed in the symbol value display area are automatically modified when emulation is terminated in one of the following ways.

  — When the emulation initiated by clicking the <STEP>, <STEP_OVER>, <STEP_UP>, or <CONTINUE> button in the base window is terminated

  — When the emulation initiated by clicking the <Step>, <Step_over>, or <Continue> button of the DISASSEMBLE window is terminated

  — When the emulation initiated by the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

- Symbol value modification
  Symbol values in the symbol value display area can be modified directly at the hexadecimal display field and symbol value display field.

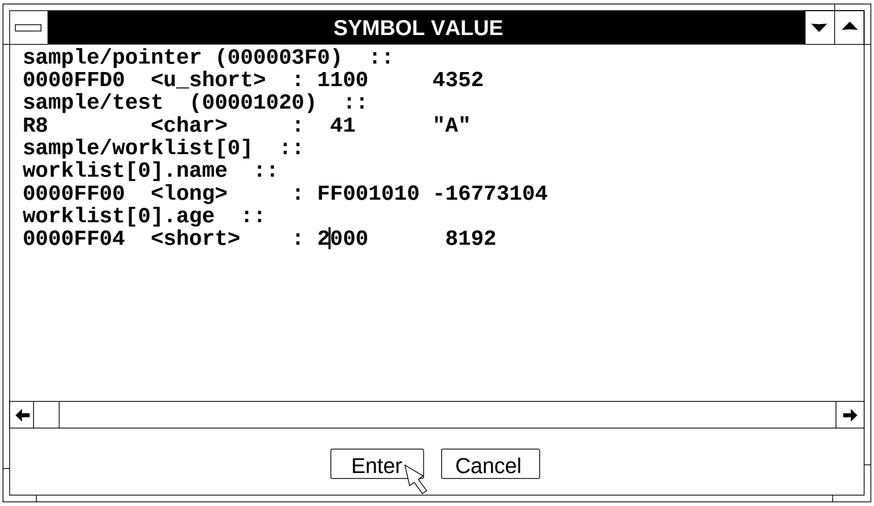
  — Modifications
  Symbol values can be modified using the procedure shown in figure 5-68.

  (1) After moving the mouse pointer to the hexadecimal display field or symbol value display field which is to be modified, click the left button and produce a cursor (|).
  When performing modification using only the keyboard, press the (Tab) key and produce a cursor in the symbol value display area. Then move the cursor to the data to be modified with the arrow keys.

  (2) Enter new data from the keyboard. After modifying the data, the cursor will move right by one digit for each new digit entered. Data cannot be entered beyond the symbol display size.

  (3) To move the cursor to the next data to be modified, use the mouse or enter the arrow keys.

  (4) Clicking the <Enter> button actually performs symbol value modifications.



```
sample/pointer (000003F0)  ::
0000FFD0  <u_short>  : 1100     4352
sample/test  (00001020)  ::
R8        <char>     :  41      "A"
sample/worklist[0]  ::
worklist[0].name  ::
0000FF00  <long>     : FF001010 -16773104
worklist[0].age  ::
0000FF04  <short>    : 2000      8192
```

Figure 5-68 Symbol Value Modification Procedure (SYMBOL VALUE Window)

— Usable characters

(1) Only hexadecimal data can be entered in the hexadecimal display field.
Hexadecimal characters: Numbers (0 to 9) and alphabetical characters (A to F and a to f)

(2) Table 5-16 lists the characters which can be entered in the symbol value display field.

Table 5-16 Symbol Value Display Field Characters (SYMBOL VALUE Window)

Usable Characters	Description
Numbers (0 to 9)	Used for decimal data
Minus (–)	Changes a value from positive to negative
Plus (+)	Changes a value from negative to positive
Space key	Deletes the decimal character located to the left of the cursor and shifts the characters located to the right of the cursor one digit to the left
All characters	All displayable characters can be entered

— Symbol value modification example

Example 1: Modifying char pointer variable xyz from abc to XYZ in the symbol value display field

① Move the mouse pointer to character a and click the left button. A cursor (|) is displayed.

```
sample/xyz     (0000EF00)     ::
00001000    <char>      :       61626300  "|abc."
```

② Enter X from the keyboard. a is modified to X and the cursor moves to the next character on the right. At this point, the hexadecimal display field contents are automatically modified.

```
sample/xyz     (0000EF00)     ::
00001000    <char>      :       58626300  "X|bc."
```

③ Enter Y and Z from the keyboard to perform modifications in the same way.

```
sample/xyz     (0000EF00)     ::
00001000    <char>      :       58595A00  "XYZ|."
```

④ Click the <Enter> button to actually write new data to memory.

Figure 5-69 Modification Example of a char Pointer Variable (SYMBOL VALUE Window)

Example 2: Modifying long-type integer variable abc from 120 to –4 in the symbol value display field

① Move the mouse pointer to digit 1 and click the left button. A cursor (|) is displayed.

```
sample/abc      ::
00001100    <long>      :     00000078       |120
```

② Enter a minus (–) from the keyboard. A minus sign is added to the beginning of the value. At this point, the hexadecimal display field contents are automatically modified.

```
sample/abc      ::
00001100    <long>      :     FFFFFF88       –|120
```

③ Enter 4 from the keyboard. The value is modified and the cursor moves to the next character on the right.

```
sample/abc      ::
00001100    <long>      :     FFFFFE5C       –4|20
```

④ Enter the (SPACE) key from the keyboard twice to delete 2 and 0.

```
sample/abc      ::
00001100    <long>      :     FFFFFFFC       –4|
```

⑤ Click the <Enter> button to actually write new data to memory.

Figure 5-70 Modification Example of a long-Type Integer Variable (SYMBOL VALUE Window)

Example 3: Modifying double floating-point variable fdata from 123.4 to –123450 in the symbol value display field

① Move the mouse pointer to digit 1 and click the left button. A cursor (|) is displayed.

```
sample/fdata     ::
00001200  <double>  :  405ED9999999999A  |1.234000e+002
```

② Enter a minus (–) from the keyboard. A minus sign is added to the beginning of the value. At this point, the hexadecimal display field contents are automatically modified. and the <Enter> button can be clicked because shadow display has been cancelled.

```
sample/fdata  ::
00001200  <double>  :  C05ED9999999999A  –|1.234000e+002
```

③ Press the right arrow key (➔ key) to move the cursor right by five characters.

```
sample/fdata  ::
00001200  <double>  :  C05ED9999999999A  –1.234|000e+002
```

④ Enter 5 from the keyboard. The value is modified and the cursor moves to the next character on the right.

```
sample/fdata  ::
00001200  <double>  :  C05EDCCCCCCCCCCD  –1.2345|00e+002
```

⑤ Press the right arrow key (➔ key) to move the cursor right by five characters.

```
sample/fdata  ::
00001200  <double>  :  C05EDCCCCCCCCCCD  –1.234500e+00|2
```

⑥ Enter 5 from the keyboard.

```
sample/fdata  ::
00001200  <double>  :  C0FE23A000000000  –1.234500e+005|
```

⑦ Click the <Enter> button to actually write new data to memory.

Figure 5-71 Modification Example of a double Floating-Point Variable (SYMBOL VALUE Window)

Example 4: Modifying short-type six-dimensional array variable "argument" from 100 to 0 in the hexadecimal display field

① Move the mouse pointer to digit 6 and click the left button. A cursor (|) is displayed.

```
sample/argument[0][1][2][3][4][5]  ::
00001300  <short>  :  00|64  100
```

② Enter 0 from the keyboard. The value is modified and the cursor moves to the next character on the right. At this point, the symbol value display field contents are automatically modified.

```
sample/argument[0][1][2][3][4][5]  ::
00001300  <short>  :  000|4  4
```

③ Enter 0 from the keyboard.

```
sample/argument[0][1][2][3][4][5]  ::
00001300  <short>  :  0000|  0
```

④ Click the <Enter> button to actually write new data to memory.

Figure 5-72 Modification Example of a Short-Type Six-Dimensional Array Variable (SYMBOL VALUE Window)

Notes

(1) If the symbol such as static variable, local variable, and register variable in the symbol value display area exceeds the scope during program execution, the following message,

*** Symbol is not SCOPE"

is displayed. A symbol outside the scope during program execution and having the same name can be defined but only the symbol within the scope is displayed.

(2) If the symbol has not been allocated or if the symbol allocation has been cancelled during program execution, the following message,

*** Symbol is not allocated

is displayed.

(3) If the symbol in the symbol value display area is to be allocated to an illegal address, the following message

*** Symbol is ILLEGAL ADDRESS

is displayed.

(4) Symbol values for the externally defined variable symbol or external reference variable symbol are always displayed regardless of the scope. If there is a symbol having the same name within the scope, the symbol value of the symbol within the scope is displayed.

(5) If the SP (stack pointer) is modified during local variable display, the displayed symbol value cannot be guaranteed.

(6) When the program is optimized by the C compiler, the local variables cannot be displayed even if they are assigned to the symbol area correctly. Accordingly, care must be taken by referring to assembly instructions before modifying local variables while the program is optimized by the C compiler.

(7) The following must be noted when specifying the symbol in the C source file optimized by the C compiler.

(a) If there are symbols declared with the same name inside and outside a function, information on multiple symbols may be displayed depending on the emulation stop address.

(b) Locations not related to the target symbol must not be referenced or modified.

(8) The following must be noted when modifying the bit-field symbols.

(a) In the hexadecimal display field, the symbol value is displayed in hexadecimal using the display size indicated in table 5-13. However, only the bit-field bits within the range assigned to the bit field can be modified.

(b) In the symbol value display field, the symbol value corresponding to the bit-field size is displayed as decimal data. When modifying this value, decimal data having a value within the specified bit-field size must be entered.

Restrictions

(1) If a symbol to be assigned requires more than 20 kbytes, the following message

```
1032: symbol assignment area too long
```

is displayed instead of the symbol values.

(2) The display format is <pointer> when symbols that have the following attributes are displayed. Contents of the address indicated by the pointer are not displayed.

- A pointer-type symbol for a struture or union member
- A symbol that has pointer and array attributes

(3) A function-type symbol is displayed as a pointer to the function value type for the size from the program start address to the function-type symbol. Note that symbol values cannot be displayed for the void-type function-type symbols.

Related Functions

<DISPLAY>, <STEP>, <STEP_OVER>, <STEP_UP>, <CONTINUE>, and <STOP> buttons
GUI commands: [Symbol - Display], [Go...], and [Disassemble]
Emulator commands: STEP, STEP_OVER, and GO

5.6.8 Setting Source Area (Alt + Ctrl + A)

[Source - Setting...] SOURCE SETTING dialog box

Overview

Specifies the font, font size, and tab size of the source program to be displayed in the source area.

Window

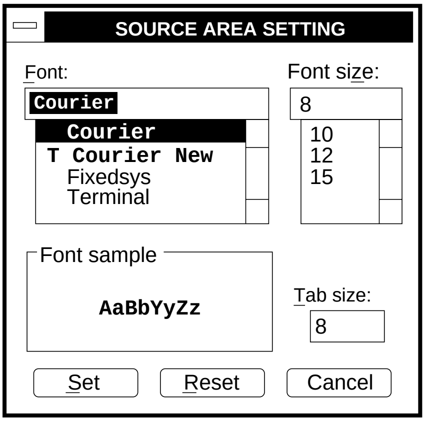


Figure 5-73 SOURCE SETTING Dialog Box

Operation

- [Font:] list box
  Selects a source program font to be displayed in the source area from the list box. Only the fixed-pitch fonts installed using the Windows control panel are displayed in the list box. In this GUI, Courier or Fixed System font is recommended. Default at system initiation is Courier.

- [Font size:] list box
  Selects the size of the font specified in [Font:] from the list box. In True-type fonts, sizes from 4 to 25 can be entered. Default at system initiation is Courier 8.

- [Font sample]
  Displays an example of the font specified by [Font:] and [Font size:].

- [Tab size:] text box
  Specifies 1 to 32 as a tab size of a source program to be displayed in the source area. Default at system initiation is 8.

- <Set>button
  Clicking the <Set> button closes the SOURCE SETTING dialog box. In this case, the settings specified in the SOURCE SETTING dialog box are stored and the source area can be redisplayed according to the stored setting.

- <Reset>button
  Clicking the <Reset> button resets the settings in the SOURCE SETTING dialog box, closes the SOURCE SETTING dialog box, and redisplays the source area according to the default settings.

- <Cancel>button
  Clicking the <Cancel> button closes the SOURCE SETTING dialog box without storing the settings. In this case, the source area display is not modified.

Function

- Font settings
  The font and font size of the source program to be displayed in the source area are specified. The user can use this function to change the current font to a desired font or font size. The modified font settings are valid until they are reset. Even if the source area display contents change according to program execution, the font and font size are not changed.

- Tab size setting
  Specifies the tab size of the source program to be displayed in the source area. The user can use this function to change the current tab setting to a desired tab setting. The modified tab settings are valid until they are reset. Even if the source area display contents change according to program execution, the tab size is not changed.

Related Functions

[Source - Display...] command and source area

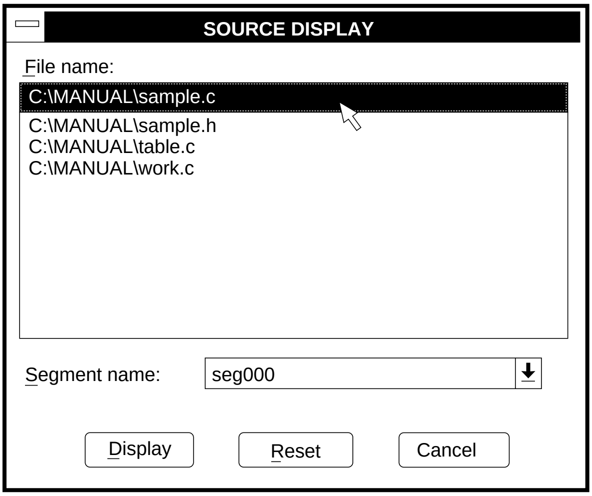
5.6.9 Displaying and Selecting a Source File (Alt + A)

[Source - Display...] SOURCE DISPLAY dialog box

Overview

Lists all source files included in the load module, selects a source file, and displays the file in the source area of the base window.

Window



C:\MANUAL\sample.c
C:\MANUAL\sample.h
C:\MANUAL\table.c
C:\MANUAL\work.c

Segment name: seg000

Figure 5-74 SOURCE DISPLAY Dialog Box

Operation

- [File name:] list box
  Displays all source file names included in the load module. Each file name is expressed as the complete file name from the route directory.

  Clicking a displayed source file with the mouse highlights the name. Click the <Display> button to display the selected source file contents in the source area of the base window.

- [Segment name:] drop-down list box
  Displays segment names composing the load module. Opening the drop-down list box and clicking a displayed segment name displays all source files included in that segment in the [File name:] list box.

  Specifying [All segments] displays source files included in all segments in the [File name:] list box.

- <Display> button
  Clicking the <Display> button displays the source file selected in the source file name display area in the source area of the base window.

- <Reset> button
  Clicking the <Reset> button displays the section of the source file pointed to by the program counter (PC).

- <Cancel> button
  Clicking the <Cancel> button closes the SOURCE DISPLAY dialog box.

Function

The source file name display area displays the following type of files:

- C source files
- Assembly source files
- Include files (such as header files)

Related Functions

[Source-Setting...] command and source area

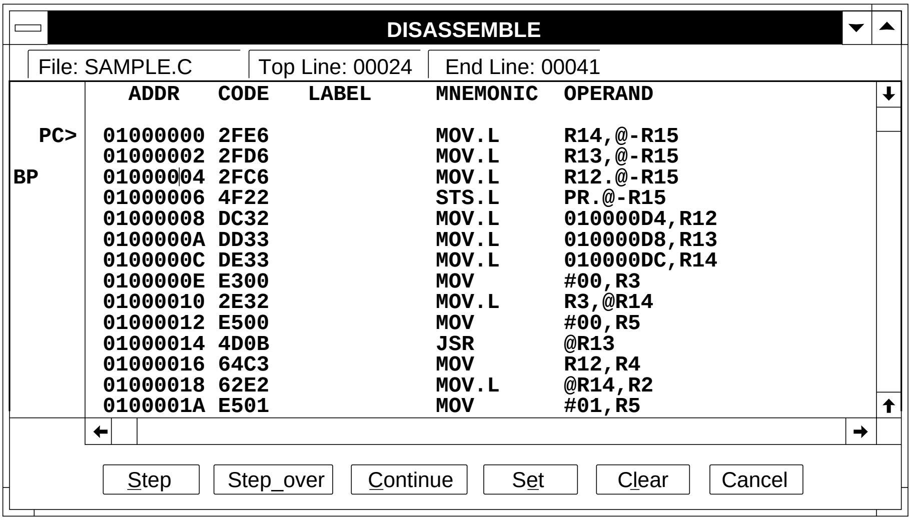
5.6.10 Disassembling and Executing Programs (Alt + D)

[Disassemble] DISASSEMBLE window

Overview

Disassembles and displays the source file displayed in the source area of the base window, and enables assembly-language level program debugging.

Window



File: SAMPLE.C Top Line: 00024 End Line: 00041

```
          ADDR     CODE    LABEL      MNEMONIC  OPERAND

  PC>    01000000 2FE6               MOV.L     R14,@-R15
         01000002 2FD6               MOV.L     R13,@-R15
BP       01000004 2FC6               MOV.L     R12.@-R15
         01000006 4F22               STS.L     PR.@-R15
         01000008 DC32               MOV.L     010000D4,R12
         0100000A DD33               MOV.L     010000D8,R13
         0100000C DE33               MOV.L     010000DC,R14
         0100000E E300               MOV       #00,R3
         01000010 2E32               MOV.L     R3,@R14
         01000012 E500               MOV       #00,R5
         01000014 4D0B               JSR       @R13
         01000016 64C3               MOV       R12,R4
         01000018 62E2               MOV.L     @R14,R2
         0100001A E501               MOV       #01,R5
```

Figure 5-75 DISASSEMBLE Window

Operation

- Source file information area
  Displays the contents of the source information area of the base window.

  — [File name:]
  Displays the name of the file displayed in the source area of the base window.

  — [Top line:]
  Displays the first line number displayed in the source area.

  — [End line:]
  Displays the last line number displayed in the source area.

- Disassemble display area
  Displays disassembly results for the section of the source file displayed in the source area of the base window. The headers have the following meanings:

ADDR:	Address
CODE:	Instruction code (2 bytes)
LABEL:	Label
MNEMONIC:	Instruction mnemonic
OPERAND:	Instruction operand

- Program counter display area
  Displays the PC mark (PC>) indicating the address pointed to by the current program counter (PC).

- <Step> button
  Executes one program line pointed to by the current program counter (PC) and then stops. When the line includes a subroutine call, program execution stops at the first line of the subroutine.

- <Step_over> button
  Executes one program line pointed to by the current program counter (PC) and then stops. When the line includes a subroutine call, program execution stops after executing the subroutine.

- <Continue> button
  Executes the program from the line pointed to by the current program counter (PC). Execution continues until a line including the cursor in the source area is reached.

- <Set> button
  Sets a breakpoint to an instruction which is pointed to by the cursor.

- <Clear> button
  Clears a breakpoint set at an instruction which is pointed to by the cursor.

- <Cancel> button
  Clicking the <Cancel> button closes the DISASSEMBLE window.

Function

- Display of source file disassembly
  The information displayed in the disassemble display area depends on the type of the file in the source area, as shown in table 5-17.

Table 5-17 File Type and Disassembly Display (DISASSEMBLE Window)

Source File Type	Disassemble Display Area	Source File Information Area
C source file, Assembly source file	Disassembles and displays the section displayed in the source area	File name, first line number, last line number
None	Disassembles and displays 16 instructions from the current program counter (PC)	None

- Automatic modification of disassemble display area
  The contents of the disassemble display area are automatically modified when emulation is terminated in one of the following ways:

  — When the emulation initiated by clicking the <STEP>, <STEP_OVER>, <STEP_UP>, or <CONTINUE> button in the base window is terminated

  — When the emulation initiated by clicking the <Step>, <Step_over>, or <Continue> button of the DISASSEMBLE window is terminated.

  — When the emulation initiated by the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

- Assembly-language-level debugging function
  Using the buttons in the DISASSEMBLE window, enables assembly-language-level programs to be debugged. When a source program is displayed in the source area, the source program in the assembly language level is displayed in the source area in realtime during assembly-language-level program execution.

  If a breakpoint is set in the DISASSEMBLE window, a BP mark is also displayed on the corresponding program line in the source area. If a breakpoint is set to a line which is not the start line of the program using the DISASSEMBLE window, the breakpoint cannot be canceled using the <CLEAR> button in the base window. The breakpoint must be cleared using the <Clear> button in the DISASSEMBLE window or the [Break...] command. If multiple breakpoints are specified on a line, a BP mark on the source line may not be cleared until all breakpoints specified on the line are canceled.

Related Functions

<STEP>, <STEP_OVER>, <STEP_UP>, <CONTINUE>, and <STOP> buttons
GUI command: [Go...]
Emulator commands: STEP, STEP_OVER, and GO

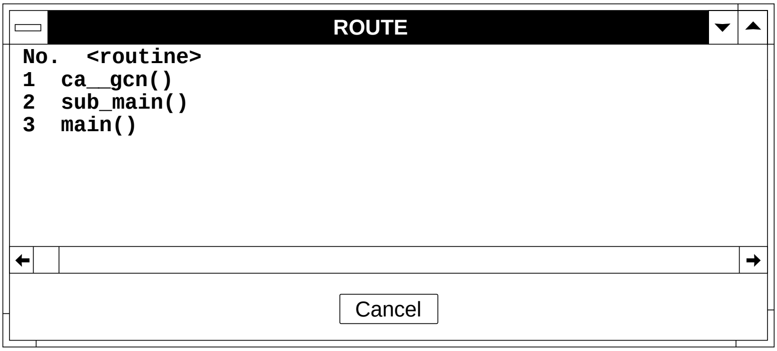
5.6.11 Displaying Function Call Sequence (Alt + U)

[Route] ROUTE window

Overview

Displays the function call sequence up to the function pointed to by the current program counter (PC).

Window



```
No.  <routine>
1  ca__gcn()
2  sub_main()
3  main()
```

Figure 5-76 ROUTE Window

Operation

- Function call sequence display area
  Displays the function call sequence in terms of the number and function name up to the function pointed to by the current program counter (PC). Up to 64 functions can be displayed.

- <Cancel> button
  Clicking the <Cancel> button closes the ROUTE window.

Function

- Function call sequence display
  Displays the function call sequence from the function pointed to by the current program counter to the initial call. The [ROUTE] command is effective only for C programs.

  However, if a return address of a function in this sequence does not point to another function, or if a function call in this sequence is written in assembly language, the call sequence is displayed from the function pointed to by the program counter up to the last function that can be detected. If interrupt processing is declared with a #pragma interrupt statement in C language, the function call sequence up to the declared function in which an interrupt occurred is displayed when an interrupt occurs in a C program. If an interrupt occurs in an assembly program, the function call sequence up to the interrupt processing function is displayed.

- Automatic function call sequence modification
  While the ROUTE window is open, information in the function call sequence display area is modified automatically when emulation is terminated in one of the following ways.

  — When the emulation initiated by pressing the <STEP>, <STEP_OVER>, <STEP_UP>, or <CONTINUE> button is terminated

  — When the emulation initiated by clicking the <Step>, <Step_over>, or <Continue> button of the DISASSEMBLE window is terminated

  — When the emulation initiated with the [Go...] command is terminated

  — When the emulation initiated by the STEP, STEP_OVER, or GO command entered in the command area is terminated

  — When emulation is forcibly terminated by clicking the <STOP> button

  If the program counter (PC) or stack pointer (SP) is modified after emulation termination, or if the instruction located at the start address of a function terminates emulation, the function call sequence cannot be displayed correctly in the function call sequence display area. In this case, executing one line of the program, for example, with the <STEP> button, will enable the function call sequence to be displayed correctly.

Note

To display a function call sequence correctly using the [ROUTE] command, the PR register must be initialized to -1 before executing the program.

Related Functions

<STEP>, <STEP_OVER>, <STEP_UP>, <CONTINUE>, and <STOP> buttons
GUI commands: [Go...] and [Disassemble]
Emulator commands: STEP, STEP_OVER, and GO

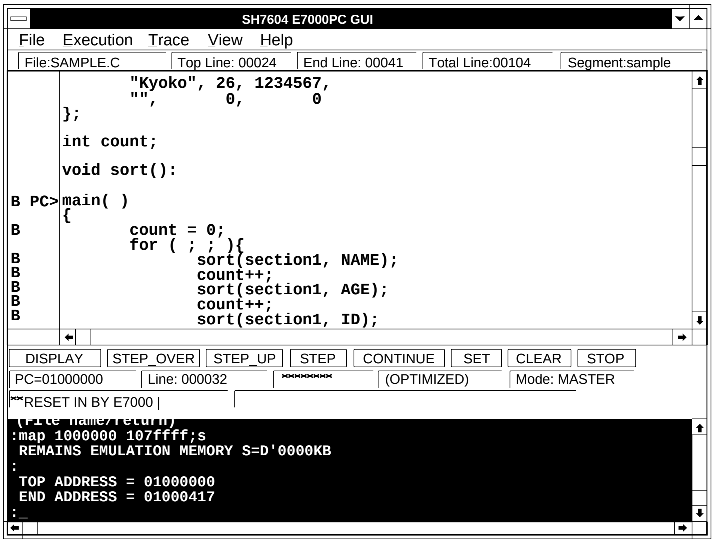
5.6.12 Controlling Command Area (Ctrl + F6)

[Command area]

Overview

Modifies the command area size of the base window.

Window



File:SAMPLE.C | Top Line: 00024 | End Line: 00041 | Total Line:00104 | Segment:sample

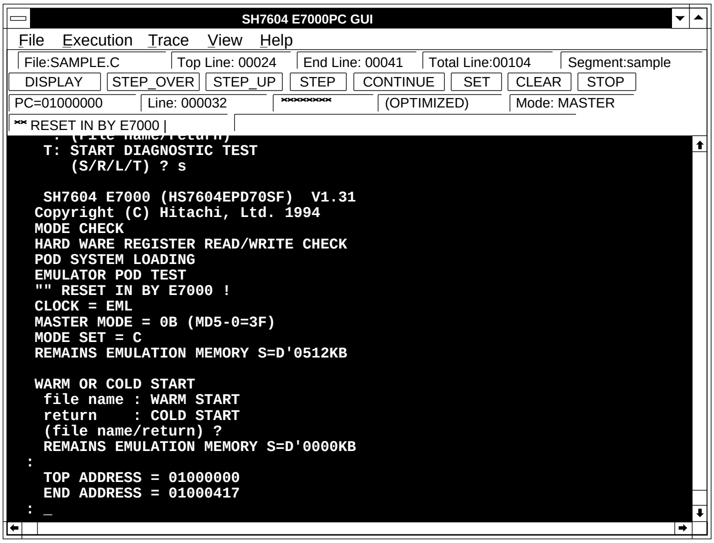
```
               "Kyoko", 26, 1234567,
               "",       0,       0
       };

       int count;

       void sort():

B PC>  main( )
       {
B              count = 0;
               for ( ; ; ){
B                      sort(section1, NAME);
B                      count++;
B                      sort(section1, AGE);
B                      count++;
B                      sort(section1, ID);
```

PC=01000000 | Line: 000032 | ******** | (OPTIMIZED) | Mode: MASTER

** RESET IN BY E7000 |

```
 (File name/return)
:map 1000000 107ffff;s
 REMAINS EMULATION MEMORY S=D'0000KB
:
 TOP ADDRESS = 01000000
 END ADDRESS = 01000417
:_
```

(a) small



File:SAMPLE.C | Top Line: 00024 | End Line: 00041 | Total Line:00104 | Segment:sample

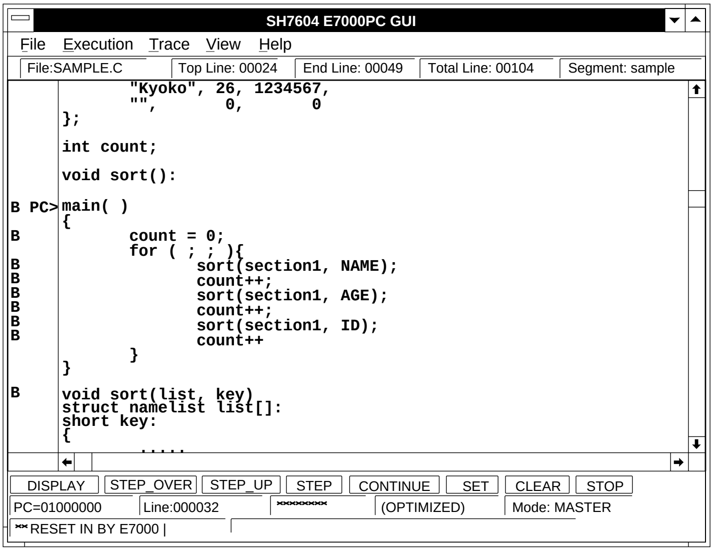
PC=01000000 | Line: 000032 | ******** | (OPTIMIZED) | Mode: MASTER

** RESET IN BY E7000 |

```
     : (file name/return)
    T: START DIAGNOSTIC TEST
       (S/R/L/T) ? s

    SH7604 E7000 (HS7604EPD70SF)  V1.31
   Copyright (C) Hitachi, Ltd. 1994
   MODE CHECK
   HARD WARE REGISTER READ/WRITE CHECK
   POD SYSTEM LOADING
   EMULATOR POD TEST
   "" RESET IN BY E7000 !
   CLOCK = EML
   MASTER MODE = 0B (MD5-0=3F)
   MODE SET = C
   REMAINS EMULATION MEMORY S=D'0512KB

   WARM OR COLD START
    file name : WARM START
    return    : COLD START
    (file name/return) ?
    REMAINS EMULATION MEMORY S=D'0000KB
  :
    TOP ADDRESS = 01000000
    END ADDRESS = 01000417
  : _
```

(b) full

Figure 5-77 Command Area Display



File:SAMPLE.C | Top Line: 00024 | End Line: 00049 | Total Line: 00104 | Segment: sample

```
               "Kyoko", 26, 1234567,
               "",       0,       0
        };

        int count;

        void sort():

B PC>   main( )
        {
B              count = 0;
               for ( ; ; ){
B                     sort(section1, NAME);
B                     count++;
B                     sort(section1, AGE);
B                     count++;
B                     sort(section1, ID);
B                     count++
               }
        }

B       void sort(list, key)
        struct namelist list[]:
        short key:
        {
                .....
```

PC=01000000 | Line:000032 | ×××××××× | (OPTIMIZED) | Mode: MASTER

×× RESET IN BY E7000 |

(c) none

Figure 5-77 Command Area Display (cont)

Operation

- Command area size modification

  The size of the command area in the base window can be modified to one of the following: (small), (full), or (none). The command area size cycles in the order of (small), (full), and (none). The current command area size is displayed on the menu in the command area of the [View] menu.

  (a) small

  Default size at system initiation. Both the source area and command area are displayed in the base window.

  (b) full

  The largest sized area. The source area is closed and is not assigned to the base window. This size is effective when using emulator commands.

  (c) none

  The command area is closed and the source area is enlarged in the base window. This size is effective in maximizing the acquisition of source file information, accomplished by closing the command area while emulator commands are not used.

5.7 Help Function

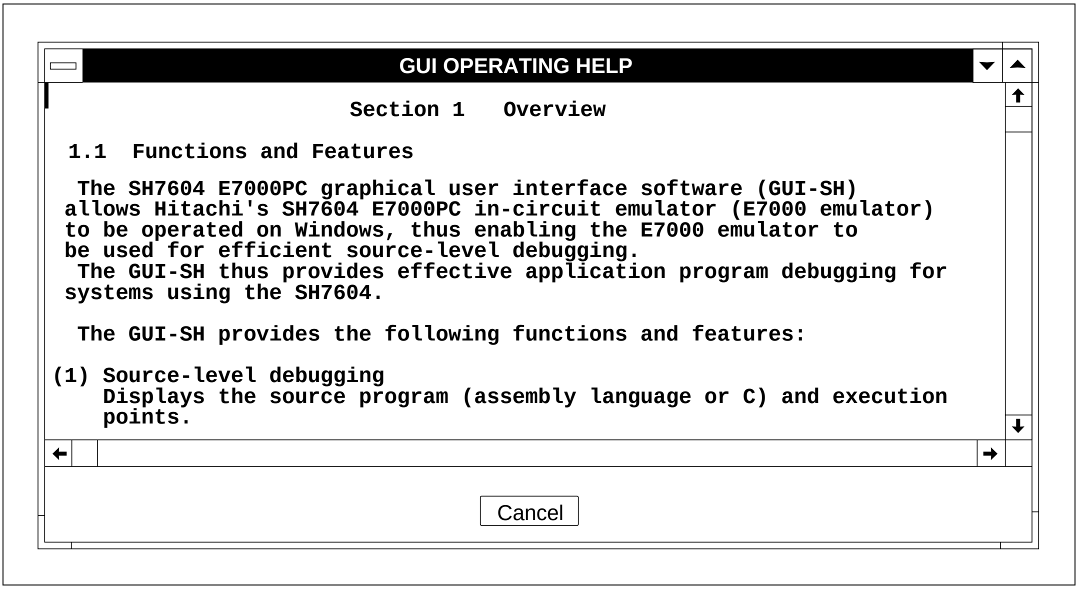
5.7.1 Displaying GUI Operating Help (F1)

[GUI operating help] GUI OPERATING HELP Window

Overview

Displays the GUI basic operations.

Window



```
                    Section 1   Overview

 1.1  Functions and Features

  The SH7604 E7000PC graphical user interface software (GUI-SH)
 allows Hitachi's SH7604 E7000PC in-circuit emulator (E7000 emulator)
 to be operated on Windows, thus enabling the E7000 emulator to
 be used for efficient source-level debugging.
  The GUI-SH thus provides effective application program debugging for
 systems using the SH7604.

  The GUI-SH provides the following functions and features:

(1) Source-level debugging
    Displays the source program (assembly language or C) and execution
    points.
```

Figure 5-78 GUI OPERATING HELP Window

Operation

- Help display area
  Displays the GUI basic operations.

- <Cancel> button
  Clicking the <Cancel> button closes the GUI OPERATING HELP window.

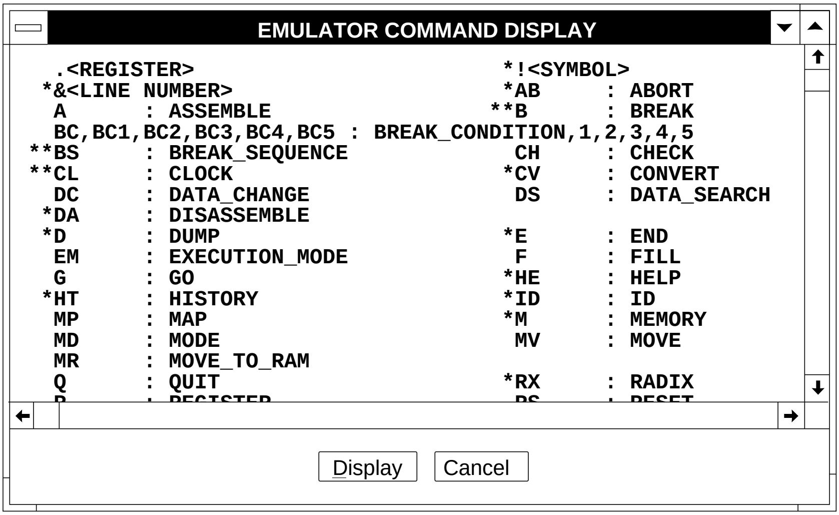
5.7.2 Displaying Emulator Commands (Alt + F1)

[Emulator command display] EMULATOR COMMAND DISPLAY Window

Overview

Displays emulator commands. Also displays help information on the specified command.

Window



```
   .<REGISTER>                          *!<SYMBOL>
  *&<LINE NUMBER>                       *AB     : ABORT
   A       : ASSEMBLE                  **B      : BREAK
   BC,BC1,BC2,BC3,BC4,BC5 : BREAK_CONDITION,1,2,3,4,5
 **BS      : BREAK_SEQUENCE              CH     : CHECK
 **CL      : CLOCK                      *CV     : CONVERT
   DC      : DATA_CHANGE                 DS     : DATA_SEARCH
  *DA      : DISASSEMBLE
  *D       : DUMP                       *E      : END
   EM      : EXECUTION_MODE              F      : FILL
   G       : GO                         *HE     : HELP
  *HT      : HISTORY                    *ID     : ID
   MP      : MAP                        *M      : MEMORY
   MD      : MODE                        MV     : MOVE
   MR      : MOVE_TO_RAM
   Q       : QUIT                       *RX     : RADIX
   R       : REGISTER                    RS     : RESET
```

Figure 5-79 EMULATOR COMMAND DISPLAY Window

Operation

- Emulator command display area
  Displays E7000PC emulator commands. Clicking the <Display> button after double-clicking the name or the abbreviation of a displayed emulator command, displays detailed information on that command.

- <Display> button
  Clicking the <Display> button after selecting a command name in the emulator command display area, displays the command input format in the EMULATOR COMMAND HELP window.

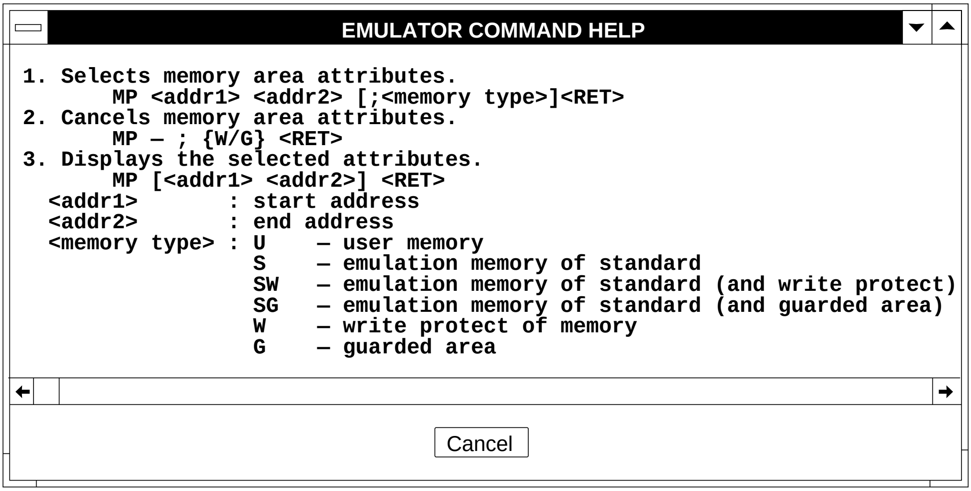
- <Cancel> button
  Clicking the <Cancel> button closes the EMULATOR COMMAND DISPLAY window.

Function

- Emulator command display
  Both the full command name and abbreviation are displayed for each command. Asterisks located before the abbreviation have the following meanings:

*:	Can be executed in parallel mode
**:	Can be displayed in parallel mode
No *:	Cannot be executed in parallel mode

- Emulator command input format display
  Input formats of emulator commands are displayed in the EMULATOR COMMAND HELP window shown in figure 5-80.



```
1. Selects memory area attributes.
       MP <addr1> <addr2> [;<memory type>]<RET>
2. Cancels memory area attributes.
       MP — ; {W/G} <RET>
3. Displays the selected attributes.
       MP [<addr1> <addr2>] <RET>
  <addr1>       : start address
  <addr2>       : end address
  <memory type> : U    — user memory
                  S    — emulation memory of standard
                  SW   — emulation memory of standard (and write protect)
                  SG   — emulation memory of standard (and guarded area)
                  W    — write protect of memory
                  G    — guarded area
```

Figure 5-80 EMULATOR COMMAND HELP Window

— Help display area
Displays the input format of the E7000PC emulator command specified in the EMULATOR COMMAND DISPLAY window.

— <Cancel> button
Clicking the <Cancel> button closes the EMULATOR COMMAND HELP window.

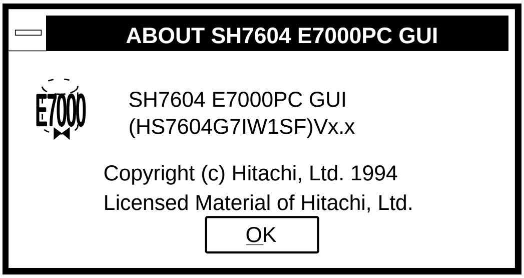
5.7.3 Describing GUI-SH (None)

[About GUI-SH...] ABOUT Window

Overview

Displays information on the GUI-SH.

Window



ABOUT SH7604 E7000PC GUI

SH7604 E7000PC GUI
(HS7604G7IW1SF)Vx.x

Copyright (c) Hitachi, Ltd. 1994
Licensed Material of Hitachi, Ltd.

Figure 5-81 ABOUT Window

Operation

- GUI-SH information display area
  Displays the product number, type number, version, copyright, and license of the GUI-SH.

- <Cancel> button
  Clicking the <Cancel> button closes the ABOUT window.

Appendix A E7000PC Emulator Command List

Table A-1 lists the E7000PC emulator commands and shows which commands can be executed in the command area.

Table A-1 E7000PC Emulator Commands

Command	Function	Usable /Unusable in Command Area
.<register>	Modifies and displays register contents	Usable
!<symbol> or &<symbol>	Displays symbol value	Usable
ABORT	Terminates emulation in parallel mode	Usable
ASSEMBLE	Assembles program one line at a time	Usable
BREAK	Sets, displays, and cancels PC breakpoints	Usable
BREAK_ CONDITION1,2,3,4,5	Sets, displays, and cancels hardware break conditions	Usable
BREAK_ SEQUENCE	Sets, displays, and cancels PC breakpoints with pass sequence specification	Usable
CHECK	Tests SH7604 pin status	Usable
CLOCK	Sets and displays clock	Usable
COMMAND_CHAIN	Inputs E7000PC commands from a file	Usable in INPUT command
CONVERT	Converts data	Usable
DATA_CHANGE	Replaces memory data	Usable
DATA_SEARCH	Searches for memory data	Usable
DISASSEMBLE	Disassembles and displays memory contents	Usable
END	Cancels parallel mode	Usable
EXECUTION_ MODE	Specifies and displays execution mode	Usable
FILL	Writes data to memory	Usable
GO	Executes realtime emulation	Usable
HELP	Displays all commands and command format	Usable

Table A-1 E7000PC Emulator Commands (cont)

Command	Function	Usable /Unusable in Command Area
HISTORY	Displays all commands input	Unusable
ID	Displays E7000PC reset message	Usable
MAP	Specifies and displays memory attributes	Usable
MEMORY	Displays and modifies memory contents	Usable
MODE	Specifies and displays the SH7604 operating mode	Usable
MOVE	Transfers memory contents	Usable
MOVE_TO_RAM	Moves ROM contents to emulation memory	Usable
PRINT	Sets or cancels output device for command result display	Unusable in E7000PC
QUIT	Terminates E7000PC system program	Usable
RADIX	Specifies and displays radix for numeric input	Usable
REGISTER	Displays register contents	Usable
RESET	Resets SH7604	Usable
RESULT	Displays execution results	Usable
SHORT_SYMBOL	Defines a short format for a symbol and displays current symbol definition	Unusable
STATUS	Displays E7000PC execution status	Usable
STEP	Performs single-step execution	Usable
STEP_ INFORMATION	Specifies and displays information during single-step execution	Usable
STEP_OVER	Performs single-step execution except for subroutines	Usable
SYMBOL	Defines, displays, and deletes symbols	Definition function is unusable

Table A-1 E7000PC Emulator Commands (cont)

Command	Function	Usable /Unusable in Command Area
TRACE	Displays trace buffer contents	Usable
TRACE_ CONDITION	Specifies, displays, and cancels trace conditions	Usable
TRACE_MEMORY	Specifies, displays, and cancels trace data address	Usable
TRACE_MODE	Specifies and displays trace information acquisition mode	Usable
TRACE_SEARCH	Searches for and displays trace information	Usable
HOST	Specifies and displays host system interface parameters	Unusable in E7000PC
LOAD	Loads program from host system — Transparent mode and local mode	Usable
SAVE	Saves program in host system — Transparent mode and local mode	Usable
TERMINAL	Transfers to terminal mode — Transparent mode	Unusable in E7000PC
TRANSFER	Transfers file to and from host system — Transparent mode and local mode	Unusable in E7000PC
VERIFY	Verifies memory contents against host system file — Transparent mode and local mode	Usable
INTFC_LOAD	Loads program from host system — Remote mode	Unusable in E7000PC
INTFC_SAVE	Saves program in host system — Remote mode	Unusable in E7000PC
INTFC_TRANSFER	Transfers file to and from host system — Remote mode	Unusable in E7000PC
INTFC_VERIFY	Verifies memory contents against host system file — Remote mode	Unusable in E7000PC
ASC	Specifies the transferred file type as ASCII	Unusable in E7000PC

BIN	Specifies the transferred file type as binary	Unusable in E7000PC
BYE	Terminates the FTP interface (Re-connection is performed by the OPEN command)	Unusable in E7000PC

Table A-1 E7000PC Emulator Commands (cont)

Command	Function	Usable /Unusable in Command Area
CD	Modifies the file directory name of the FTP server	Unusable in E7000PC
CLOSE	Disconnects the FTP interface (Re-connection is performed by the OPEN command)	Unusable in E7000PC
FTP	Connects via the FTP interface	Unusable in E7000PC
LAN	Displays E7000PC IP address	Unusable in E7000PC
LAN_HOST	Specifies, modifies, and displays the IP address of the host system connected via the FTP interface	Unusable in E7000PC
LAN_LOAD	Loads a program from the host system via the FTP interface	Unusable in E7000PC
LAN_SAVE	Saves a program in the host system via the FTP interface	Unusable in E7000PC
LAN_TRANSFER	Transfers a file to and from the host system connected via the FTP interface	Unusable in E7000PC
LAN_VERIFY	Verifies user system memory contents against the host system file connected via the FTP interface	Unusable in E7000PC
LS	Displays the host system directory connected via the FTP interface	Unusable in E7000PC
OPEN	Connects via the FTP interface	Unusable in E7000PC
PWD	Displays the current directory name of the host system connected via the FTP interface	Unusable in E7000PC
STA	Displays the file type to be transferred	Unusable in E7000PC
LOGOUT	Disconnects from the Telnet	Unusable in E7000PC

Limitations:

- The following E7000PC monitor commands cannot be input from the command area:
  L (Sets an E7000PC IP address)
  T (Loads and initiates the diagnostic program)

- It is not possible to re-display previously input command parameters by entering "<command name>.".

- Control codes other than (Ctrl + C) cannot be input from the command area. Accordingly, display cannot be controlled using control codes (TRACE, MEMORY, and DISASSEMBLE emulator commands).

- With the LOAD command, a GUI-SH option can be specified in addition to the function of the emulator. Figure A-1 shows the command syntax including the GUI-SH option specification.

```
LOAD [<offset>][;<load module type>]:<file name>[ <GUI-SH option>](Enter)
```

<GUI-SH option>: Option usable only in the GUI-SH when SYSROF (R) is specified as load module type.

A: Loads debugging information of the load module file to the GUI-SH and loads the load module file to the E7000PC (default).

E: Does not load debugging information of the load module file to the GUI-SH and loads the load module file to the E7000PC.

G: Loads debugging information of the load module file to the GUI-SH and does not load the load module file to the E7000PC.

Figure A-1 LOAD Command Syntax (Emulator Command)

For details on <offset>, <load module type>, and <file name>, refer to the SH7604 Emulator User's Manual.

- If the file specified with the SAVE command already exists, the file is automatically overwritten.

- The command file can be executed from the command area using the INPUT command. Figure A-2 shows the command syntax of the INPUT command.

```
INPUT <file name> (Enter)
```

<file name>: A name of a command file to be executed in the command area.

Figure A-2 INPUT Command Syntax

If the INPUT command is executed within the command file, an error occurs but command file execution continues.

Appendix B Error Messages

This section describes error messages displayed in the message box when an error occurs during GUI-SH operation or E7000PC emulator operation.

B.1 Error Message Levels

GUI-SH error messages are grouped into two types as follows:

1000–	:	Operation error messages Displayed during GUI-SH operation
2000–	:	System error messages Displayed when an error occurs in the system under which the GUI-SH is operating

B.2 List of Error Messages

This section describes error messages in detail. Each error message is described in the following format.

Error Message Description Format

Error No.	Error Message	Error Description (P) Solution

B.2.1 Operation Error Messages

Table B-1 Operation Error Messages

Error No.	Error Message	Error Description and Solution
1002	file name not specified	A file name is not specified. (P) Specify a file name.
1004	file not found	The specified file could not be found. (P) Specify a correct file name.
1005	File already exists. Overwrite ? <file name>	The specified file already exists. (P) Specify whether the file specified is to be overwritten or not.
1008	load module target error	The file specified in the LOAD frame is not an SH7604 load module. (P) Specify a correct file name.
1009	debug information too many	The number of defined symbols exceeds the limit (16,777,215) that can be defined in the GUI-SH. (P) Reduce the number of defined symbols to within the limit.
1011	symbol name not specified	No symbol is specified. (P) Specify a symbol.
1013	symbol not found	The specified symbol could not be found. (P) Specify a correct symbol.
1016	address not specified	No address is specified. (P) Specify an address.
1017	value not specified	No data is specified. (P) Specify data.
1019	condition not specified	No condition is specified. (P) Specify conditions.
1021	command line too long	The entered command line is too long. (P) Enter and specify the command line within 78 characters.
1022	can not use command in SH7604 E7000PC GUI	The command cannot be entered from the command area. (P) This command must not be entered from the command area.

Table B-1 Operation Error Messages (cont)

Error No.	Error Message	Error Description and Solution
1023	invalid operation in parallel mode	The operation attempted is not supported in emulation or in parallel mode. (P) The operation cannot be executed in emulation or in parallel mode. Stop program execution (change the prompt to ":").
1024	syntax error	An input or operation error has occurred. (P) Enter commands or other information correctly as specified in the manuals.
1025	invalid value	Invalid data is entered. (P) Enter correct data.
1026	invalid address	An invalid address is entered. (P) Enter a correct address.
1027	invalid token in command	The command syntax is invalid. (P) Enter a command with the correct syntax from the command area.
1029	modified data too much	Memory and symbol modification in the MEMORY DUMP and SYMBOL VALUE windows exceed 256. (P) Press the Enter key and start entering again from the 257th data item.
1030	can not routing	Function call sequence analysis cannot be performed in the ROUTE window. (P) Check the restrictions on function call sequence analysis described in section 5.6.11, Displaying Function Call Sequence.
1034	can not cancel	The window operation cannot be terminated. (P) If an error has occurred in the OUTPUT TO window, click the End button to terminate the file output.
1035	debug information not included	The load module's debugging information is not loaded in the GUI-SH. Thus, the SOURCE DISPLAY and ROUTE windows cannot be used. (P) Reload the load module program having debugging information.
1036	too many break point	The number of breakpoints specified in the BREAK dialog box exceeds 255. (P) Cancel a breakpoint already set to set the breakpoint desired.

Table B-1 Operation Error Messages (cont)

Error No.	Error Message	Error Description and Solution
1037	symbol assignment area too large	The assignment size of a symbol displayed in the SYMBOL VALUE window exceeds 20,000 bytes. (P) Reduce symbol assignment size to 20,000 bytes or less.
1038	invalid operation in sub-command mode	The operation cannot be performed when the prompt (?) is displayed in the command area. (P) Change the E7000PC emulator operating state (change the prompt from "?" to ":") from the command area.
1043	symbol name already exists	The specified symbol name has already been defined in the SYMBOL VALUE window. (P) To display existing symbol, close the SYMBOL VALUE window once. Reopen the SYMBOL VALUE window and select the symbol.
1046	internal management of GUI-SH is executed. Please waiting ...	GUI-SH internal processing is being executed. (P) Wait for completion of GUI-SH internal processing.
1047	effective symbol not found	The specified or input symbol is not defined or the specified symbol assignment area is a register. (P) The symbol cannot be entered or specified. Specify a value or another symbol.
1049	too many display symbol	The number of symbols displayed in the SYMBOL VALUE window exceeds 16 (maximum number). (P) No more symbols can be displayed. Terminate the SYMBOL VALUE window and redefine the symbol.
1051	create file error	The text in the window cannot be output. (P) The disk has no space left or is write-protected. Check the disk.
1052	command file open error	The command file cannot be opened. (P) Specify a correct command file.
1053	logging file open error	The logging file cannot be opened. (P) The disk has no space left or is write-protected. Check the disk.
1054	nothing with data	The specified window has no data to output to the file. (P) Specify a window that contains text data.

Table B-1 Operation Error Messages (cont)

Error No.	Error Message	Error Description and Solution
1056	segment name not specified	The <DISPLAY> button was clicked without a segment name specified in the segment information area of the SOURCE DISPLAY window. (P) Specify a segment name with the mouse and then click the <DISPLAY> button.
1057	double definition	The specified watch point has already been set. [P] Set another watch point in the WATCH SETTING window.
1058	can not step up	The <STEP_UP> button cannot be used when the current program counter (PC) is pointing to at this address. [P] Move the PC to an address where the <STEP_UP> button can be used. The <STEP_UP> button cannot be used for a ROM program.
1059	INPUT command in command file	The INPUT command was entered in a command file. [P] Do not enter the INPUT command.
1062	can not display this symbol value	The selected symbol cannot be displayed in the SYMBOL VALUE window. [P] Refer to restrictions in section 5.6.7, Displaying and Modifying Symbol Values, for details on undisplayable symbols.

B.2.2 System Error Messages

Table B-2 System Error Messages

Error No.	Error Message	Error Description and Solution
2006	file open error <file name>	The specified file cannot be opened. (P) A system error may have occurred. Check system operation.
2007	file read error <file name>	The specified file cannot be read. (P) A system error may have occurred. Check system operation.
2008	file write error <file name>	The specified file cannot be written to a disk. (P) The disk has no space left to write the file or a system error has occurred. Open up enough space for the file or check system operation.
2009	file close error <file name>	The specified file cannot be closed. (P) The disk has no space left to close the file or a system error has occurred. Open up enough space for the file or check system operation.
2010	can not get memory	There is not enough memory space to store symbol information during system operation. (P) Reduce the amount of debugging information for the load module or increase memory space.
2012	internal error	An internal error has occurred. (P) A system error may have occurred. Check system operation.
2013	allocate memory error	There is not enough memory for system operation. (P) Terminate other concurrent application(s) or increase the memory space available for Windows.
2014	source file open error	The source file corresponding to the load module cannot be opened. (P) Either load module is damaged or the source program has been moved to another directory since the load module was generated. Regenerate the load module.

Table B-2 System Error Messages (cont)

Error No.	Error Message	Error Description and Solution
2015	command file I/O error	An I/O error has occurred during command file input. (P) The command file may be damaged. Check the command file.
2016	logging file I/O error	An I/O error has occurred during logging file output. (P) The disk has no space left or is write-protected. Check the disk.
2017	interface error	An interface error has occurred between the PC and the E7000PC emulator (P) Check the interface.
2018	number of line is system overflow	The number of lines has exceeded the maximum allowable in the system. (P) Lines beyond number 32,767 cannot be referenced.

Appendix C   Command File

C.1 Command File Description

A file consisting of E7000PC emulator commands is called a command file. E7000PC emulator commands can be automatically executed by specifying a previously created command file with the [Input from...] or INPUT command.

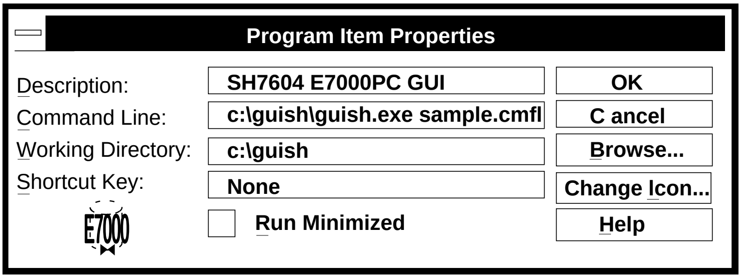
Figure C-1 shows an example of a command file. Command files can be created by inputting commands in the same format as input from the command area.

```
MAP 1000000 101FFFF;E
LOAD :SAMPLE.ABS
.PC 1000000
.R15 1020000
BREAK 100034C
GO
```

Figure C-1 Command File Example

C.2 Command File Execution at GUI-SH Initiation

If a command file name is specified on the command line at GUI-SH initiation, the command file can be executed automatically. Figure C-2 shows a command file name specification example. If icon modification is selected from the icon menu ([File]) in the program manager window, a dialog box as shown in figure C-2 is displayed. In the dialog box, a file name to be executed by the GUI-SH (guish.exe), a space, and a command file name (sample.cmf in figure C-2) are entered. After entering the above items, clicking the GUI-SH icon automatically initiates the GUI-SH and executes a command file. The automatic command file execution starts after the E7000 emulator system file has been loaded and a prompt is displayed in the command area.



Description: SH7604 E7000PC GUI

Command Line: c:\guish\guish.exe sample.cmfl

Working Directory: c:\guish

Shortcut Key: None

□ Run Minimized

Figure C-2 Command File Execution at GUI-SH Initiation

Appendix D Initial Setting File (guish.ini)

The GUI-SH reads the initial setting file (guish.ini) at initiation, and operates according to the contents of the file.

By modifying part of the file contents, the GUI-SH operating environment can be changed.

Figure D-1 shows the contents of the initial setting file (guish.ini). While the GUI-SH is operating, part of the contents is automatically modified.

```
    [GUISH]

    backupfile=
    gihelpfile=guishop.hlp

    baseaddr=XXXXX
```

Figure D-1 Initial Setting File (guish.ini) Contents

- backupfile=
  When a backup file exists, the file name is set by the absolute path name. When the setting information for debugging is stored with the [QUIT...] command, the GUI-SH automatically sets the file name. The user must not change this setting.
  When no backup file exists, no file name is set.

- gihelpfile=guishop.hlp
  The name of the help file (guishop.hlp), which is displayed with the [GUI operating help] command. The help file must be in the directory where the GUI-SH is installed. For details of the install directory, refer to section 2.4, Executing Installer Program.

- baseaddr=XXXXX
  The starting address of the memory address range assigned for the IBM-PC interface board. If no address is specified, the installer automatically searches for the starting address. Accordingly, there is usually no need to set the address, but when using two or more IBM-PC interface boards, the user must set the address.
  The address must be set as follows:

  Example:

  When the memory address range for the IBM-PC interface board is D000:0000 to D3FF:000F

  baseaddr=D0000

  To directly set the address range, set a memory address range with the EMMExclude in the [386Enh] section of the SYSTEM.INI file. For details, refer to section 2.5, System File Modification.