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

RENESAS
Renesas Technology Corp.

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

SH7729/SH7709A E8000 Emulator HS7729EDD81H

User's Manual

Renesas Microcomputer
Development Environment
System

Renesas Electronics
www.renesas.com

Rev.1.0 2000.09

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi’s or any third party’s patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party’s rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi’s sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi’s sales office for any questions regarding this document or Hitachi semiconductor products.

IMPORTANT INFORMATION

READ FIRST

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

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

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

Emulator Product:

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

- Emulator station
- Device control board
- Evaluation chip board
- Cable

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

Purpose of the Emulator Product:

This emulator product is a software and hardware development tool for systems employing the Hitachi microcomputer HD6417729 or HD6417709A (hereafter referred to as MCU). By exchanging the device control board and evaluation chip board, this emulator product can also be used for systems using other microcomputers. This emulator product must only be used for the above purpose.

Limited Applications:

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

Improvement Policy:

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

Target User of the Emulator Product:

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

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

LIMITED WARRANTY

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

DISCLAIMER

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

State Law:

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

The Warranty is Void in the Following Cases:

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

All Rights Reserved:

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

Other Important Things to Keep in Mind:

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

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

Figures:

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

Limited Anticipation of Danger:

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

SAFETY PAGE

READ FIRST

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

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

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

DEFINITION OF SIGNAL WORDS

⚠

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

⚠ DANGER

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

⚠ WARNING

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

⚠ CAUTION

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

CAUTION

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

NOTE emphasizes essential information.

⚠ WARNING

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

1. **Carefully handle the emulator product to prevent receiving an electric shock because the emulator product has a DC power supply. Do not repair or remodel the emulator product by yourself for electric shock prevention and quality assurance.**
2. **Always switch OFF the emulator and user system before connecting or disconnecting any CABLES or PARTS.**
3. **Always before connecting, make sure that pin 1 on both sides are correctly aligned.**
4. **Supply power according to the power specifications and do not apply an incorrect power voltage. Use only the provided AC power cable. Use only the specified type of fuse.**

Warnings on Emulator Usage

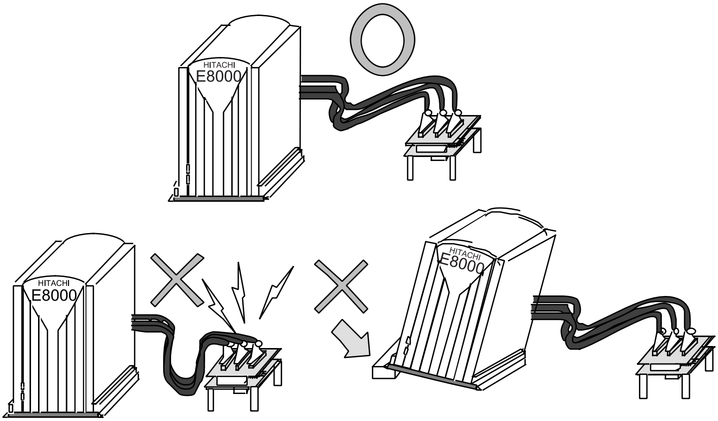
Warnings described below apply as long as you use this emulator. Be sure to read and understand the warnings below before using this emulator. Note that these are the main warnings, not the complete list.

⚠ **WARNING**

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

CAUTION

Place the emulator station and EV-chip board so that the trace cables are not bent or twisted. A bent or twisted cable will impose stress on the user interface leading to connection or contact failure. Make sure that the emulator station is placed in a secure position so that it does not move during use nor impose stress on the user interface.



CAUTION

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

Preface

Thank you for purchasing the emulator for the Hitachi microcomputer SH7729/SH7709A.

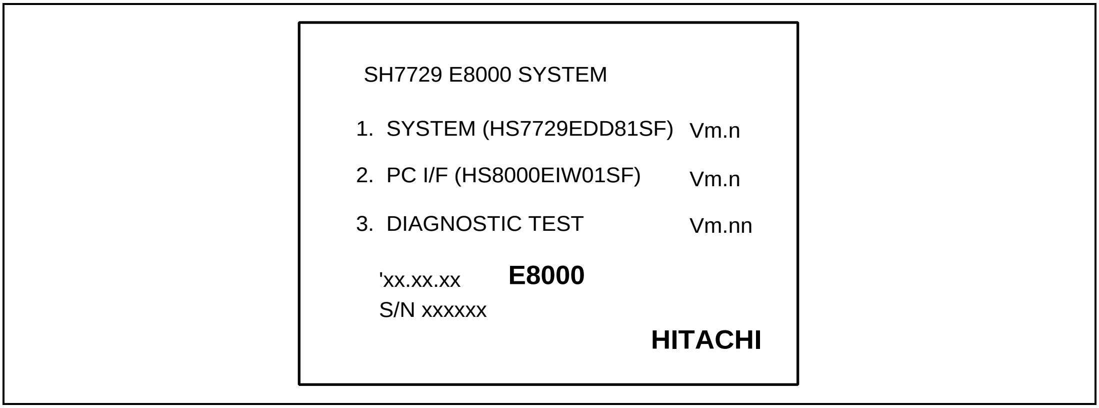
CAUTION

Read section 3, Preparation before Use in Part I, E8000 Guide of this user's manual before using the emulator product. Incorrect operation will damage the user system, the emulator product, and the user program.

The emulator is an efficient software and hardware development tool for systems based on Hitachi microcomputer SH7729/SH7709A. By exchanging the device control board and the evaluation chip board, this emulator can also be used for systems using other microcomputers.

This manual describes the emulator functions and operations. Please read this manual carefully in order to gain a full understanding of the emulator's performance. In particular, be sure to read section 1.2, Warnings, in Part I, E8000 Guide.

A 3.5-inch system floppy disk in PC 1.44-MB format is packaged together with the evaluation chip board.



SH7729 E8000 SYSTEM

1. SYSTEM (HS7729EDD81SF) Vm.n
2. PC I/F (HS8000EIW01SF) Vm.n
3. DIAGNOSTIC TEST Vm.nn

'xx.xx.xx **E8000**
S/N xxxxxx

HITACHI

Figure E8000 System Disk

Before using the system disk, back up it to a floppy disk according to the instructions in the manuals of the personal computer and the operating system.

Install (copy) the system disk to the personal computer connected to the emulator. For details on the copy procedure, refer to section 3.7, System Program Installation in Part I, E8000 Guide.

Related Manuals:

HS7729EBH81H Manual
LAN Board Manual
Description Notes on Using the PC Interface Board (HS6000EII01H)
Description Notes on Using the PCI Interface Board (HS6000EIC01H)
Description Notes on Using the PC Card Interface (HS6000EIP01H)
Description Notes on Using the LAN Adapter (HS6000ELN01H)
SuperH RISC engine C/C++ Compiler User's Manual
SH Series Cross Assembler User's Manual
H Series Linkage Editor, Librarian, Object Converter User's Manual
Integration Manager User’s Manual
SH7729/SH7709A E8000 Hitachi Debugging Interface User's Manual

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

2. Ethernet is a registered trademark of Xerox Corporation.

3. Microsoft®, Windows®, and MS-DOS are registered trademarks of Microsoft Coroporation in the United States and/or in other countries.

Contents

Part I E8000 Guide

Part II Emulator Function Guide

Part III Appendix

Figures

Tables

Part I E8000 Guide

Section 1 Overview

1.1 Overview

This system is an efficient software and hardware development support tool for application systems using the HD6417729 (SH7729) or HD6417709A (SH7709A) (hereafter referred to as MCU) microcomputer developed by Hitachi, Ltd.

The MCU incorporates the cache memory, the internal X/Y memory (SH7729 only), memory management unit (MMU), the timer required for system configuration, the realtime clock, the interrupt controller, the user break controller, the bus state controller, the internal DMAC (direct memory access controller), the serial communication interface (SCI), the A/D converter, the D/A converter, etc. as the peripheral function other than the high-speed CPU and the digital signal processing (DSP) expanded functions (SH7729 only).

The emulator operates in place of the MCU and performs realtime emulation of the user system. The emulator also provides functions for efficient hardware and software debugging.

The emulator consists of an emulator (E8000) station, a device control board, and an evaluation chip board. The evaluation chip board is connected to the user system via an IC socket (figure 1.1).

By connecting a PC interface board (option; each for ISA bus, PCI bus, or PCMCIA bus) to the host computer, SH7729/SH7709A E8000 Hitachi debugging interface (hereafter referred to as HDI) can be used. LAN adapter (option) or LAN board (option) are connected to the emulator via network. Connecting the LAN adapter enables debugging using the HDI (option). For details on PC interface boards (each for ISA bus, PCI bus or PCMCIA bus specifications) and LAN adapter, refer to their description notes.

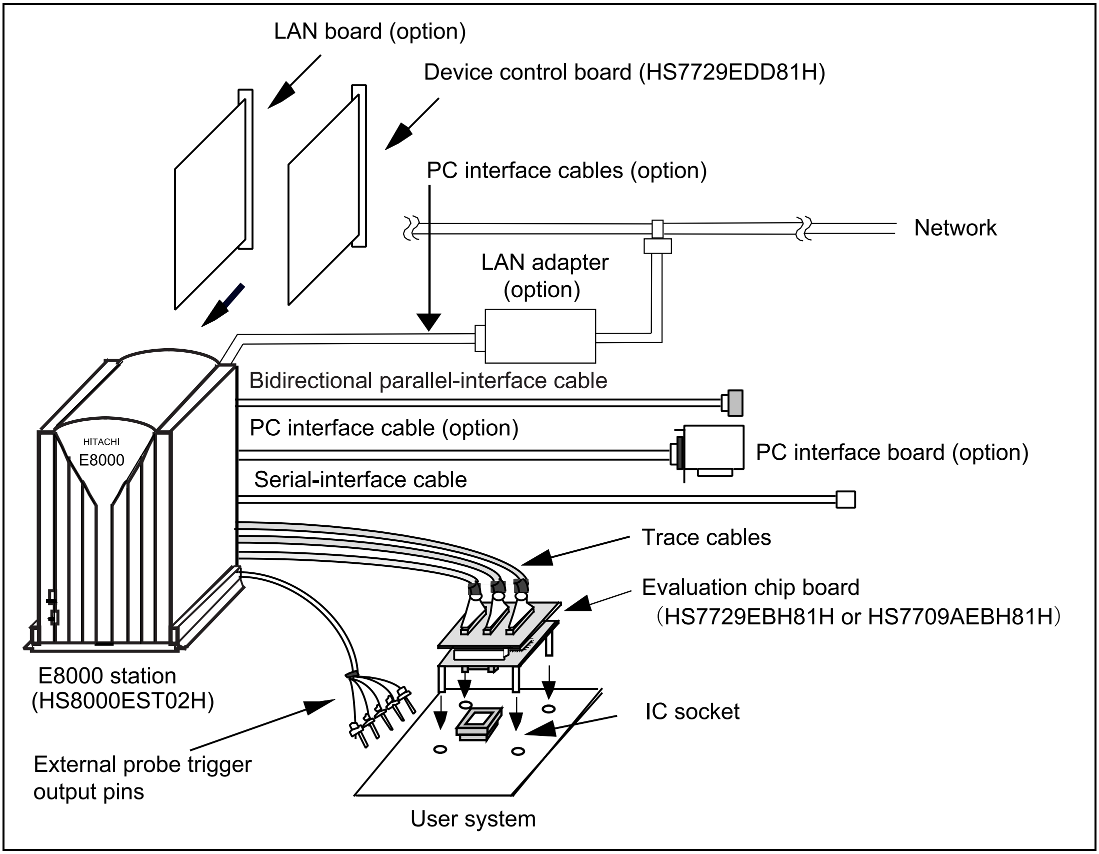


Figure 1.1 Emulator for the SH7729/SH7709A (Connected via the IC Socket)

The emulator provides the following features:

- Realtime emulation of the MCU at 66 MHz
- A wide selection of emulation commands, promoting efficient system development
- On-line help functions to facilitate command usage without a manual
- Efficient debugging enabled by variable break functions and a mass-storage trace memory (128 kilo-cycles)
- Command execution during emulation, for example
  - Trace data display
  - Emulation memory display and modification
- Measurement of subroutine execution time and count for evaluating the execution efficiency of user programs
- 8-Mbyte standard emulation memory for use as a substitute user-system memory
- An optional LAN board for interfacing with workstations, enabling high-speed downloading (12 Mbytes/min) of user programs

  The LAN board contains Ethernet (10BASE5) and Cheapernet (10BASE2) interfaces.
- An optional LAN adapter for connecting to the emulator through 10BASE or 100BASE interface, enabling the SH7729/SH7709A E8000 Hitachi Debugging Interface (option) can be loaded into the PC to enable:
  - Graphic display operations in a multi-window environment
  - Source-level debugging
- A PC interface board (for the ISA bus, PCI bus, or PCMCIA bus) connected to the host computer through the PC interface cable, enabling high-speed downloading (12 Mbytes/min) of user programs. The SH7729/SH7709A E8000 Hitachi Debugging Interface (option) can be loaded into the PC to enable:
  - Graphic display operations in a multi-window environment
  - Source-level debugging

1.2 Warnings

CAUTION

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

1. Check all components with the component list after unpacking the emulator.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the emulator is to be used:
   - Make sure that the internal cooling fans on the sides of the E8000 station must be at least 20 cm (8") away from walls or other equipment.
   - Keep out of direct sunlight or heat. Refer to section 1.3, Environmental Conditions.
   - Use in an environment with constant temperature and humidity.
   - Protect the emulator from dust.
   - Avoid subjecting the emulator to excessive vibration. Refer to section 1.3, Environmental Conditions.
4. Protect the emulator from excessive impacts and stresses.
5. Before using the emulator's power supply, check its specifications such as power output, voltage, and frequency. For details of the power supply, refer to section 1.3, Environmental Conditions.
6. When moving the emulator, take care not to vibrate or otherwise damage it.
7. After connecting the cable, check that it is connected correctly. For details, refer to section 3, Preparation before Use.
8. Supply power to the emulator and connected parts after connecting all cables. Cables must not be connected or removed while the power is on.
9. For details on differences between the MCU and the emulator, refer to section 2, Differences between the MCU and the Emulator in Part II, Emulator Function Guide.

1.3 Environmental Conditions

CAUTION

The following environmental conditions must be satisfied when using the emulator. Failure to do so will damage the user system and the emulator. The USER PROGRAM will be LOST.

Observe the conditions listed in table 1.1 when using the emulator.

Table 1.1 Environmental Conditions

Item	Specifications	
Temperature	Operating:	+10 to +35°C
	Storage:	–10 to +50°C
Humidity	Operating:	35 to 80% RH, no condensation
	Storage:	35 to 80% RH, no condensation
Vibration	Operating:	2.45 m/s^2 max.
	Storage:	4.9 m/s^2 max.
	Transportation:	14.7 m/s^2 max.
AC input power	Voltage:	AC100-120 V/200-240 V ± 10%
	Frequency:	50/60 Hz
	Power consumption:	200 VA
Ambient gases	There must be no corrosive gases present.	

1.4 Components

The emulator consists of the E8000 station, device control board, and evaluation chip board. Check all components after unpacking. If any component is missing, contact the sales agency from which the emulator was purchased.

1.4.1 E8000 Emulator Station

Table 1.2 lists the E8000 station components.

Table 1.2 E8000 Station Components (HS8000EST02H)

Classification	Item	Quantity	Remarks
Hardware	E8000 station	1	Power supply, control board, and trace board are installed
	Trace cable	3	Length: 50 cm
	AC power cable	1	
	Serial-interface cable	1	RS-232C interface
	Parallel-interface cable	1	Conforms to IEEE-P1284
	External probe	1	
	Fuse	1	Spare (3 A or T3.15A corresponding to CE marking)
Manual	HS8000EST02H description notes	1	HS8000EST02HE

1.4.2 Device Control Board and Evaluation Chip Board

Tables 1.3 and 1.4 list the device control board and evaluation chip board components. For details, refer to each user’s manual.

Table 1.3 Device Control Board Components

Classification	Item	Quantity	Remarks
Hardware	Device control board	1	One board, to be installed in the E8000 station
	External probe	1	Probe input: 4 Run/break state output: 1 Trigger output: 1 GND: 2
Software	3.5-inch floppy disk	1	E8000 system program
Manual	HS7729EDD81H description notes	1	HS7729EDD81HE

Table 1.4 Evaluation Chip Board Components for the SH7729

Item	Product No.	Quantity	Remarks
Evaluation chip board	HS7729EBH81H	1	User interface: IC socket (QFP208)
HS7729EBH81H description notes	HS7729EBH81HE	1	

Table 1.5 Evaluation Chip Board Components for the SH7709A

Item	Product No.	Quantity	Remarks
Evaluation chip board	HS7709AEBH81H	1	User interface: IC socket (QFP208)
HS7709AEBH81H description notes	HS7709AEBH81HE	1	

1.4.3 Options

In addition to the E8000 station and Evaluation chip board components, the options listed in table 1.5 are also available. Refer to each option manual for details on these optional components.

Table 1.6 Optional Component Specifications

Item	Model Name	Specifications
LAN board	HS7000ELN01H or HS7000ELN02H	• TCP/IP communications protocol • Ethernet (10BASE5) • Cheapernet (10BASE2) Only command line interface can be used. When using a router or a gateway, use HS7000ELN02H.
PC interface board	HS6000EII01H	ISA bus Can be used only under the HDI interface environment
PCI interface board	HS6000EIC01H	PCI bus Can be used only under the HDI interface environment
PCMCIA card	HS6000EPI01H	PCMCIA bus Can be used only under the HDI interface environment
LAN adapter	HS6000ELN01H	• TCP/IP communications protocol • 10BASE-T • 100BASE-Tx Can be used only under the HDI interface environment

Section 2 Components

2.1 Emulator Hardware Components

The emulator consists of an E8000 station, an SH7729/SH7709A device control board, and an SH7729/SH7709A evaluation chip board, as shown in figure 2.1. By installing a PC interface board (option; ISA bus, PCI bus, or PCMCIA bus), or a LAN adapter, the Hitachi debugging interface (HDI) can be used for debugging. By installing a LAN board (option) or a LAN adapter, the emulator can be connected to a workstation via the LAN interface. For details on the PC interface board (option; ISA bus, PCI bus, or PCMCIA bus specifications) and the LAN adapter, refer to the description notes on each product.

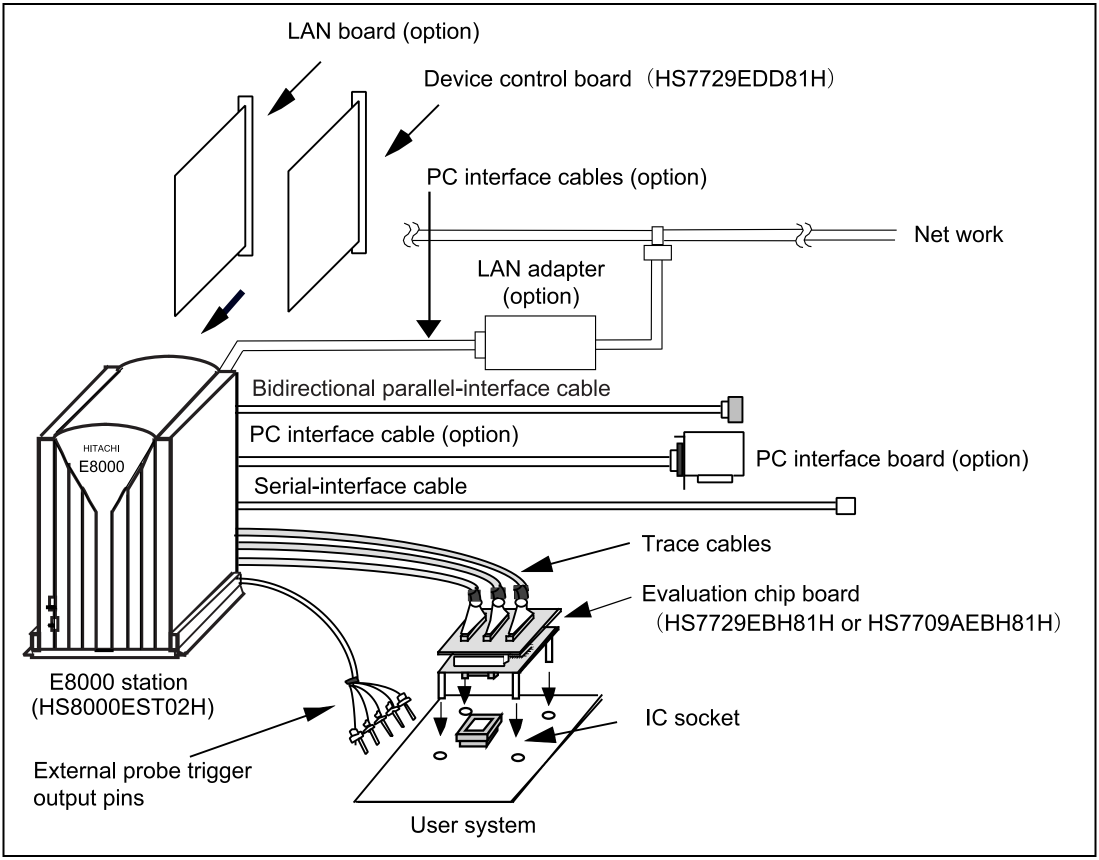

LAN board (option)
Device control board（HS7729EDD81H）
PC interface cables (option)
Net work
LAN adapter (option)
Bidirectional parallel-interface cable
PC interface cable (option)
PC interface board (option)
Serial-interface cable
HITACHI
E8000
Trace cables
Evaluation chip board （HS7729EBH81H or HS7709AEBH81H）
E8000 station (HS8000EST02H)
IC socket
External probe trigger output pins
User system

Figure 2.1 Emulator Hardware Components

2.1.1 E8000 Station Components

Front Panel:

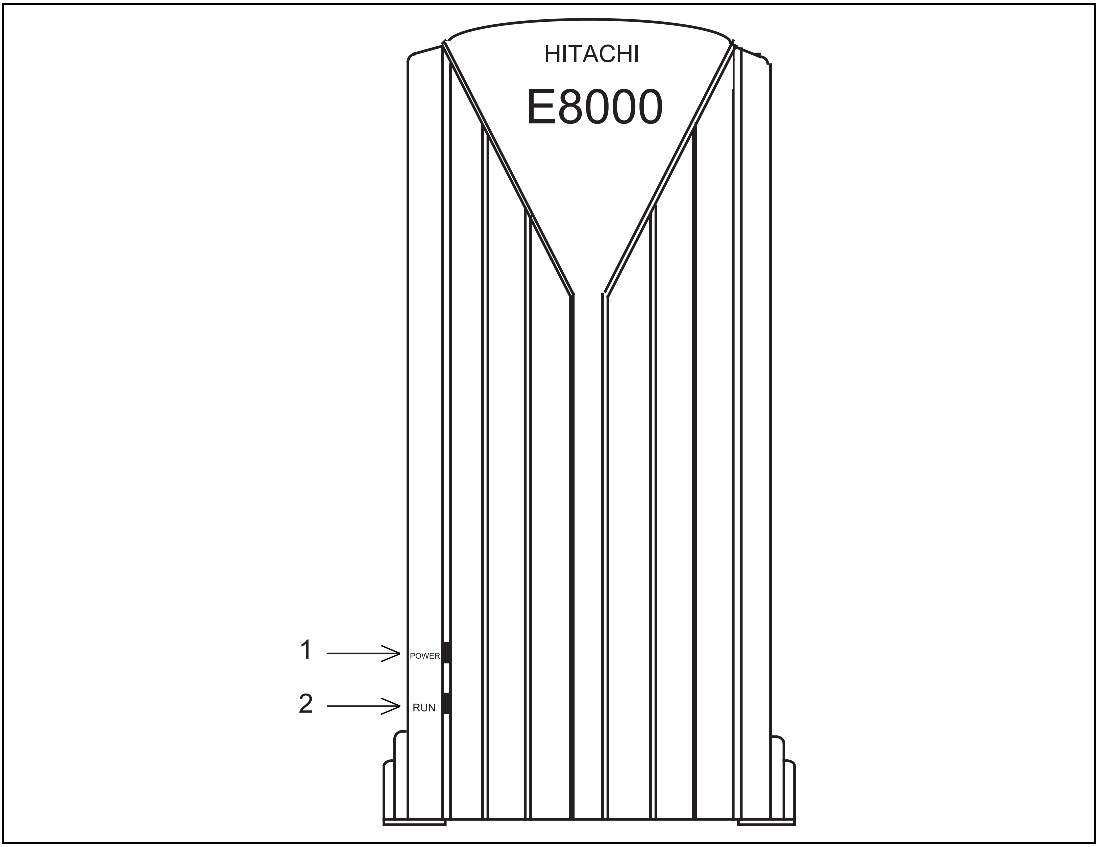


Figure 2.2 E8000 Station Front Panel

1. POWER lamp: Is lit up when the E8000 station power is on.
2. RUN lamp: Is lit up when the user program is running.

Rear Panel:

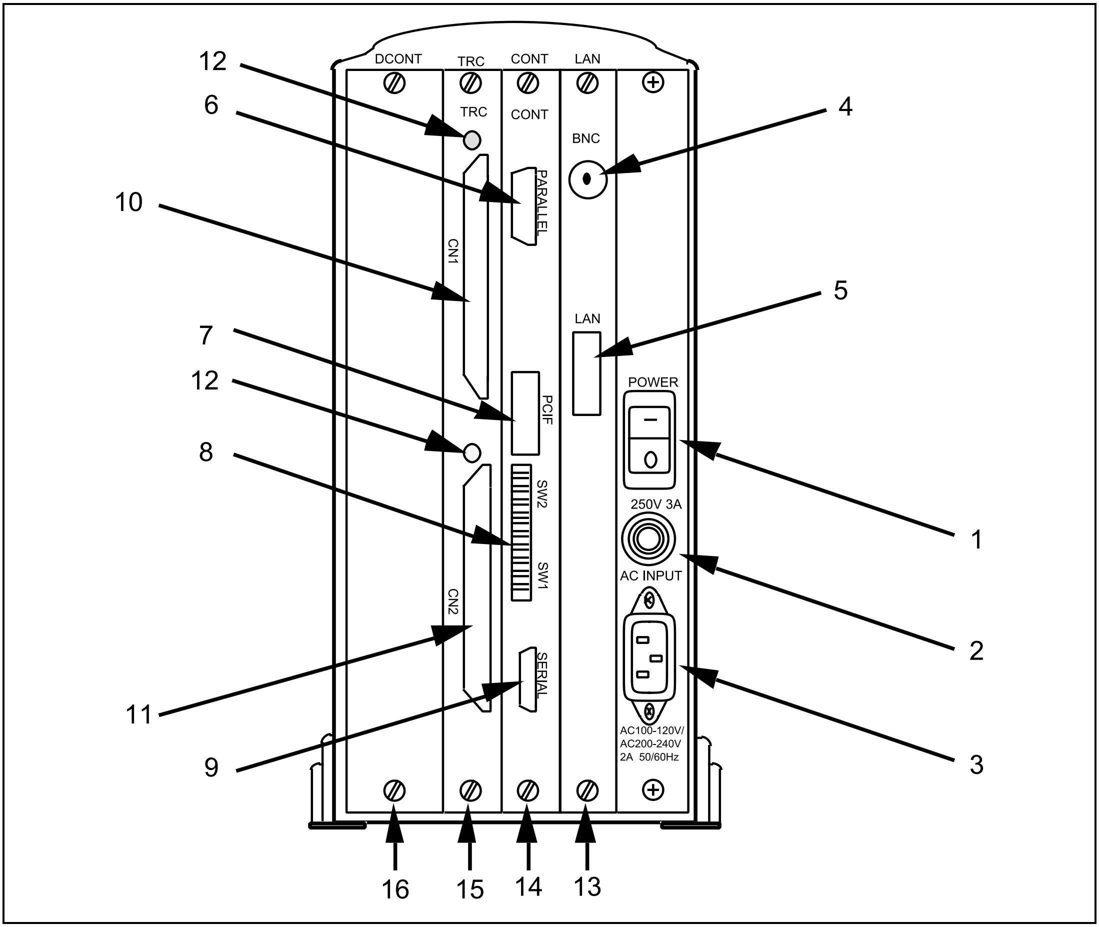


Figure 2.3 E8000 Station Rear Panel

1. Power switch: Turning this switch to I (input) supplies power to the emulator (E8000 station and evaluation chip board).
2. Fuse box: Contains a 3-A 250-V AC fuse.
3. AC power connector: For a AC100-120 V/200-240 V power supply.
4. Cheapernet connector: For a Cheapernet cable. Marked BNC.
5. Ethernet connector: For an Ethernet cable. Marked LAN.
6. Parallel-interface connector: For a parallel-interface cable with the PC interface board. Conforms to IEEE-P1284 (ECP mode). Marked PARALLEL.
7. PC interface cable connector: For the PC interface cable which connects the host computer to the E8000 station. A PC interface board (ISA bus, PCI bus, or PCMCIA bus) or LAN adapter can be connected. Marked PCIF.
8. Host interface switches: For selecting the host interface. Specifies the connection of the LAN interface, RS-232C interface, or PC interface board. When the RS-232C interface is used, the data-bit length, stop-bit length, or parity-setting transfer rate can be switched. Marked SW1 and SW2.
9. Serial-interface connector: For RS-232C communication with a host computer. Marked SERIAL.
10. Station to evaluation chip board interface connector CN1: For trace cable 1 which connects the E8000 station to the evaluation chip board.
11. Station to evaluation chip board interface connector CN2: For trace cable 2 which connects the E8000 station to the evaluation chip board.
12. Trace cable mis-insertion inhibiting seal: Prevents a trace cable from being inserted into the wrong place.
13. LAN-board slot: For installing the optional LAN board.
14. Control board slot: For installing the control board.
15. Trace board slot: For installing the trace board.
16. Device control board slot: For installing the device control board (depends on the target device).

2.1.2 Device Control Board Components

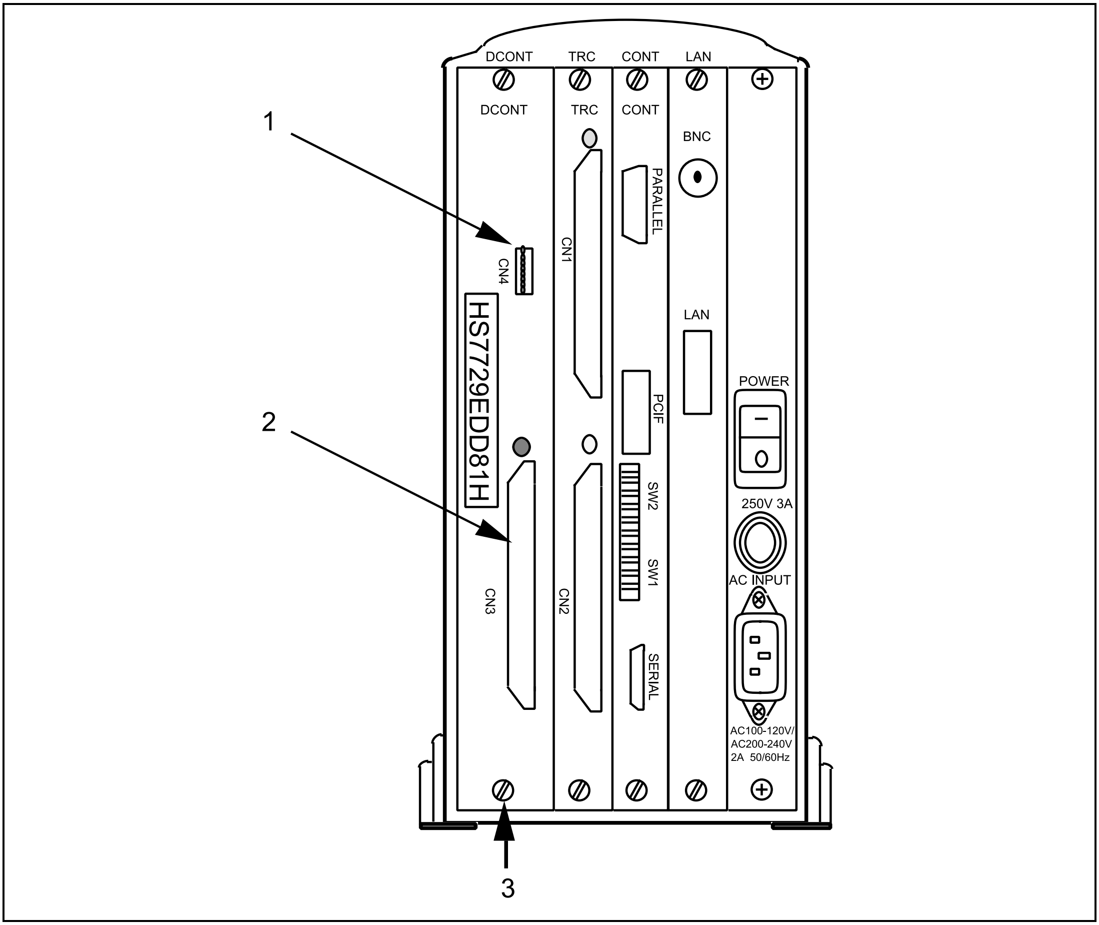


Figure 2.4 Device Control Board

1. External probe connector CN4: For connecting to the external probe.
2. Station to evaluation chip board interface connector CN3: For trace cable 3 which connects the E8000 station to the evaluation chip board.
3. Device control board slot: For installing the device control board (depends on the target device).

2.1.3 Evaluation Chip Board Configuration

The name of each part in the evaluation chip board (HS7729EBH81H or HS7709AEBH81H) is shown below.

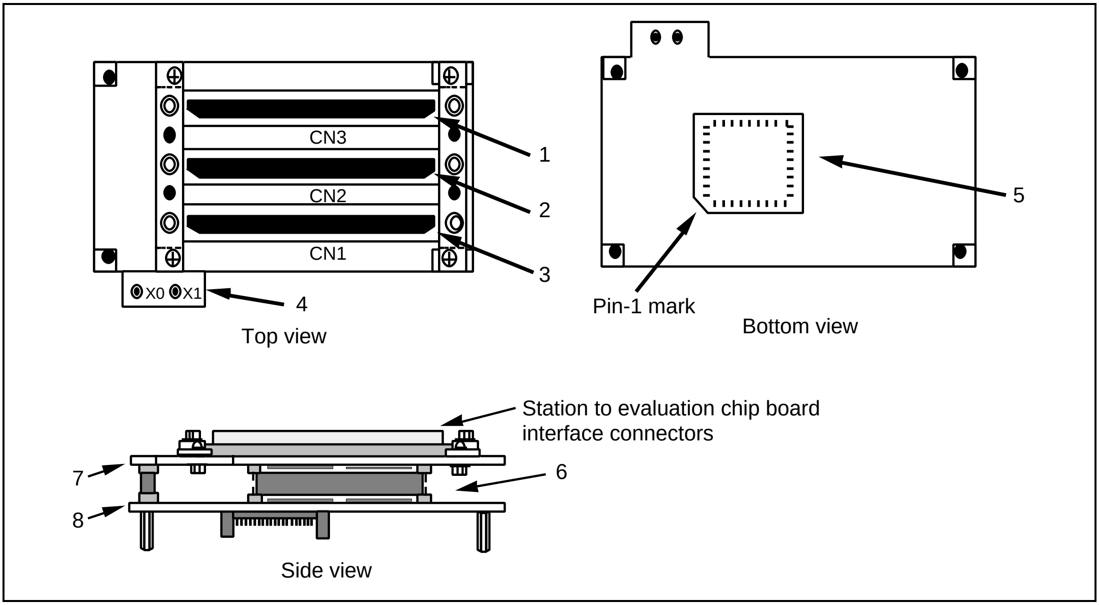


Figure 2.5 Evaluation Chip Board (HS7729EBH81H or HS7709AEBH81H)

1. Station to evaluation chip board interface connector CN3: For trace cable 3 which connects the emulator to the evaluation chip board.
2. Station to evaluation chip board interface connector CN2: For trace cable 2 which connects the emulator to the evaluation chip board.
3. Station to evaluation chip board interface connector CN1: For trace cable 1 which connects the emulator to the evaluation chip board.
4. Crystal oscillator terminals: For installing a crystal oscillator to be used as an external clock source for the MCU.
5. User system or user system interface cable connector: For connecting the user system or user system interface cable.
6. Board connector: For connecting HS7410PWB20H and HS7729PWB30H or HS7709APW30H.
7. HS7410PWB20H: Includes connectors for interfacing with the E8000 station.

8. HS7729PWB30H*[1]: SH7729 is incorporated. The IC socket to connect to the user system is installed.

9. HS7709APWB30H*[2]: SH7709A is incorporated. The IC socket to connect to the user system is installed.

Notes:

1. The generic term for items 1 to 8 above is HS7729EBH81H.
2. The generic term for items 1 to 7 and 9 above is HS7709AEBH81H.

2.2 Emulator Software Components

The emulator's software components are illustrated in figure 2.6. The evaluation chip board contains a 3.5-inch floppy disk. The system disk files are described in table 2.1.

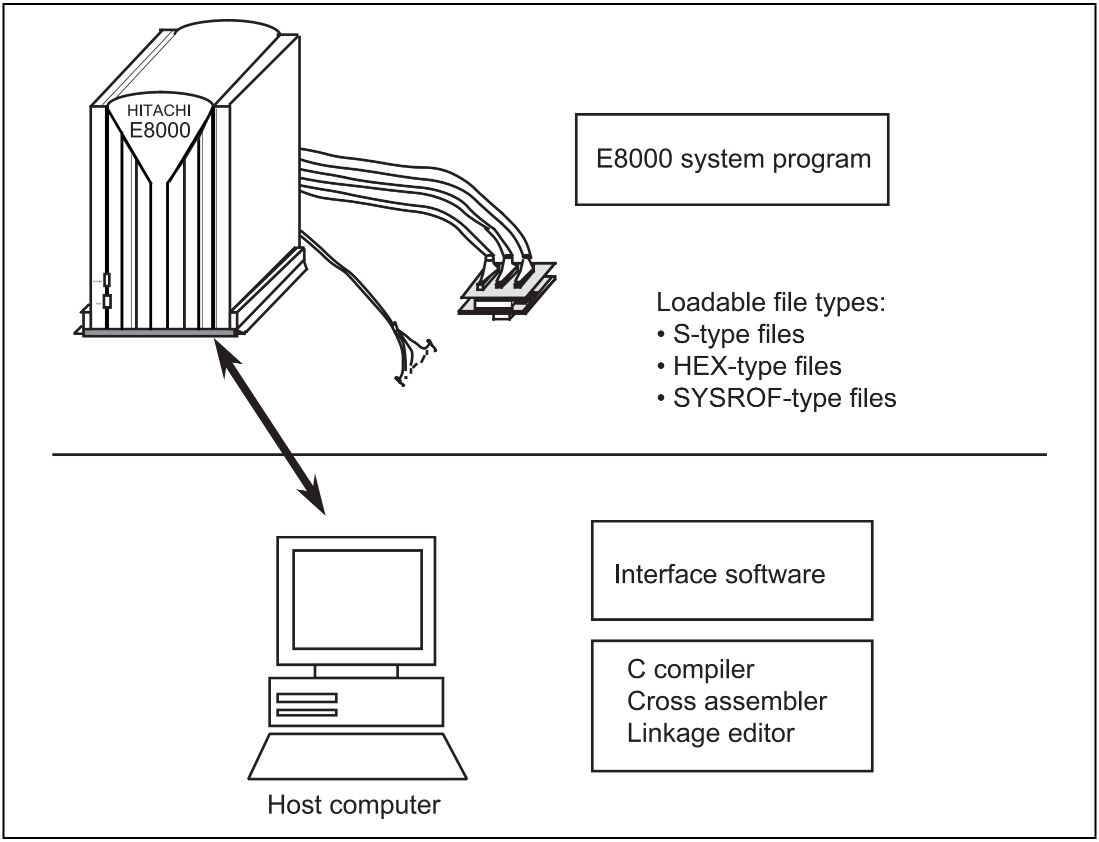


Figure 2.6 Emulator Software Components

Table 2.1 Contents of E8000 System Disk

File Name	Contents	Description
E8000.SYS	E8000 system program	Controls the evaluation chip board and processes commands, such as emulation commands. Loaded into the emulator memory.
SHDCT729.SYS	SH7729 control program	Controls the MCU in the evaluation chip board. Loaded into the emulator memory.
SHCNF729.SYS	Configuration file	Contains MCU operating mode and MAP information.
IPW.EXE	Interface program	Operates on the Microsoft® Windows® 3.1 and Windows® 95 of the host computer, and communicates with the emulator.
DIAG.SYS	Diagnostic program	Loaded into the emulator station memory for testing and maintenance.
SETUP.CC*	Load file	Loads files E8000.SYS, SHDCT729.SYS, and SHCNF729.SYS to the emulator memory.

Note: See section 3.7, System Program Installation.

2.3 System Configuration

The E8000 station can be connected to the host computer (a personal computer or a workstation) via a LAN interface (optional LAN board or LAN adapter) when a workstation is connected, and an RS-232C interface (optional PC interface board) when a personal computer is connected.

2.3.1 System Configuration Using a LAN Interface

By installing an optional LAN board in the E8000 station, the emulator can communicate with a workstation using a LAN interface. The LAN board contains connectors for both Cheapernet (10BASE2) and Ethernet (10BASE5). The system configuration using a LAN interface is shown in figure 2.7.

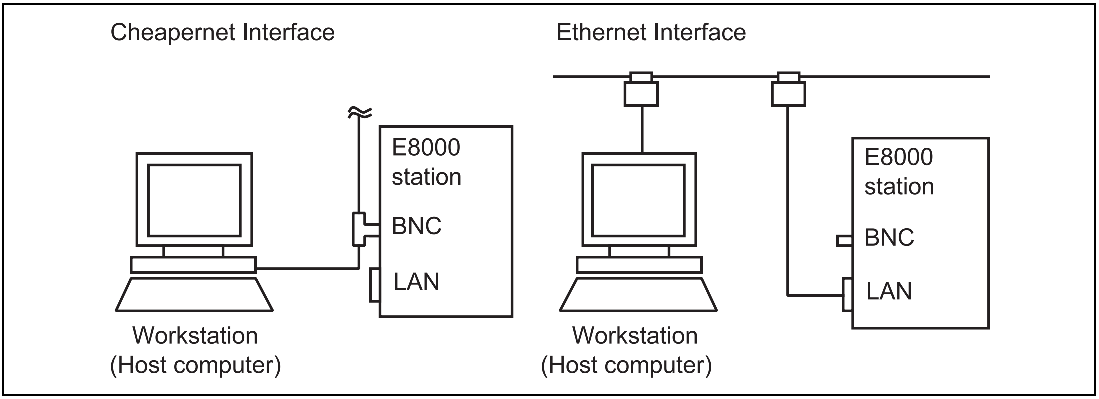


Figure 2.7 System Configuration Using a LAN Interface

Cheapernet Interface: This is achieved by connecting a coaxial cable (referred to as the Cheapernet thin-wire cable) between the BNC connector on the LAN board and the workstation.

Ethernet Interface: This is achieved by connecting transceivers and transceiver cables between the D-SUB connector on the LAN board and the workstation.

2.3.2 System Configuration Using an RS-232C or Bidirectional Parallel Interface

Using an RS-232C interface or a bidirectional parallel interface, the E8000 station can be connected to a personal computer. Figure 2.8 shows the system configuration using the RS-232C or bidirectional parallel interface.

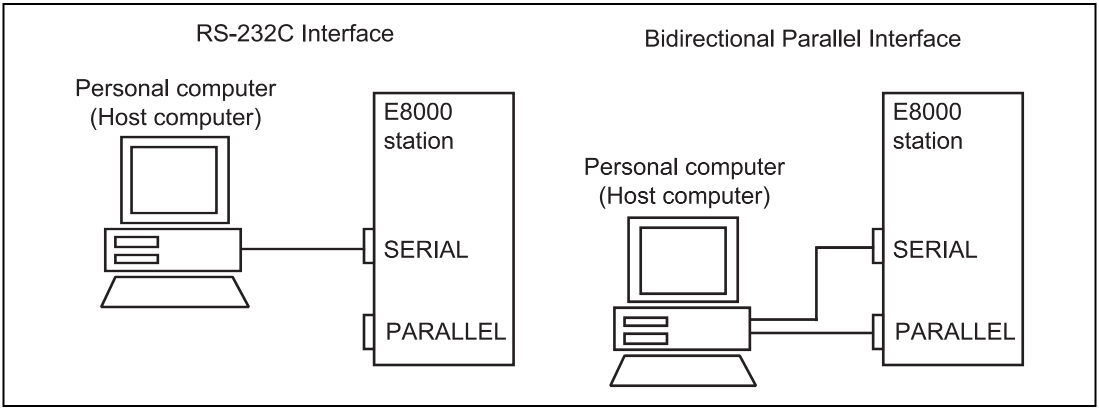


Figure 2.8 System Configuration Using an RS-232C or Bidirectional Parallel Interface

2.3.3 System Configuration Using a PC Interface Board

The E8000 station can be connected to a personal computer via a PC interface board (options; ISA bus, PCI bus, or PCMCIA bus). Install the PC interface board to the extension slot of the personal computer, and connect the interface cable supplied with the PC interface board to the E8000 station. Figure 2.9 shows the system configuration using the PC interface board. For information on using the PC interface board, refer to the description notes of the PC interface board in use.

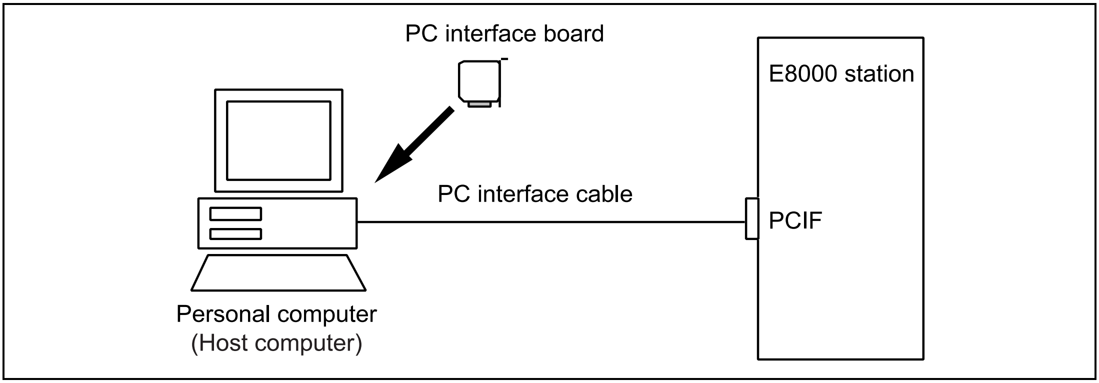


Figure 2.9 System Configuration Using a PC Interface Board

2.3.4 System Configuration Using a LAN Adapter

The E8000 station can be connected to a workstation via the LAN adapter. For description on using the LAN adapter, refer to its description notes.

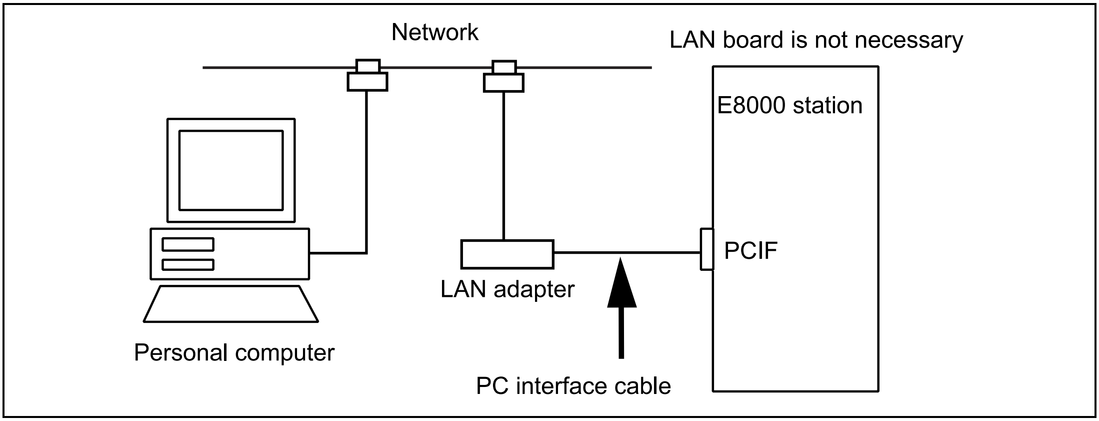


Figure 2.10 System Configuration Using a LAN Adapter

Section 3 Preparation before Use

3.1 Emulator Preparation

CAUTION

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

Unpack the emulator and prepare it for use as follows. Here, the LAN board, PC interface board (ISA bus), RS-232C, and bidirectional parallel interface cable are described. For other preparations such as the PCI bus, PCMCIA bus, or LAN adapter, refer to the description notes.

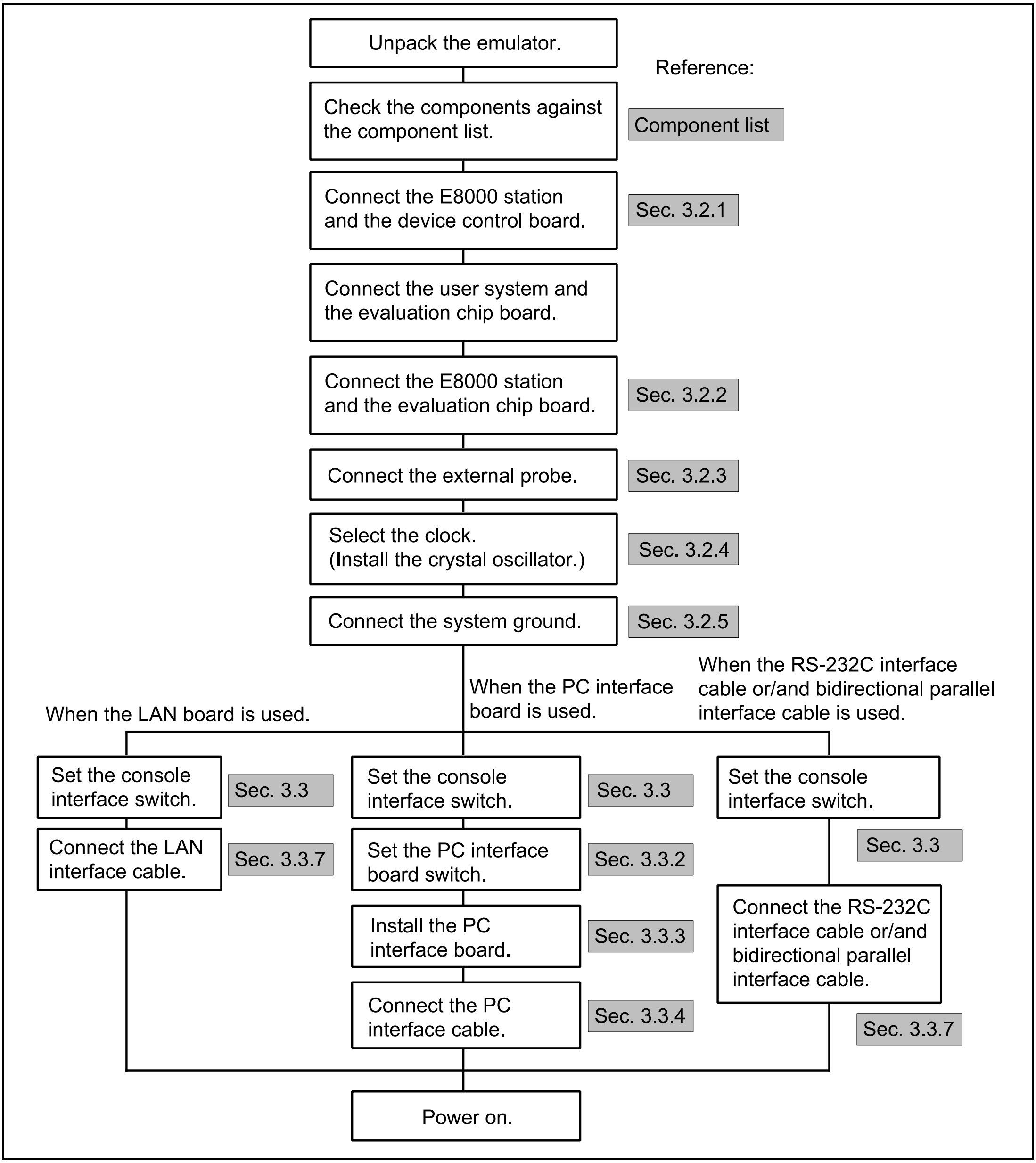

Unpack the emulator.
Reference:
Check the components against the component list.
Component list
Connect the E8000 station and the device control board.
Sec. 3.2.1
Connect the user system and the evaluation chip board.
Connect the E8000 station and the evaluation chip board.
Sec. 3.2.2
Connect the external probe.
Sec. 3.2.3
Select the clock. (Install the crystal oscillator.)
Sec. 3.2.4
Connect the system ground.
Sec. 3.2.5
When the LAN board is used.
When the PC interface board is used.
When the RS-232C interface cable or/and bidirectional parallel interface cable is used.
Set the console interface switch.
Sec. 3.3
Connect the LAN interface cable.
Sec. 3.3.7
Set the console interface switch.
Sec. 3.3
Set the PC interface board switch.
Sec. 3.3.2
Install the PC interface board.
Sec. 3.3.3
Connect the PC interface cable.
Sec. 3.3.4
Set the console interface switch.
Sec. 3.3
Connect the RS-232C interface cable or/and bidirectional parallel interface cable.
Sec. 3.3.7
Power on.

Figure 3.1 Emulator Preparation Flow Chart

3.2 Emulator Connection

3.2.1 Connecting the Device Control Board

At shipment, the device control board is packed separately from the E8000 station. Connect the device control board to the E8000 station according to the following procedure. Also, use the following procedure to connect them after removing the device control board from the E8000 station to change the device control board.

WARNING!

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

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the left side of the E8000 station's front panel is not lit.
2. Remove the AC power cable of the E8000 station from the outlet (if the cable is connected to the outlet).
3. Remove the back panel from the E8000 station. For the slot to which the device control board is to be connected, DCONT is marked.
4. Connect the device control board to the E8000 station. When connecting the board, prevent the upper or lower side of the board from lifting off the connector. Alternately tighten the screws on both sides of the board.

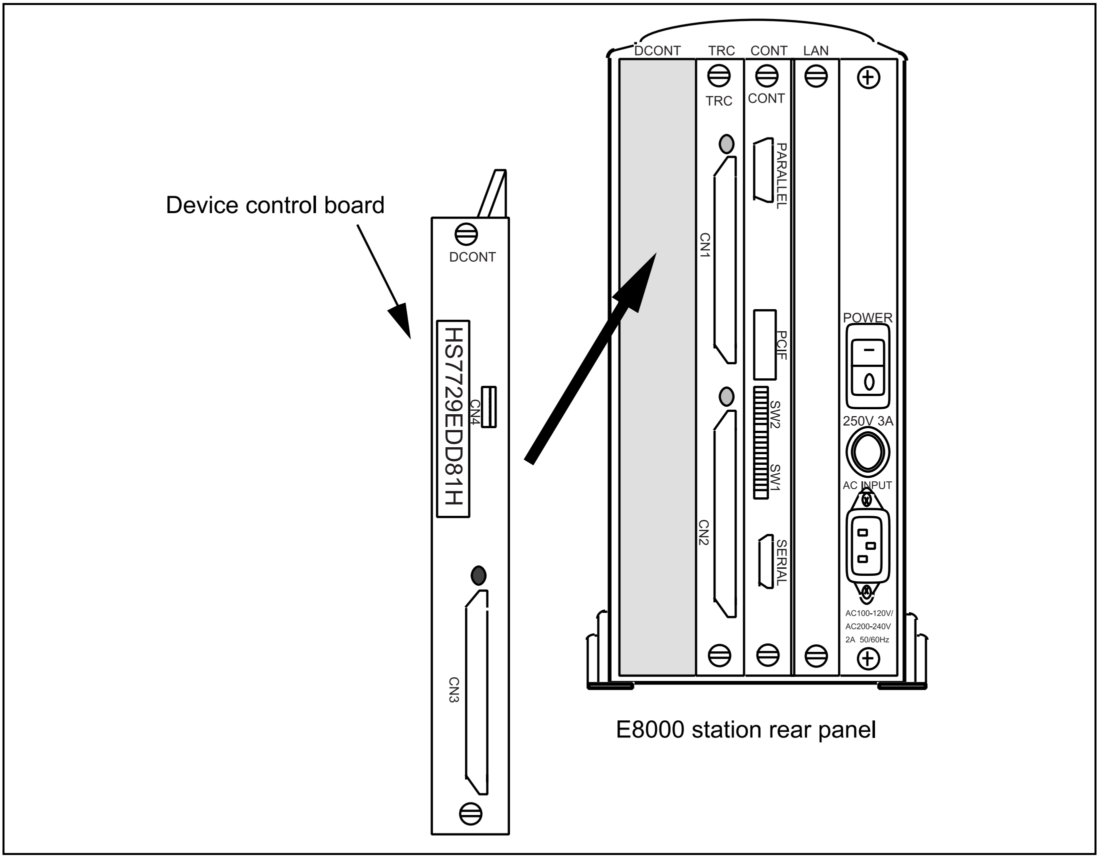


Figure 3.2 Connecting the Device Control Board

3.2.2 Connecting the Evaluation Chip Board

At shipment, the evaluation chip board is packed separately from the E8000 station. Use the following procedure to connect the evaluation chip board to the E8000 station, or to disconnect them when moving the emulator:

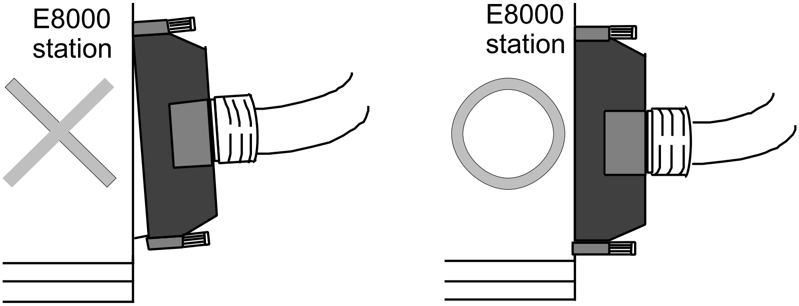
WARNING!

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

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the left side of the E8000 station's front panel is not lit.
2. Remove the AC power cable of the E8000 station from the outlet (if the cable is connected to the outlet).

WARNING!

When connecting the cable, ensure that the upper (A) or lower (B) side of the cable does not lift off the connector. Alternately tighten the screws on both sides of the cable while gradually pushing the cable toward the connector. Failure to do so will result in a FIRE HAZARD, damage the user system and emulator, and will result in PERSONAL INJURY. The USER PROGRAM will be LOST.



3. Connect the trace cables into the station to evaluation chip board interface connectors (CN1, CN2, and CN3) on the E8000 station's rear panel. Confirm that the shape of the trace-cable plug matches that of the station to evaluation chip board interface connector before connecting. Also note which trace cable is connected to which E8000-station connector so that the other end of the trace cable is connected to the matching connector number on the evaluation chip board. After the connection is completed, alternately tighten the screws on both sides of the trace cable to prevent the upper or lower side of the trace cable from lifting off the connector. Figure 3.3 shows how to correctly connect the trace cables to the E8000 station connectors.

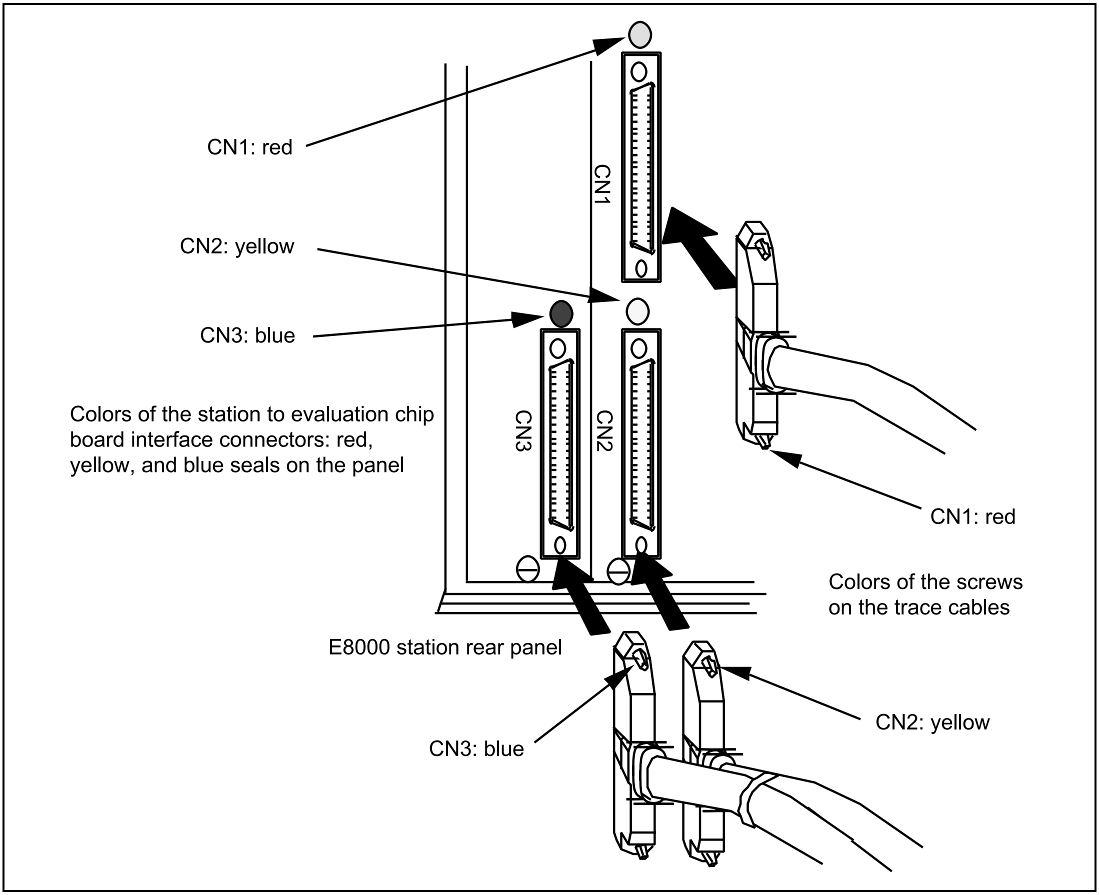

CN1: red
CN1
CN2: yellow
CN3: blue
Colors of the station to evaluation chip board interface connectors: red, yellow, and blue seals on the panel
CN3
CN2
CN1: red
Colors of the screws on the trace cables
E8000 station rear panel
CN2: yellow
CN3: blue

Figure 3.3 Connecting Trace Cables to the E8000 Station

Note: At shipment, the trace cable screws are colored to prevent an insertion error (CN1: red, CN2: yellow, CN3: blue). In addition, trace cables CN2 and CN3 to be connected to the E8000 station are bound into a bundle, and trace cables CN1, CN2, and CN3 to be connected to the evaluation chip board are bound into a bundle. Check for the number of cables bound into a bundle and the colors for connectors when connecting the cables.

WARNING!

Make sure the connector shapes and numbers are correctly matched when connecting trace cables to the station to evaluation chip interface connectors. Failure to do so will damage the emulator.

4. Connect the trace cables to the station to evaluation chip board interface connectors CN1, CN2, and CN3 on the evaluation chip board. Confirm that each trace cable connected to a connector on the E8000 station is also connected to its corresponding station to evaluation chip board interface connector on the evaluation chip board. Connect the cables using the same method as in step 3. Figure 3.4 shows how to connect the trace cables to the evaluation chip board interface connectors.

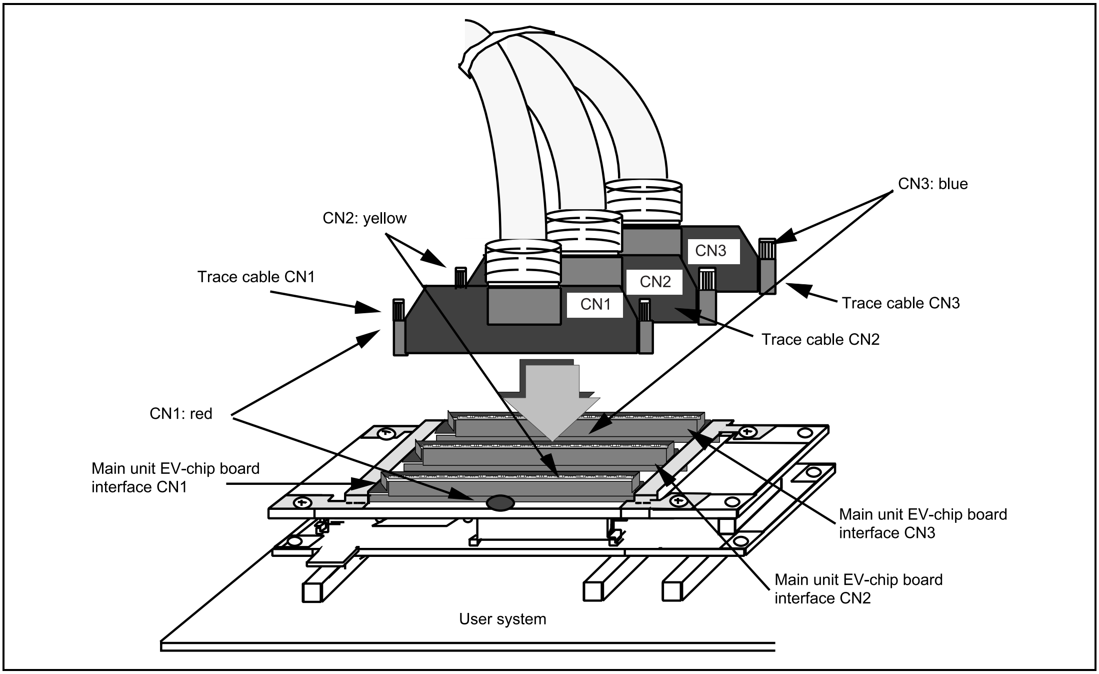

CN3: blue
CN2: yellow
CN3
Trace cable CN1
CN2
CN1
Trace cable CN3
Trace cable CN2
CN1: red
Main unit EV-chip board interface CN1
Main unit EV-chip board interface CN3
Main unit EV-chip board interface CN2
User system

Figure 3.4 Connecting Trace Cables to the Evaluation Chip Board

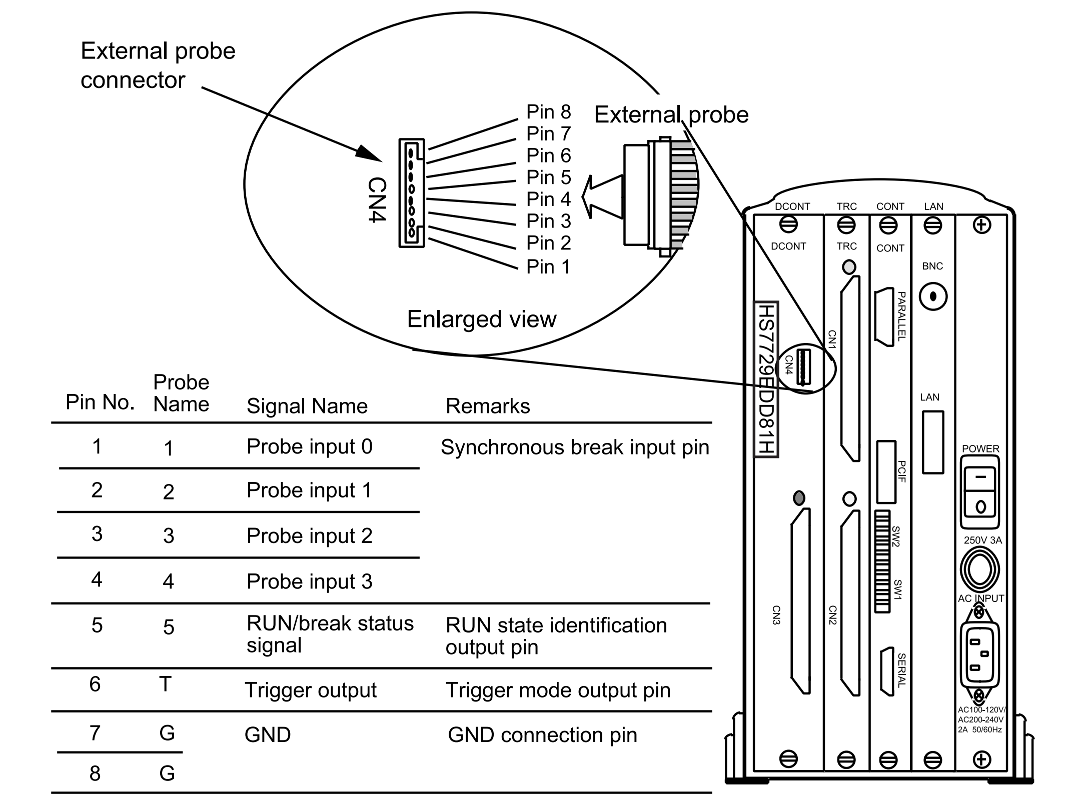
Note: For the connection between the evaluation chip board and the user system, refer to section 3, Connecting the Evaluation Chip Board to the User System, in the Evaluation Chip Board (HS7729EBH81HE) User's Manual.

3.2.3 Connecting the External Probe

CAUTION

Check the external probe direction and connect the external probe to the emulator station correctly. Incorrect connection will damage the probe or connector.

When an external probe is connected to the emulator probe connector on the emulator station's rear panel, it enables external signal tracing and multibreak detection. Figure 3.6 shows the external probe connector.



Pin No.	Probe Name	Signal Name	Remarks
1	1	Probe input 0	Synchronous break input pin
2	2	Probe input 1	
3	3	Probe input 2	
4	4	Probe input 3	
5	5	RUN/break status signal	RUN state identification output pin
6	T	Trigger output	Trigger mode output pin
7	G	GND	GND connection pin
8	G		

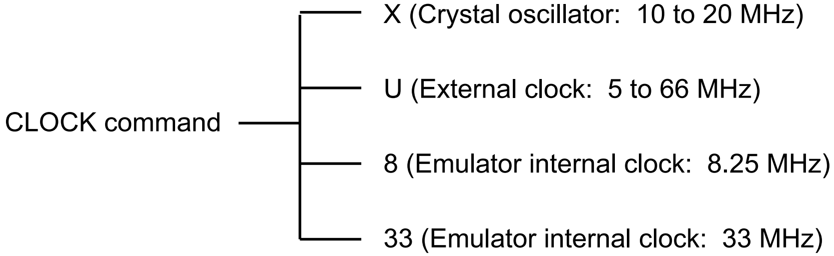
Figure 3.5 External Probe Connector

3.2.4 Selecting the Clock

This emulator supports three types of clock for the MCU: a crystal oscillator attached on the evaluation chip board, external clock input from the user system, and the emulator internal clock. The clock is specified with the CLOCK command.

Note, however, when the user system power is turned on while the clock operating mode 7 (CKIO input) is specified by the emulator, the input clock from the MCU is the clock input from the CKIO pin. The clock cannot be switched with the CLOCK command.

This emulator can use a clock source of up to 66 MHz (quadruple of external clock frequency 16.5 MHz) as the MCU clock input.


CLOCK command
X (Crystal oscillator: 10 to 20 MHz)
U (External clock: 5 to 66 MHz)
8 (Emulator internal clock: 8.25 MHz)
33 (Emulator internal clock: 33 MHz)

Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Use one that has the same frequency as that of the user system. When using a crystal oscillator as the MCU clock source, the frequency must be from 10 to 20 MHz. The multiplied value of the input clock must not exceed 66 MHz.

WARNING!

Always switch OFF the emulator and user system before connecting or disconnecting the CRYSTAL OSCILLATOR. The USER PROGRAM will be LOST.

Use the following procedure to install the crystal oscillator:

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the left side of the E8000 station's front panel is not lit.
2. Attach the crystal oscillator into the terminals on the evaluation chip board (figure 3.6).

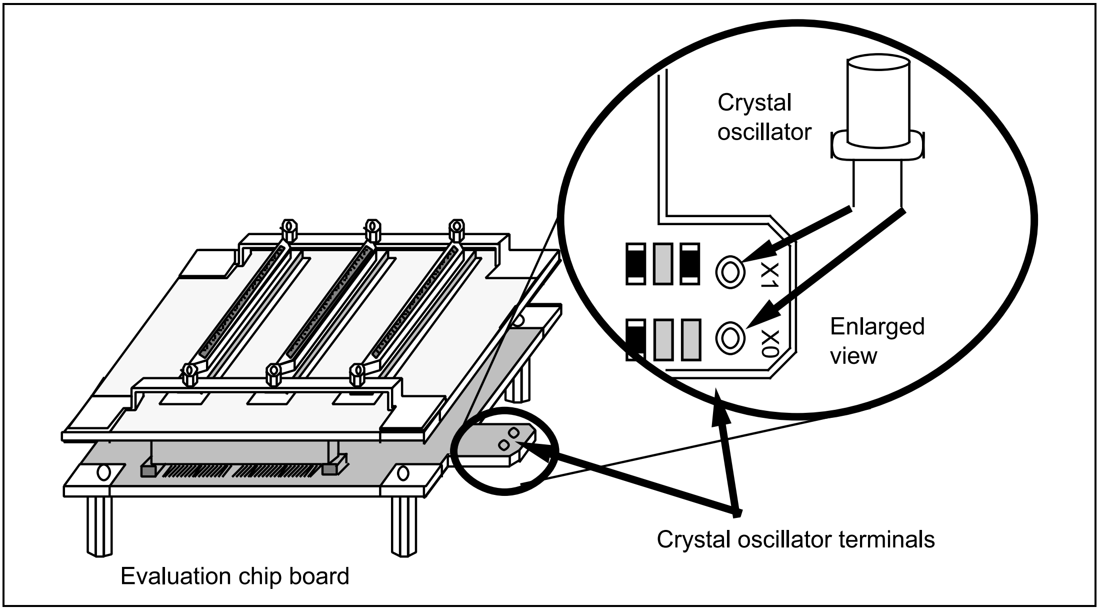


Figure 3.6 Installing the Crystal Oscillator

3. Turn on the emulator power and then the user system power. X (crystal oscillator) will then be automatically specified in the CLOCK command.

Using the crystal oscillator enables execution of the user program at the user system's operating frequency, even when the user system is not connected.

External Clock: Use the following procedure to select the external clock.

1. Check that the emulator power switch is turned off. Ensure that the power lamp on the left side of the E8000 station's front panel is not lit.
2. Connect the evaluation chip board to the user system and supply a clock through the EXTAL pin from the user system.
3. Turn on the emulator power and then the user system power. U (external clock) will then be automatically specified in the CLOCK command.

Emulator Internal Clock: Specify 8 (8.25 MHz) or 33 (33 MHz) with the CLOCK command.

Reference:

When the emulator system program is initiated, the emulator automatically selects the MCU clock source according to the following priority:

1. When the clock operating mode is 7, the clock is input from the CKIO pin.
2. External clock when supplied from the user system
3. Crystal oscillator when attached to the evaluation chip board
4. Emulator internal clock; when the clock operating mode is 1, a 8.25-MHz clock is selected, and when the clock operating mode is 0 or 3, a 33-MHz clock is selected.

3.2.5 Connecting the System Ground

WARNING!

Separate the frame ground from the signal ground at the user system. Failure to do so will result in a FIRE HAZARD or ELECTROCUTION and will damage the user system and the emulator or will result in PERSONAL INJURY.

The emulator's signal ground is connected to the user system's signal ground via the evaluation chip board. In the E8000 station, the signal ground and frame ground are connected (figure 3.8). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground.

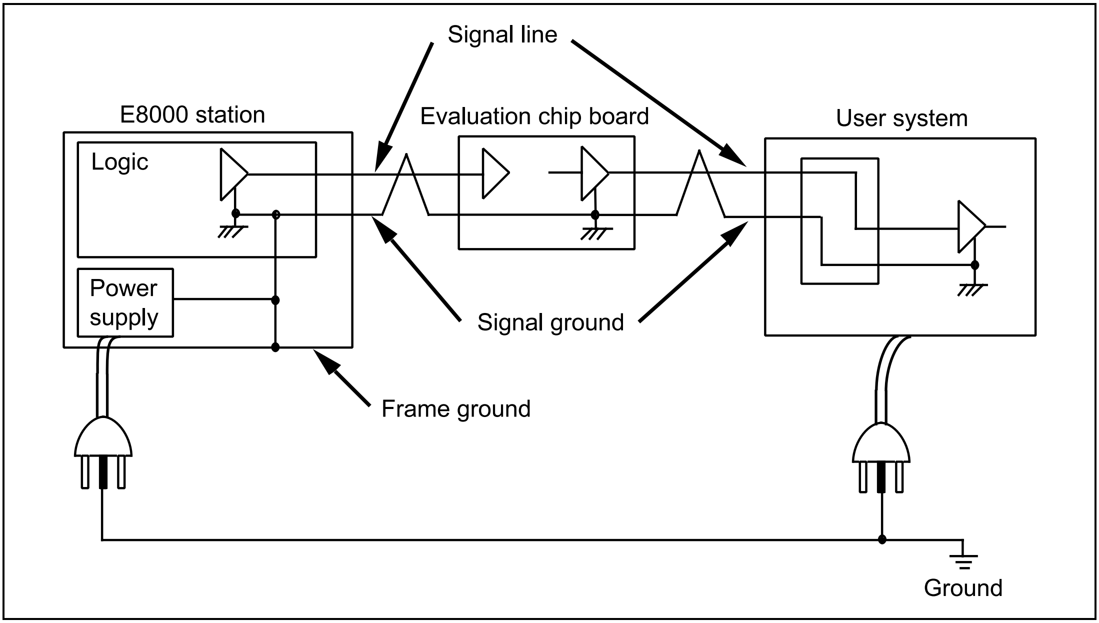


Figure 3.7 Connecting the System Ground

WARNING!

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

If it is difficult to separate the signal ground from the frame ground, insert the user system power cable and the emulator's power cable into the same outlet (figure 3.8) so that the ground lines of the cables are maintained at the same ground potential.

The user system must be connected to an appropriate ground so as to minimize noise and the adverse effects of ground loops. When connecting the evaluation chip board and the user system, confirm that the ground pins of the evaluation chip board are firmly connected to the user system's ground.

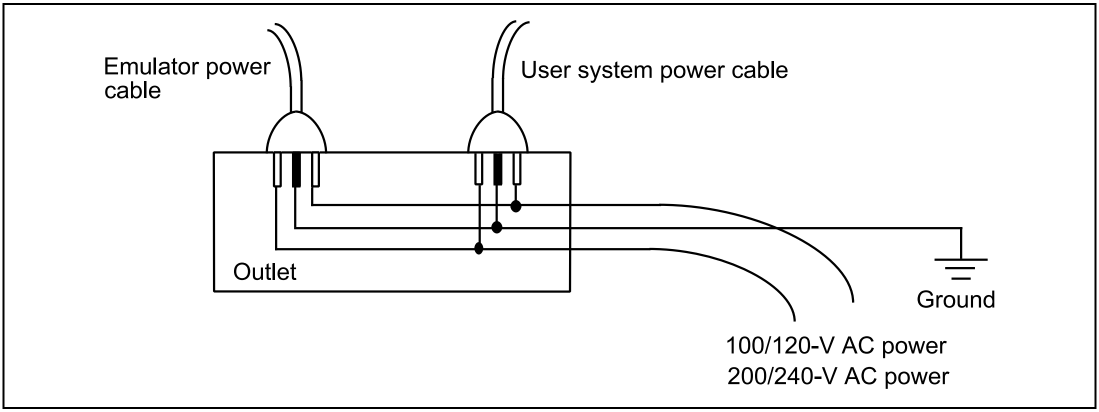


Figure 3.8 Connecting the Frame Ground

3.3 System Connection

The following describes the procedure for connecting the emulator to a host computer (work station or a personal computer). See figure 2.3 for the connector arrangement in the E8000 station.

Console Interface Setting: The settings of the transfer rate, data-bit length, stop-bit length, and parity can be changed. Use console interface switches SW1 and SW2 on the back of the E8000 station to change the settings. Switches SW1 and SW2 also include switches for the use of the console interface, the LAN interface or the PC interface.

The console interface consists of 16 switches (eight switches in both SW1 and SW2), as shown in figure 3.9. The switch state becomes on when the switches are pushed to the left, and the state becomes off when the switches are pushed to the right.

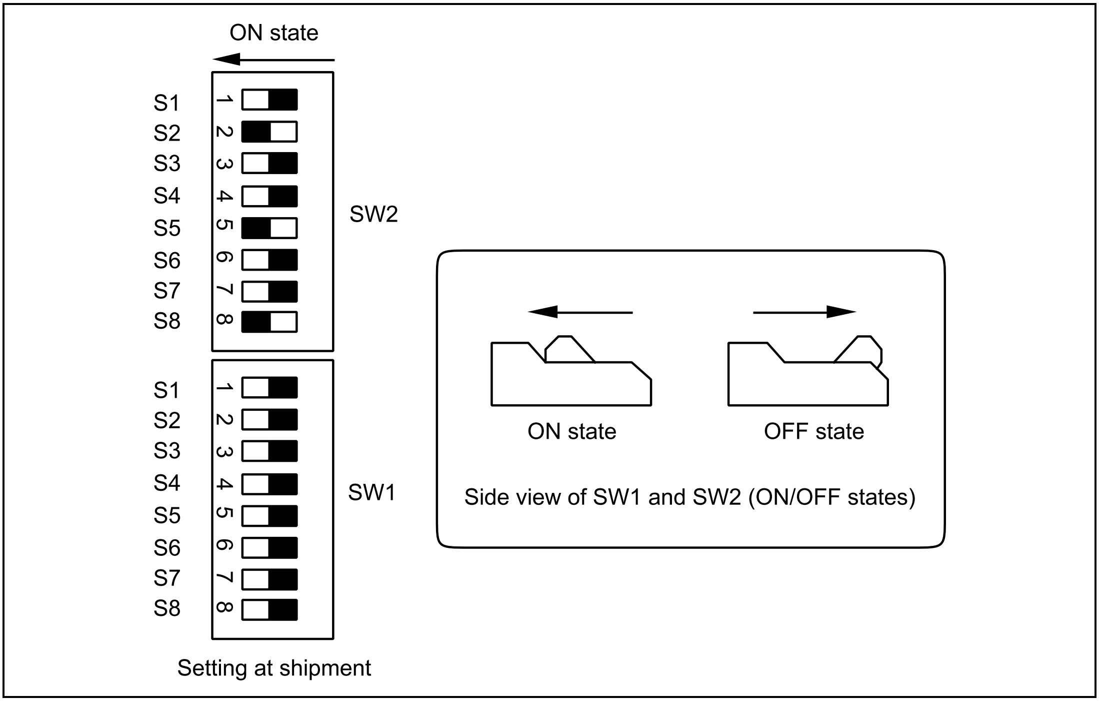


Figure 3.9 Console Interface Switches

To change the console interface settings, turn switches S1 to S8 on or off in the console interface switches SW1 and SW2. Table 3.1 lists the console interface settings and the corresponding setting states.

CAUTION

Be sure to turn off the power supply before changing the settings of console interface switches SW1 and SW2.

Table 3.1 Console Interface Settings *[1]

Transfer Rate (SW2)	S3	S2	S1
2400 BPS	OFF	OFF	OFF
4800 BPS	OFF	OFF	ON
9600 BPS	OFF	ON	OFF (Setting at shipment)
19200 BPS	OFF	ON	ON
38400 BPS	ON	OFF	OFF

Stop-bit Length (SW2)	S4
1 bit	OFF (Setting at shipment)
2 bits	ON

Bit Length (SW2)	S5
7 bits	OFF
8 bits	ON (Setting at shipment)

Parity (SW2)	S6
None	OFF (Setting at shipment)
Parity	ON

Even/odd Parity (SW2)	S7
1 bit	OFF (Setting at shipment)
2 bits	ON

Note: Effective only when there is a parity.

Flow Control (Protocol) (SW2)	S8
CTS, RTS	OFF
X-ON/OFF	ON (Setting at shipment)

Table 3.1 Console Interface Settings (cont)

Automatic System Program Initiation (Quit & Warm Start) (SW1)	S4
Cold start	OFF (Setting at shipment)
Warm start	ON

Console/LAN/PC Interface (SW1) *[2]	S7	S8
Console	OFF	OFF (Setting at shipment)
LAN board	OFF	ON
PC interface board (ISA bus)	ON	ON
PC interface board (PCI bus)	ON	ON
PC interface board (PCMCIA bus)	ON	ON
LAN adapter	ON	ON

Notes: 1. Switches S1, S2, S3, S5, and S6 of SW1 are not used. Use these switches with the off state. Console interface settings must be performed before the E8000 station power is turned on.

2. If the settings of the console interface (S7 and S8 of SW1) are incorrect, the initiation of the E8000 station cannot be confirmed on the screen. After turning off the E8000 station power, correct the interface settings. See section 3.5, Power-On Procedures for Emulator.

3.3.1 PC Interface Board Specifications

Table 3.2 lists the ISA-bus PC interface board specifications. For the PCI-bus and PCMCIA-bus interface boards, refer to their description notes.

Table 3.2 PC Interface Board Specifications

Item	Specifications
Available personal computer	ISA-bus specification PC, or compatible machine
System bus	ISA-bus specification
Memory area	16 kbytes
Memory area setting	Can be set at every 16 kbytes in the range from H'C0000 to H'EFFFF with a switch.

3.3.2 Switch Settings of the PC Interface Board

Memory-Area Setting: The PC interface board uses a 16-kbyte memory area on the perosnal computer. The memory area to be used must be allocated to the memory area on the personal computer with a switch on the PC interface board. Any 16 kbytes in the range of H'C0000 to H'EFFFF can be allocated (figure 3.10). Addresses to be allocated must not overlap the memory addresses of other boards. An overlap will cause incorrect operation.

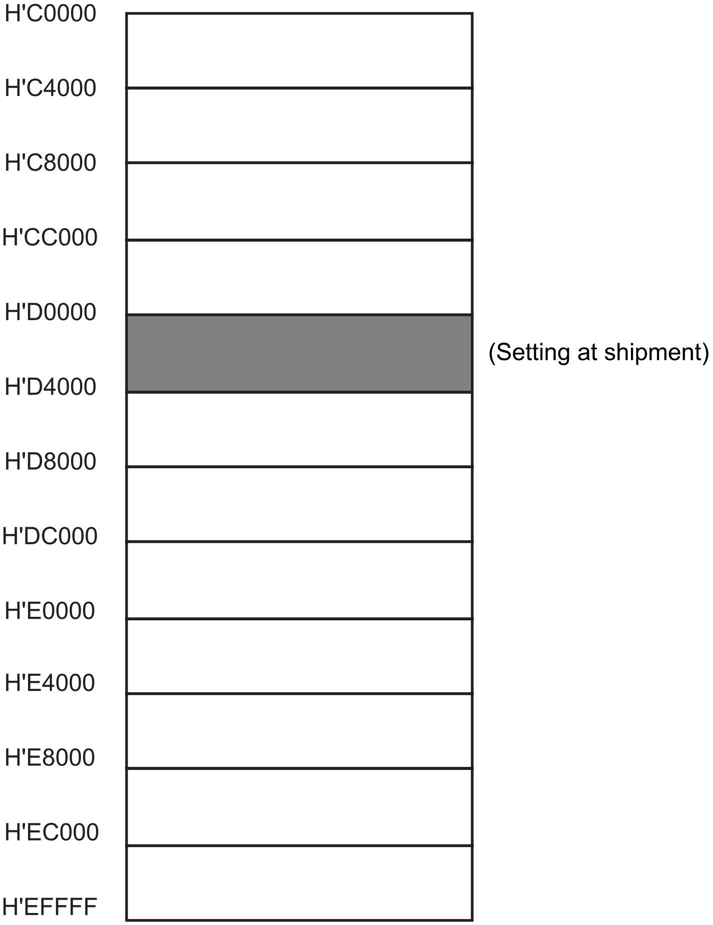


Figure 3.10 Allocatable Memory Area of PC Interface Board

Switch Setting: A rotary switch is installed on the PC interface board (figure 3.11). The switch is used to set the memory-area allocation. Table 3.3 lists the switch setting states. The switch setting at emulator shipment is No. 4 (memory area H'D0000 to H'D3FFF).

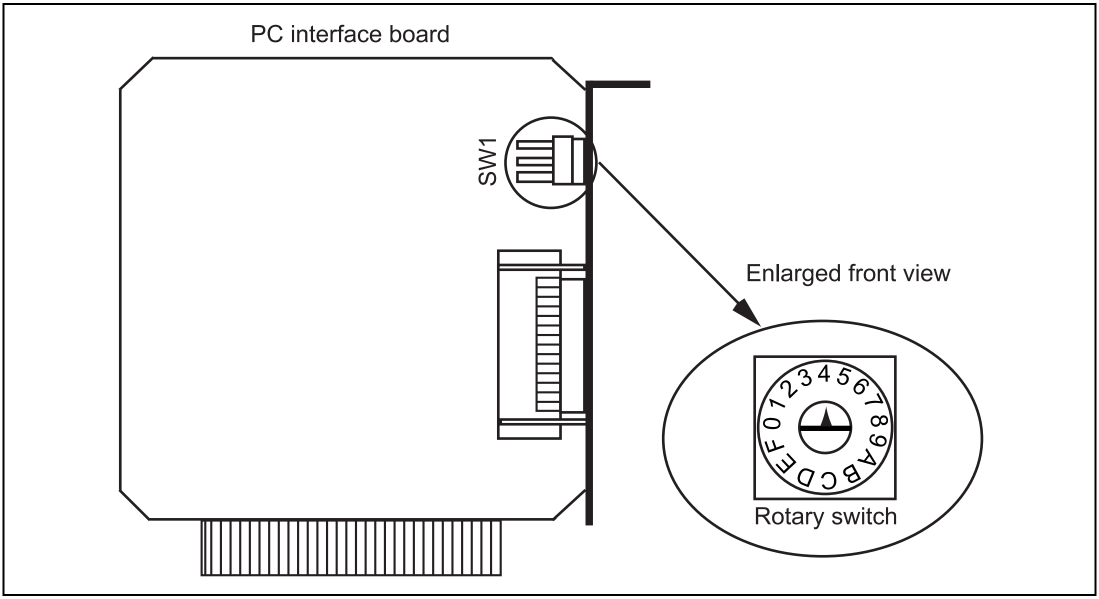


Figure 3.11 PC Interface Board Switch

Table 3.3 Switch Settings for Memory Areas

Switch Setting	Memory Area	Switch Setting	Memory Area
0	H'C0000 to H'C3FFF	8	H'E0000 to H'E3FFF
1	H'C4000 to H'C7FFF	9	H'E4000 to H'E7FFF
2	H'C8000 to H'CBFFF	A	H'E8000 to H'EBFFF
3	H'CC000 to H'CFFFF	B	H'EC000 to H'EFFFF
4 (setting at shipment)	H'D0000 to H'D3FFF	C	Not used
5	H'D4000 to H'D7FFF	D	Not used
6	H'D8000 to H'DBFFF	E	Not used
7	H'DC000 to H'DFFFF	F	Not used

Note: When C to F of the switch are set, memory areas cannot be allocated. Set one of 0 to B.

3.3.3 Installing the PC Interface Board

WARNING!

Always switch OFF the personal computer (host computer) and peripheral devices connected to the personal computer before installing the PC interface board. Failure to do so will result in a FIRE HAZARD and will damage the personal computer, interface board, and peripheral devices, or will result in PERSONAL INJURY.

Remove the cover of the personal computer and install the PC interface board in the ISA-bus specification extension slot. Tighten the screw after confirming that the PC interface cable can be connected to the board.

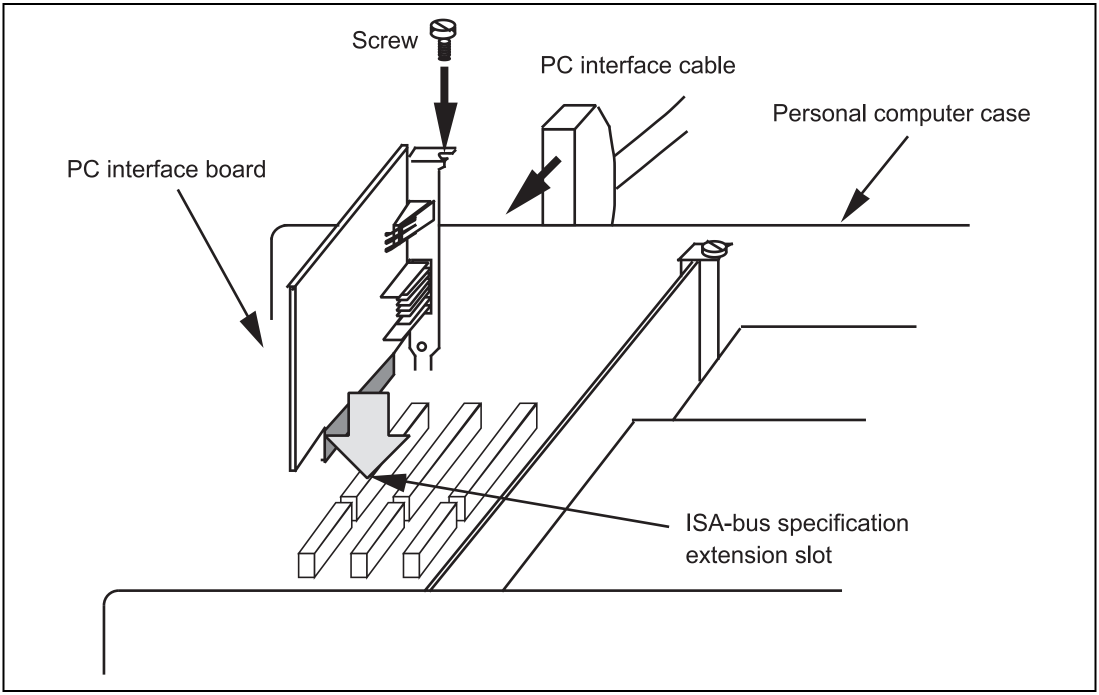


Figure 3.12 Installing the PC Interface Board

3.3.4 Connecting the E8000 Station to the PC Interface Board

WARNING!

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

Before using the emulator, connect the E8000 station to the PC interface board with the PC interface cable supplied, as shown in figure 3.13.

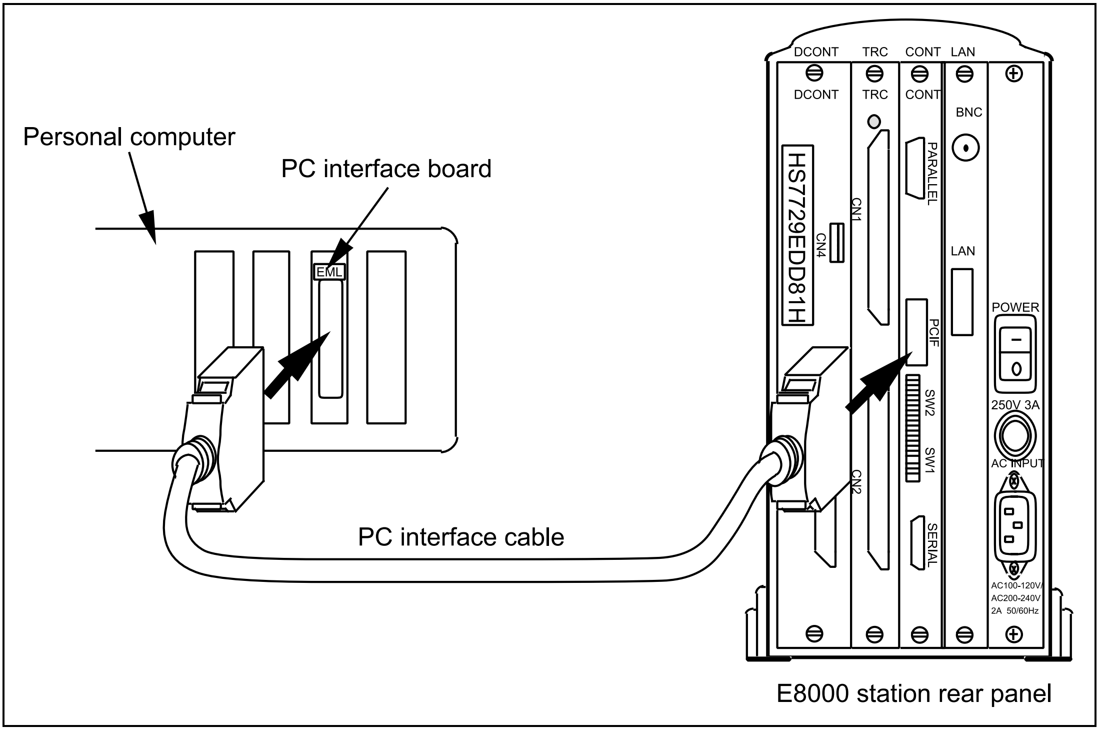


Figure 3.13 Connecting the E8000 Station to the PC Interface Board

3.3.5 Connecting to a Personal Computer

WARNING!

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

This section describes how to set the personal computer interface when the emulator is connected to a personal computer. The personal computer connector (marked SERIAL) is located on the E8000 station's rear panel. Connecting this connector to a personal computer via the RS-232C interface cable enables data transfer between the emulator and the personal computer. Table 3.4 lists the personal computer interface specifications.

The system program can be loaded to the E8000 station memory with the bidirectional parallel interface. At this time, confirm that the printer driver is specified by the personal computer settings. Use a personal computer to which the bidirectional parallel interface can be applied. See section 3.7, System Program Installation.

Table 3.4 Personal Computer Interface Specifications

Item	Specifications
Signal level	RS-232C High: +5 to +15 V Low: –5 to -15 V
Transfer rate	2400/4800/9600/19200/38400 bits per second (BPS)
Synchronization method	Asynchronous method
Start-bit length	1 bit
Data-bit length	7/8 bits
Stop-bit length	1/2 bits
Parity	Even/odd or none
Control method	X-ON/X-OFF control, RTS/CTS control

Personal Computer Interface Settings at Emulator Start Up: When the emulator is turned on, or when the emulator system program is initiated, the personal computer interface settings are determined by the console interface switches in the same way as in the console interface (the control method will be X-ON/X-OFF control).

Changing the Personal Computer Interface Settings: The transfer rate, data-bit length, stop-bit length, parity, and control method can be changed with the console interface switch. For the personal computer connector pin assignments and signal names, refer to Appendix A, Connectors.

3.3.6 Connecting to a LAN Interface

WARNING!

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

The LAN board for the emulator supports Ethernet (10BASE5) and Cheapernet (10BASE2) interfaces conforming to Ethernet specifications V2.0. The LAN board communicates with a workstation according to the TCP/IP protocol, and the workstation transfers files and commands according to the FTP/TELNET protocol. The LAN board specifications at each layer of the OSI model are as follows.

Physical and Data Link Layers: The LAN board communicates with Ethernet and Cheapernet. Table 3.5 shows the Ethernet and Cheapernet specifications.

Table 3.5 Ethernet and Cheapernet Specifications

Item	Ethernet	Cheapernet
Transfer rate	10 Mbits/second	10 Mbits/second
Maximum distance between segments	500 m	185 m
Maximum network length	2500 m	925 m
Maximum number of nodes in one segment	100	30
Minimum distance between nodes	2.5 m	0.5 m
Network cable	Diameter: 0.4 inch (1.02 cm) 50-Ω shielded coaxial cable	Diameter: 0.25 inch (0.64 cm) 50-Ω shielded coaxial cable (RG-58A/U)
Network connector	N-type connector	BNC connector
Transceiver cable	Diameter: 0.38 inch (0.97 cm) Ethernet cable to be connected to the 15-pin D-SUB connector	

Network Layer:

- IP (Internet Protocol)
  - Transmits and receives data in datagram format.
  - Does not support IP options.
  - Does not have subnet mask functions when HS7000ELN01H is used. Supports subnet mask functions when HS7000ELN02H is used.
  - Does not support broadcast communications.
- ICMP (Internet Control Message Protocol)

  Supports only echo reply functions.
- ARP (Address Resolution Protocol)

  Calculates Ethernet addresses from IP addresses by using broadcast communications.

Transport Layer:

- TCP (Transmission Control Protocol)

  Logically connects the emulator to the workstation.
- UDP (User Datagram Protocol)

  Not supported.

Session, Presentation, and Application Layers:

- FTP (File Transfer Protocol)

  The emulator operates as a client.
- TELNET (Teletype Network)

  The emulator operates as a server.

Note: The emulator communicates through routers or gateways for the HS7000ELN02H, but not for the HS7000ELN01H.

3.3.7 System Connection Examples

WARNING!

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

System configuration examples are shown below.

For LAN interface, Ethernet interface and Cheapernet interface is described.

Ethernet Interface: The LAN board of the emulator has a 15-pin D-SUB connector for the Ethernet transceiver cables. Figure 3.14 shows an example of the Ethernet system configuration. Use commercially available Ethernet transceivers and transceiver cables. Table 3.6 shows a recommended transceiver and transceiver cable.

Note: When using the LAN interface, refer to section 3.5.1, Power-On Procedures for LAN Interface, and set the IP address, router, and subnet mask value.

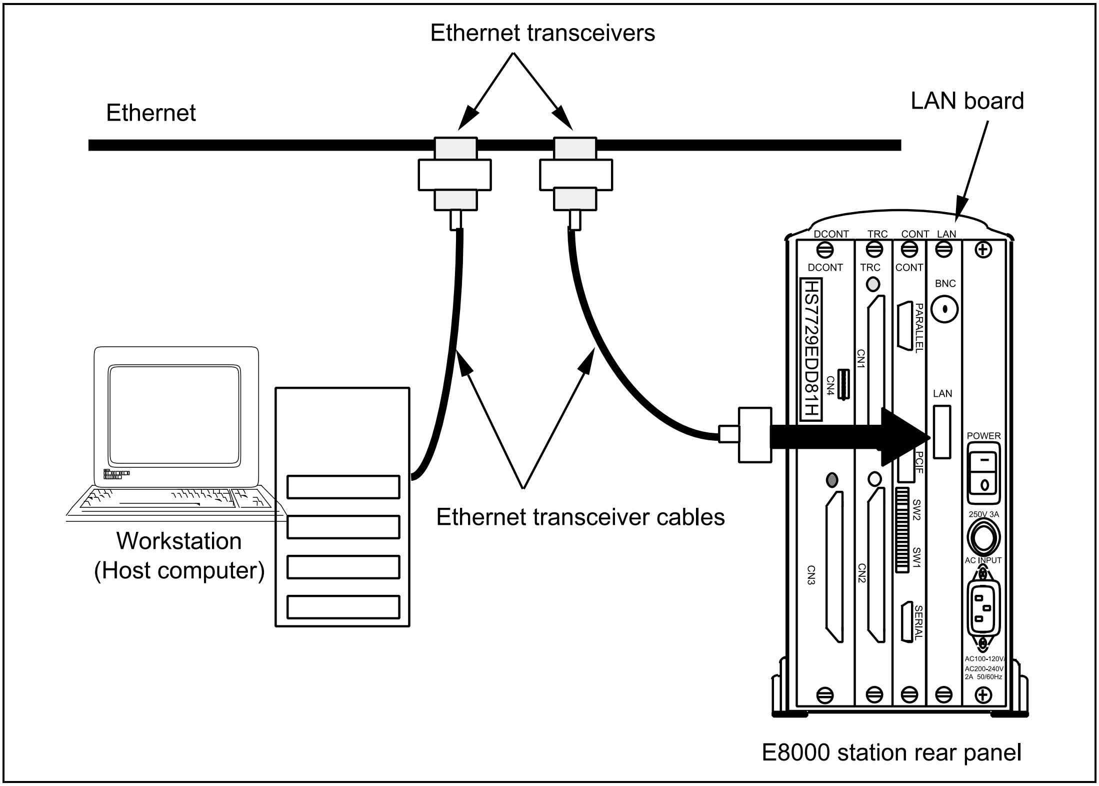


Figure 3.14 Ethernet Interface

Table 3.6 Recommended Transceiver and Transceiver Cable

Item	Product Type	Manufacturer
Transceiver	HBN-200 series	Hitachi Cable, Ltd.
Transceiver cable	HBN-TC-100	Hitachi Cable, Ltd.

For setting up the Ethernet interface, refer to the LAN board user's manual.

Cheapernet Interface: The LAN board of the emulator incorporates a transceiver and a BNC connector for a Cheapernet interface. Figure 3.15 shows an example of the Cheapernet system configuration. Use a commercially available Cheapernet BNC T-type connector with a characteristic impedance of 50 Ω and a RG-58A/U thin-wire cable or its equivalent. Table 3.7 shows a recommended BNC T-type connector and thin-wire cable.

Note: If a connector or a cable with a characteristic impedance other than 50 Ω is used, the impedance mismatch will cause incorrect data transmission and reception.

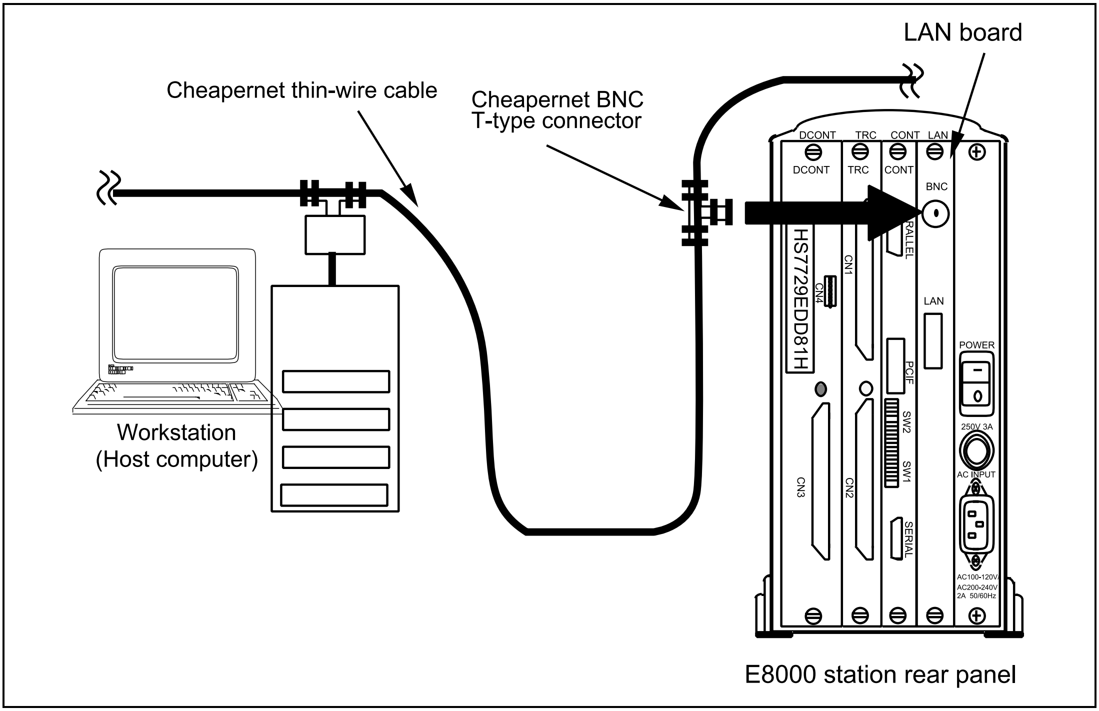


Figure 3.15 Cheapernet Interface

Table 3.7 Recommended BNC T-Type Connector and Thin-Wire Cable

Item	Product Type	Manufacturer
BNC T-type connector	HBN-TA-JPJ	Hitachi Cable, Ltd.
Thin-wire cable	HBN-3D2V-LAN	Hitachi Cable, Ltd.

For setting up Cheapernet, refer to the LAN board manual.

RS-232C Interface: Figure 3.16 shows the E8000 station connected to the personal computer via an RS-232C for a serial interface.

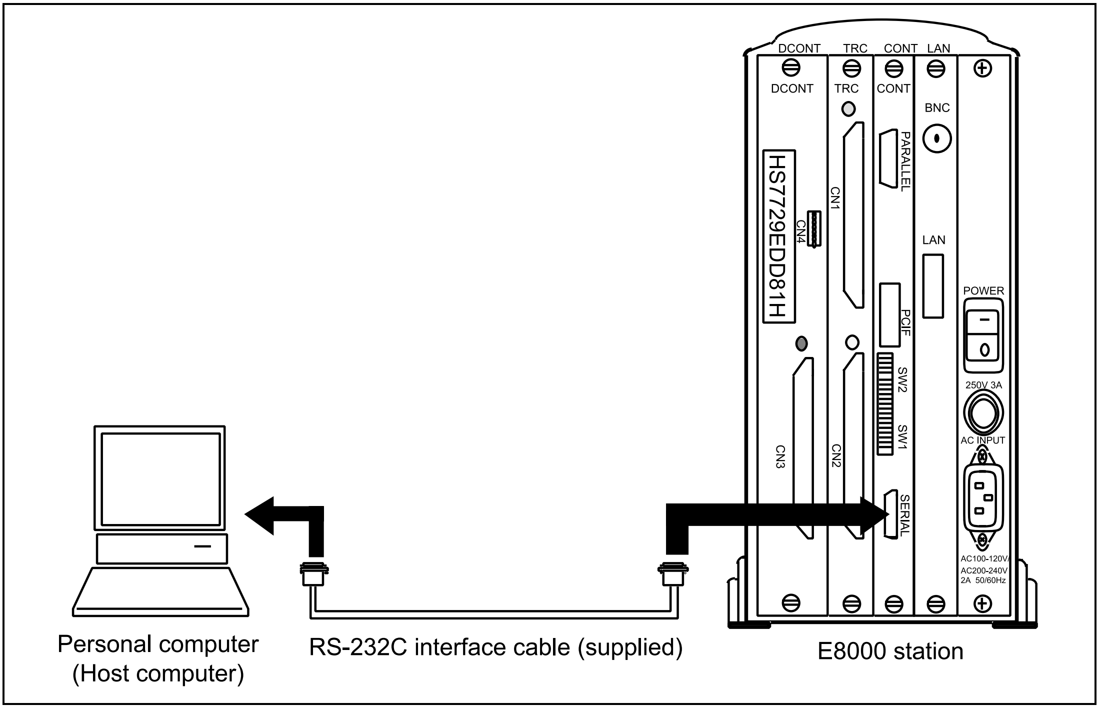


Figure 3.16 RS-232C Interface

Parallel Interface: Figure 3.17 shows the E8000 station connected to a personal computer via a parallel cable for a parallel interface. When using the parallel interface, connect not only the parallel interface cable but also the RS-232C cable. It is impossible to use only the bidirectional parallel interface cable. The parallel interface enables higher-speed installation of the system program and higher-speed load, save, or verification of the user program as compared with the RS-232C interface.

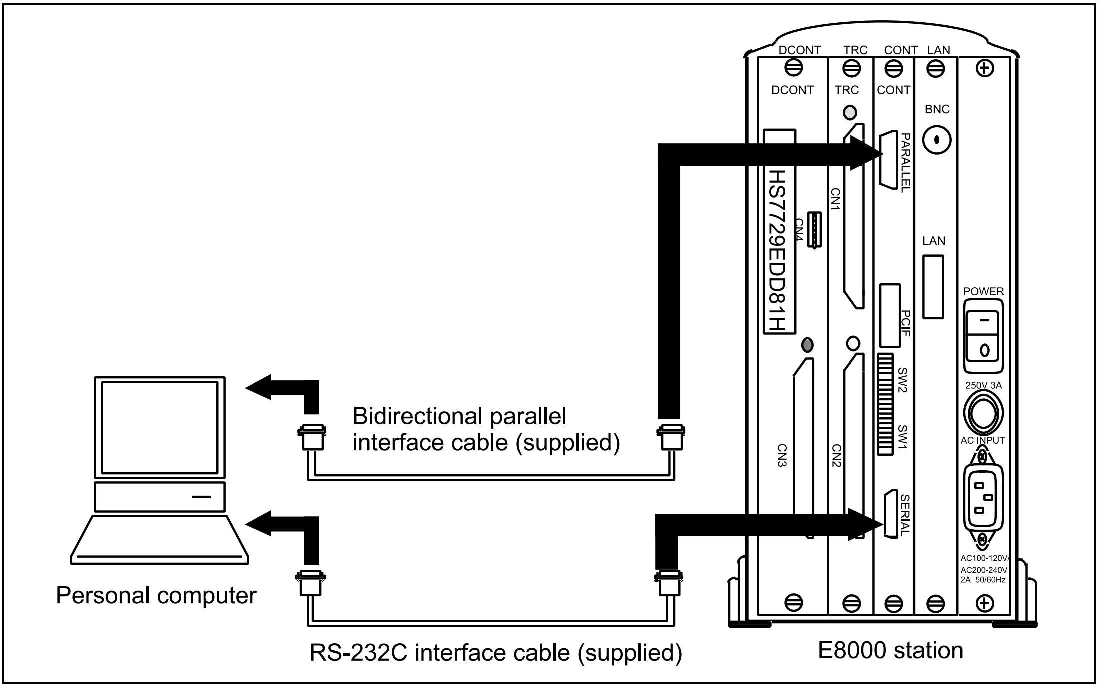


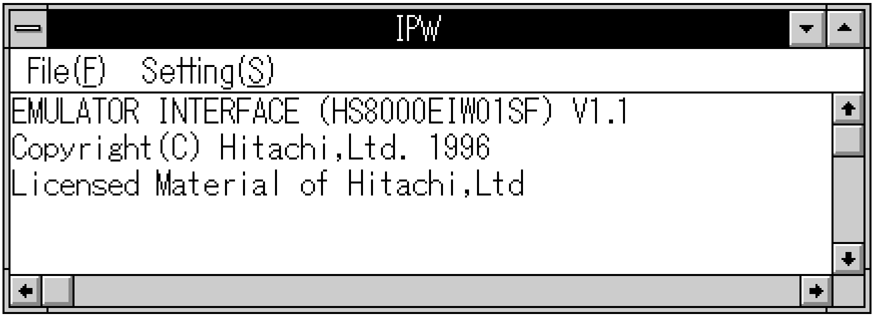
Figure 3.17 Bidirectional Parallel Interface

3.4 Operation Procedures of Interface Software IPW

Interface software IPW is used when the emulator is connected to the host computer via the RS-232C interface. Interface software IPW runs on the Microsoft® Windows® 3.1 and Windows® 95.

3.4.1 Installation and Initiation of Interface Software IPW

Make a copy of file IPW.EXE in the system disk to a folder. The directory containing the copied folder will become the current directory. Double clicking the IPW icon initiates interface software IPW and displays the IPW window shown in figure 3.18.



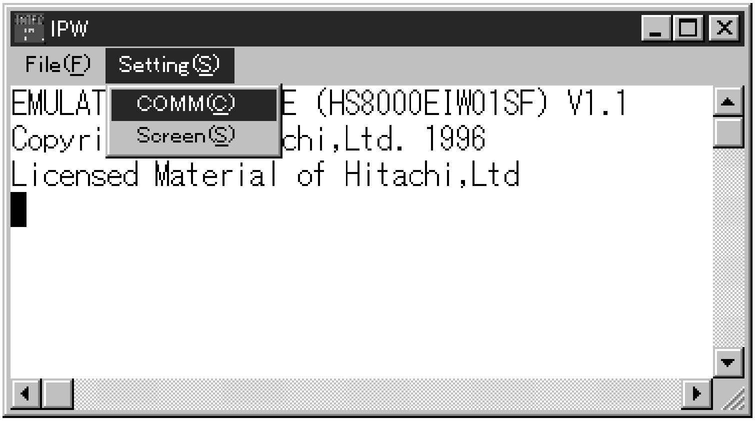
```
EMULATOR INTERFACE (HS8000EIW01SF) V1.1
Copyright(C) Hitachi,Ltd. 1996
Licensed Material of Hitachi,Ltd
```

Figure 3.18 IPW Window

Note: Microsoft and Windows are registered trademarks of Microsoft Corporation.

3.4.2 Interface Software IPW Settings

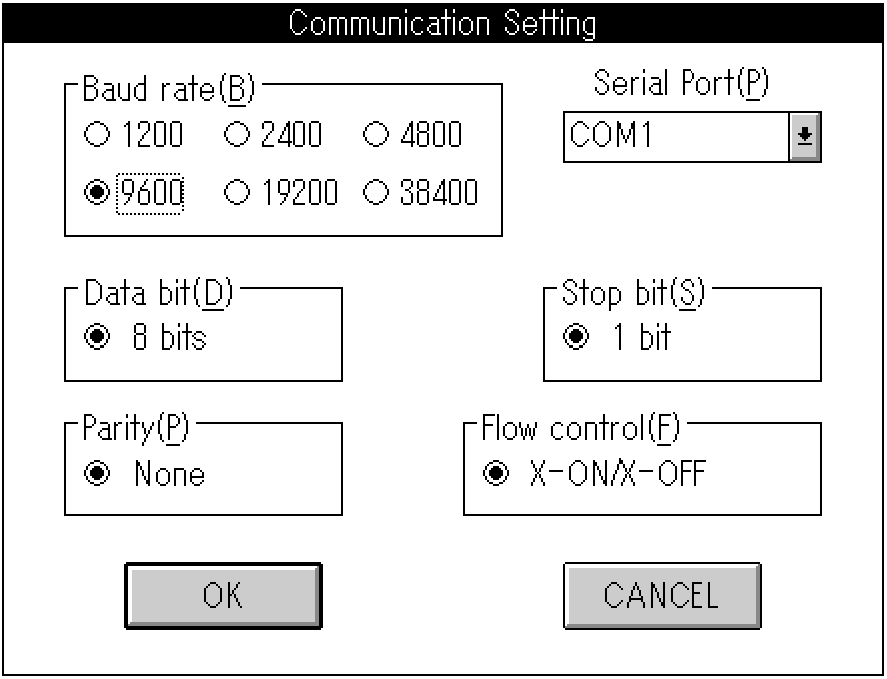
The procedures for operating interface software IPW are shown in the following. Figure 3.19 shows the File menu and Setting menu locations in the interface software IPW display.



```
EMULAT          E (HS8000EIW01SF) V1.1
Copyri          chi,Ltd. 1996
Licensed Material of Hitachi,Ltd
```

Figure 3.19 File Menu and Setting Menu

1. Clicking COMM in the Setting menu displays the Communication Setting box (figure 3.20). The Communication Setting box can also be displayed by pressing (Alt) + S keys and then the C key. Set the communications conditions to be the same as those of the DIP switches on the E8000 station rear panel.



Communication Setting

Baud rate(B)
○ 1200 ○ 2400 ○ 4800
● 9600 ○ 19200 ○ 38400

Serial Port(P): COM1

Data bit(D): ● 8 bits

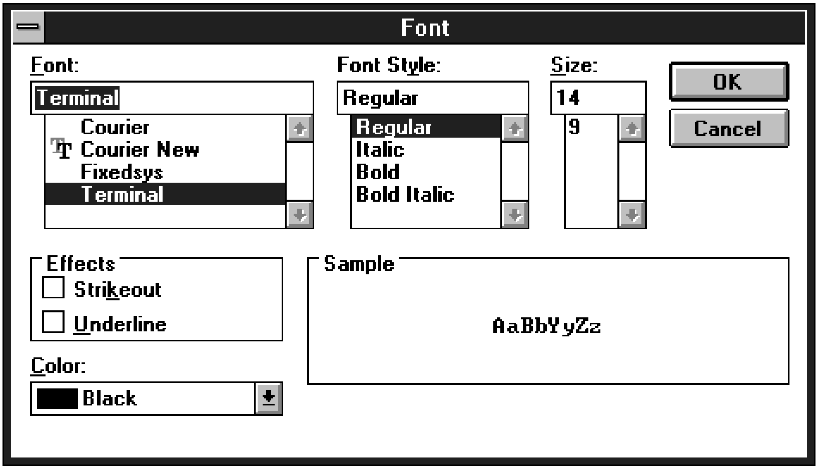
Stop bit(S): ● 1 bit

Parity(P): ● None

Flow control(F): ● X-ON/X-OFF

Figure 3.20 Communication Setting Box

2. Selecting Screen in the Setting menu displays the Screen Setting box (figure 3.21). The Screen Setting box can also be displayed by pressing (Alt) + S keys and then the S key.



Font: Terminal

Font Style: Regular

Size: 14

Effects

□ Strikeout

□ Underline

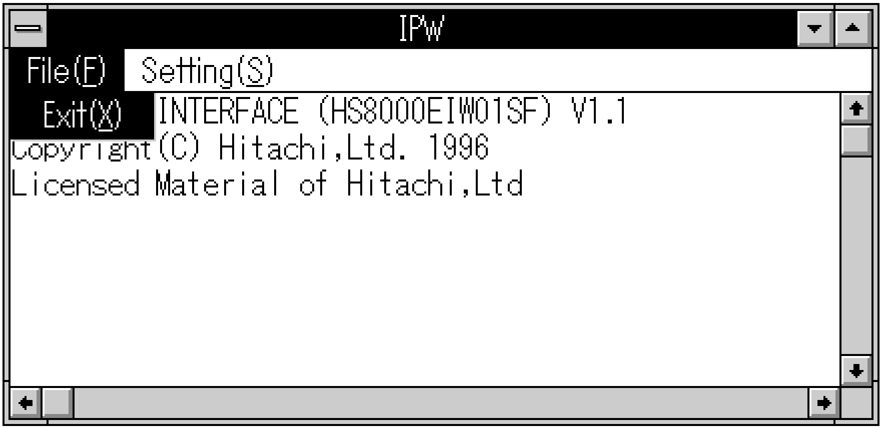
Color: Black

Figure 3.21 Screen Setting Box

3. Clicking Exit in the File menu terminates interface software IPW. Interface software IPW can also be terminated by pressing (Alt) + F keys and then the X key (figure 3.22). Note that in the following conditions a termination request is ignored and interface software IPW will not be terminated.

   — File transfer between the emulator and host computer

   — Automatic command input from a file



```
 INTERFACE (HS8000EIW01SF) V1.1
Copyright(C) Hitachi,Ltd. 1996
Licensed Material of Hitachi,Ltd
```

Figure 3.22 Exit Menu

Note: Set communication setting and screen setting in the Setting menu immediately after IPW initiation because they are not saved at IPW termination.

3.4.3 Debugging Support Functions

Interface software IPW supports the following two debugging functions.

- Automatic command input from a host computer file
- Logging acquisition

The start of automatic command input or start and end of logging acquisition can be specified when the emulator is in command input wait state (the emulator prompt is # or :).

Automatic Command Input: The file from which commands are to be input (command file) is specified with < and <file name> when the emulator is in command input wait state. Do not insert a space between < and <file name>.
Example: :***<FILENAME (RET)***
Commands are sequentially read from the specified command file and transferred to the emulator. As in the following example, when the command file is specified, commands in that file are sequentially executed. Commands requiring further input, such as the MEMORY command, can be read from a file and executed.

Example:

File contents:

```
f 400000 43ffff 0;l
m 400000;l
aaaaaaaa
55555555
12345678
.
d 400000;l
```

Execution results:

```
:f 400000 43ffff 0;l
:m 400000;l
 00400000   00000000 ? aaaaaaaa
 00400004   00000000 ? 55555555
 00400008   00000000 ? 12345678
 0040000C   00000000 ? .
:d 400000;l
<ADDRESS>           <   D   A   T   A   >     <ASCII CODE>
00400000 AAAAAAAA 55555555 12345678 00000000 “....UUUU.4Vx....”
00400010 00000000 00000000 00000000 00000000 “................”
00400020 00000000 00000000 00000000 00000000 “................”
```

The command file reading does not terminate until the end of the file is detected, or the (CTRL) + C keys are pressed. If the (CTRL) + C keys are pressed, the command being executed is terminated and the message below is displayed. According to the input reply, command file reading is continued or terminated.

INTFC ERROR - STOP COMMAND CHAIN? (Y/N) : (a) (RET)

(a) Y: Terminate
N: Continue

Logging: When logging acquisition is specified, not only are command inputs, execution results, and error messages after the specification displayed on the console, but they are output to the file specified with FILENAME.
Logging is specified with > and characters when the emulator is in command input wait state. Do not insert a space between > and characters.

- To overwrite FILENAME:

  :>***FILENAME (RET)***
- To add to FILENAME:

  :>>***FILENAME (RET)***
- To terminate logging to FILENAME:

  :>***– (RET)***

To overwrite the existing file, enter Y when the following message is displayed.

INTFC ERROR - FILE ALREADY EXISTS
OVERWRITE? (Y/N) : ***(a) (RET)***

(a) Y: Overwrites the existing file with the new file
N: Terminates command execution

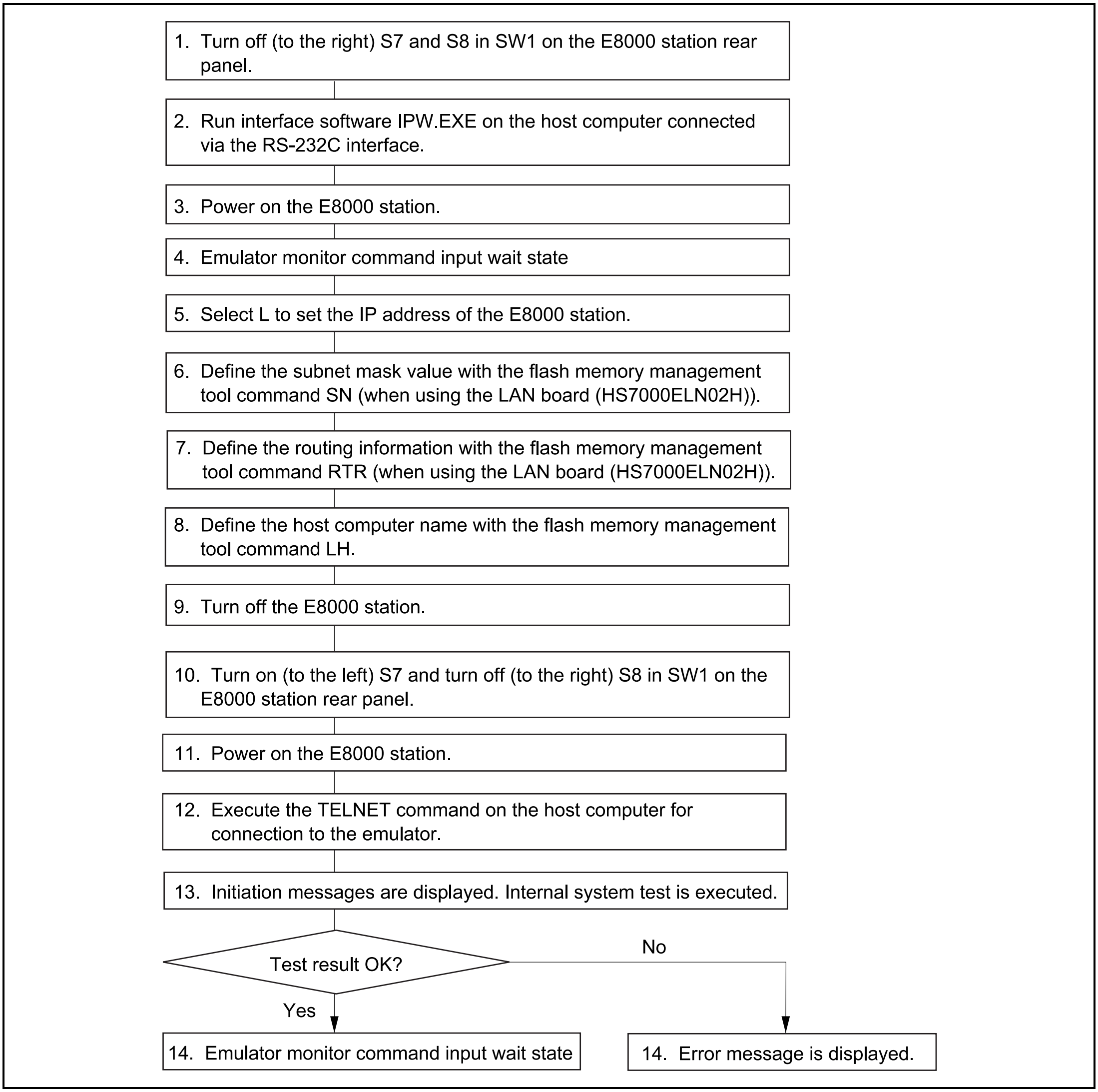
Addresses during load, save, or verification cannot be logged.

3.5 Power-On Procedures for Emulator

The emulator power-on procedures differ in each system configuration. Power on the emulator in the appropriate way for the system configuration, as shown below.

3.5.1 Power-On Procedures for LAN Interface

Figure 3.23 shows the power-on procedures when the LAN interface is used.



1. Turn off (to the right) S7 and S8 in SW1 on the E8000 station rear panel.
2. Run interface software IPW.EXE on the host computer connected via the RS-232C interface.
3. Power on the E8000 station.
4. Emulator monitor command input wait state
5. Select L to set the IP address of the E8000 station.
6. Define the subnet mask value with the flash memory management tool command SN (when using the LAN board (HS7000ELN02H)).
7. Define the routing information with the flash memory management tool command RTR (when using the LAN board (HS7000ELN02H)).
8. Define the host computer name with the flash memory management tool command LH.
9. Turn off the E8000 station.
10. Turn on (to the left) S7 and turn off (to the right) S8 in SW1 on the E8000 station rear panel.
11. Power on the E8000 station.
12. Execute the TELNET command on the host computer for connection to the emulator.
13. Initiation messages are displayed. Internal system test is executed.

14. Emulator monitor command input wait state

14. Error message is displayed.

Figure 3.23 Power-On Procedures for LAN Interface

The following describes the power-on procedures when using the LAN interface.

1. Check that S7 and S8 in console interface switch SW1 on the E8000 station rear panel are turned off (to the right).

2. Run interface software IPW.EXE on the host computer connected to the emulator via the RS-232C interface.

3. Turn on the power switch at the E8000 station rear panel.

4. The emulator waits for an emulator monitor command input. (For details, refer to section 3.6, Monitor Commands.)

5. Specify the emulator IP address.

   The optional LAN board supports the TCP/IP protocol. When the host computer is connected to the emulator via the LAN interface, the IP address (internet address) of the emulator must be specified with emulator monitor command L.

   Press L and then the (RET) key. The set IP address is displayed. Make sure the IP address is correct. The 32-bit IP address, which is generally expressed in hexadecimal, is displayed in four bytes in decimal. For example, when the IP address has been specified as H'80010101 (H' represents hexadecimal), the emulator will display the IP address as follows and wait for a new IP address input.

   ```
   : IP ADDRESS = 128.1.1.1  :  _
   ```

   Enter a new IP address to change the displayed IP address. When changing the IP address with emulator monitor command L, restart the emulator.

   The host name and IP address of the emulator must be specified in the network database for the host computer. Normally, the network management tool of the host computer is used. For details, refer to the host computer user's manual.

6. Define the subnet mask value when using the LAN board (HS7000ELN02H).

   When the F command (flash memory management tool initiation) is entered while the emulator waits for an emulator monitor command input, the emulator displays prompt FM> and waits for a flash memory management tool command (refer to table 3.9).

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ?  F (RET)
FM>
```

Next define the subnet mask value.

```
FM>  SN <subnet mask value>;C (RET)
```

Enter Q (RET) to terminate the flash memory management tool.

```
FM>  Q (RET)
```

7. Set the routing information with the flash memory management tool command RTR when the LAN board HS7000ELN02H is used to connect the host computer in a different network to the emulator. A maximum of ten routing information can be defined. Enter the number to be defined, and then the IP address and the network number of the router.

   ```
   FM> RTR (RET)
   *** NO ENTRY DATA
   PLEASE SELECT NO. (1-10/L/E/Q/X) ? 1 (RET)
   01  IP ADDRESS                    ? <router IP address> (RET)
   01  NET ID                        ? <network number> (RET)
   ```

   Enter E (RET) and terminate the RTR command to enable the input contents and save the settings in the emulator.

   ```
   PLEASE SELECT NO. (1-10/L/E/Q/X) ?  E (RET)
   LAN CONFIGURATION FILE WRITE OK (Y/N) ?  Y (RET)
   FM>_
   ```

   Enter Q (RET) to terminate the flash memory management tool.

   ```
   FM> Q (RET)
   ```

8. Store the host name and IP address of the host computer in the emulator.

   To transfer data between the host computer and emulator, initiate the FTP server to connect the host computer to the emulator. Before the FTP server is initiated, the host name and IP address of the host computer must be stored in the emulator flash memory. The following describes how to specify the host name and IP address.

   When the F command (flash memory management tool initiation) is entered while the emulator waits for an emulator monitor command, the emulator displays prompt FM> and waits for a flash memory management tool command (refer to table 3.9).

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ?  F (RET)
FM>
```

   Next enter the LH command, and the following message is displayed.

```
FM>  LH (RET)
 NO   <HOST NAME>   <IP ADDRESS>      NO     <HOST NAME>   <IP ADDRESS>
 01     xxxxxx      xxx.xxx.xxx.xxx   02       xxxxxx      xxx.xxx.xxx.xxx
 03     xxxxxx      xxx.xxx.xxx.xxx   04       xxxxxx      xxx.xxx.xxx.xxx
 05     xxxxxx      xxx.xxx.xxx.xxx   06       xxxxxx      xxx.xxx.xxx.xxx
 07     xxxxxx      xxx.xxx.xxx.xxx   08       xxxxxx      xxx.xxx.xxx.xxx
 09     xxxxxx      xxx.xxx.xxx.xxx
        E8000 IP ADDRESS = xxx.xxx.xxx.xxx
PLEASE SELECT NO.(1-9/L/E/Q/X) ? _
```

   Up to nine pairs of host names and IP addresses can be specified. Input a number from 1 to 9. The emulator prompts the host name. Enter a name with up to 15 characters. After that, the emulator prompts the IP address.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? 1 (RET)
01 HOST NAME   xxxxxx   <name of host computer> (RET)
01 IP ADDRESS   xxx.xxx.xxx.xxx   <IP address of host computer> (RET)
```

   After the IP address has been specified, the emulator will prompt for another selection number. When connecting more than one host computer, continue specifying the host names and IP addresses. To confirm the specifications, enter L (RET) as follows.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? L (RET)
 NO   <HOST NAME>   <IP ADDRESS>       NO    <HOST NAME>   <IP ADDRESS>
 01     xxxxxx      xxx.xxx.xxx.xxx    02      xxxxxx      xxx.xxx.xxx.xxx
 03     xxxxxx      xxx.xxx.xxx.xxx    04      xxxxxx      xxx.xxx.xxx.xxx
 05     xxxxxx      xxx.xxx.xxx.xxx    06      xxxxxx      xxx.xxx.xxx.xxx
 07     xxxxxx      xxx.xxx.xxx.xxx    08      xxxxxx      xxx.xxx.xxx.xxx
 09     xxxxxx      xxx.xxx.xxx.xxx
      E8000 IP ADDRESS = xxx.xxx.xxx.xxx
PLEASE SELECT NO.(1-9/L/E/Q/X) ? _
```

To terminate input, enter E, Q, or X followed by (RET).

Entering E (RET) saves the new specifications in the emulator flash memory, initiates the LAN board, and terminates LH command execution.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? E (RET)
LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

Entering Q (RET) saves the new specifications in the emulator flash memory without initializing the LAN board, and terminates LH command execution.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? Q (RET)
LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

Entering X (RET) terminates LH command execution without saving the new specifications.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? X (RET)
FM>
```

When the emulator waits for a flash memory management tool command (prompt FM>), entering Q (RET) terminates the flash memory management tool.

```
FM>  Q (RET)
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ? _
```

9. Turn off the E8000 station.

10. Check that S7 and S8 in console interface switch SW1 on the E8000 station rear panel are turned off (to the right) and on (to the left), respectively.

11. Turn on the power switch at the E8000 station rear panel.

12. Execute the TELNET command on the host computer.

13. The following messages are displayed and the internal system tests are executed.

```
E8000 MONITOR (HS8000EST02SR) Vm.n
Copyright (C) Hitachi, Ltd. 1995
Licensed Material of Hitachi, Ltd.

TESTING
  RAM             0123
```

14. If no error occurs, the emulator waits for an emulator monitor command.

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ? _
```

Refer to section 3.6.1, Emulator Monitor Initiation, for details on operations after emulator power-on and section 3.8, E8000 System Program Initiation, for details on emulator system initiation.

3.5.2 Power-On Procedures for RS-232C Interface

Figure 3.24 shows the power-on procedures when the RS-232C interface is used.

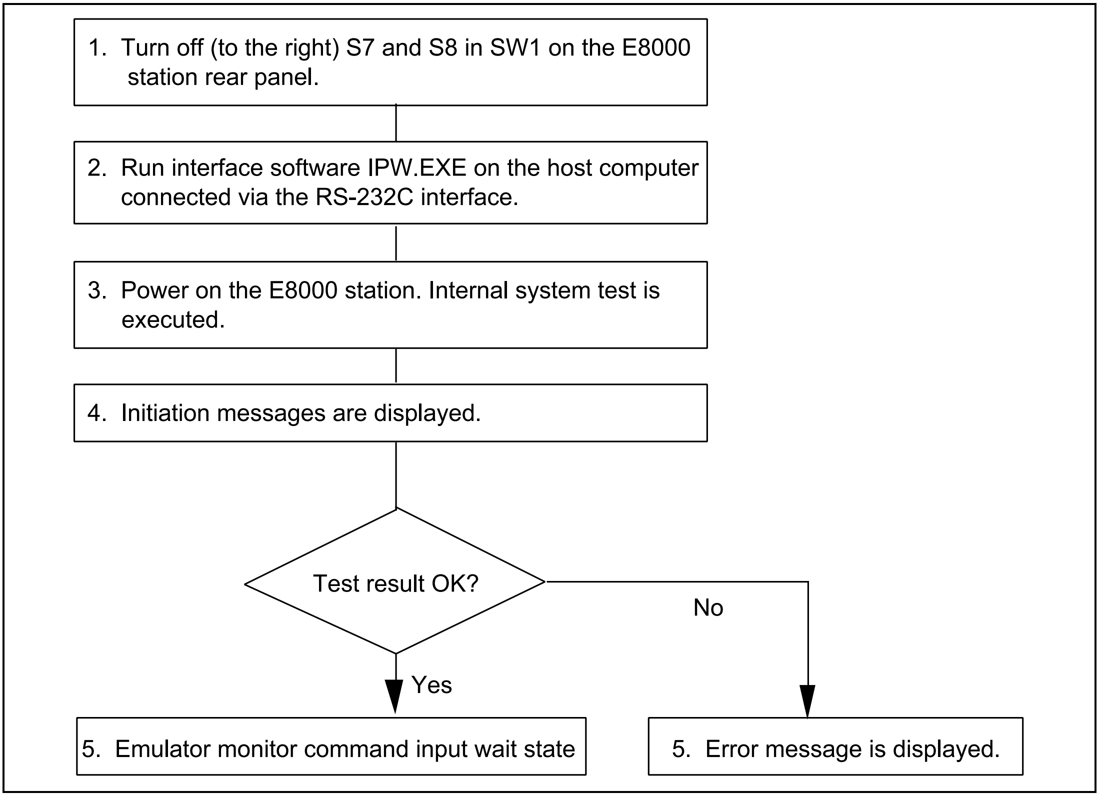

1. Turn off (to the right) S7 and S8 in SW1 on the E8000 station rear panel.
2. Run interface software IPW.EXE on the host computer connected via the RS-232C interface.
3. Power on the E8000 station. Internal system test is executed.
4. Initiation messages are displayed.
Test result OK?
No
Yes
5. Emulator monitor command input wait state
5. Error message is displayed.

Figure 3.24 Power-On Procedures for RS-232C Interface

Refer to section 3.6.1, Emulator Monitor Initiation, for details on operations after emulator power-on and section 3.8, E8000 System Program Initiation, for details on emulator system initiation.

3.6 Emulator Monitor Commands

3.6.1 Emulator Monitor Initiation

The emulator supports the four monitor commands listed in table 3.8. These commands initiate the E8000 system program, manage flash memory, set an IP address for LAN interface, and execute the diagnostic program. After turned on, the emulator displays the following monitor initiation message and waits for an emulator monitor command input.

Display Message:

```
E8000 MONITOR (HS8000EST02SR) Vm.n
Copyright (C) Hitachi, Ltd. 1995
Licensed Material of Hitachi, Ltd.

TESTING
  RAM          0123

START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ? _
```

Table 3.8 Emulator Monitor Commands

Command	Function	Remark
S	E8000 system program initiation	
F	Flash memory management tool initiation	
L	Emulator IP address setting	
T	Diagnostic program initiation	

3.6.2 S [S] Initiates the E8000 system program

Command Format

- Initiation S (RET)

Description

- Initiation
  Initiates the E8000 system program.

Example

To initiate the E8000 system program:

```
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ? S (RET)

SH7729 E8000(HS7729EDD81SF) Vm.n
Copyright (C) Hitachi, Ltd. 1998
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
HARDWARE REGISTER READ/WRITE CHECK
FIRMWARE SYSTEM LOADING
EMULATOR FIRMWARE TEST
** RESET BY E8000!
CLOCK=8.25MHz
MPU NAME=SH7729  MODE=19 (MD5-0=3F)
MONITOR START ADDRESS=03FC0000  MONITOR BUS WIDTH=32BIT
AREA2 MEMORY=NORMAL  AREA3 MEMORY=NORMAL  SDRAM TYPE=1MB-16BIT
DRAM BUS WIDTH=32BIT
CS2/PTK[0] PIN=CS2  CS3/PTK[1] PIN=CS3  CS4/PTK[2] PIN=CS4
CS5/CE1A/PTK[3] PIN=CS5  STATUS[1:0]/PTJ[7:6] PIN=STATUS[1:0]
WE[2:3]/PTK[6:7] PIN=WE[2:3]  BS/PTK[4] PIN=BS
CAS[LL:HH]/PTJ[2:5] PIN=CAS[LL:HH]
CAS[LL:2H]/PTJ[2:3, PTE[3, 6] PIN=CAS[LL:2H]
JTAG CLOCK=16.5MHz
REMAINING EMULATION MEMORY S=8MB
:
```

3.6.3 F [F] Initiates the flash memory management tool

Command Format

- Flash memory management tool initiation F (RET)

Description

- Initiates flash memory management tool

  Initiates the flash memory management tool. The flash memory management tool can use the commands listed in table 3.9.

Table 3.9 Flash Memory Management Tool Commands

Command	Function
DIR	Displays system file loading status
LH	Defines the host name and IP address of the host computer to be connected
Q	Terminates the flash memory management tool
RTR	Defines routing information for remote network
SL	Loads the E8000 system program
SN	Defines the subnet mask value

Note: The RTR and SN commands can be used only when the LAN board HS7000ELN02H is used.

Example

To initiate the flash memory management tool:

```
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ? F (RET)
FM>
```

DIR [DIR] Displays system file loading status

Command Format

- Display DIR (RET)

Description

- Display

  Displays system-file loading status. Displays OK for correctly loaded system file, NG for abnormally loaded on, and NO for not loaded.

Example

To display system file loading status:

```
FM>DIR (RET)
 <FILE ID> <STATUS>
  SYS        OK
  CONF       OK
  LAN        NO
  FIRM       OK
  TRON       NO
  DIAG       OK
  INI        OK
  MON        OK
FM>
```

LH [LH] Defines the host name and IP address of the host computer

Command Format

- Definition LH (RET)

Description

- Definition

  Defines the host name and IP address of the host computer. Enter the host name and IP address as follows after the specified number is entered and the emulator prompts them:

```
PLEASE SELECT NO. (1-9/L/E/Q/X) ? <definition number> (RET)
01  HOSTNAME xxxxxx                 <host name> (RET)
01  IP ADDRESS xxx.xxx.xxx.xxx       <IP address> (RET)
```

- Display

  Entering L (RET) displays the list of the defined host computer.

- Initiation

  Entering E (RET) saves the new specifications in the emulator flash memory, and initiates the LAN board.

- Termination

  Entering Q (RET) saves the new specifications in the emulator flash memory without initializing the LAN board, and terminates LH command execution. Entering X (RET) terminates LH command execution without saving the new specifications.

Example

To define the host name of the host computer as host and its IP address as 128.1.1.1:

```
FM>LH (RET)
 PLEASE SELECT NO.(1-9/L/E/Q/X) ? 1 (RET)
 01 HOST NAME xxxxxx           host (RET)
 01 IP ADDRESS xxx.xxx.xxx.xxx 128.1.1.1 (RET)
 PLEASE SELECT NO.(1-9/L/E/Q/X) ? L (RET)
 NO <HOST NAME> <IP ADDRESS>      NO <HOST NAME> <IP ADDRESS>
 01  host        128.1.1.1        02
 03                               04
 05                               06
 07                               08
 09
     E8000 IP ADDRESS = xxx.xxx.xxx.xxx
 PLEASE SELECT NO.(1-9/L/E/Q/X) ? E (RET)
 LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

Q [Q] Terminates the flash memory management tool

Command Format

- Termination Q (RET)

Description

- Termination
  Terminates the flash memory management tool.

Example

To terminate the flash memory management tool:

```
FM>Q (RET)
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ?
```

RTR [RTR] Defines the remote network routing information

Command Format

- Definition RTR (RET)

Description

- Definition

  Defines the remote network routing information. Enter the IP address and network number as follows after the specified number is entered and the emulator prompts them:

```
FM> RTR (RET)
 PLEASE SELECT NO. (1-10/L/E/Q/X) ? <definition number> (RET)
 IP ADDRESS                      ? <router IP address> (RET)
 NET ID                          ? <network number> (RET)
```

- Display

  Entering L (RET) displays the list of the defined host computer.

- Initiation

  Entering E (RET) saves the new specifications in the emulator flash memory, and initiates the LAN board.

- Termination

  Entering Q (RET) saves the new specifications in the emulator flash memory without initializing the LAN board, and terminates LH command execution. Entering X (RET) terminates LH command execution without saving the new specifications.

Example

To define router IP address 128.1.2.1 for network number 128.1.2.0 as the routing information:

```
FM>RTR (RET)
 PLEASE SELECT NO.(1-10/L/E/Q/X) ? 1 (RET)
 IP ADDRESS                  ? 128.1.2.1 (RET)
 NET ID                      ? 128.1.2.0 (RET)
 PLEASE SELECT NO.(1-10/L/E/Q/X) ? L (RET)
 NO <IP-ADDRESS>     <NET-ID>           NO <IP-ADDRESS>   <NET-ID>
 01  128.1.2.1        128.1.2.0
 PLEASE SELECT NO.(1-10/L/E/Q/X) ? E (RET)
 LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

SL [SL] Loads the system program

Command Format

- Load SL (RET)

Description

- Load

  Loads the system program.

Example

To load the system program:

```
FM>SL (RET)
 SELECT LOAD No. (1:PC or 2:WS) ? 1 (RET)
 SELECT INTERFACE (1:RS-232C or 2:PARALLEL) ? 2 (RET)
 LOAD E8000 SYSTEM FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\E8000.SYS (RET)
 LOAD CONFIGURATION FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\SHCNF729.SYS (RET)
 LOAD FIRMWARE FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\SHDCT729.SYS (RET)
 LOAD ITRONDEBUGGER FILE OK (Y/N) ? N (RET)
 LOAD DIAGNOSTIC FILE OK (Y/N) ? N (RET)
FM>
```

SN [SN] Defines the subnet mask value

Command Format

- Definition SN <subnet mask value> (RET)
- Save SN <subnet mask value> [;C] (RET)
- Display SN (RET)

Description

- Definition

  Defines the subnet mask value.

  ```
  FM>SN <subnet mask value> (RET)
  FM>
  ```

- Save

  Saves the setting specifications in the E8000 station when the C option is specified.

  ```
  FM>SN <subnet mask value>; C (RET)
   LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
  FM>
  ```

- Display

  Displays the subnet mask value.

  ```
  FM>SN (RET)
   SUB-NET-MASK xxx. xxx. xxx. xxx (H'xx. H'xx. H'xx. H'xx)
  ```

Examples

1. To define 255.255.255.0 as the subnet mask value and save the setting specifications in the E8000 station:

```
FM>SN 255.255.255.0;C (RET)
 LAN CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

2. To display the subnet mask value:

```
FM>SN (RET)
 SUB-NET-MASK 255.255.255.0  (H'FF.H'FF.H'FF.H'00)
FM>
```

3.6.4 L [L] Sets the emulator IP address

Command Format

- Setting L (RET)

Description

- Setting

  Sets the emulator IP address.

Example

To set the IP address of the E8000 station to 128.1.1.1:

```
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ? L (RET)

E8000 IP ADDRESS = 0.0.0.0 : 128.1.1.1 (RET)
```

3.6.5 T [T] Initiates the diagnostic program

Command Format

- Initiation T (RET)

Description

- Initiation

  Initiates the diagnostic program.

Example

To initiate the emulator diagnostic program:

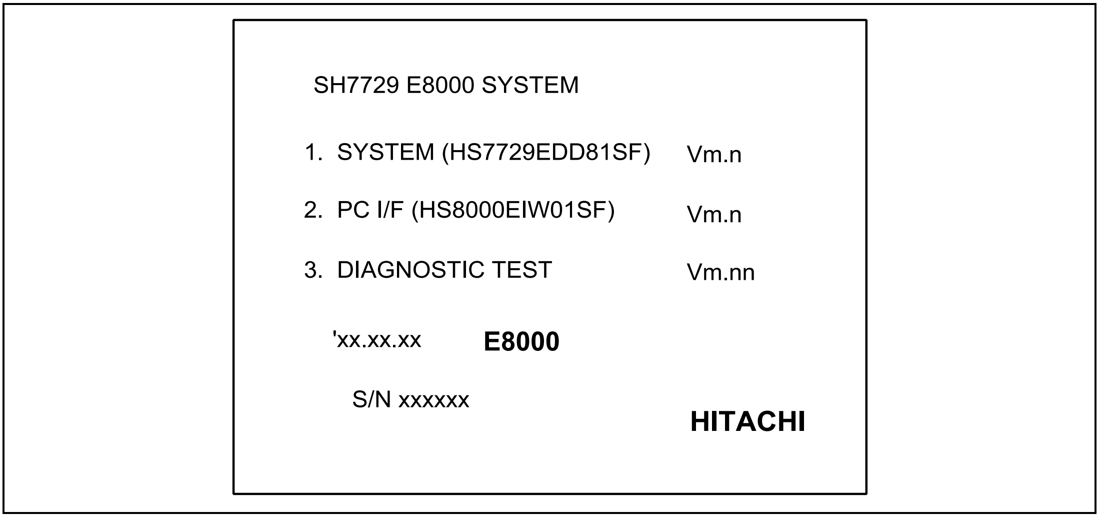
```
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ? T (RET)

E8000 TEST & MAINTENANCE PROGRAM (DIAG.SYS)
Version No.=x.xx     xx/xx/1999
Copyright (C) Hitachi, LTD 1999
Please,key in TEST PARAMETER
OPERATION TEST EXECUTE (Y/N) ?
```

3.7 System Program Installation

3.7.1 E8000 System Disk

The emulator contains one floppy disk.



SH7729 E8000 SYSTEM

1. SYSTEM (HS7729EDD81SF) Vm.n
2. PC I/F (HS8000EIW01SF) Vm.n
3. DIAGNOSTIC TEST Vm.nn

'xx.xx.xx **E8000**

S/N xxxxxx

HITACHI

Figure 3.25 E8000 System Disk

The E8000 system disk with a 1.44-Mbyte format is for PC. This floppy disk contains the following six files:

- E8000.SYS
- SHCNF729.SYS
- SHDCT729.SYS
- SETUP.CC
- IPW.EXE
- DIAG.SYS

E8000.SYS, SHCNF729.SYS, and SHDCT729.SYS are system programs that must be installed to the emulator flash memory with emulator monitor command F (flash memory management tool initiation). SETUP.CC is a file for writing the system programs via the parallel interface. IPW.EXE is a file containing interface software that runs on the Microsoft® Windows® 3.1 and Windows® 95, and must be installed to the host computer memory.

3.7.2 Installation

To use the emulator, the E8000 system program must be installed in the emulator flash memory. Load the E8000 system program to flash memory with the system program writing file or with the flash memory management tool using the emulator monitor commands.

Automatic System Program Load by Bidirectional Parallel Interface: If the emulator is connected to the host computer via the bidirectional parallel interface and the E8000 system disk is inserted in drive A of the host computer, the E8000 system program can be automatically loaded with the system program writing file SETUP.CC in the following procedures. It takes approximately one minute.

Operations — **Display Message**

1. Initiate IPW in the E8000 system floppy disk.
2. Power on the emulator. For details on the power-on procedures, refer to section 3.5.2, Power-On Procedures for RS-232C Interface.
3. Emulator monitor command prompt

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

4. Enter <A:\SETUP.CC (RET) in the monitor command input wait state.

```
(S/F/L/T) ? <A:\SETUP.CC (RET)
```

5. After the system program writing file completes loading the system program, the emulator re-enters the monitor command input wait state.

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

6. Installation is completed.

Manual System Program Load by Bidirectional Parallel Interface: To use the emulator, files E8000.SYS, SHCNF729.SYS, and SHDCT729.SYS must be installed in the emulator flash memory.

If the emulator is connected to the host computer via the bidirectional parallel interface, the E8000 system program can be loaded with the following procedures. Note that the E8000 system disk is assumed to be inserted in drive A of the host computer. It takes approximately one minute.

Operations — **Display Message**

1. Initiate IPW in the E8000 system floppy disk.
2. Power on the emulator. For details on the power-on procedures, refer to section 3.5.2, Power-On Procedures for RS-232C Interface.
3. Emulator monitor command prompt

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

4. Enter F (RET) to initiate the flash memory management tool. The emulator displays prompt FM> and waits for a flash memory management tool command.

```
(S/F/L/T) ? F (RET)
FM>
```

5. Enter SL (RET) to load the system program.

```
FM> SL (RET)
```

6. Enter 1 (RET) to select PC as the host computer type, and 2 (RET) to select parallel interface as the interface method.

```
 SELECT LOAD No. (1:PC or 2:WS) ? 1 (RET)
 SELECT INTERFACE (1:RS-232C or 2:PARALLEL) ? 2 (RET)
```

7. Enter Y (RET) to allow system program E8000.SYS to be loaded. Then enter the parallel transfer command to load E8000.SYS in drive A on the host computer to the emulator flash memory.

```
 LOAD E8000 SYSTEM FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\E8000.SYS (RET)

:COMPLETED
```

Operations | **Display Message**

8. Enter Y (RET) to allow configuration file SHCNF729.SYS to be loaded. Then enter the parallel transfer command to load SHCNF729.SYS in drive A on the host computer to the emulator flash memory.

```
 LOAD CONFIGURATION FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\SHCNF729.SYS (RET)

:COMPLETED
```

9. Enter Y (RET) to allow firmware file SHDCT729.SYS to be loaded. Then enter the parallel transfer command to load SHDCT729.SYS in drive A on the host computer to the emulator flash memory.

```
 LOAD FIRMWARE FILE OK (Y/N) ? Y (RET)
 INPUT COMMAND : #B:A:\SHDCT729.SYS (RET)

:COMPLETED
```

10. Enter N (RET) not to load the ITRON debugger.

```
 LOAD ITRON DEBUGGER FILE OK (Y/N) ? N
(RET)
```

11. Enter N (RET) not to load the diagnostic program.

```
 LOAD DIAGNOSTIC FILE OK (Y/N) ? N (RET)
```

12. Enter DIR (RET) to check whether the necessary files have been loaded.

```
FM> DIR (RET)
  <FILE ID> <STATUS>
   SYS        OK
   CONF       OK
   LAN        NO
   FIRM       OK
   TRON       NO
   DIAG       NO
   INI        OK
   MON        OK
```

13. Enter Q (RET) to terminate the flash memory management tool.

```
FM> Q (RET)

START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

14. Installation is completed.

Manual System Program Load by RS-232C Interface: To use the emulator, files E8000.SYS, SHCNF729.SYS, and SHDCT729.SYS must be installed in the emulator flash memory.

If the emulator is connected to the host computer via the RS-232C interface, the E8000 system program can be loaded with the following procedures. Note that the E8000 system disk is assumed to be inserted in drive A of the host computer. It takes approximately 20 minutes.

Operations **Display Message**

1. Initiate IPW in the E8000 system floppy disk.
2. Power on the emulator. For details on the power-on procedures, refer to section 3.5.2, Power-On Procedures for RS-232C Interface.
3. Emulator monitor command prompt

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

4. Enter F (RET) to initiate the flash memory management tool. The emulator displays prompt FM> and waits for a flash memory management tool command.

```
(S/F/L/T) ? F (RET)
FM>
```

5. Enter SL (RET) to load the system program.

```
FM> SL (RET)
```

6. Enter 1 (RET) to select PC as the host computer type, and 1 (RET) to select RS-232C (serial) interface as the interface method.

```
SELECT LOAD No. (1:PC or 2:WS) ? 1 (RET)
SELECT INTERFACE (1:RS-232C or 2:PARALLEL) ? 1 (RET)
```

7. Enter the directory containing the system file. In this example, A:\ (RET) is entered.

```
INPUT SYSTEM DIRECTORY : A:\ (RET)
```

8. Enter Y (RET) to allow system program E8000.SYS to be loaded in the emulator flash memory. Then enter system program file name E8000.SYS.

```
LOAD E8000 SYSTEM FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : E8000.SYS (RET)
COMPLETED
```

Operations **Display Message**

9. Enter Y (RET) to allow configuration file SHCNF729.SYS to be loaded in the emulator flash memory. Then enter configuration file name SHCNF729.SYS.

```
LOAD CONFIGURATION FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : SHCNF729.SYS (RET)
COMPLETED
```

10. Enter Y (RET) to allow firmware file SHDCT729.SYS to be loaded in the emulator flash memory. Then enter firmware file name SHDCT729.SYS.

```
LOAD FIRMWARE FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : SHDCT729.SYS (RET)
COMPLETED
```

11. Enter N (RET) to not load the ITRON debugger.

```
 LOAD ITRON DEBUGGER FILE OK (Y/N) ? N
(RET)
```

12. Enter N (RET) to not load the diagnostic program.

```
LOAD DIAGNOSTIC FILE OK (Y/N) ? N (RET)
```

13. Enter DIR (RET) to check whether the necessary files have been loaded.

```
FM> DIR (RET)
  <FILE ID> <STATUS>
   SYS       OK
   CONF      OK
   LAN       NO
   FIRM      OK
   TRON      NO
   DIAG      NO
   INI       OK
   MON       OK
```

14. Enter Q (RET) to terminate the flash memory management tool.

```
FM> Q (RET)

START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

15. Installation is completed.

Manual System Program Load by LAN Interface: To use the emulator, files E8000.SYS, SHCNF729.SYS, and SHDCT729.SYS must be installed in the emulator flash memory.

If the emulator is connected to the host computer via the LAN interface, the E8000 system program can be loaded with the following procedures. Transfer all files on the system floppy disk to the host computer using the FTP before installation. For details on the transfer method, refer to the host-computer user's manual. It takes approximately one minute.

Operations — **Display Message**

1. Power on the emulator. For details on the power-on procedures, refer to section 3.5.1, Power-On Procedures for LAN Interface. Confirm the emulator monitor command prompt is displayed.

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

2. Enter F (RET) to initiate the flash memory management tool. The emulator displays prompt FM> and waits for a flash memory management tool command.

```
(S/F/L/T) ? F (RET)
FM>
```

3. Enter SL (RET) to load the system program.

```
FM> SL (RET)
```

4. Enter 2 (RET) to select WS as the host computer type since the LAN interface is used.

```
 SELECT LOAD No. (1:PC or 2:WS) ? 2
(RET)
```

5. Enter the host computer name. In this example, hostname is entered.

```
 INPUT SYSTEM LOADING HOST NAME :
hostname (RET)
```

6. Enter the user name. In this example, username is entered.

```
 INPUT USER NAME : username (RET)
```

7. Enter the password. In this example, password is entered.

```
 INPUT PASS WORD : password (RET)
```

8. Enter the directory containing the system file. In this example, (RET) is entered to select the current directory of the host computer.

```
 INPUT SYSTEM DIRECTORY : (RET)
```

Operations **Display Message**

9. Enter Y (RET) to allow system program E8000.SYS to be loaded in the emulator flash memory. Then enter system program file name E8000.SYS.

```
LOAD E8000 SYSTEM FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : E8000.SYS (RET)
COMPLETED
```

10. Enter Y (RET) to allow configuration file SHCNF729.SYS to be loaded in the emulator flash memory. Then enter configuration file name SHCNF729.SYS.

```
LOAD CONFIGURATION FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : SHCNF729.SYS (RET)
COMPLETED
```

11. Enter Y (RET) to allow firmware file SHDCT729.SYS to be loaded in the emulator flash memory. Then enter firmware file name SHDCT729.SYS.

```
LOAD FIRMWARE FILE OK (Y/N) ? Y (RET)
INPUT FILE NAME : SHDCT729.SYS (RET)
COMPLETED
```

12. Enter N (RET) to not load the ITRON debugger.

```
 LOAD ITRON DEBUGGER FILE OK (Y/N) ? N
(RET)
```

13. Enter N (RET) to not load the diagnostic program.

```
LOAD DIAGNOSTIC FILE OK (Y/N) ? N (RET)
```

14. Enter DIR (RET) to check whether the necessary files have been loaded.

```
FM> DIR (RET)
  <FILE ID> <STATUS>
   SYS       OK
   CONF      OK
   LAN       NO
   FIRM      OK
   TRON      NO
   DIAG      NO
   INI       OK
   MON       OK
```

Operations / **Display Message**

15. Enter Q (RET) to terminate the flash memory management tool.

```
FM> Q (RET)

START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

16. Installation is completed.

3.8 E8000 System Program Initiation

When the emulator is turned on while S4 in DIP SW1 is turned off (to the right) and a manual system program load method is selected, the emulator enters monitor command input wait state, and the E8000 system program must be loaded and initiated by monitor commands. If S4 in DIP SW1 has been turned on (to the left) and the automatic system program load method is selected, the E8000 system program is automatically loaded and initiated.

3.8.1 Initiation on Emulator Monitor

If S is entered, followed by (RET), when the emulator is in monitor command input wait state, the E8000 system program in the emulator flash memory is initiated.

Display at E8000 System Program Initiation:

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ? S (RET)

  SH7729 E8000 (HS7729EDD81SF) Vm.n
  Copyright (C) Hitachi, Ltd. 1998
  Licensed Material of Hitachi, Ltd.

  CONFIGURATION FILE LOADING
  HARD WARE REGISTER READ/WRITE CHECK
  FIRMWARE SYSTEM LOADING
  EMULATOR FIRMWARE TEST
  ** RESET BY E8000 !
  CLOCK=8.25MHz
 MPU NAME=SH7729  MODE=19 (MD5-0=3F)
 MONITOR START ADDRESS=A3FC0000 MONITOR BUS WIDTH=32BIT
 AREA2 MEMORY=NORMAL AREA3 MEMORY=NORMAL  SDRAM TYPE=1MB-16BIT
 DRAM BUS WIDTH=32BIT
 CS2/PTK[0] PIN=CS2  CS3/PTK[1] PIN=CS3  CS4/PTK[2] PIN=CS4
 CS5/CE1A/PTK[3] PIN=CS5  STATUS[1:0]/PTJ[7:6] PIN=STATUS[1:0]
 WE[2:3]/PTK[6:7] PIN=WE[2:3]  BS/PTK[4] PIN=BS
 CAS[LL:HH]/PTJ[2:5] PIN=CAS[LL:HH]
 CAS[LL:2H]/PTJ[2:3],PTE[3:6] PIN=CAS[LL:2H]
 JTAG CLOCK=16.5MHz
 REMAINING EMULATION MEMORY S=8MB
 :
```

3.8.2 Automatic Initiation of E8000 System Program

If S4 in DIP SW1 has been turned on (to the left) and the automatic system program load method is selected, the E8000 system program is automatically loaded and initiated, and the emulator waits for an emulation command.

Display at Power On:

```
(Power on)

  E8000 MONITOR (HS8000EST02SR) Vm.n
  Copyright (C) Hitachi, Ltd. 1995
  Licensed Material of Hitachi, Ltd.

  TESTING
  RAM            0123

  SH77295 E8000 (HS7729EDD81SF) Vm.n
  Copyright (C) Hitachi, Ltd. 1998
  Licensed Material of Hitachi, Ltd.

  CONFIGURATION FILE LOADING
  HARD WARE REGISTER READ/WRITE CHECK
  FIRMWARE SYSTEM LOADING
  EMULATOR FIRMWARE TEST
  ** RESET BY E8000 !
CLOCK=8.25MHz
 MPU NAME=SH7729  MODE=19 (MD5-0=3F)
 MONITOR START ADDRESS=A3FC0000 MONITOR BUS WIDTH=32BIT
 AREA2 MEMORY=NORMAL AREA3 MEMORY=NORMAL  SDRAM TYPE=1MB-16BIT
 DRAM BUS WIDTH=32BIT
 CS2/PTK[0] PIN=CS2  CS3/PTK[1] PIN=CS3  CS4/PTK[2] PIN=CS4
 CS5/CE1A/PTK[3] PIN=CS5  STATUS[1:0]/PTJ[7:6] PIN=STATUS[1:0]
 WE[2:3]/PTK[6:7] PIN=WE[2:3]  BS/PTK[4] PIN=BS
 CAS[LL:HH]/PTJ[2:5] PIN=CAS[LL:HH]
 CAS[LL:2H]/PTJ[2:3],PTE[3:6] PIN=CAS[LL:2H]
 JTAG CLOCK=16.5MHz
 REMAINING EMULATION MEMORY S=8MB
:
```

If the E8000 system program is automatically initiated without being loaded to the emulator flash memory, after displaying an error message, the emulator enters monitor command input wait state. Make sure to load the E8000 system program to the emulator flash memory before initiation.

```
  *** E8000 SYSTEM PROGRAM NOT FOUND
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ? _
```

Section 4 Operating Examples

4.1 Emulator Operating Examples

This section covers explanations on how to operate the emulator using examples. Sections 4.2, Basic Examples and 4.3, Application Examples are based on the following user program. These examples assume that the emulator is connected to the host computer by a LAN interface and is used with a TELNET connection.

```
  ADDR      CODE           MNEMONIC    OPERAND
00201000    E00A           MOV         #0A,R0
00201002    E101           MOV         #01,R1
00201004    E201           MOV         #01,R2
00201006    D405           MOV.L       0020101C,R4
00201008    6323           MOV         R2,R3
0020100A    321C           ADD         R1,R2
0020100C    2426           MOV.L       R2,@-R4
0020100E    6133           MOV         R3,R1
00201010    70FF           ADD         #FF,R0
00201012    8800           CMP/EQ      #00,R0
00201014    8BF8           BF          00201008
00201016    0009           NOP
00201018    AFFE           BRA         00201018
0020101A    0009           NOP
0020101C    0020           .DATA.W     0020
0020101E    FFFC           .DATA.W     FFFC
```

Store the user program in the host computer before initiating the emulator and download it to the emulator. In these examples, the IP address is set to 128.1.1.1.

CAUTION

In these examples, the IP address is set to 128.1.1.1 to 128.1.1.10. For the actual host computer, an IP address available on the network connected to the emulator must be specified. If an unavailable IP address is specified, the network will have problems.

4.2 Basic Examples

4.2.1 Connecting to the LAN Host Computer

The following host name and IP address are examples. Specify the actual host computer name and IP address of the host computer.

Operations / **Display Message**

1. Specify the host name and IP address of the host computer to which the emulator is to be connected by the LAN interface. Enter the F command to initiate the flash memory management tool in the monitor command input wait state.

```
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/F/L/T) ? F (RET)
FM>_
```

2. Enter LH (RET) to store the host name and IP address of the host computer.

```
FM>LH (RET)
NO <HOST NAME> <IP ADDRESS>      NO <HOST
01                               02
03                               04
05                               06
07                               08
09
    E8000 IP ADDRESS = 128.1.1.1
PLEASE SELECT NO.(1-9/L/E/Q/X) ? _
```

3. Enter 1 as the selection number, HITACHI (RET) as the host name, and 128.1.1.10 (RET) as the IP address. After that, the emulator prompts the user to select another number.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? 1 (RET)
01 HOST NAME  HOST_A    ? HITACHI (RET)
01 IP ADDRESS 128.1.1.1 ? 128.1.1.10 (RET)
PLEASE SELECT NO.(1-9/L/E/Q/X) ? _
```

4. Enter E (RET) to enable the settings and to exit interactive mode.

```
PLEASE SELECT NO.(1-9/L/E/Q/X) ? E (RET)
```

5. The emulator confirms whether to save the settings in the configuration file with the above settings.

```
LAN CONFIGURATION FILE WRITE OK (Y/N) ? _
```

6. Enter Y (RET) to save the settings.

```
CONFIGURATION FILE WRITE OK (Y/N) ? Y (RET)
FM>
```

7. Enter Q (RET) to terminate the flash memory management tool and enter the monitor command input wait state.

```
FM>Q (RET)

 START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
     (S/F/L/T) ? _
```

8. Enter S (RET) to re-initiate the emulator. The emulator is re-initiated, and waits for an emulation command.

```
     (S/F/L/T) ? S (RET)

 SH7729 E8000 (HS7729EDD81SF) Vm.n
 Copyright (C) Hitachi, Ltd. 1998
 License Material of Hitachi, Ltd.

 CONFIGURATION FILE LOADING
 HARDWARE REGISTER READ/WRITE CHECK
 FIRMWARE SYSTEM LOADING
 EMULATOR FIRMWARE LOADING
 EMULATOR FIRMWARE TEST
 ** RESET BY E8000 !
 CLOCK = 8.25MHz
 MPU NAME=SH7729  MODE=19 (MD5-0=3F)
 MONITOR START ADDRESS=03FC0000 MONITOR BUS
 AREA2 MEMORY=NORMAL AREA3 MEMORY=NORMAL
 DRAM BUS WIDTH=32BIT
 CS2/PIK[0] PIN=CS2  CS3/PIK[1] PIN=CS3
 CS5/CE1A/PIK[3] PIN=CS5  STATUS[1:0]/
 WE[2:3]/PIK[6:7] PIN=WE[2:3]  BS/PIK[4] PIN=BS
 CAS[LL:HH]/PIJ[2:5] PIN=CAS[LL:HH]
 CAS[LL:2H]/PTJ[2:3] PTE[3, 6] PIN=CAS[LL:2H]
 JTAG CLOCK=16.5MHz
 REMAINING EMULATION MEMORY S=8MB
 :
```

4.2.2 Specifying the MCU Operating Mode

Specify the emulator operating mode by the following procedures. SH7729 is specified as the CPU, the operating mode is 19, monitor program start address is H’A0100000, bus width is 16 bits, area 2 is SDRAM, area 3 is SDRAM, SDRAM type is 2-Mbyte 16 bits, CS2 to CS6 is used as CS pin, STATUS[1:0]/PTJ[7:6] pin is used as STATUS[1:0], WE[2:3]/PTK[6:7] pin as WE[2:3], BS/PTK[4] pin as BS pin, CAS[LL:HH]/PTJ[2:5] as CAS[LL:HH] pin, CAS[LL:2H]/PTJ[2:3], PTE[3,6] pin as CAS[LL:HH] pin, and JTAG clock is 4.12 MHz.

Operations	Display Message
1. Enter MD;C (RET) to specify the emulator operating mode.	`:`***`MD;C (RET)`***
2. .To use the SH7729, enter 1 (RET).	`CPU NAME (1:SH7729,2:SH7709A) ?` ***`1 (RET)`***
3. As the operating mode, enter 19 (RET).	`E8000 MODE (0-3F) ?` ***`19 (RET)`***
4. As the monitor program start address, enter A0100000 (RET).	`MONITER START ADDR ?` ***`A0100000 (RET)`***
5. As the bus width, enter 16BIT (RET).	`MONITER BUS WIDTH (1:16BIT,2:32BI)` ***`1 (RET)`***

6. SDRAM is installed at area 2, so enter 3 (RET).	`AREA2 MEMORY (1:NORMAL,2:DRAM,3:SDRAM) ?`
	`3 (RET)`

7. SDRAM is installed at area 3, so enter 3 (RET).	`AREA3 MEMORY (1:NORMAL,2:DRAM,3:SDRAM) ?`
	`3 (RET)`

8 SDRAM type is 1-Mbyte 16 bits, so enter 1 (RET).	`SDRAM TYPE (1:1MB-16BIT, –) ? 1 (RET)`

9. To use CS2/PTK[0] pin, CS3/PTK[1] pin, CS4/PTK[2] pin, CS5/CE1A/PTK[3] pin, and CS6/CEIB pin as the CS pin, enter 1 (RET).	`CS2/PTK[0] PIN (1:CS2,2:PTK[0]) ?` ***`1 (RET)`***
	`CS3/PTK[1] PIN (1:CS2,2:PTK[0]) ?` ***`1 (RET)`***
	`CS4/PTK[2] PIN (1:CS2,2:PTK[1]) ?` ***`1 (RET)`***
	`CS5/CE1A/PTK[3] PIN (1:CS2,2:CE1A,3:–) ?`
	`1 (RET)`
	`CS6/CE1B PIN (1:CS2,2:1:CE1B)?` ***`1 (RET)`***

10. To use STATUS[1:0]/PTJ[7:6] pin as STATUS[1:0], enter 1 (RET).	`STATUS [1:0]/PTJ [7:6] PIN (1:STATUS[1:–)`
	`1 (RET)`

11. To use WE[2:3]/PTK[6:7] pin as WE[2:3] pin, enter 1 (RET).

```
WE[2:3]/PTK[6:7] PIN (1:WE[2:3],
2:PTK[6:7]) ?                                1 (RET)
```

12. To use BS/PTK[4] pin as BS pin, enter 1 (RET).

```
BS/PTK[4] PIN (1:BS,2:PTK[4]) ?    1 (RET)
```

13. To use CAS[LL:HH]/PTJ[2:5] pin as the CAS[LL:HH] pin, enter 1 (RET).

```
CAS[LL:HH]/PTJ[2:5] PIN (1:CAS[LL:HH],
2:PTJ[2:5]) ?                                1 (RET)
```

14. To use CAS[LL:2H]/PTJ[2:3], PTE[3,6] pin as the CAS[LL:2H] pin, enter 1 (RET).

```
CAS[LL:2H]/PTJ[2:3],PTE[3.6] PIN
(1:CAS[LL:2H],2:PTJ[2:3],PTE[3:6]) ?
                                             1 (RET)
```

15. To use 4.12 MHz JTAG clock, enter 3 (RET).

```
JTAG CLOCK
(1:16.5MHz/2:8.25MHz/3:4.12MHz) ? 3 (RET)
```

16. When the above items are all entered, the confirmation message to store this state into the emulator flash memory is displayed. When Y is entered, the items are stored, and when the emulator is initiated after configuration information storage, it emulates in the stored operating mode. When N is entered, the MODE command is terminated, and the emulator enters command input wait state.

```
CONFIGRATION STORE OK (Y/N)? Y (RET)
```

17. The E8000 system program terminates after the MCU operating mode is set, and must then be re-initiated.

```
START E8000
  S:START E8000
  F:FLASH MEMORY TOOL
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
      (S/F/L/T) ?  _
```

18. To re-initiate, enter S (RET).

```
    (S/F/L/T) ? S (RET)
```

4.2.3 Allocating Standard Emulation Memory and Specifying Attributes

To load the user program to memory and run the user program, allocate standard emulation memory by the following procedures:

Operations / **Display Message**

1. Enter MAP 200000 5FFFFF;S (RET) to allocate standard emulation memory to addresses H'200000 to H'5FFFFF.

```
:MAP 200000 5FFFFF;S (RET)
```

2. The message shown on the right, which indicates that memory allocation has been completed is displayed.

```
REMAINING EMULATION MEMORY S=4MB
```

3. Enter MAP (RET) to display the attributes of all the memory areas.

```
:MAP (RET)
 00200000-005FFFFF;S
 INTERNAL I/O 04000000-04FFFFFF
E0000000-FFFFFFFF
 REMAINING EMULATION MEMORY S=4MB
:
```

4.2.4 Loading the User Program

Connect the emulator to the host computer using the FTP server and load the user program by the following procedures. This example assumes that in host computer HITACHI, the user name is defined as E8000 and its password as PASSWORD.

Operations — **Display Message**

1. Enter FTP HITACHI (RET) to connect the emulator to the host computer using the FTP server.

```
:FTP HITACHI (RET)
```

2. The emulator asks for the user name. Enter E8000 (RET).

```
 Username: E8000 (RET)
```

3. The emulator asks for the password. Enter PASSWORD (RET).

```
 Password: PASSWORD (RET)
```

4. The message shown on the right, which indicates that the emulator and the host computer have been connected is displayed. The prompt becomes FTP>.

```
 login command success
FTP>
```

5. To load program PROGRAM.MOT, enter LAN_LOAD ;S:PROGRAM.MOT (RET). This example assumes that the load module is S type.

```
FTP> LAN_LOAD ;S:PROGRAM.MOT (RET)
```

6. While loading, the address to which the program is being loaded is displayed, as shown on the right.

```
LOADING ADDRESS = xxxxxxxx
```

7. When the program has been loaded, the start address of the program (TOP ADDRESS) and its end address (END ADDRESS) are displayed.

```
TOP ADDRESS = 00201000
END ADDRESS = 0020101F
```

8. Entering BYE (RET) terminates the FTP server connection. The message shown on the right is displayed.

```
FTP>BYE (RET)
 bye command success
 :
```

4.2.5 Executing the Program

Execute the loaded program by the following procedures:

Operations **Display Message**

1. Set the initial values of the registers. Enter .SP (RET) to set the stack pointer (SP register) to H'0020FFFC.

```
:.SP (RET)
 R15(SP)=xxxxxxxx ? 0020FFFC (RET)
 PC=xxxxxxxx ? _
```

2. The emulator asks for the program counter value. Enter 201000 (RET) as the program counter value.

```
 PC=xxxxxxxx ? 201000 (RET)
```

3. The emulator then asks for the status register value. In this example, other registers need not to be set or changed, therefore, enter . (RET) to exit this interactive mode.

```
 SR=xxxxxxxx:-PIB0000000000----IIII00-? .
(RET)
```

4. Enter GO (RET) to execute the loaded program from the address pointed to by the PC. While the program is executed, the current program counter value is displayed.

```
:GO (RET)

**PC=00201018
```

5. Enter the (BREAK) key to terminate program execution.

```
(BREAK)
```

6. The contents of the program counter, status register, control registers, general registers R0 to R15, and DSP registers are displayed at GO command termination. RUN-TIME shows the duration of program execution from GO command execution to (BREAK) key entry. BREAK KEY shows that execution has been terminated because the (BREAK) key was entered.

```
PC=00201018 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=A0000000 MACH=000
RS=00000000 RE=00000000 MOD=00000000
R0-7  00000000 00000059 00000090 00000059
R8-15 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:-----------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:00M:01S:667859US:200NS
 +++:BREAK KEY
```

4.2.6 Setting a Software Breakpoint

Execution of the GO command can be stopped immediately before executing a particular address by setting a software breakpoint by the following procedures:

Operations **Display Message**

1. Enter BREAK 201010 (RET) to terminate the GO command immediately before executing the instruction at address H'201010.

```
:BREAK 201010 (RET)
```

2. Restart program execution from address H'201000. This can be done in two ways: one is to first set the program counter to H'201000, then enter the GO command to execute the program, and the other is to enter the start address directly.

```
:.PC 201000 (RET)
:GO (RET)

or

:GO 201000 (RET)
```

3. The GO command execution terminates immediately before the instruction at address H'201010 is executed. The data shown on the right is displayed. BREAKPOINT shows that the GO command execution was terminated due to a software breakpoint.

```
PC=00201010 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:00M:00S:000012US:800NS
 +++:BREAKPOINT
:
```

4.2.7 Executing a Single Step

A single step can be executed using the single-step function by the following procedures:

Operations / **Display Message**

1. The program counter points to the next address to be executed when the GO command terminates. Entering STEP (RET) here executes only a single instruction.

```
:STEP (RET)
```

2. The information shown on the right is displayed. 00201010 ADD #FF,R0 shows the address and mnemonic code executed by the STEP command, and STEP NORMAL END shows that single-step execution has terminated.

```
PC=00201012 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 00201010               ADD    #FF, R0
 +++:STEP NORMAL END
:
```

3. To continue single-step execution, enter (RET). Until another command is entered, the STEP command continues by entering (RET).

```
:(RET)
PC=00201014 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 00201012              CMP/EQ    #00, R0
 +++:STEP NORMAL END
```

4.2.8 Setting Hardware Break Conditions

Various hardware break conditions can be specified by the following procedures:

Operations / **Display Message**

1. Enter BREAK- (RET) to cancel the software breakpoint.

```
:BREAK- (RET)
```

2. To confirm the cancellation, execute the BREAK command (enter BREAK (RET)). *** 45: NOT FOUND shows that no software breakpoint is set.

```
:BREAK (RET)
 ***45:NOT FOUND
:
```

3. To specify that program execution should terminate when data is written to address H'0020FFF8, enter BREAK_CONDITION_UBC1 A=0020FFF8 W (RET).

```
:BREAK_CONDITION_UBC1 A=0020FFF8 W (RET)
```

4. Enter GO 201000 (RET) to start executing the program from address H'201000.

```
:GO 201000 (RET)
```

5. When the break condition is satisfied, the information shown on the right is displayed. BREAK CONDITION UBC1 shows that GO command execution has terminated because the break condition was satisfied.

```
PC=00201014 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=A0000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:00M:00S:000024US:000NS
 +++:BREAK CONDITION UBC1
:
```

4.2.9 Displaying Trace Information

Trace information acquired during program execution can be displayed in various ways as follows:

Operation **Display Message**

1. To display the instruction mnemonic information, enter TRACE (RET).

```
:TRACE (RET)
```

```
 IP            ADDR                             MNEMONIC    OPERAND
 *-D'000008 00201000                            ADD         #FF,R0
 *-D'000007 00201002                            CMP/EQ      #00,R0
 *-D'000006 00201004                            BF          00201008
 *-D'000005 00201008                            MOV         R2,R3
 *-D'000004 0020100A                            ADD         R1,R2
 *-D'000003 0020100C                            MOV.L       R2,@-R4
 *-D'000002 0020100E                            MOV         R3,R1
 *-D'000001 00201010                            ADD         #FF,R0
 * D'000000 00201012                            CMP/EQ      #00,R0
```

2. To display the trace information in bus-cycle units, enter TRACE ;B (RET).

```
:TRACE ;B (RET)
```

```
      BP        AB        DB     RW   IRL    NMI RES BRQ VCC PRB  TIME_STAMP
   -D'000009 00201000 E00AE101  R  111111  1   1   1   1  1111  00H00M00S
   -D'000008 00201004 E201D405  R  111111  1   1   1   1  1111  00H00M00S
   -D'000007 00201008 6323321C  R  111111  1   1   1   1  1111  00H00M00S
   -D'000006 0020100C 24266133  R  111111  1   1   1   1  1111  00H00M00S
   -D'000005 0020101C 0F100000  R  111111  1   1   1   1  1111  00H00M00S
   -D'000004 00201010 70FF0000  R  111111  1   1   1   1  1111  00H00M00S
   -D'000003 00201014 8BF80009  R  111111  1   1   1   1  1111  00H00M00S
   -D'000002 0020FFF8 00000002  W  111111  1   1   1   1  1111  00H00M00S
   -D'000001 00201018 AFFE0009  R  111111  1   1   1   1  1111  00H00M00S
```

3. To temporarily stop the trace information display, enter (CTRL) + S. To continue the trace information display, enter (CTRL) + Q.
   (CTRL) + S and (CTRL) + Q are also effective with other information displays.

```
:TRACE ;B (RET)
 (CTRL) + S (stops trace information display)
 (CTRL) + Q (restarts trace information
                 display)
```

4.3 Application Examples

4.3.1 Break with Pass Count Condition

The pass count condition can be set to a breakpoint by the following procedures:

Operations **Display Message**

1. Enter BREAK 201012 5 (RET) to terminate program execution immediately after address H'201012 is passed five times.

```
:BREAK 201012 5 (RET)
```

2. To start execution from address H'201000, enter GO 201000 (RET).

```
:GO 201000 (RET)
```

3. When address H'201012 is passed five times, the data shown on the right is displayed and GO command execution terminates.

```
PC=00201012 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:00M:00S:000075US:200NS
 +++:BREAKPOINT
:
```

4. Entering BREAK (RET) displays the breakpoint address, the specified count, and the pass count, as shown on the right. The pass count is cleared when the GO command is entered again.

```
:BREAK (RET)
 <ADDR>    <CNT>    <PASS>
 00201012   0005     0005
:
```

4.3.2 Conditional Trace

The acquisition of trace information during program execution can be limited by the following procedures:

Operations | **Display Message**

1. Enter BREAK - (RET) to cancel the breakpoint set in the example of section 4.3.1, Break with Pass Count Condition.

```
:BREAK - (RET)
```

2. Enter TRACE_CONDITION_A1 A=201010:201014 ;R (RET) to get trace information only while the program counter is between addresses H'201010 and H'201014.

```
:TRACE_CONDITION_A1 A=201010:201014;R (RET)
```

3. Enter GO 201000 (RET) to start executing the program, then the (BREAK) key to terminate the program execution.

```
:GO 201000(RET)
 ** PC = 00201010

(BREAK)
PC=00201018 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:00M:04S:753923US:200NS
 +++:BREAK KEY
:
```

4. Enter TRACE ;B (RET) to display the trace information acquired under the specified condition. :***TRACE;B (RET)***

```
    BP       AB        DB     RW    IRL     NMI  RES  BRQ   VCC   PRB       TIME_STAMP
-D'000019 00201010 70FF8800  R   111111   1     1    1     1   1111   00H00M00S000000
-D'000018 00201014 8BF80009  R   111111   1     1    1     1   1111   00H00M00S000000
```

5. Enter TRACE_CONDITION_A1 - (RET) to cancel the trace acquisition condition. :***TRACE_CONDITION_A1 - (RET)***

4.3.3 Parallel Mode

During program execution in parallel mode, the memory contents can be displayed or modified by the following procedures:

Operations / **Display Message**

1. After executing the GO command, enter (RET) to move to parallel mode.

```
:GO 201000 (RET)
 ** PC = xxxxxxxx
(RET)
#                    (Moves to parallel mode)
```

2. Enter DUMP 202000 20200F (RET) to display the memory contents from addresses H'202000 to H'20200F in parallel mode.

```
#DUMP 202000 20200F (RET)
           (Dump display)
           . . .
```

3. Enter MEMORY 201019 FD (RET) to modify the memory contents of address H'201019 to H'FD in parallel mode.

```
#MEMORY 201019 FD (RET)
# _
```

4. To exit from parallel mode, enter END (RET).

```
#END (RET)
 ** PC = xxxxxxxx
```

5. To terminate program execution, enter the (BREAK) key.

```
** PC = xxxxxxxx
(BREAK)
** PC = xxxxxxxx
PC=00201018 SR=700000F0:-P1B000000000
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=00000000 MACH=00000000 MACL=0000
RS=00000000 RE=00000000 MOD=00000000
R0-7 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000
R0_BANK0-R3_BANK0 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000
DSR=000000F1:------------------------
A0G=00 A0=00000000 M0=00000000 X0=00
A1G=00 A1=00000000 M1=00000000 X1=00
 RUN-TIME=D'0000H:03M:48S:662384US:000NS
 +++:BREAK KEY
: _
```

6. Enter DISASSEMBLE 201000 20101F (RET) to confirm that the program has been changed by memory modification in parallel mode.

```
:DISASSEMBLE 201000 20101F (RET)
```

```
   ADDR       CODE                     MNEMONIC      OPERAND
 00201000     E00A                       MOV         #0A,R0
 00201002     E101                       MOV         #01,R1
 00201004     E201                       MOV         #01,R2
 00201006     D405                       MOV.L       0020101C,R4
 00201008     6323                       MOV         R2,R3
 0020100A     321C                       ADD         R1,R2
 0020100C     2426                       MOV.L       R2,@-R4
 0020100E     6133                       MOV         R3,R1
 00201010     70FF                       ADD         #FF,R0
 00201012     8800                       CMP/EQ      #00,R0
 00201014     8BF8                       BF          00201008
 00201016     0009                       NOP
 00201018     AFFD                       BRA         00201016 (Modified)
 0020101A     0009                       NOP
 0020101C     FFFF                       .DATA.W     FFFF
 0020101E     0000                       .DATA.W     0000
```

4.3.4 Searching Trace Information

A particular part of the acquired trace information can be searched for, using the TRACE_SEARCH command as follows:

Operation	Display Message
Enter TRACE_SEARCH A=201018 (RET) to display the parts of trace information in which the address bus value is H'201018.	:***`TRACE_SEARCH A=201018 (RET)`***

```
   BP       AB       DB     RW    IRL     NMI RES BRQ VCC PRB  TIME_STAMP
-D'004091 00201018 AFFD0009 R  11111111  1   1   1   1  1111 00H00M00S000000US000NS
-D'004090 00201018 AFFD0009 R  11111111  1   1   1   1  1111 00H00M00S000000US000NS
-D'004067 00201018 AFFD0009 R  11111111  1   1   1   1  1111 00H00M00S000000US000NS
         . . .
```

Part II Emulator Function Guide

Section 1 Emulator Functions

1.1 Precautions concerning the MODE Command Settings

When debugging the user system using the emulator, the MCU operating mode and the user system information must be set with the MODE command. If the user system information and the MODE command setting does not match,

```
EMULATOR FIRMWARE TEST
***INVALID FIRMWARE SYSTEM
```

is displayed, and the emulator initiation message stops. Press (CTRL) + C keys (forced break), so that the emulator enters the command input wait state. When the prompt (:) is displayed, enter the MODE command as follows:

:***MODE;C(RET)***

The user system information can be set in the interactive mode. For details, refer to section 7.2.27, MODE [MD], Specifies and Displays MCU Operating Mode.

1.1.1 MODE Command Setting

CAUTION

1. The monitor program area occupies 128 kbytes of user area.
2. The monitor program area must be allocated to the P2 area (cache or TLB invalid area), and after the address H’A0040000.

Monitor Program Allocation:

When the emulator is used, the monitor program area uses 128 kbytes of the user area. The monitor program must be allocated to an area in which the user program is not used with a 256-kbytes boundary. When the input message of the MODE command,

MONITOR START ADDR ?

is displayed, enter the start address of the monitor program. The monitor program occupies 128 kbytes of the user area from the input address. Figure 1.1 shows how to allocate the monitor program.

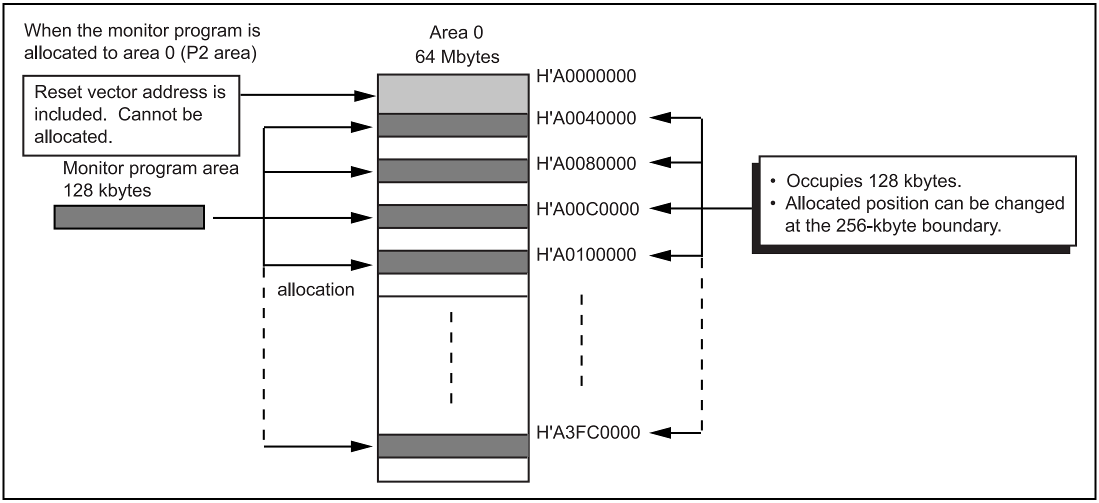

When the monitor program is allocated to area 0 (P2 area)
Area 0
64 Mbytes
Reset vector address is included. Cannot be allocated.
Monitor program area
128 kbytes
allocation
H'A0000000
H'A0040000
H'A0080000
H'A00C0000
H'A0100000
H'A3FC0000
• Occupies 128 kbytes.
• Allocated position can be changed at the 256-kbyte boundary.

Figure 1.1 Monitor Program Allocation

Note: Do not set a software breakpoint in the monitor program.

Memory Type:

When using the emulator, the memory type installed in areas 2 and 3 must be specified using the MODE command. When the message,

AREA2 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM) ?
AREA3 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM, 4:EDO-DRAM) ?

is displayed, enter the memory type installed in areas 2 and 3. When the SDRAM or DRAM is used, the SDRAM type (capacity and bus width) and DRAM type (bus width) must also be specified when the following message is displayed:

SDRAM TYPE (1:1MB-16BIT or 4MB-16BIT/:2 2MB-8BIT or 8MB-8BIT/3:4MB-4BIT/4:256KB-16BIT or 2MB-32BIT) ?

DRAM TYPE (1:16BIT, 2:32BIT) ?

Multiplexed Signals:

The usage of the multiplexed signals listed in table 1.1 must be selected using the MODE command when they are used by the user system.

Table 1.1 Multiplexed Signals

Pin No.	Signal Name	Pin No.	Signal Name	Pin No.	Signal Name
98	CS2/PTK[0]	91	WE2/DQMUL/ ICIORD/PTK[6]	113	CASHH/PTJ[5]
99	CS3/PTK[1]	92	WE3/DQMUL/ ICIOWR/PTK[7]	116	CAS2L/PTE[6]
100	CS4/PTK[2]	87	BS/PTK[4]	117	CAS2H/PTE[3]
101	CS5/CE1A/PTK[3]	108	CASLL/CASL/PTJ[2]		
157	STATUS[0]/PTJ[6]	110	CASLH/CASU/PTJ[3]		
158	STATUS[1]/PTJ[7]	112	CASHL/PTJ[4]		

CS2/PTK[0] PIN(1:CS2, 2:PTK[0]) ?
CS3/PTK[1] PIN(1:CS3, 2:PTK[1]) ?
CS4/PTK[2] PIN(1:CS4, 2:PTK[2]) ?
CS5/CE1A/PTK[3] PIN(1:CS5/2:CE1A or PTK[3]) ?
STATUS[1:0]/PTJ[7:6] PIN(1:STATUS[1:0]/2:PTJ[7:6])
WE[2:3]/PTK[6:7] PIN(1:WE[2:3], 2:PTK[6:7]) ?
BS/PTK[4] PIN(1:BS, 2:PTK[4]) ?
CAS[LL:HH]/PTJ[2:5] PIN(1:CAS[LL:HH]. 2:PTJ[2:5]) ?
CAS[LL:2H]/PTJ[2:3], PTE[3, 6] PIN(1:CAS[LL:2H], 2:PTJ[2:3], PTE[3:6]) ?

When the input messages are displayed, enter the number of the signal to be used.

Hitachi-UDI:

The MCU internal Hitachi-UDI (Hitachi user debugging interface) is used for firmware transfer and execution of the GO and MEMORY commands. The Hitachi-UDI clock value can be selected by responding to the MODE command input message:

JTAG CLOCK (1:16.5MHz/2: 8.25MHz/3: 4.12MHz) ?

However, the clock value must be less than half of the MCU internal clock (the clock generated from the MCU internal divider 1).

1.1.2 Precautions when Debugging the User System

Trace:

1. Traced data cannot be disassembled.
2. Trace information cannot be acquired for an internal bus. However, when the AUD trace function is used, trace information can be acquired for a branch source and a branch destination address including the internal bus.
3. The AUD CLK value used for AUD trace must satisfy the following relationship with respect to the MCU internal clock.

   Internal clock/4 • AUD CLK • internal clock
4. When a port pin that is multiplexed with data is used, data input or output from the port pin is displayed as the traced data.
5. Even when the refresh cycle occurs while the SDRAM is being accessed, SDRAM CYCLE is displayed as the trace display.

Emulation memory:

1. Emulation memory and user memory cannot be allocated to the same CS area.
2. Insert software waits by using wait control register 2 (WCR2) of the bus state controller when using emulation memory. The number of waits to insert depends on the external clock as shown below. However, the number of waits must be 10 when the emulation memory is allocated to the same area as the monitor program.

CKIO (MHz)	to 15	20 to 25	33	40 to 45	50 to 55	60	66
Number of waits	1	2	3	4	6	8	10

3. When the emulation memory is allocated to two areas, the bus width of each emulation memory must be made the same by using the bus state controller setting.
4. When the emulation memory is allocated to the same area (CSx) as the monitor program, do not let CSx overlap the monitor program area set with the MODE command (256-kbyte boundary).

User Interface:

1. The pins listed in table 1.2 are used by the emulator, and are not connected to the user system.

Table 1.2 Signals Used by the Emulator

Pin No.	Signal Name	Pin No.	Signal Name
4	XTAL2	139	TCK/PTF[4]/PINT[12]
5	EXTAL2	146	CAP1
120	TDO/PTE[0]	149	CAP2
127	ASEMD0/PTG[6]	145	VCC-PLL1
128	ASEBKAK/PTG[5]	150	VCC-PLL2
136	TRST/PTF[7]/PINT[15]	147	VSS-PLL1
137	TMS/PTF[6]/PINT[14]	148	VSS-PLL2
138	TDI/PTF[5]/PINT[13]		

2. The pins listed in table 1.3 are used by the emulator when the AUD trace function is used, and released to the user when the AUD trace function is not used.

Table 1.3 Signals Used for AUD Trace Function

Pin No.	Signal Name	Pin No.	Signal Name
135	AUDATA[0]/PTG[0]	130	AUDATA[3]/PTG[3]
133	AUDATA[1]/PTG[1]	151	AUDCK/PTH[6]
131	AUDATA[2]/PTG[2]	94	AUDSYNC/PTE[7]

3. The MCU signals at the end of the evaluation chip board are all directly conneted to the user system. However, a delay occurs for the RESETP, RESETN, NMI, WAIT, and BREQ signals because they are connected to the user system via the evaluation chip board. The delay time is shown in table 1.4.

Table 1.4 Delay Time for Signals Connected via the Evaluation Chip Board

Signal Name	Delay Time (ns)
RESETP	18.9
RESETM	10.1
NMI	10.1
BREQ	10.1
WAIT	6.7

4. When the MCU accesses the emulator firmware, the RD signal and WE0 to WE3 signals, as well as the address bus and data bus, are activated. These signals are directly connected to the user system. Therefore, when using them for bus control, design the user system so that no bus collisions occur.

Performance Function:

When using the performance function, condition setting must be done on the external bus. When an internal bus is set as a condition, the execution time cannot be measured.

Hardware Break Function:

1. During user program execution, the user break controller cannot be used.
2. The BREAK_CONDITION_UBC command must be used to set a hardware break in a user program in the SDRAM memory area.

Hardware Standby Function:

No hardware standby function is supported.

Clock Operating Mode:

1. Clock operating modes 2 and 4 are not supported. When using the crystal oscillator, install the crystal oscillator in the socket on the evaluation chip board, and use either clock operating mode 0, 1, or 3.
2. When the clock operating mode is switched from some other value to 7, be sure to turn off the E8000 station then restart it, to avoid collisions between the CKIO output and the CKIO input from the user system. If the user power supply is turned off while in clock operating mode 7, turn the E8000 station off, then restart it after supplying power to the user system, to avoid collisions of CKIO.

1.2 Overview

The emulator is an efficient software and hardware development tool for systems based on Hitachi microcomputer SH7729/SH7709A (hereafter called MCU).

The MCU incorporates a vast range of functions on a single chip. In addition to high-speed CPU and DSP (SH7729 only), these include peripheral functions such as an interrupt controller, user break controller (UBC), bus state controller, DMAC, timer, realtime clock, watchdog timer, serial communication interface 0/1/2, Hitachi user debugging interface (Hitachi-UDI), advanced user debugger (AUD), I/O port, X/Y memory (SH7729 only), A/D converter, and D/A converter.

Table 1.5 MCU Functions

Support Device	SH7729	SH7709A
Maximum controllable memory size	4 Gbytes	4 Gbytes
Maximum external bus width	32 bits	32 bits
DSP	32-bit/40-bit internal bus	None
MMU	4 Gbytes	4 Gbytes
X/Y memory	8 kbytes each	None
Cache memory	16 kbytes	16 kbytes
DMAC	4 ch	4 ch
Interrupt controller	External interrupt source: 7 (NMI, IRL0 to IRL3, IRQ4, IRQ5)	
User break controller	2 ch	2 ch
Bus state controller	Internal	Internal
Watchdog timer	Internal	Internal
Timer	2 ch	2 ch
RTC	Internal	Internal
A/D converter	10 bits x 8 ch	10 bits x 8 ch
D/A converter	8 bits x 2 ch	8 bits x 2 ch
Hitachi user debug interface	Internal	Internal
Advanced user debugger	Internal	Internal
Serial communication interface	Asynchronous/synchronous: 3 ch	
I/O port	I/O port: 59, input-only port: 37, output-only port: 3	

1.3 Overview

The emulator operates on the user system in the same way as the MCU, and enables realtime emulation of the user system with functions for debugging hardware and software.

The emulator consists of an emulator (E8000) station, and an evaluation chip board. The evaluation chip board is directly connected to the user system.

1.4 Specifications

The main features of the emulator are its emulation functions and its host computer interface functions, as listed in tables 1.6 and 1.7, respectively.

Table 1.6 Emulation Functions

Command Type	Command	Function	Reference section
Realtime emulation	GO	Performs realtime emulation in the following cases. The operating frequency is 66.0 MHz at max. • Executes until a hardware or software break condition is satisfied, or until the (CTRL) + C or (BREAK) key is pressed. • Cycle-reset mode: Executes while the reset signal is sent to the MCU at fixed intervals. This mode is effective to observe waveforms immediately after reset. • Parallel mode: Displays trace data and modifies memory contents during emulation.	7.2.21
	EXECUTION_ MODE	Specifies execution mode.	7.2.19

Table 1.6 Emulation Functions (cont)

Command Type	Command	Function	Reference section
Break condition setting	BREAK_CONDITION_UBC	Sets hardware break conditions (1). • Normal break: Execution is forcibly stopped when the specified conditions are satisfied (a maximum of two points). — Address bus value or data bus value — PC (program counter) value — Read/write condition — Pass count specification (for BREAK_CONDITION_UBC1 only) • Mask specification for address and data conditions — Bit-by-bit specification is enabled for address, PC, or data conditions.	7.2.8
	BREAK_CONDITION_A,B,C	Sets hardware break conditions (2). • Execution is forcibly stopped when the specified conditions are satisfied (a maximum of 24 points). — Address bus value or data bus value — Read/write condition — Delay count (1 channel) — Pass count specification (8 channels) — External probe value — System control signals (NMI, IRL0 to IRL3, IRQ4, IRQ5) — NOT condition	7.2.7
	BREAK_CONDITION_A,B,C	External probe signal • PRB1 to PRB4	
	BREAK	Sets software break conditions. • Sets up to 255 breakpoints. • Sets pass count.	7.2.6

Table 1.6 Emulation Functions (cont)

Command Type	Command	Function	Reference section
Trace data acquisition and display	TRACE	Displays the following data for each bus cycle: • Address bus value • Data bus value • I/O control signals • External probe value • Time stamp (20 ns, 1.6 µs, 52 µs)	7.2.39
	TRACE_ CONDITION_ A,B,C	Sets, displays, and cancels trace condition. • Traces data only when a condition is satisfied (a maximum of 24 points). — Address bus value (NOT condition) — Read/write condition — External probe value — System control signal — NOT condition — Delay count • Stops trace when a trace stop condition is satisfied (a maximum of 24 points). — Address bus value or data bus value — Read/write condition — External probe value — System control signals — NOT condition — Delay count	7.2.40
		• Low pulse is output from the trigger output terminal when conditions are satisfied. — Address bus value or data bus value — Read/write condition — External probe value — System control signals — NOT condition — Delay count	7.2.40

Table 1.6 Emulation Functions (cont)

Command Type	Command	Function	Reference section
Trace data acquisition and display (cont)	TRACE_ SEARCH	Searches for trace data. • Address bus value • Data bus value • Read/write condition • System control signals • External probe value • Time stamp (20 ns, 1.6 µs, 52 µs)	7.2.43
	TRACE_MODE	Sets and displays trace information acquisition mode	7.2.42
Performance	PERFORMANCE _ANALYSIS 1 to 8	A maximum of four measurement modules Time intervals: 20 ns (6 hours), 406 ns (124 hours), and 1.6 µs (488 hours) A maximum of 65,535 execution count measurements • Subroutine measurement — Subroutine execution count • Time measurement — POINT-POINT time	7.2.29
Single-step execution	STEP, STEP_OVER, STEP_ INFORMATION	Executes one step at a time, and displays the following. • Instruction mnemonic • Memory contents • Register contents Displays the above data for a specified routine until a specified address is reached. The above operations are performed for a specified number of steps or until a specified address is reached. Specifies information to be displayed during single-step execution. Executes subroutine in one step.	7.2.36, 7.2.38, 7.2.37

Table 1.6 Emulation Functions (cont)

Command Type	Command	Function	Reference section
Memory access	MEMORY, DUMP	• Displays or modifies memory contents. • Displays or modifies memory contents in 1-, 2-, or 4-byte units. • Displays memory contents.	7.2.26, 7.2.17
	MAP	Specifies memory attributes in a 4-Mbyte unit. • Memory on user system • Write protected (emulation memory only) • Standard emulation memory — Standard: 8 Mbytes provided	7.2.25
	FILL	Writes data in specified pattern.	7.2.20
	DATA_SEARCH, DATA_CHANGE	Searches for and replaces data in specified pattern.	7.2.15, 7.2.14
Clock selection	CLOCK	• Selects emulator internal clock EML (8.25 MHz or 33 MHz). • Selects user system clock USER (5 to 66 MHz). • Selects crystal oscillator of evaluation chip board XTAL (10 to 20 MHz). • CKIO input mode.	7.2.11
Register access	REGISTER	Displays and modifies register contents.	7.2.32
Line assembly	ASSEMBLE	Assembles instruction mnemonics and specifies memory contents.	7.2.4
Disassembly	DISASSEMBLE	Disassembles memory contents.	7.2.16
Execution time measurement	GO	Measures GO command execution time. • Measures total run time	7.2.21
Test functions	FILL	Reads or writes the specified data to the memory.	7.2.20
	CHECK	Tests MCU input signals.	7.2.10

Table 1.6 Emulation Functions (cont)

Command Type	Command	Function	Reference section
Command input		Automatically inputs from file. Enables editing with cursor keys. Copies immediately preceding line. Copies operand of previous command.	
	RADIX	Enables value input in binary, octal, decimal, or hexadecimal. (Default can be specified.)	7.2.31
Results display	RESULT	Displays emulation results.	7.2.34
Others	MOVE	Transfers memory contents. • Memory to memory	7.2.28
	CONVERT	Converts number display. • Displays in binary, octal, decimal, hexadecimal, ASCII, or fixed-point values.	7.2.13
	STATUS	Displays emulator operating status.	7.2.35
	GO	Monitors emulation. • Monitors emulation status at constant intervals and displays the emulation status.	7.2.21
	RESET	Inputs the reset signal to the MCU.	7.2.33
	MODE	Sets and displays the MCU operating mode.	7.2.27
	HELP	Displays all commands.	7.2.22
	HISTORY	Displays the history of the input command.	7.2.23
	ALIAS	Alias function • Defines aliases.	7.2.3
	. <register>	Displays and changes registers of the MCU peripheral modules.	7.2.1
	ID	Displays versions of the system program.	7.2.25
	ABORT	Stops emulation in parallel mode.	7.2.2
	END	Cancels parallel mode.	7.2.18
	QUIT	Quits system program.	7.2.30

Table 1.7 Host Computer Interface Functions

Command Type	Command	Function	Reference Section
Serial interface	INTFC_LOAD	Loads program from host computer.	8.2.1
	INTFC_SAVE	Saves program in host computer.	8.2.2
	INTFC_VERIFY	Verifies memory contents against host computer files.	8.2.3
Bi-directional parallel interface	LOAD	Loads program from host computer.	8.2.4
	SAVE	Saves program in host computer.	8.2.5
	VERIFY	Verifies memory contents against host computer files.	8.2.6

1.5 Realtime Emulation

The emulator enables realtime emulation with a clock frequency of 66 MHz for the MCU with no wait states. Realtime emulation consists of the following three modes:

- Normal mode: Executes only emulation.
- Cycle reset mode: Forcibly inputs the RESETP signal to the MCU periodically.
- Parallel mode: Enables the user to display and modify memory contents and display trace information during user program execution.

The user can select the mode which best suits the user's debugging needs. The following describes each of these modes.

1.5.1 Normal Mode

Normal Mode Function: This mode executes only user program emulation. Until a break condition is satisfied, the emulator executes the user program. When a hardware break condition or software break condition is satisfied, the emulator stops the program execution. When a number of times or sequential break for the software break condition is specified, the emulator stops, only for a moment, the program execution every time the specified address is passed, and then resumes program execution.

Normal Mode Specification: Specifying no option with the GO command sets normal mode.

1.5.2 Cycle Reset Mode

Cycle Reset Mode Function: The emulator inputs the RESETP signal to the MCUafter a specified time during realtime emulation and repeats the execution from the reset state. When the RESETP signal is input to the MCU, a low-level pulse is output to the trigger output probe concurrently. This function is useful to observe the waveform from the initial state, such as power-on-reset, to a specified time.

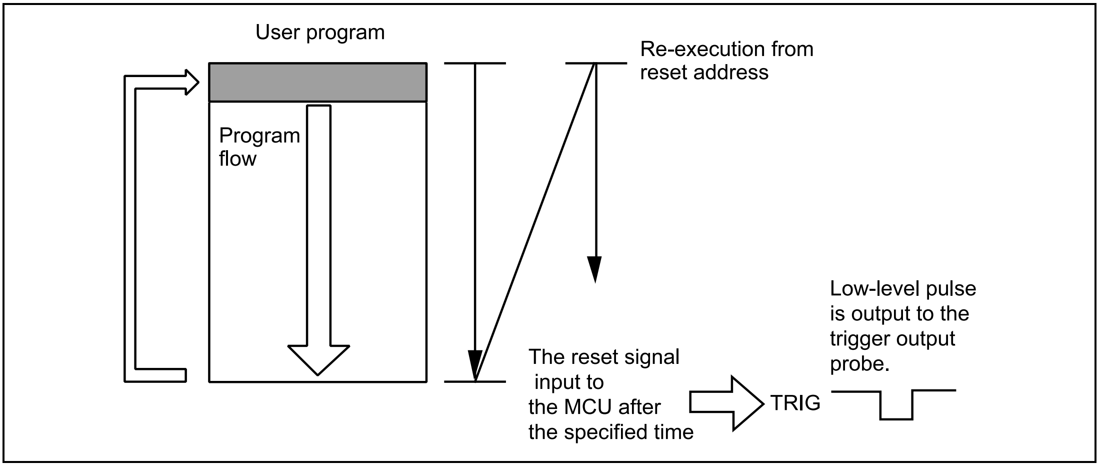

User program
Program flow
Re-execution from reset address
The reset signal input to the MCU after the specified time
TRIG
Low-level pulse is output to the trigger output probe.

Figure 1.2 Cycle Reset Mode

Cycle Reset Mode Specification: Set "R=n" as a GO command option to specify cycle reset mode. For details, refer to section 7.2.21, GO.

Emulation Stop: In cycle reset mode, hardware break conditions and software break conditions are invalid. To stop emulation, press the (CTRL) + C keys or the (BREAK) key.

Trigger Signal Output Timing in Cycle Reset Mode: In cycle reset mode, the RESETP signal is output to the MCU regardless of the MCU operating status when the time specified by the command has elapsed. Figure 1.3 shows the timing in which the TRIG signal is output to the trigger output probe in cycle reset mode.

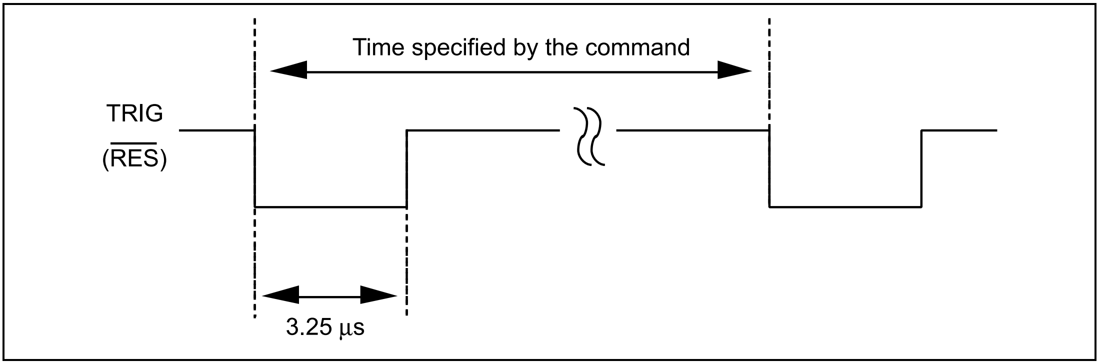


Figure 1.3 Trigger Signal Output Timing

1.5.3 Parallel Mode

Parallel Mode Function: In parallel mode, the emulator can display and modify memory or display trace information during realtime emulation. However, during memory contents display or modification, realtime emulation cannot be performed.

Parallel Mode Specification: Parallel mode can be activated during GO command realtime emulation by any of the following methods. Figure 1.4 shows transition to parallel mode.

- Press the (RET) key
- Press the space key
- Satisfy a trace stop condition specified by the TRACE_CONDITION_A,B,C command

If any of the above occurs, the emulator will display a prompt (#) and enter parallel mode command input wait state. Emulation, however, continues without interruption. Input the END (E) command to return to the normal mode. Input the ABORT (AB) command to stop user program execution in the parallel mode.

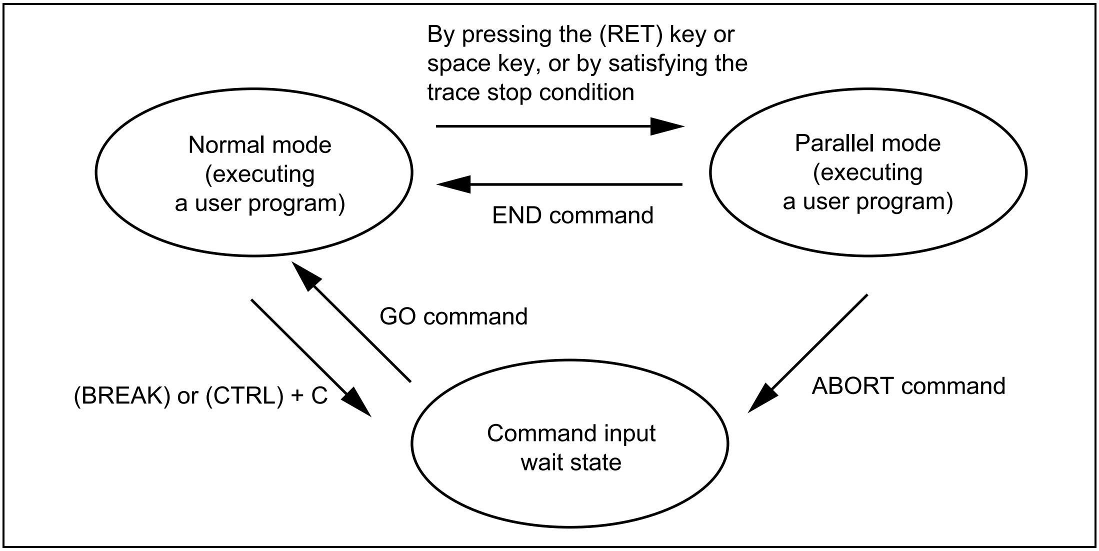

By pressing the (RET) key or space key, or by satisfying the trace stop condition
Normal mode (executing a user program)
Parallel mode (executing a user program)
END command
GO command
(BREAK) or (CTRL) + C
ABORT command
Command input wait state

Figure 1.4 Transition to Parallel Mode

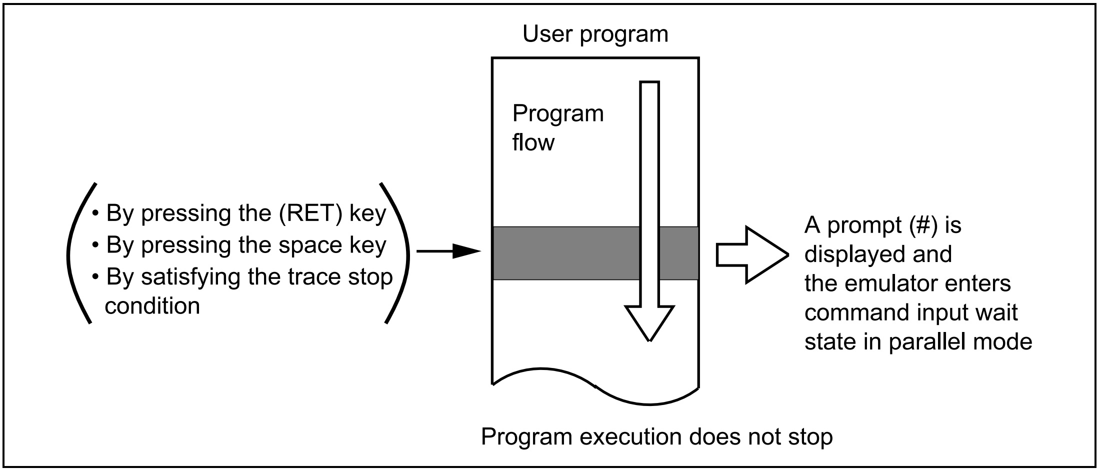

User program
Program flow
• By pressing the (RET) key
• By pressing the space key
• By satisfying the trace stop condition
A prompt (#) is displayed and the emulator enters command input wait state in parallel mode
Program execution does not stop

Figure 1.5 Parallel Mode

Note that debugging differs in parallel mode operation depending on the method used to activate it, as follows.

- By pressing the (RET) key or satisfying a trace stop condition
  - The emulator stops acquiring trace information as soon as parallel mode is entered.
  - The emulator can execute multiple commands entered by the user in parallel mode. The parallel mode continues even after the command execution is terminated.
  - The END command terminates the parallel mode and returns the emulator to normal mode (displays the current PC). At this time, the emulator restarts trace information acquisition.

- By pressing the space key
  - The emulator continues trace information acquisition; however, while the emulator executes the TRACE, TRACE_CONDITION_A,B,C or TRACE_SEARCH command, it acquires no trace information.
  - In parallel mode, the emulator returns to normal mode after one command execution and displays the current PC. At this time, if trace information acquisition has stopped, the emulator restarts acquisition.

Commands usable in parallel mode are listed in table 7.1.

Notes: 1. When memory (standard emulation memory or internal I/O) is accessed with the MEMORY command, DUMP command, or DISASSEMBLE command in parallel mode, there are some restrictions with respect to user program execution.

- Standard emulation memory

  When accessing standard emulation memory in parallel mode, the user program temporarily halts. This pause lasts for about 3.28 ms during user system clock operation. Therefore, realtime emulation cannot be performed.

- Internal X-RAM/Y-RAM and I/O

  When accessing internal X-RAM, Y-RAM, or internal I/O, the user program temporarily halts. This pause lasts for about 3.28 ms during user system clock operation. Therefore, realtime emulation cannot be performed.

- In the above two cases, the emulator pauses at the following timing.

  — MEMORY command: At each memory access

  — DUMP command: In 16-byte units

  — DISASSEMBLE command: In 4-byte units

2. During execution of the TRACE, TRACE_SEARCH, or TRACE_CONDITION_A,B,C command, the emulator stops trace information acquisition.

3. The emulator cannot enter parallel mode when executing emulation in the cycle reset mode (R option of GO command).

1.6 Break Function

The following four methods are useful to stop emulation. The break function can be used regardless of the MCU's operating mode.

- Hardware break: Caused by the MCU's signal status as specified
- Software break: Caused by a program counter
- Forced break: Caused by pressing the (CTRL) + C keys or the (BREAK) key
- Write protected/guarded break: Caused by writing to a write-protected or guarded area

1.6.1 Hardware Break

A hardware break can be specified using the BREAK_CONDITION_UBC1,2,3 command or BREAK_CONDITION_A,B,C commands. Specifiable break conditions are listed in table 1.8.

Table 1.8 Specifiable Hardware Break Conditions

Condition	BREAK_ CONDI- TION _C (1 to 8)	BREAK_ CONDI- TION _B (1 to 8)	BREAK_ CONDI- TION _A (1 to 8)	BREAK_ CONDI- TION _UBC3	BREAK_ CONDI- TION _UBC2	BREAK_ CONDI- TION _UBC1
Address condition	O	O	O		O	O
Data condition		O	O			O
PC condition					O	O
ASID condition					O	O
Read/write condition		O	O		O	O
Access type specification					O	O
Probe condition		O	O			
External interrupt condition		O	O			
Pass count						O
Delay count specification *		O				
Sequential break					O	O
I/O access condition				O		
LDTLB condition				O		

Note: Only the BREAK_CONDITION_B7 can be specified for the delay count specification.

The following describes the break conditions.

Address Bus Value: A break occurs when the address bus value matches the specified condition.

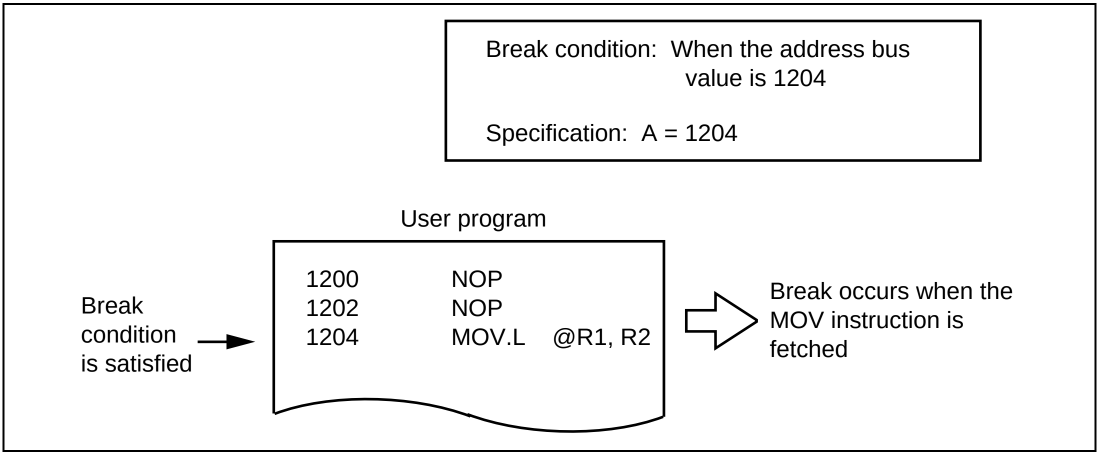

Break condition: When the address bus value is 1204
Specification: A = 1204
User program
1200 NOP
1202 NOP
1204 MOV.L @R1, R2
Break condition is satisfied
Break occurs when the MOV instruction is fetched

Figure 1.6 Break with Address Bus Value

Data Bus Value: A break occurs when the data bus value matches the specified condition.

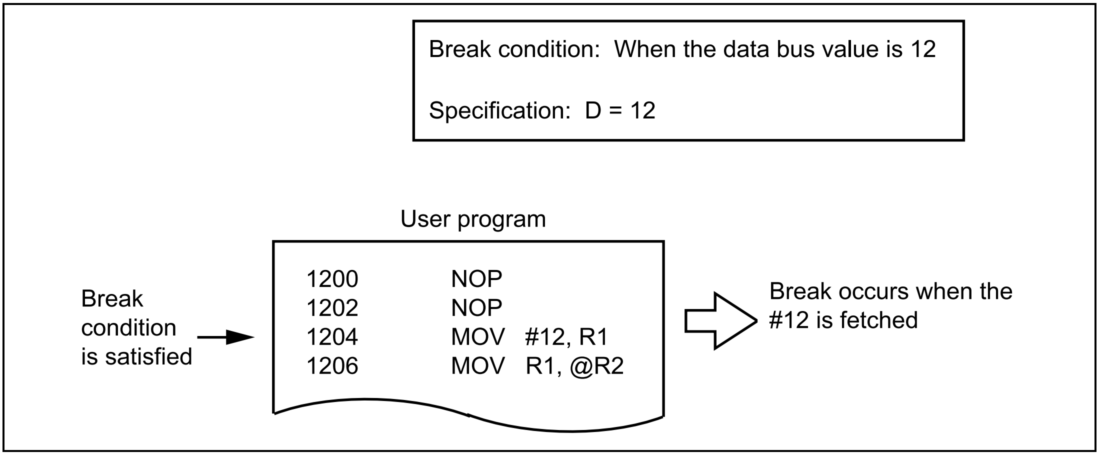

Break condition: When the data bus value is 12
Specification: D = 12
User program
1200 NOP
1202 NOP
1204 MOV #12, R1
1206 MOV R1, @R2
Break condition is satisfied
Break occurs when the #12 is fetched

Figure 1.7 Break with Data Bus Value

Read/Write Condition: A break occurs when the MCU's RW signal levels match the specified conditions. Usually, the read/write condition is specified together with the address or data conditions.

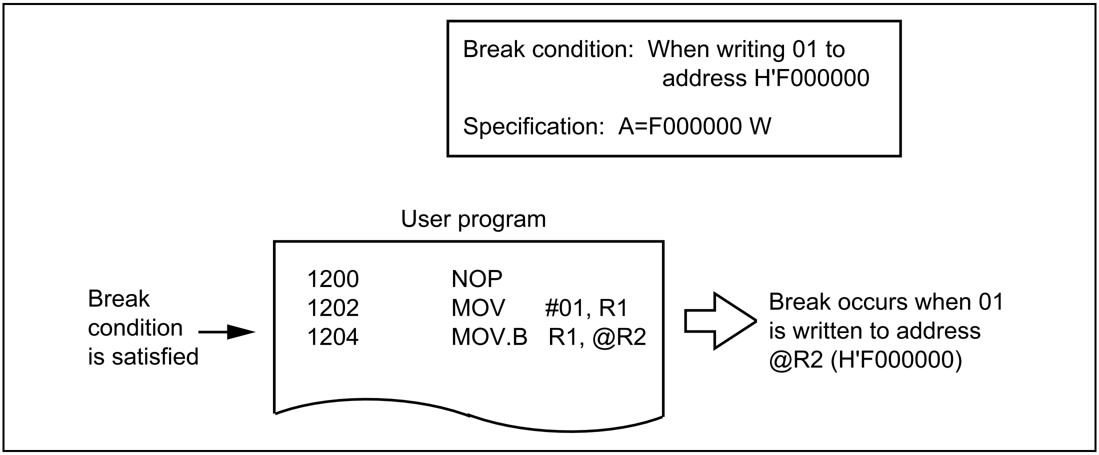

Break condition: When writing 01 to address H'F000000
Specification: A=F000000 W
User program
1200 NOP
1202 MOV #01, R1
1204 MOV.B R1, @R2
Break condition is satisfied
Break occurs when 01 is written to address @R2 (H'F000000)

Figure 1.8 Break with Read/Write

Delay Count and Number of Times the Break Condition is Satisfied: These functions can only be specified with the BREAK_CONDITION_UBC1* and BREAK_CONDITION_B7 commands.

In delay count specification, a break occurs when the above break condition (address bus value, data bus value, or read/write condition) is satisfied and the emulator executes the bus cycle a specified number of times. Specify this condition in combination with any of the above break conditions. A maximum of 32767 bus cycles can be specified for the BREAK_CONDITION_B command delay count.

Note: For the BREAK_CONDITION_UBC1 command, only a satisfaction count can be specified.

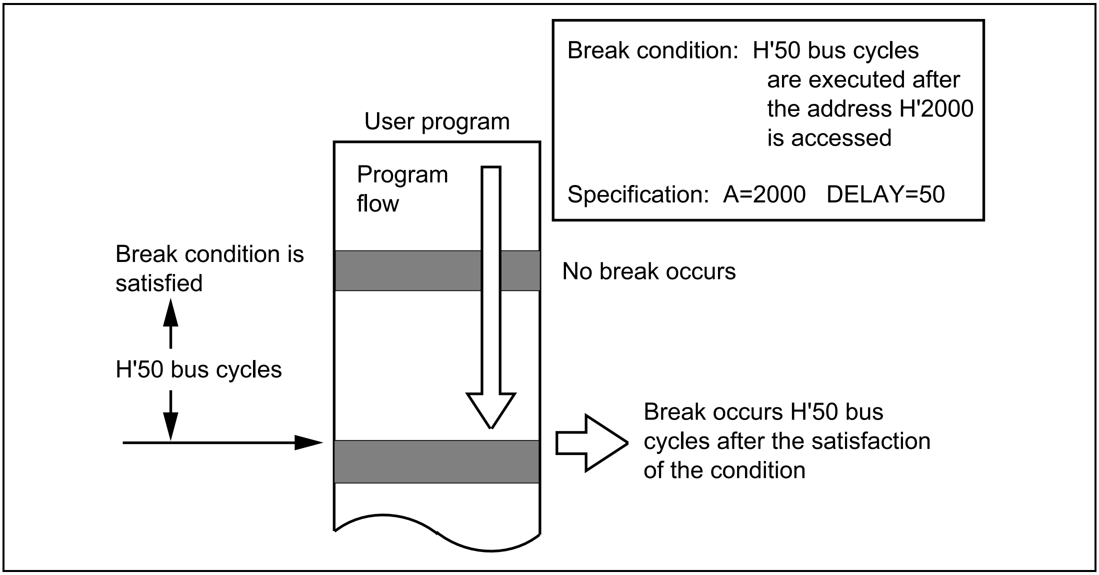


Figure 1.9 Break with Delay Count Specification

In satisfaction count specification, a break occurs when the above break condition (address bus value, data bus value, or read/write condition) is satisfied for a specified number of times. Specify this condition in combination with any of the above break conditions. When the BREAK_CONDITION_B command is used, a maximum satisfaction count of 32767 can be specified. When the BREAK_CONDITION_UBC1 command is used, maximum satisfaction count of 4095 can be specified. This condition can be specified in combination with any of the above break conditions, the satisfaction count, and the delay count specification. In this case, a break occurs when the break condition is satisfied the specified number of times and the number of bus cycles specified by the delay count specification is executed.

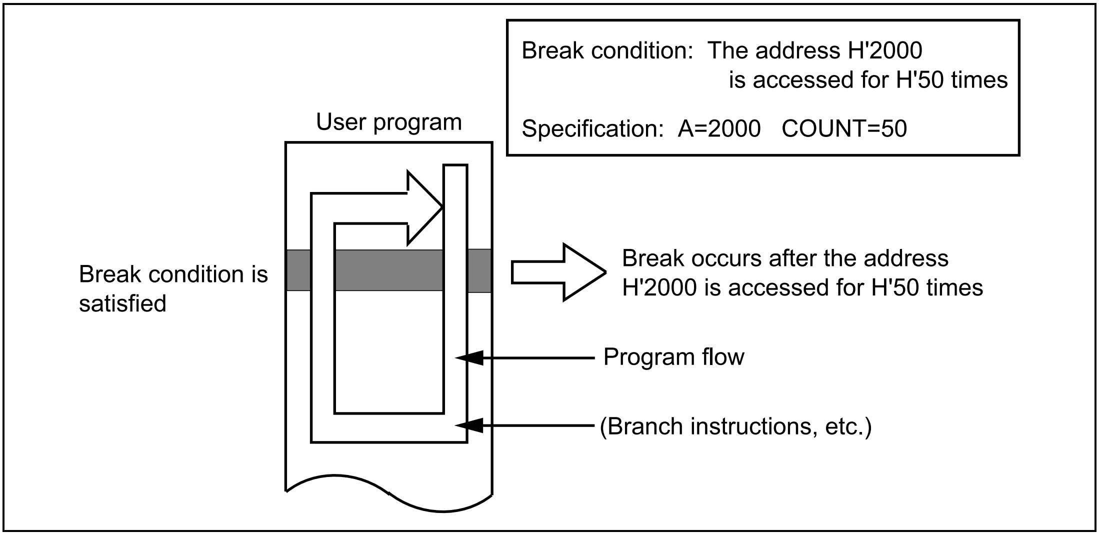

Break condition: The address H'2000
is accessed for H'50 times
Specification: A=2000 COUNT=50
User program
Break condition is satisfied
Break occurs after the address H'2000 is accessed for H'50 times
Program flow
(Branch instructions, etc.)

Figure 1.10 Break with Satisfaction Count Specification

PC Value (BREAK_CONDITION_UBC1,2): A break occurs when the program counter (PC) value satisfies the specified condition. The break timing depends on the ;P option setting as follows:

- PC value without option ;P (PC=1000): Break after execution
  A break occurs after the instruction at the specified address is executed.
- PC value followed by option ;P (PC=1000;P): Break before execution
  A break occurs before the instruction at the specified address is executed.

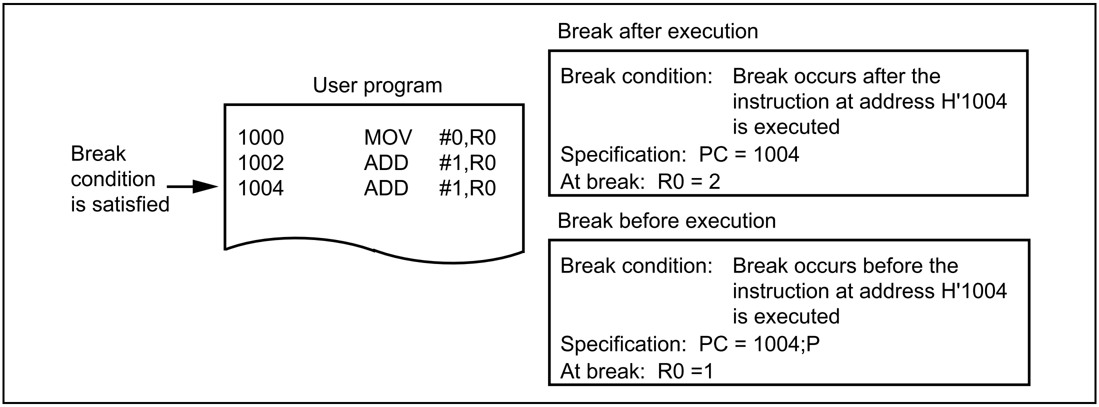

Break condition is satisfied
User program
1000 MOV #0,R0
1002 ADD #1,R0
1004 ADD #1,R0
Break after execution
Break condition: Break occurs after the instruction at address H'1004 is executed
Specification: PC = 1004
At break: R0 = 2
Break before execution
Break condition: Break occurs before the instruction at address H'1004 is executed
Specification: PC = 1004;P
At break: R0 =1

Figure 1.11 Break with PC Value Specification

Sequential Break Condition (BREAK_CONDITION_UBC1,2): In sequential break mode, a break occurs when hardware break conditions UBC2 and UBC1 have been satisfied in that order.

Specify break conditions with the BREAK_CONDITION_UBC1 and 2* command. In this case, specifiable conditions are address bus values, data bus values, program counter (PC) value, ASID value, read/write conditions, or access type value.

When executing the user program, specify the mode option of the GO command as a sequential break option (;SB). Unless the option is specified, a sequential break does not occur. In this case, a break occurs whenever each break condition is satisfied.

- Sequential break mode

  When break condition UBC2 and then break condition UBC1 are satisfied, a break occurs.

Note: When the sequential break option (;SB) of the GO command is specified while the BREAK_CONDITION_UBC1 or 2, or both are not specified, the error message below will be displayed. At this time, a user program will not be executed.
***** 35:CAN NOT USE THIS MODE**

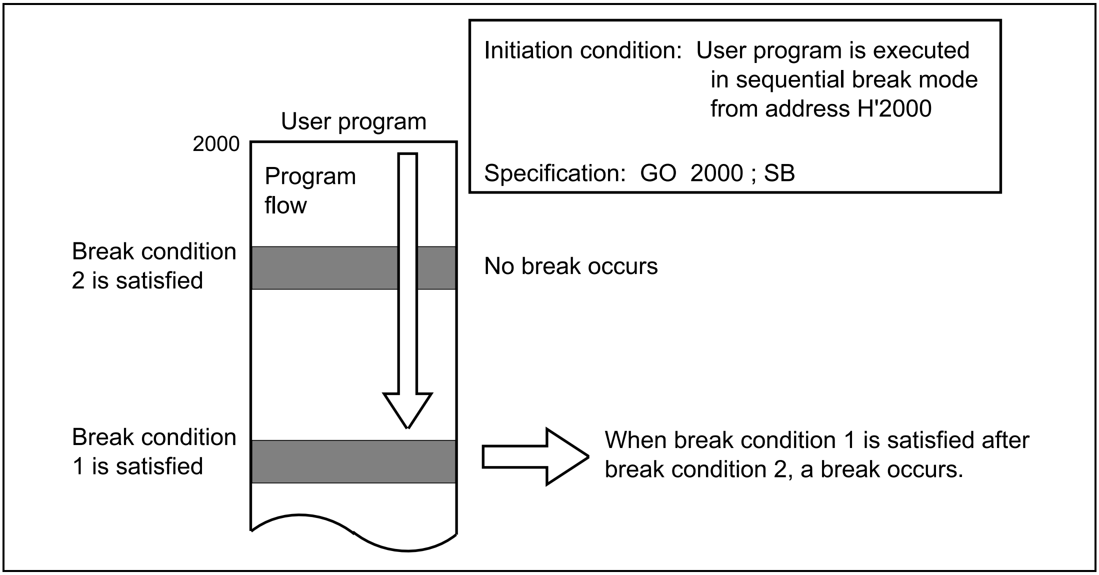

Initiation condition: User program is executed in sequential break mode from address H'2000
Specification: GO 2000 ; SB
User program
2000
Program flow
Break condition 2 is satisfied
No break occurs
Break condition 1 is satisfied
When break condition 1 is satisfied after break condition 2, a break occurs.

Figure 1.12 Break with Sequential Specification

1.6.2 Software Break

The contents at the specified address are replaced with a break instruction. The program execution stops when the break instruction is executed. The replaced instruction at the address is not executed. After the GO command is executed, the contents at the specified address will be replaced with a break instruction and the user program will be executed. When the user program execution stops, the break instruction will be replaced again with the contents at the specified address. Therefore, the contents at the specified address can be accessed immediately after the user program execution, using the DISASSEMBLE command or the DUMP command. However, note that a break instruction will be read if the memory contents at the address to which a break is specified are accessed in the parallel mode.

No software break must be specified immediately after a delayed branch instruction (at a slot instruction). If specified, a slot invalid instruction interrupt will occur at the branch instruction execution, and a break will not occur.

The software break can be performed in the following two ways:

- Normal break
- Sequential break

Normal Break: A break occurs before executing the instruction at a breakpoint specified with the BREAK command. At this time, the following can be specified:

- Number of break points: 255 points (max)
- Number of times the break condition is satisfied: A break occurs after executing the instruction at which a breakpoint is set a specified number of times. The maximum number to specify is 65,535 (H'FFFF).

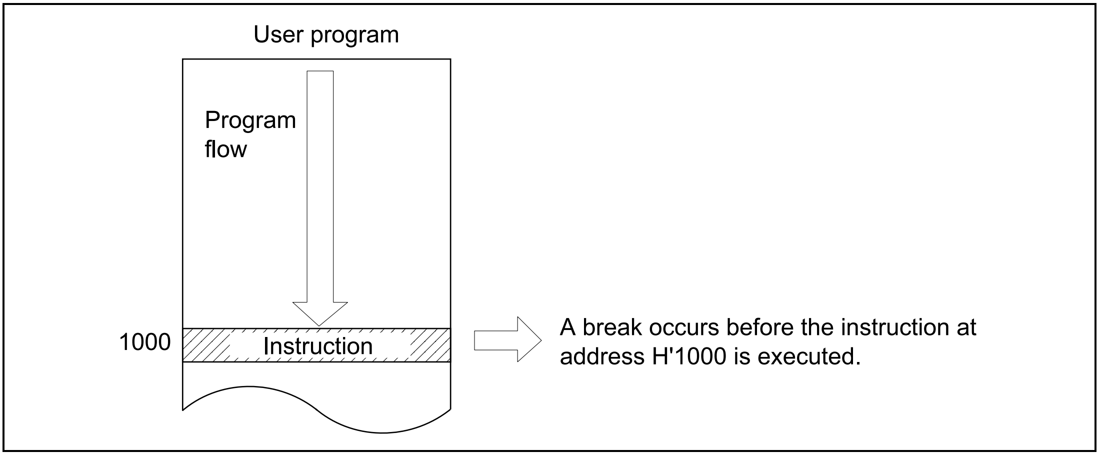

User program
Program flow
1000
Instruction
A break occurs before the instruction at address H'1000 is executed.

Figure 1.13 Normal Break (Software Break)

Note: When specifying the satisfaction count as one of the normal break, emulator firmware performs processing every time the program passes the break condition address. As a result, the program will not operate in realtime. When the program passes the break condition address, the emulator executes the instruction at the address for one step then returns to user program execution. At this time, the BREAK_CONDITION_UBC2 settings becomes invalid because the BREAK_CONDITION_UBC2 is used to perform the step execution of the break address.

Sequential Break: A sequential break occurs (seven pass points max) when certain conditions are satisfied in a specified order. A reset point can be specified in addition to these pass points.

If the reset point is passed, all sequential break conditions up to that point become invalid and the emulator rechecks from the first break condition.

Figure 1.14 illustrates the usual sequential break and figure 1.15 describes a sequential break when a reset point is specified.

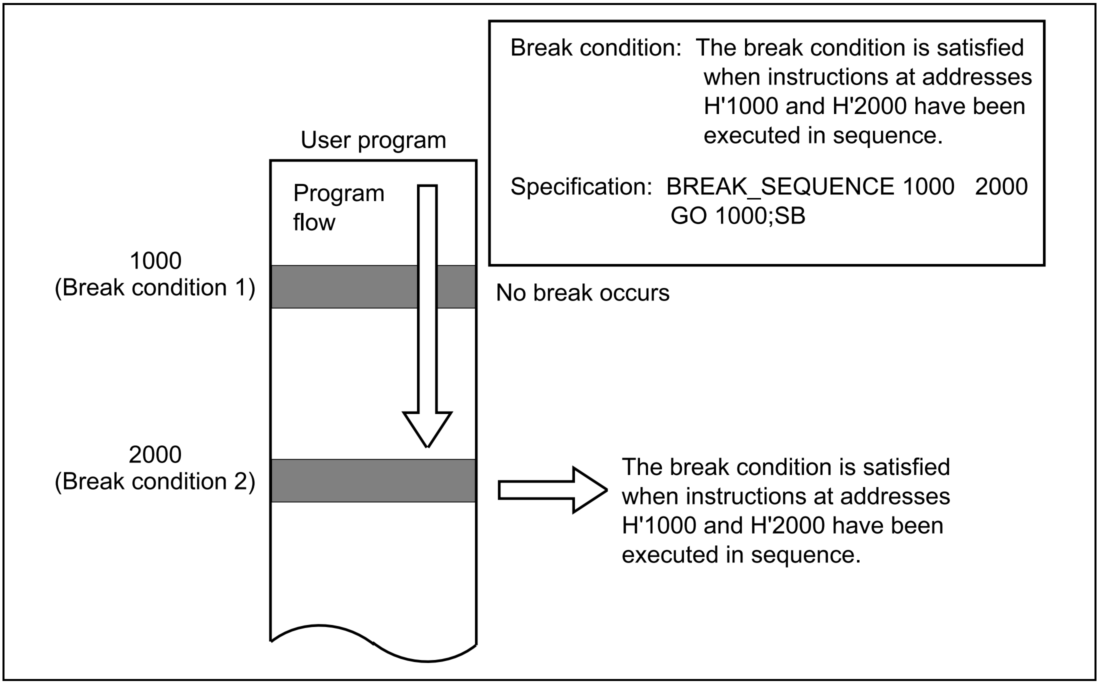

Break condition: The break condition is satisfied when instructions at addresses H'1000 and H'2000 have been executed in sequence.
Specification: BREAK_SEQUENCE 1000 2000
GO 1000;SB
User program
Program flow
1000
(Break condition 1)
No break occurs
2000
(Break condition 2)
The break condition is satisfied when instructions at addresses H'1000 and H'2000 have been executed in sequence.

Figure 1.14 Sequential Break

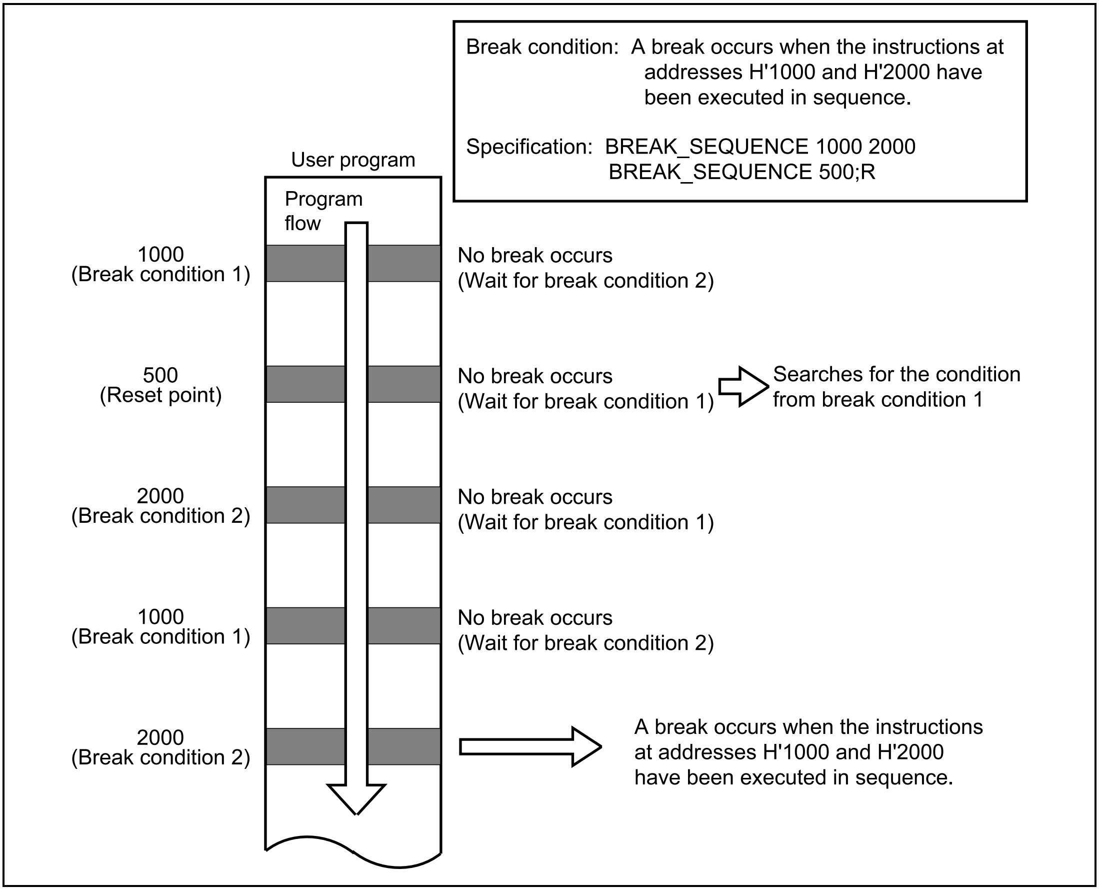

Break condition: A break occurs when the instructions at addresses H'1000 and H'2000 have been executed in sequence.
Specification: BREAK_SEQUENCE 1000 2000
BREAK_SEQUENCE 500;R
User program
Program flow
1000 (Break condition 1)
No break occurs (Wait for break condition 2)
500 (Reset point)
No break occurs (Wait for break condition 1)
Searches for the condition from break condition 1
2000 (Break condition 2)
No break occurs (Wait for break condition 1)
1000 (Break condition 1)
No break occurs (Wait for break condition 2)
2000 (Break condition 2)
A break occurs when the instructions at addresses H'1000 and H'2000 have been executed in sequence.

Figure 1.15 Sequential Break (Reset Point Specification)

Note: When specifying the sequential break (BREAK_SEQUENCE), emulator firmware performs processing every time the program passes the pass point or reset point. As a result, the program will not operate in realtime. When the program passes the pass point or reset point, the emulator executes the instruction at the address for one step then returns to program execution. The BREAK_CONDITION_UBC2 settings are invalid at pass point or reset point execution.

1.6.3 Forced Break

Pressing the (CTRL) + C keys or the (BREAK) key stops program execution.

1.7 Realtime Trace Function

The emulator can trace MCUexternal bus information (called a bus trace hereafter) during realtime emulation without affecting the user system. The emulator can fetch external bus information concerning the address or data, and an external probe value up to 131,070 bus cycles. Trace information is referenced with the TRACE command. Display of this information enables a check of the executed user program flow. The AUD trace function (the on-chip advanced user debugger) can be used to acquire the address of the branch source and branch destination. By using the AUD trace function, internal bus information can be traced.

External bus trace information:

- Address bus: 32 bits
- Data bus: 32 bits
- External probe: Four
- Time stamp value: 32 bits
- MCU control I/O signal: 11
- The number of bus cycle clocks

AUD trace information:

- Attribute of the address and instruction (branch or destination)
- Instruction address
- Instruction mnemonic and operand

Emulator displays trace information in the following ways:

- Displays the bus trace information as a mnemonic in bus cycle units.
- Displays the AUD trace information in branch instruction execution units.
- Displays both bus trace information and AUD trace information.
- Searches for the specified information and displays it when the TRACE_SEARCH command is used.

Restrictions on Using the AUD trace function:

1. The AUD trace outputs the differences between newly output branch source/destination addresses and the previously output branch source/destination addresses. If the previous branch source/destination address is the same as the upper 16 bits, the lower 16 bits are output. If it matches the upper 24 bits, the lower 8 bits are output. If it matches the upper 28 bits, the lower 4 bits are output. The emulator usually regenerates the 32-bit address from these differences and displays it. If the emulator cannot display the 32-bit address, it displays the difference from the previously displayed 32-bit address.
2. During an AUD realtime trace, some trace information may not be acquired if the cache is on, the CPU clock is 33 MHz, the AUD clock is 30 MHz, and there are nine NOP instructions followed by an endless loop of BRA instruction at the tenth instruction.
3. During user program execution, when the power-on reset (low RESETP signal) is input, trace information cannot be acquired after the input. To acquire trace information after a power-on reset, break at the reset vector, then re-execute.
4. The AUD clock must be lower than the CPU clock and higher than a quarter of the CPU clock.

1.7.1 Trace Timing

The trace information acquisition timing in a bus trace differs depending on the memory type accessed.

- RAM (other than DRAM or SDRAM), ROM, and PCMCIA: Inactivated timing of the RD and WEx signals
- DRAM: Inactivated timing of the CASxx signal
- SDRAM: Rising edge of the CKIO clock in which CAS, RAS, or BS signal is active

In each bus cycle, the clock number is the number of clock (CLK) cycles between the end of the previous bus cycle and the end of the current bus cycle. Figure 1.16 shows an example of the RAM bus trace.

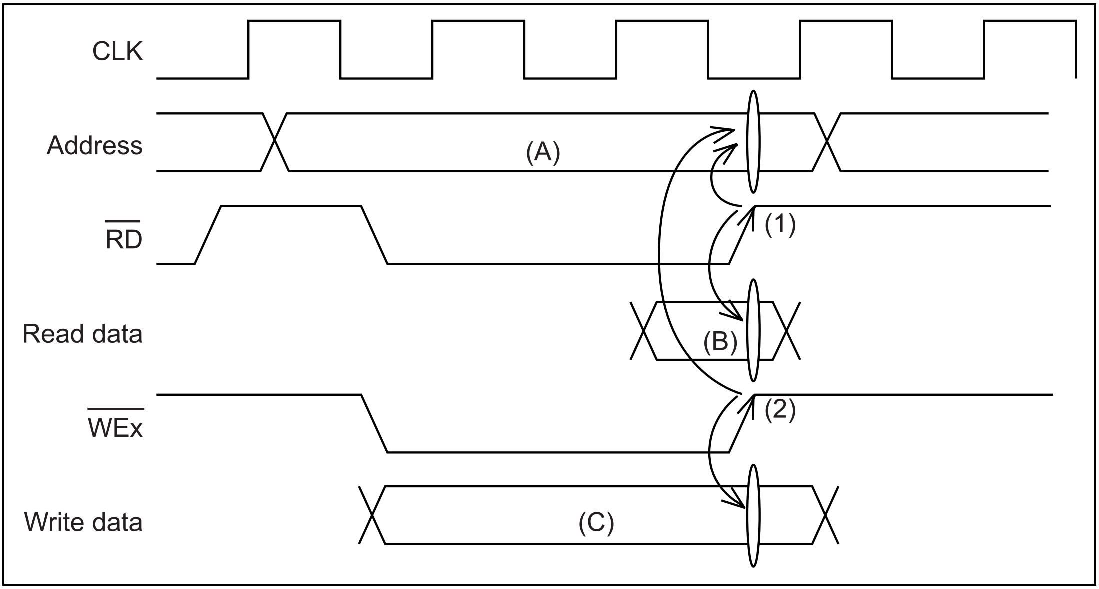


Figure 1.16 RAM Bus Trace Timing

Description:

(1): The trace acquisition timing of the read cycle.
(2): The trace acquisition timing of the write cycle.
(A): The trace acquisition timing of the address bus.
(B): The trace acquisition timing of the read data bus.
(C): The trace acquisition timing of the write data bus.
Three clocks are traced in bus cycle (A).

Figure 1.17 shows an example of the SDRAM bus trace.

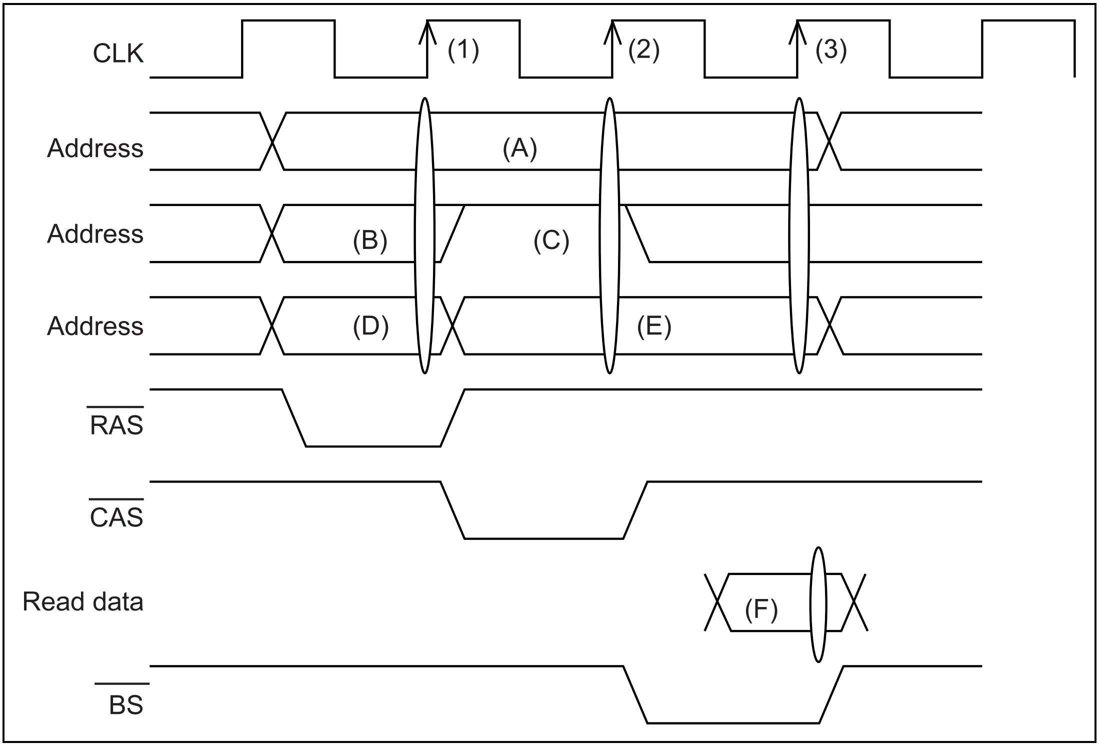


Figure 1.17 SDRAM Bus Trace Timing

Description:

(1): Trace is acquired at the rising edge of the CLK signal during the RAS signal cycle.
(2): Trace is acquired at the rising edge of the CLK signal during the CAS signal cycle.
(3): Trace is acquired at the rising edge of the CLK signal during the BS signal cycle.
(A), (B), and (D): Trace acquisition timing of the address bus during the RAS cycle.
(A), (C), and (E): Trace acquisition timing of the address bus during the CAS cycle.
(F): Trace acquisition timing of the data bus during the BS cycle.

1.7.2 Trace Condition Setting

The user can specify the following four conditions with the TRACE_CONDITION_A,B,C commands. For details, refer to section 7.2.40, TRACE_CONDITION_A,B,C. Table 1.9 shows the maximum specifiable numbers in trace mode.

- Free trace
- Range trace
- Trigger signal
- Trace stop (parallel mode)

Table 1.9 Maximum Specifiable Numbers in Trace Mode

	TRACE_ CONDITION_A	TRACE_ CONDITION_B	TRACE_ CONDITION_C	Total
Range trace	8	8	8	24
Trace stop (Parallel mode)	8	8	8	24

Free Trace: In free trace when the user program is executed as a result of the GO, STEP, or STEP_OVER command, tracing is carried out continuously for a maximum of the latest 131,070 bus cycles until a break condition is satisfied. When no parameter is given with the TRACE_CONDITION_A,B,C commands, the default is free trace. Figure 1.18 illustrates the free trace operation.

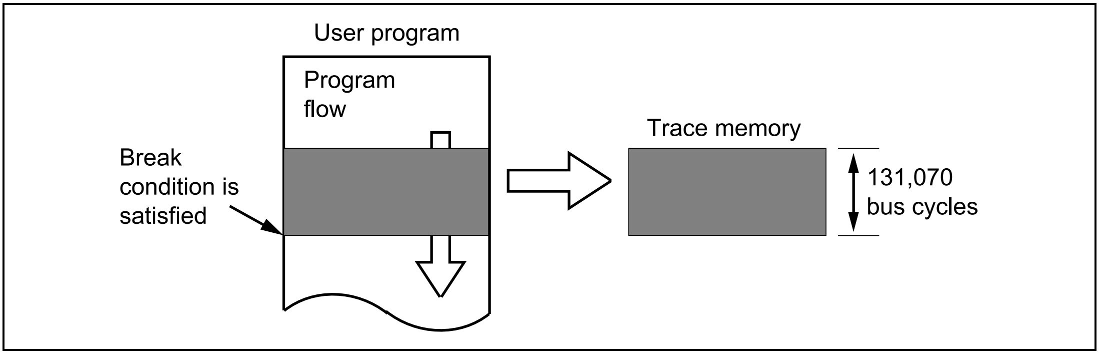


Figure 1.18 Free Trace

Range Trace: When a range trace is specified, the emulator only traces at points where specified conditions are satisfied. The following conditions can be specified.

- Address bus value (within or outside a specified range)
- Read/write condition

Figure 1.19 illustrates the trace acquisition condition.

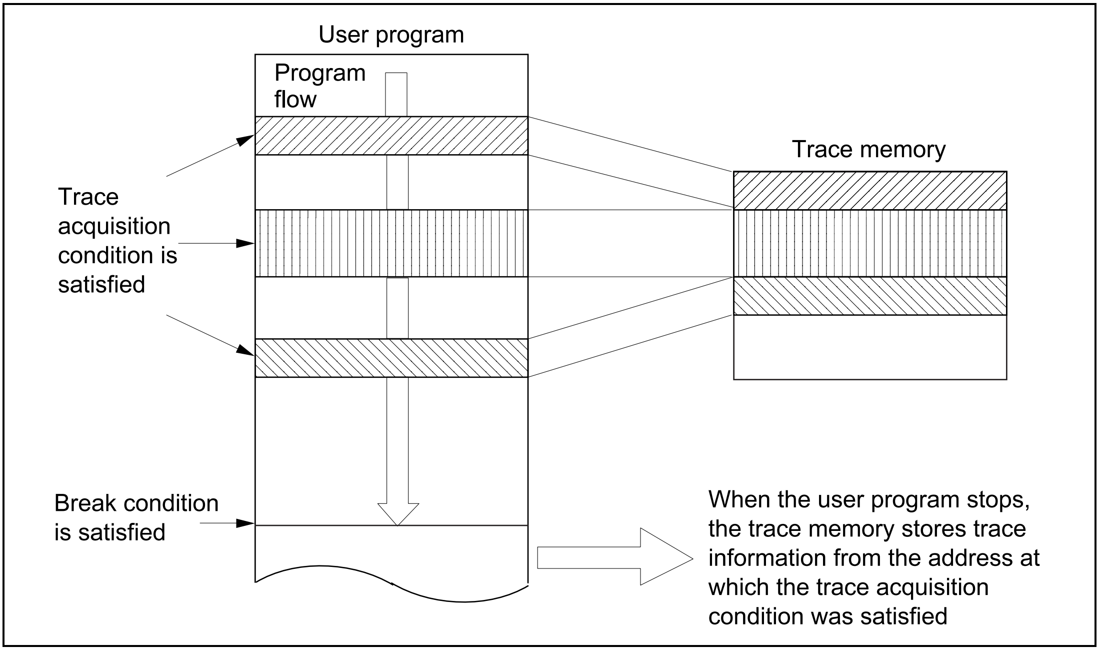

User program
Program flow
Trace memory
Trace acquisition condition is satisfied
Break condition is satisfied
When the user program stops, the trace memory stores trace information from the address at which the trace acquisition condition was satisfied

Figure 1.19 Range Trace

Trace Stop (Parallel Mode): The trace information can be displayed without terminating the realtime emulation. When a trace stop condition is specified with the TRACE_CONDITION_A,B,C command, the emulator acquires trace information until the specified condition is satisfied. At this point, trace acquisition stops and the emulator prompts for command input in parallel mode, although realtime emulation does not stop. Refer to section 1.3.3, Parallel Mode, for details. Once the trace stop conditions have been satisfied and the trace information has been displayed, the user can specify the trace stop condition again. The user can specify the following conditions.

- Address bus or data bus value
- Read/write condition
- External probe value
- External interrupt condition
- Delay count (H'1 to H'7FFF)

Figure 1.20 shows the trace stop condition specification.

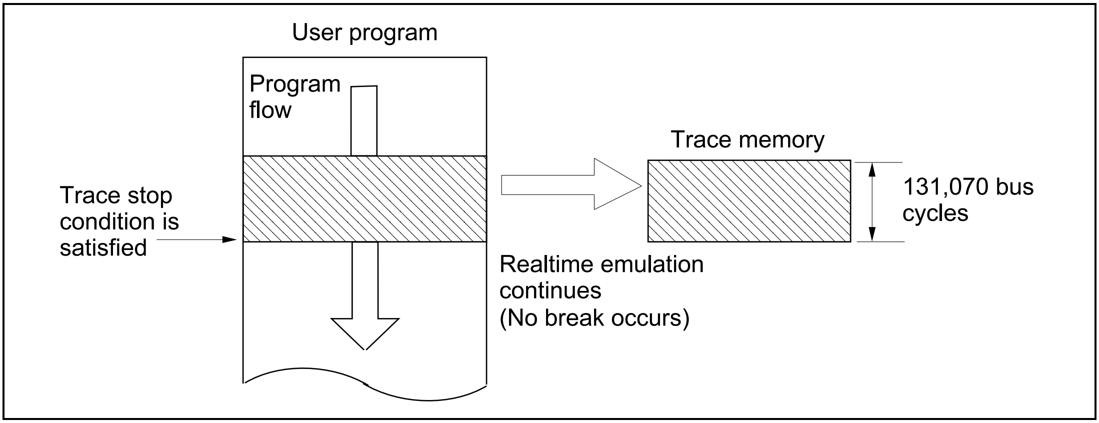

User program
Program flow
Trace stop condition is satisfied
Trace memory
131,070 bus cycles
Realtime emulation continues (No break occurs)

Figure 1.20 Trace Stop

1.7.3 Trace Display

The user can display trace information using the TRACE command. There are four display formats, as follows.

Bus Trace Display: Trace information is displayed in bus cycle units.

AUD Trace Display: Branch instruction trace information is displayed in branch instruction units.

Bus Trace and AUD Trace Display: Bus trace information are displayed together with the AUD trace information.

Search Display: The emulator searches for specified trace information and displays all the appropriate bus cycles. In this case, use the TRACE_SEARCH command.

1.8 Single-Step Function

In addition to realtime emulation, effective debugging is facilitated by the single-step function. This function displays the following information every time a program instruction is executed.

- Control registers (PC, SR, SSR, SPC, GBR, VBR, RS, RE, MOD, MACH, MACL, and PR)
- General registers (R0 to R15)
- Bank 0 general registers (R0_BANK0 to R7_BANK0)
- Bank 1 general registers (R0_BANK1 to R7_BANK1)
- MCU DSP registers (A0, A1, A0G, A1G, X0, X1, Y0, Y1, M0, M1, and DSR)
- Instruction address
- Instruction mnemonic
- Memory contents
- Termination cause

1.8.1 Single-Step Execution

Single-step execution has three modes: one in which all the instructions are displayed, one in which only branch instructions are displayed, and another in which instructions of a subroutine executed at first are displayed. To execute this function, use the STEP command, or to execute a subroutine in one step, use the STEP_OVER command.

Displaying All Instructions: The emulator displays the specified information after every instruction.

Branch Instruction Display: The information is only displayed at branch instructions listed below.

BT, BF, BRA, BSR, JMP, JSR, BT/S, BF/S, BRAF, BSRF, TRAPA, RTS, RTE

Subroutine Display: When a subroutine is called, the information for the subroutine executed at first is displayed.

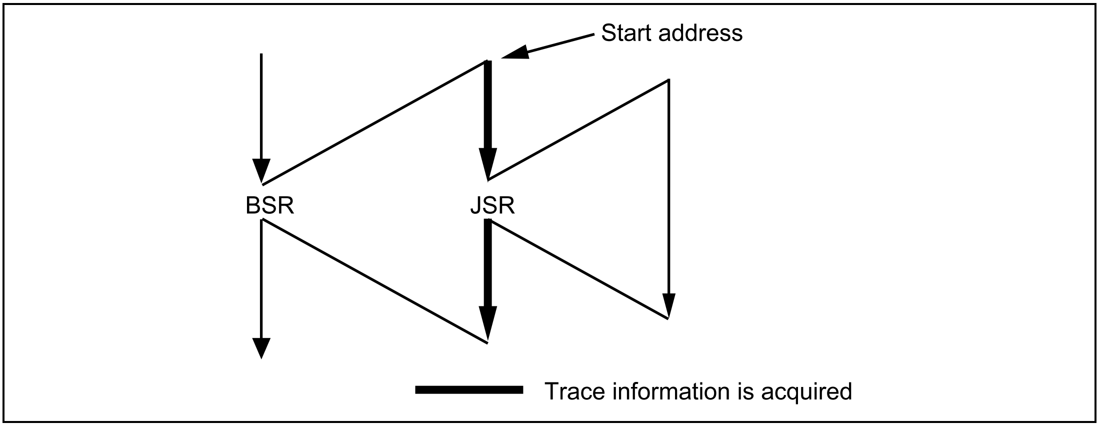


Figure 1.21   Subroutine Display

This function interrupts the execution state display at the JSR, BSR, or BSRF instruction in the designated subroutine and resumes the execution state display when the instruction placed immediately after the JSR, BSR, or BSRF instruction is executed. After that, if another JSR, BSR, or BSRF instruction is executed, the execution state display is interrupted.

Subroutine Step Execution: When executing a JSR, BSR, or BSRF instruction, the emulator treats the called subroutine as a single step. All other instructions are executed one at a time. This function is valid only in the user RAM or the emulation memory area.

1.8.2 Setting Display Information

The user can set the information displayed at each instruction using the STEP_ INFORMATION command. For details, refer to section 7.2.37, STEP_INFORMATION.

1.8.3 Termination of Single-Step Function

The single-step function stops after executing a specified number of steps from the specified start address (or the current PC address). The user can stop execution by specifying a stop address. However, the specified address must be at the start of an instruction. If the second byte of an instruction is specified (not the start of an instruction), the single-step function will not stop and execution continues for the specified number of steps.

1.9 Execution Time Measurement

1.9.1 Execution Time Measurement

GO to BREAK Time: The user can measure the user program execution time by specifying with the GO command. In this mode, the emulator measures the total execution time from when the user program is started with the GO command to when it is stopped by a break.

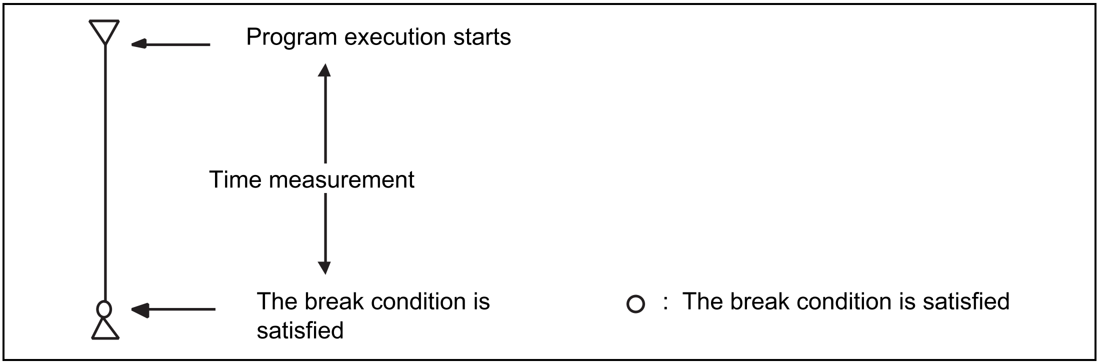

Program execution starts
Time measurement
The break condition is satisfied
○ : The break condition is satisfied

Figure 1.22 Normal Mode Time Measurement Range

1.9.2 Subroutine Time Measurement and Number of Times Measurement

The subroutine time and number of times the subroutines are executed can be measured based on the total program execution time by the PERFORMANCE_ANALYSIS command. Specify the subroutine to be measured with start and end addresses. The maximum number of subroutines which can be measured is shown in table 1.10.

Table 1.10 Maximum Number of Measurable Subroutines

Measurement Mode	Maximum Number of Measurable Subroutines
Time measurement mode 1	8
Time measurement mode 2	8
Time measurement mode 3	4

The measurement results are displayed in the following three ways:

- Numerical ratio of total execution time and specified subroutine execution time
- Bar graph indicating the ratio of total execution time and specified subroutine execution time
- Numerical value of specified subroutine execution time

For details on the PERFORMANCE_ANALYSIS command, refer to section 7.2.29, PERFORMANCE_ANALYSIS1 to 8.

Time Measurement Mode 1: The execution time and count of the subroutine specified by the start address and end address.

- Execution count measurement

  This is counted up in the end address every time the start address of the specified subroutine is passed.
- Execution time measurement

  The measurement result does not include the execution time of the subroutine called by the specified subroutine (between the start address and end address).

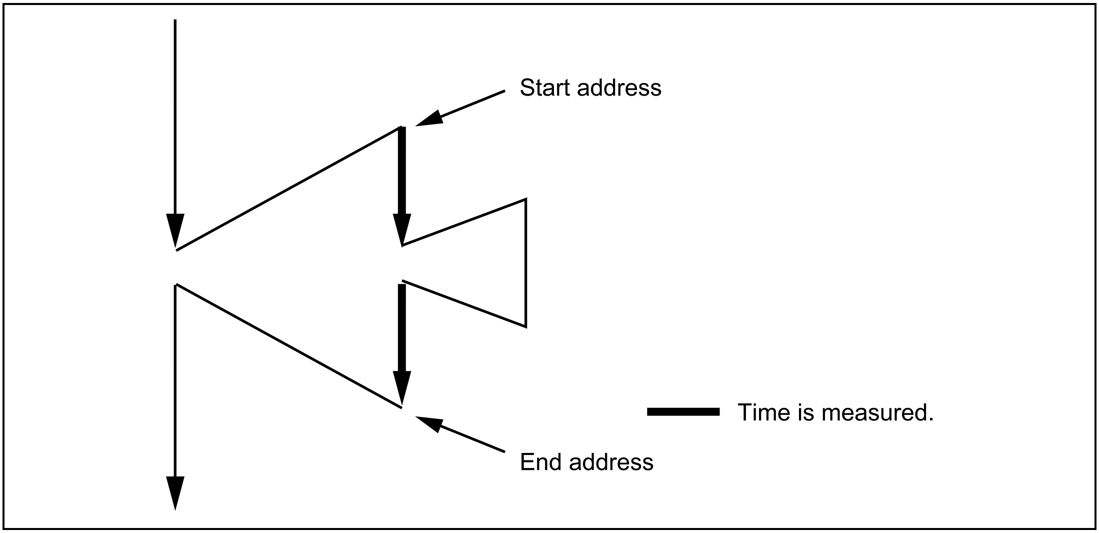


Figure 1.23 Time Measurement Mode 1

Time Measurement Mode 2: The execution time and count of the subroutine specified by the start address and end address.

- Execution count measurement

  This is counted up in the end address every time the start address of the specified subroutine is passed.
- Execution time measurement

  The measurement result includes the execution time of the subroutine called by the specified subroutine (between the start address and end address).

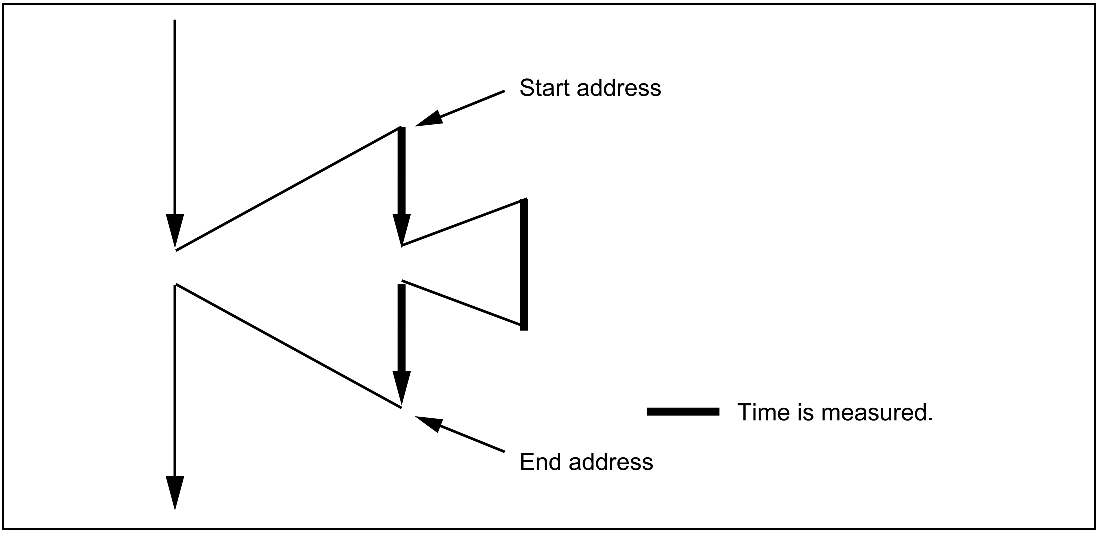


Figure 1.24 Time Measurement Mode 2

Time Measurement Mode 3: The execution time and count of the subroutine specified by the start address and end address. The combination of the channels is fixed as follows:

- PERFORMANCE_ANALYSIS1 and 2
- PERFORMANCE_ANALYSIS3 and 4
- PERFORMANCE_ANALYSIS5 and 6
- PERFORMANCE_ANALYSIS7 and 8

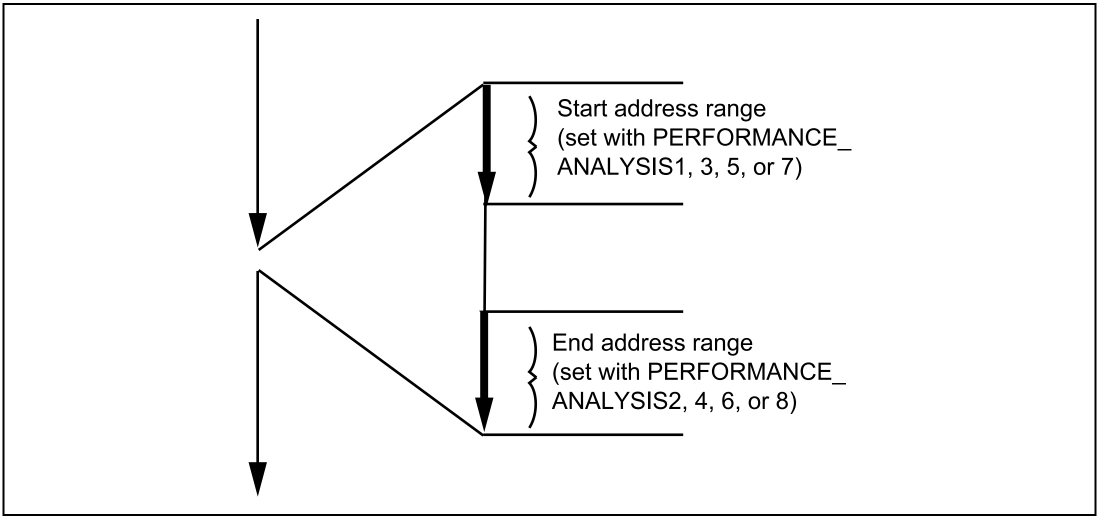


Figure 1.25 Time Measurement Mode 3

- Execution count measurement

  This is counted up in the end address every time the start address of the specified subroutine is passed.
- Execution time measurement

  The measurement starts from the program fetch cycles of the start address range and ends with the program fetch cycles of the end address range. Accordingly, the execution time of a subroutine called during this period is included.

Maximum/Minimum Subroutine Time Detection Function: This is specified in the time measurement mode 2 of PERFORMANCE_ANALYSIS_1,2,3,4. This measures the maximum/minimum execution time for a subroutine specified by the start address and end address.

Timeout Function: This compares a measured value and a user specification time during user specified subroutine execution.

- User specification time < Measured value
  User program execution breaks.
- User specification time > Measured value
  Execution time is measured.

1.10 Trigger Output

During user program execution, the emulator outputs a low-level pulse from the trigger output probe under the following two conditions.

- Trace condition satisfaction
- Hardware break condition satisfaction

When using this pulse as an oscilloscope trigger input signal, it becomes easy to adjust the user system hardware. For example, waveforms can be seen when the user program goes to a specified point.

When the trigger output is specified using the TRGB option of the EXECUTION_MODE command, a low-level pulse is output from the trigger output pin four cycles after the bus cycle in which the hardware break condition or trace condition is satisfied during emulation execution. The trigger signal is output starting from four bus cycles after the condition is satisfied until the end of the next bus cycle. If the conditions are satisfied in consecutive bus cycles, the trigger output remains low.

Note: No pulse is output from the trigger output probe when a software break condition is satisfied. In addition, a low-level pulse output timing and pulse width differ depending on each condition.

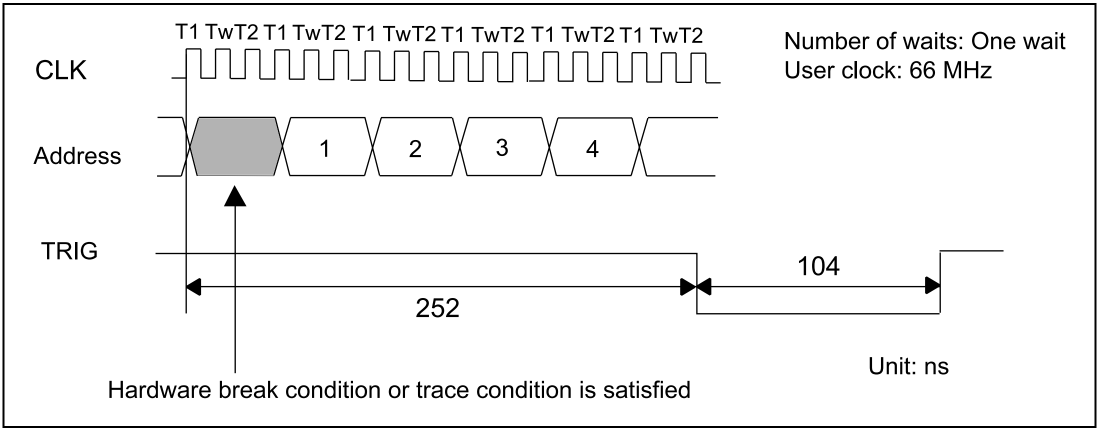

T1 TwT2 T1 TwT2 T1 TwT2 T1 TwT2 T1 TwT2 T1 TwT2
CLK
Address
1
2
3
4
TRIG
252
104
Number of waits: One wait
User clock: 66 MHz
Unit: ns
Hardware break condition or trace condition is satisfied

Figure 1.26 Pulse Output Timing

1.11 MCU Control and Status Check

The emulator is capable of switching the clock signal supplied to the MCU, outputting strobe signals when the emulation memory is accessed, checking normal operation by checking the I/O signals, and displaying the execution state. This function is effective for debugging the user system hardware.

Clock Switching: The emulation clock can be supplied from the user system clock (hereafter referred to as the user clock), the crystal oscillator installed on the evaluation chip board, and the emulator internal clock (8.25 MHz or 33 MHz). To switch the clock, refer to section 7.2.11, CLOCK, and note the following. In addition, refer to section 3.2.4, Selecting Clock in part I, E8000 Guide.

- When the clock is switched, the emulator inputs a reset signal to the MCU. This initializes the registers.
- When the user switches to the user clock and the user clock signal is not supplied, an error message is displayed and the internal clock is selected instead.
- When initiating the E8000 system program, the emulator selects the MCU clock automatically in the following order:

  — When the system is activated, clock operating mode 7 must select the CKIO input. If CKIO is not there, the external clock from the user system is normally selected. However, when the system is activated in clock operating mode 7, the following message is displayed and the system enters the command-input wait state: ** NO CLOCK. CHECK MODE

  — When an external clock is supplied from the user system, selects the user clock

  — When a crystal oscillator is installed to the evaluation chip board, selects the crystal oscillator

  — Selects the emulator internal clock (8.25 MHz or 33 MHz)

  When the clock operating mode is 1, the 8.25-MHz clock is selected, and when 0 or 3, the 33-MHz clock is selected.

Check of the I/O signals: The emulator checks the connection with the user system at system initiation. By this check, abnormalities such as short circuits of a user system interface signal can be detected. The signals to be checked are as follows:

RESETP, BREQ, WAIT, IRL0 to IRL3, IRQ4, IRQ5, and NMI

The CHECK command can check the same signals that are checked at system initiation. For details, refer to section 7.2.10, CHECK.

Emulator Execution Status Display: The emulator can display execution status information listed in table 1.11. To display the execution status, use the STATUS command. For details, refer to section 7.2.35, STATUS.

Table 1.11 Execution Status Display

Display Command	Description
RADIX=xx	Radix type
BREAK=xx	Number of breakpoints specified with the BREAK command
HOST=xx	Host-computer interface condition
CLOCK=xx	Type of clock (8.25 MHz, 33 MHz, USER, XTAL, or CKIO)
STEP_INFO=REG: (a) A: (b) SP: (c)	• Register information displayed by the STEP command • Address range displayed by the STEP command • Display size for stack contents

1.12 Emulation Monitoring Function

The emulator monitors the MCU which is being emulated during memory accesses or user program execution. Two kinds of status are monitored.

- MCU operating status
- User system power and clock status

MCU Operating Status: When executing the program with the GO command, the emulator monitors the operating status. When the status changes, the operating status display is updated. The update interval can be selected from no display, 200 ms, and 2 s with the MON option of the EXECUTION_MODE command. With this function, the user can observe the progress of the program. The operating status display and its meaning are shown in table 1.12. For details, refer to the description on operating status display, in section 7.2.21, GO.

Table 1.12 Operating Status Display

Display	Meaning
** RUNNING	The user program execution is initiated. This message is displayed once when GO command execution is started or when parallel mode is canceled. Note that this message will be deleted when **PC=xxxxxxxx is displayed.
** PC=xxxxxxxx	The program fetch address being executed is displayed with the intervals specified with the MON option of the EXECUTION_MODE command.
** VCC DOWN	User system Vcc (power voltage) is 2.65 V or less. The MCU is not operating correctly. (Displayed only when the user clock is selected.)
** RESET	RESETP signal is low. The MCU has been reset.
** WAIT A = xxxxxxxx	WAIT signal is low. The address bus value is displayed. Not displayed during memory access command execution.
** TOUT A = xxxxxxxx	The MCU stops for 1.28 ms or longer. (The address value is displayed.)
** BREQ	BREQ signal is low.
** HARDWARE STANDBY	HSTBY signal is low. However, because this signal is not input to the MCU, the MCU does not enter hardware standby mode.
** SOFTWARE STANDBY	The MCU is in the software standby mode.
** SLEEP	The MCU is in the sleep mode.

User System Power and Clock Status: The emulator monitors the user system power and clock status. If the user system power is off or the clock stops when the MCU clock is set to USER with the CLOCK command, the emulator executes the following operation according to the emulator status.

Notes: 1. If the user system power is turned off (Vcc is 2.65 V or lower), this is detected before the clock stop is detected.

2. Clock stop means that only the clock stops and the user system power remains on.

- During user program execution
  - — When the user system is turned off (Vcc is 2.65 V or lower), ** VCC DOWN is displayed. When the power is turned on again, the emulation restarts and current position of PC in the user program is displayed.
  - — When the clock stops (Vcc is 2.65 V or lower), *** 6:USER SYSTEM NOT READY is displayed and the emulator enters the command-input wait state. To operate the emulator again, restart the system program after it has been terminated by using the QUIT command.

- During the command input wait state
  - — When the user system is turned off (Vcc is 2.65 V or lower), *** 6:USER SYSTEM NOT READY is displayed and the emulator enters the command-input wait state. To operate the emulator again, restart the system program after it has been terminated by using the QUIT command. *** 6:USER SYSTEM NOT READY is displayed after the user system has been turned off and one command has been executed.
  - — When the clock stops (Vcc is 2.65 V or lower), *** 6:USER SYSTEM NOT READY is displayed and the emulator enters the command-input wait state. To operate the emulator again, restart the system program after it has been terminated by using the QUIT command.

1.13 Assembly Function

1.13.1 Overview

The ASSEMBLE command enables line assembly as shown in figure 1.27.

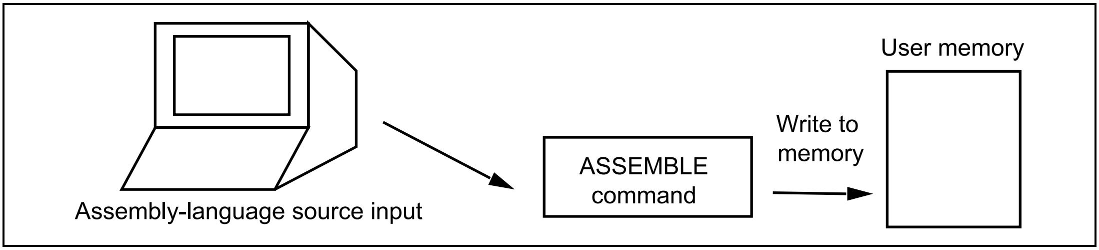


Figure 1.27 Assembly Function

Line assembly: Assembly-language source is input from the console line by line.

Refer to section 7.2.4, ASSEMBLE, for command initiation instructions.

1.13.2 Input Format

The basic instruction format is as follows.

<instruction mnemonic>[Δ<operand>,...Δ][;<comment>] (RET)

<instruction mnemonic>: Any instruction mnemonic described in the SH7729/SH7709A Series Programming Manual and any assembler directive listed in table 1.13 can be used.

<operand>: Any mnemonic described in the SH-Series Programming Manual can be used (table 1.14).

<comment>: A character string after a semicolon (;) is considered to be a comment.

[ ]: Items within square brackets ([ ]) can be omitted.
However, some <operand> values for specific instructions are required.

Δ: Indicates a space.

Notes: 1. Continuation lines cannot be input.

2. The default for radix of constants is set by the RADIX command.

Table 1.13 Assembler Directives

Directive	Operand	Description
Δ.DATA[.s]Δ	<value>[,<value>...]	• Reserves an area for initialized fixed-length data. The size of the area is equal to the unit length given by s: B (byte), W (word) or L (longword). Default size is L. • If any <value> exceeds the capacity of the size code (s), an error occurs. • A line can contain up to 40 bytes.
Δ.RES[.s]Δ	<value>	• Reserves data areas. The number of areas is given by <value>. The size of each area is given by s: B (byte), W (word) or L (longword). Default size is L. • Up to 4,294,967,295-byte area can be reserved at one time.

Table 1.14 Operand Descriptions

Format	Addressing Mode	Remarks	
Rn	Register direct	Rn:	General register name (SP can be specified instead of R15)
SR		SR:	Status register
GBR		GBR:	Global base register
VBR		VBR:	Vector base register
MACH		MACH:	Multiply and accumulate register
MACL		MACL:	Multiply and accumulate register
PR		PR:	Procedure register
SSR		SSR:	Saving status register
SPC		SPC:	Saving program counter
@Rn	Register indirect	Rn:	General register name
@Rn+	Register indirect with post-incrementation	Rn:	General register name
@-Rn	Register indirect with pre-decrementation	Rn:	General register name
@(disp, Rn)	Register indirect with displacement	disp:	Displacement value
		Rn:	General register name
@(R0, Rn)	Register indirect with index	R0,Rn:	General register name
@(disp, GBR)	GBR indirect with displacement	disp:	Displacement value
		GBR	Global base register
@(R0, GBR)	GBR indirect with index	R0:	General register name
		GBR	Global base register
@(disp, PC)	PC relative with displacement	disp:	Displacement value
		PC	PC value within vector address table
aaaa	PC relative	aaaa:	Address value (Usable with BF, BT, BRA, and BSR instructions)
#imm	Immediate	imm:	Immediate data value

Notes: 1. For the address value, immediate data value and displacement values, the formula (addition or subtraction) can be used. However, disassemble is displayed only in address value.

2. If the immediate data value is different from the specified operation size, an error occurs.

1.13.3 Disassembly

The emulator has a disassembly function to display user program contents in mnemonics. This function is performed with the DISASSEMBLE command and enables to debug without referencing to a program list. For details, refer to section 7.2.16, DISASSEMBLE.

Section 2 Differences between the MCU and the Emulator

When the emulator system is initiated, or when the emulator resets the MCU as a result of a command, such as the CLOCK command switching the clock or the RESET command, note that the general registers and part of the control registers are initialized.

Table 2.1 Differences between Initial Values of the MCU and Emulator Registers

Status	Register	Emulator	MCU
Emulator initiation (power-on)	PC	Reset vector value	Reset vector value
	R0 to R14	H'00000000	Undefined
	R15 (SP)	H'00000000	Undefined
	R0_BANK0 to R7_BANK0	H'00000000	Undefined
	R0_BANK1 to R7_BANK1	H'00000000	Undefined
	SR	H'700000F0	H'00000XFX *
	SSR	H'00000000	Undefined
	SPC	H'00000000	Undefined
	PR	H'00000000	Undefined
	VBR	H'00000000	Undefined
	GBR	H'00000000	Undefined
	RE	H'00000000	Undefined
	MOD	H'00000000	Undefined
	MACH	H'00000000	Undefined
	MACL	H'00000000	Undefined
	DSR (SH7729 only)	H'00000000	H'00000000
	A0, A1 (SH7729 only)	H'00000000	Undefined
	M0, M1 (SH7729 only)	H'00000000	Undefined
	X0, X1 (SH7729 only)	H'00000000	Undefined
	Y0, Y1 (SH7729 only)	H'00000000	Undefined
	A0G, A1G (SH7729 only)	H'00	Undefined

Note: X is an undefined value.

The emulator's user system interface is provided with pull-up resistors and a buffer, causing the signals to be delayed slightly. Also, the pull-up resistors will change high-impedance signals to high-level signals. Adjust the user system hardware accordingly. Refer to section 4, User System Interface.

The emulator can use an operating frequency of 66 MHz or lower. Note, however, that the emulator cannot use an operating frequency higher than 66 MHz. If the operating frequency is set to higher than 66 MHz, correct emulation cannot be guaranteed.

Section 3 MCU Function Support

The MCU has six operating modes. The emulator does not support operating modes 2 and 4 when the crystal oscillator is connected. This section describes how the emulator supports the MCU functions.

Note: The crystal oscillator connected to the crystal oscillator terminals X0 and X1 on the evaluation chip board is connected to the oscillator on the evaluation board to perform clock oscillation. This clock source is input to the EXTAL pin of the MCU. Note that the crystal oscillator cannot be directly connected to the EXTAL and XTAL pins of the MCU.

3.1 Operating Mode Setting

The emulator can select the clock operating mode, the bus width in the CS0 area, and the big or little endian. The operating mode is set using the MODE command.

The following shows how to set the clock operating mode, the bus width in the CS0 area, and the big or little endian. For details, refer to section 7.2.27, MODE. Table 3.1 lists the MCU clock operating mode.

Table 3.1 MCU Clock Operating Mode

Operating	Pin Setting			
Mode	MD2	MD1	MD0	Description
0	0	0	0	Inputs an external clock to the EXTAL pin, forms a waveform without multiplication at the PLL circuit 2, and outputs clock from the CKIO pin
1	0	0	1	Inputs an external clock to the EXTAL pin, multiplies the frequency by four at PLL circuit 2, and outputs clock from the CKIO pin
2	0	1	0	Operates the crystal oscillator, multiples the frequency by four at PLL circuit 2, and outputs a clock from the CKIO pin
3	0	1	1	Inputs an external clock to the EXTAL pin, forms a waveform without multiplication at PLL circuit 2, and outputs a clock from the CKIO pin
4	1	0	0	Operates the crystal oscillator, forms a waveform without multiplication at PLL circuit 2, and outputs a clock from the CKIO pin
7	1	1	1	Outputs the external clock from the CKIO pin

Note: The E8000 system program terminates after the MODE command setting. An operating mode specified using the MODE command will be valid only after the emulator is re-initiated. Therefore, the emulator must be reset after specifying an operating mode. At this time, emulator specifications such as emulation memory attributes and break point settings will not be saved. To save them, use the CONFIGURATION command. The emulator can only support clock operating modes 0, 1, 3, or 7.

The bus width setting in the CS0 area is listed in table 3.2.

Table 3.2 Bus Width Setting in the CS0 Area

MD4	MD3	Bus Width in the CS0 Area
0	0	Cannot be set
0	1	8 bits
1	0	16 bits
1	1	32 bits

The big or little endian setting is listed in table 3.3.

MD5	Big/Little Endian
0	Big endian
1	Little endian

In the emulator, the operating mode previously set is saved in the configuration file in the E8000 station. At initialization, the emulator initiates the system with the operating mode specified with the MODE command.

3.2 Memory Space

The MCU has 448 Mbytes of memory space in the CS0 to CS6 area. The standard emulation memory (8 Mbytes) can be set in 4-Mbyte units to the memory space. The emulation memory can be set in two areas by dividing the standard emulation into two parts (4 Mbytes x 2). The areas that are not set as the emulation memory are set as the user system memory. For details, refer to section 7.2.25, MAP.

- U: User system memory
- S: Standard emulation memory

The area which is set as emulation memory can be independently specified as a write-protected.

- SW: Write-protected

The above attribute setting is valid in external memory area only; they are invalid in the internal memory, and internal I/O area.

3.2.1 Internal I/O Area

If an attempt is made to access the internal I/O area, the internal I/O area in the MCU installed in the emulator is accessed regardless of the memory attribute set with the MAP command. The internal I/O area can be read from or written to by the user program or with emulator commands. When writing to the internal I/O area with an emulator command (MEMORY command), the following warning message is displayed and the emulator starts writing without verifying.

```
*** 86:  INTERNAL AREA
```

3.2.2 External Memory Area

The MCU external memory area can be set with all memory attributes that the emulator supports. Memory corresponding to the allocated attributes can be accessed by the user program or with emulator commands.

3.3 Low Power-Consumption Mode (Sleep and Standby)

To reduce power consumption, the MCU has sleep and standby modes. These modes can only be cleared with a normal clearing function. Trace information is not acquired in these modes.

3.4 Interrupts

During emulation, the user can interrupt the MCU. If an interrupt occurs while the emulator is waiting for command input, whether the interrupt is enabled or disabled can be selected.

When Interrupts Are Not Processed: Generally, interrupts are not processed in command input wait state. However, if an edge sensitive internal or external interrupt occurs while the emulator is waiting for command input, the emulator latches the interrupt and executes the interrupt processing routine when the GO command is entered.

When Interrupts Are Processed: Interrupts can be processed in command input wait state by using the BACKGROUND_INTERRUPT command. A loop program is executed in the background in command input wait state, and when an interrupt occurs, the processing for the interrupt starts. For details, refer to section 7.2.5, BACKGROUND_INTERRUPT.

Notes: 1. The loop program specified by the BACKGROUND_INTERRUPT command must be stored in the internal RAM area.

2. In interrupt processing, hardware break, software break, and access to the write-protected area or guarded area are detected and a break occurs.

3. Trace information is not acquired during interrupt processing.

3.5 Control Input Signals (RESETP, RESETM, WAIT, BREQ)

The MCU control input signals are RESETP, RESETM, WAIT and BREQ. The RESETP and RESETM signals are valid only when emulation has been started with the GO command. The WAIT and BREQ signals are valid during execution with either the MEMORY command, the GO command, STEP command, or STEP_OVER command. Therefore, while the emulator is waiting for command input, the user cannot input RESETP, RESETM, WAIT or BREQ signals to the MCU.

The RESETP, RESETM or BREQ signal inputs will be masked, using the EXECUTION_MODE command. These signals will not be input to the MCU during user program execution when the RESETP, RESETM, or BREQ signal is masked, that is, the option RES = D or BREQ = D is specified, using the EXECUTION_MODE command.

3.6 Watchdog Timer (WDT)

The WDT operates during emulation (GO or STEP command execution), and while waiting for command input.

3.7 Serial Communication Interface (SCI)

The serial communication interface signals are connected to the user system directly from the MCU on the evaluation chip board. Therefore, like the timers, the interface is valid during the command input wait state as well as emulation. For example, when data is written to the transmit data register (TDR) using the MEMORY command, after the serial communication interface output has been prepared, data is output to the TxD pin.

3.8 DMA Controller (DMAC)

The DMAC operates during the command input wait state as well as during emulation. When a transfer is requested, the DMAC executes a DMA transfer.

3.9 Hitachi User Debugging Interface (Hitachi-UDI)

The Hitachi user debugging interface (Hitachi-UDI) transfers the data. The data transfer between the chip and the external controller is executed by the command input from the external controller. However, the Hitachi-UDI cannot be used when using the emulator.

3.10 Bus State Controller

The wait state controller has a programmable wait mode and a WAIT pin input mode. The programmable wait mode is valid when the emulation memory or user external memory is accessed, but input to the user WAIT pin is only valid when user external memory is accessed. However, the EXECUTION_MODE command can be used to enable input to the user WAIT pin during emulation memory access cycles.

3.11 User Break Controller (UBC)

The UBC cannot be used.

3.12 I/O Port

The I/O port can also be used as peripheral module input/output pins or as an address/data bus. It is specified as I/O port pins according to the operating mode or internal register settings. The I/O port pins are also valid in the emulator command input wait state or during emulation.

The I/O port pins can be read from and written to by the MEMORY command.

3.13 A/D Converter

Analog I/O pins are directly connected to the user system from the MCU installed on the evaluation chip board. Therefore, they are valid in the emulator command input wait state as well as during emulation.

The A/D converter also has AVcc, AVss, and ADTRG pins. Power is supplied to AVcc (the analog power-supply pin) from the emulator, and AVss (analog GND pin) is connected to the GND of the emulator. Thus I/O port pins remain valid in the emulator command-input wait state and during emulation.

3.14 Memory Management Unit

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

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

The VP_MAP tables are supported by the following commands:

ASSEMBLE, BREAK, BREAK_CONDITION_A,B,C, BREAK_SEQUENCE, DATA_CHANGE, DATA_SEARCH, DISASSEMBLE, DUMP, FILL, MEMORY, STEP_INFORMATION, INTFC_LOAD, INTFC_SAVE, INTFC_VERIFY, LOAD, SAVE, VERIFY, LAN_LOAD, LAN_SAVE, and LAN_VERIFY

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

Example:

1. Create VP_MAP tables for translating virtual addresses H'10000 to H'10FFF to physical addresses H'4000000 to H'4000FFF and virtual addresses H'11000 to H'11FFF to physical addresses H'0 to H'FFF.

```
:VP 10000 10FFF 4000000 (RET)
:VP 11000 11FFF 0 (RET)
:VP (RET)
<VADDR_TOP>  <VADDR_END>  <PADDR_TOP>
00010000     00010FFF     04000000
00011000     00011FFF     00000000
DISABLE
```

2. Then, enable the VP_MAP tables. (When the tables are disabled, addresses are not translated.)

```
:VP ;E (RET)
:VP (RET)
<VADDR_TOP>  <VADDR_END>  <PADDR_TOP>
00010000     00010FFF     04000000
00011000     00011FFF     00000000
ENABLE
```

When file test.abs is loaded, the VP_MAP table is used as shown in figure 3.1. The test.abs file is allocated to virtual addresses H’10000 to H’12FFF. Since there is no address translation table from H’12000 to H’12FFF, the address is not translated and loaded at the same address. Virtual address H’10000 to H’10FFF is loaded to physical address H’400000 to H’4000FFF, virtual address H’11000 to H’11FFF is loaded to physical address H’0 to H’FFF, and virtual address H’12000 to H’12FFF is loaded to physical address H’12000 to H’12FFF.

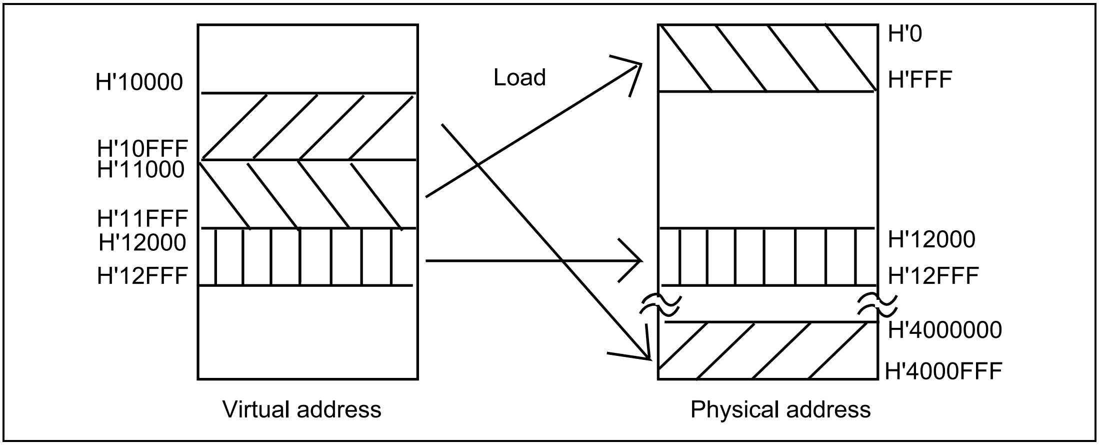


Figure 3.1   Address Translation according to VP_MAP Tables

```
:IL ;R:TEST.ABS (RET)
ALL SYMBOL LOAD (Y/N)? Y(RET)
TOP ADDRESS = 00010000
END ADDRESS = 00012FFF
:
```

Address specification for each command is shown in table 3.1.

- When V, P, or ASID is omitted from the address specification of a command:

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

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

  The address is not translated.

Note: When the VP_MAP command setting is valid for the INTFC_LOAD, INTFC_VERIFY, LOAD, or VERIFY commands, do not specify an offset, P, V, or <ASID value>.

Table 3.3 Address Translation Tables

	VP_MAP		MMU		
Command Address Spcifica-tion*	**Enabled/ Disabled**	**Within/ Outside the Range**	**Enabled/ Disabled**	**Within/Outside the TLB Range**	**Table Used for Translation**
V, P, ASID is omitted	Enabled	Within the range	Enabled	Within the range	Translated according to the VP_MAP table
				Outside the range	Translated according to the VP_MAP table
			Disabled	Within/outside the range	Translated according to the VP_MAP table
		Outside the range	Enabled	Within the range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within/outside the range	Not translated
	Disabled	Within/ outside the range	Enabled	Within the range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within/outside the range	Not translated
V option	Enabled/ disabled	Within/ outside the range	Enabled	Within the range	Translated according to the TLB table
				Outside the range	TLB error
			Disabled	Within the range	Translated according to the TLB table
				Outside the range	TLB error
P option	Enabled/ disabled	Within/ outside the range	Enabled/disabled	Within/outside the range	Not translated

Section 4 User System Interface

The emulator is connected to the user system via the evaluation chip board. Probe signal trace and break can be enabled by connecting four external probes to the user system.

The trigger output probe can output a low-level pulse as an oscilloscope trigger signal. For details, refer to section 1.10, Trigger Output.

1. **User System Interface Circuits:** The circuits that interface the MCU in the evaluation chip board to the user system include buffers and resistors, as described below. When connecting the emulator to a user system, adjust the user system hardware compensating for FANIN, FANOUT, and propagation delays.

   The AC timing values when using the emulator are shown in table 4.1

Note: The values with the emulator connected, in table 4.1, are measurements for reference but are not guaranteed values.

Table 4.1 Bus Timing when Using the Emulator (Bus Clock: 66 MHz)

	MCU Specifications (ns)		
Item	**Min**	**Max**	**Values with Emulator Connected (ns)**
tAD	1	13	3
tBSD	•	12	3
tCSD	1	12	4
tNMIS	10	•	20
tRWD	1	12	3
tRSD	•	12	4
tRESPS	23	•	32
tWDD	•	14	2
tWED	•	12	5

The basic bus cycle (two states) is shown in figure 4.1. The user system interface circuits connected to the user system are shown in figure 4.2.

2. The following signal lines are connected directly with the user system and used in the emulator. If there is much load on the user system, the emulator may not function correctly. Where possible, use buffered signals other than those for SDRAM access in the user system.

   A0 to A25, D0 to D31, WE0 to WE3, RD, CASHH to CASLL, RASxx, and CKIO

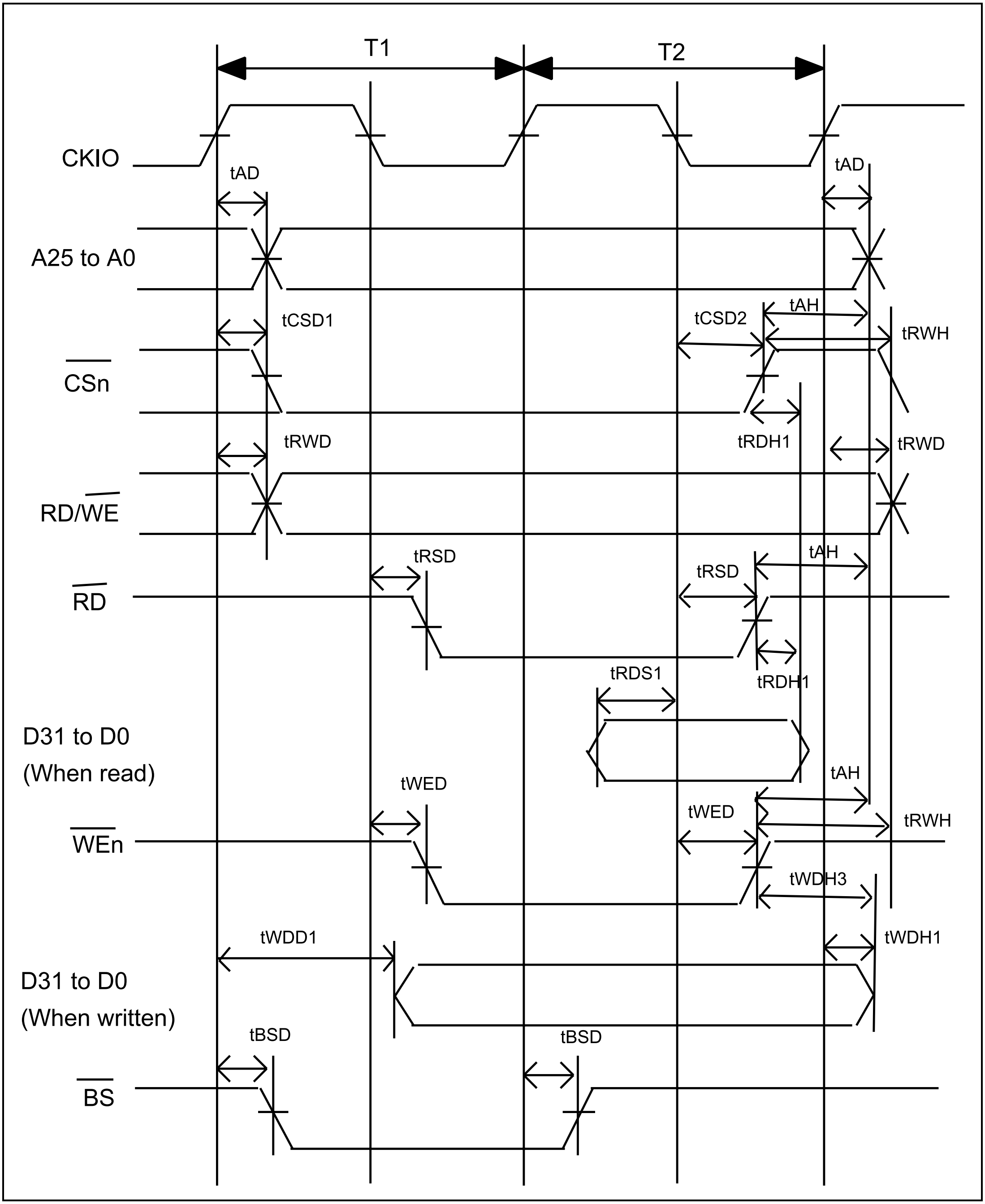


Figure 4.1 Basic Bus Cycle

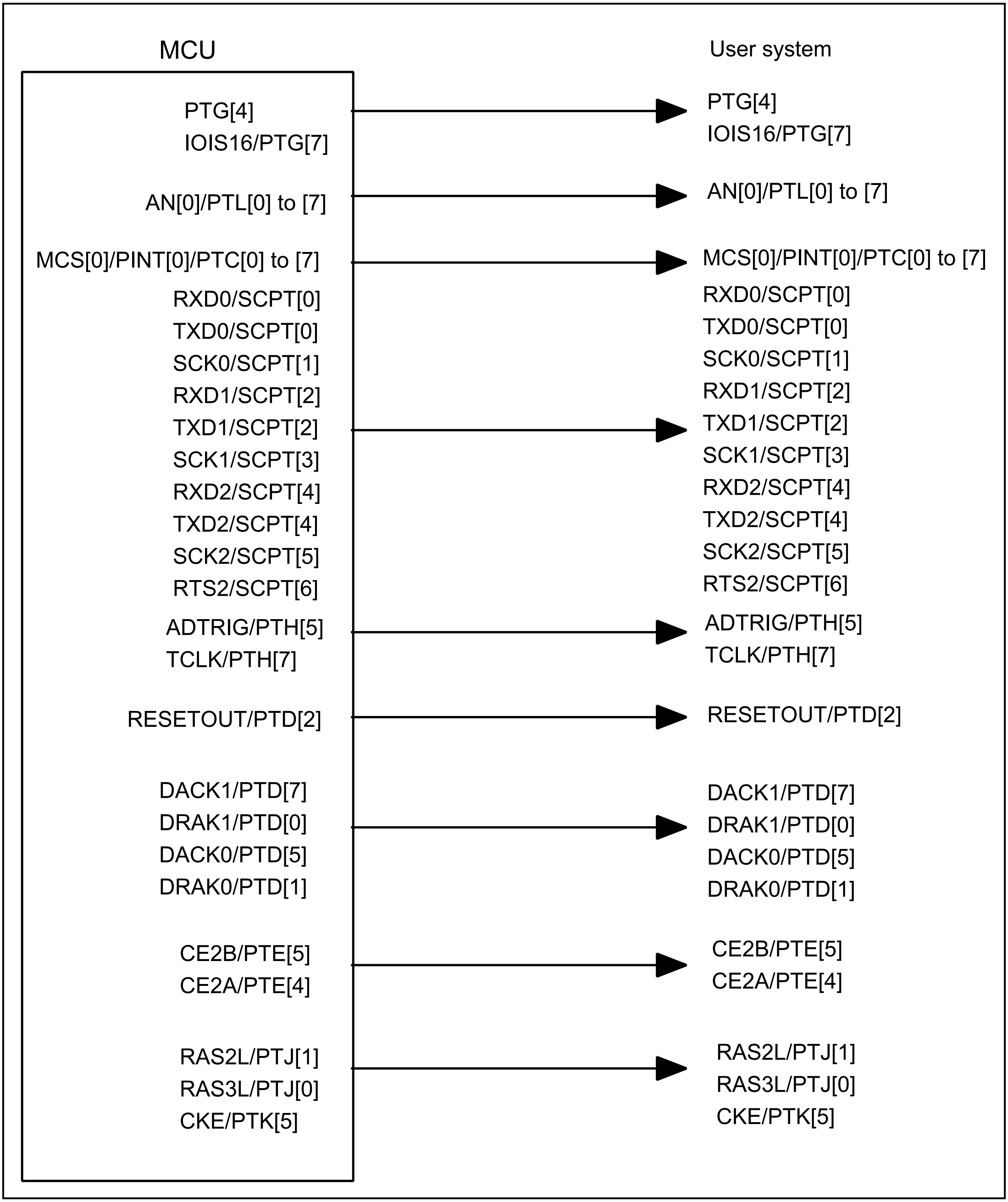


Figure 4.2 User System Interface Circuits

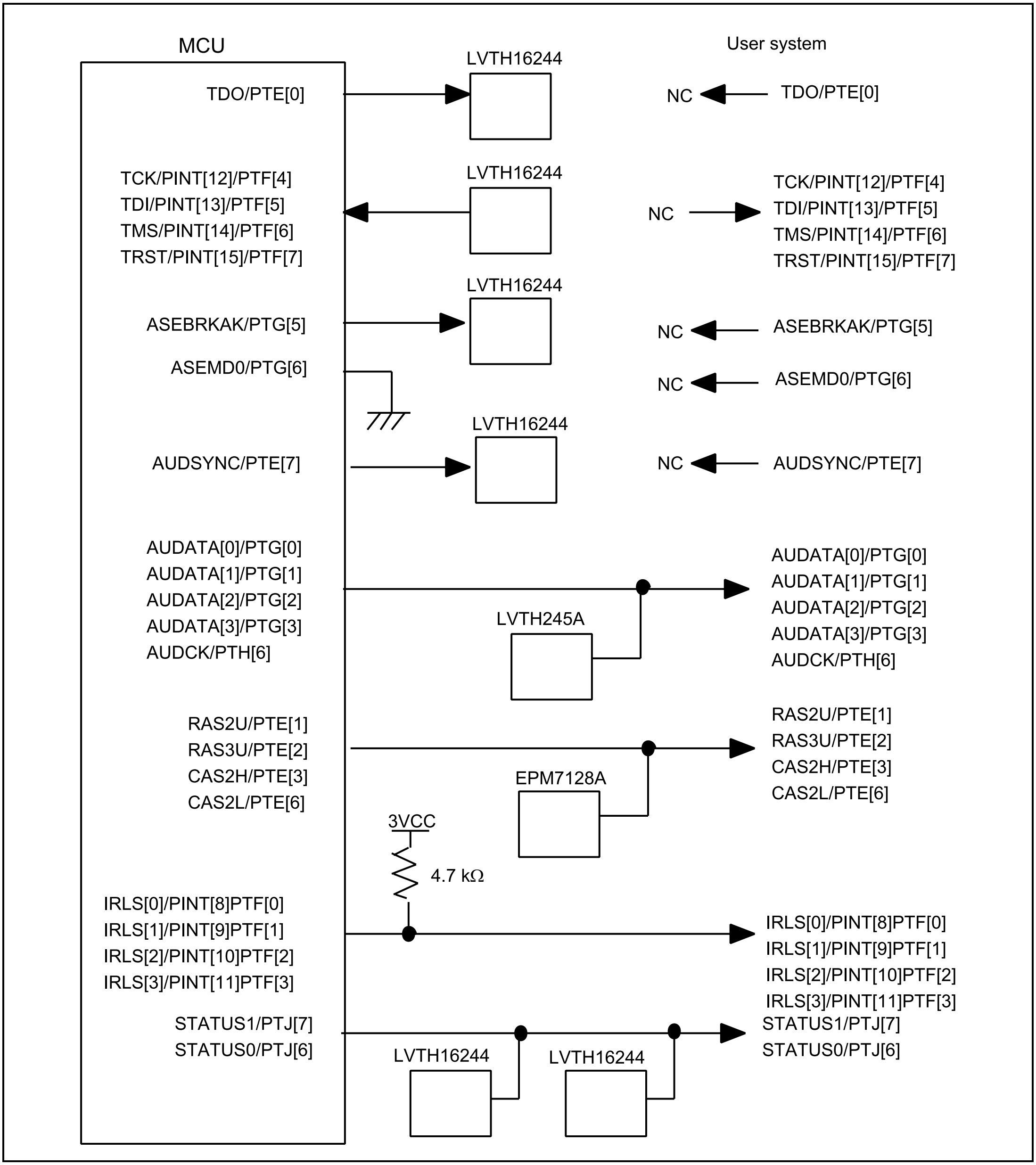


Figure 4.2 User System Interface Circuits (cont)

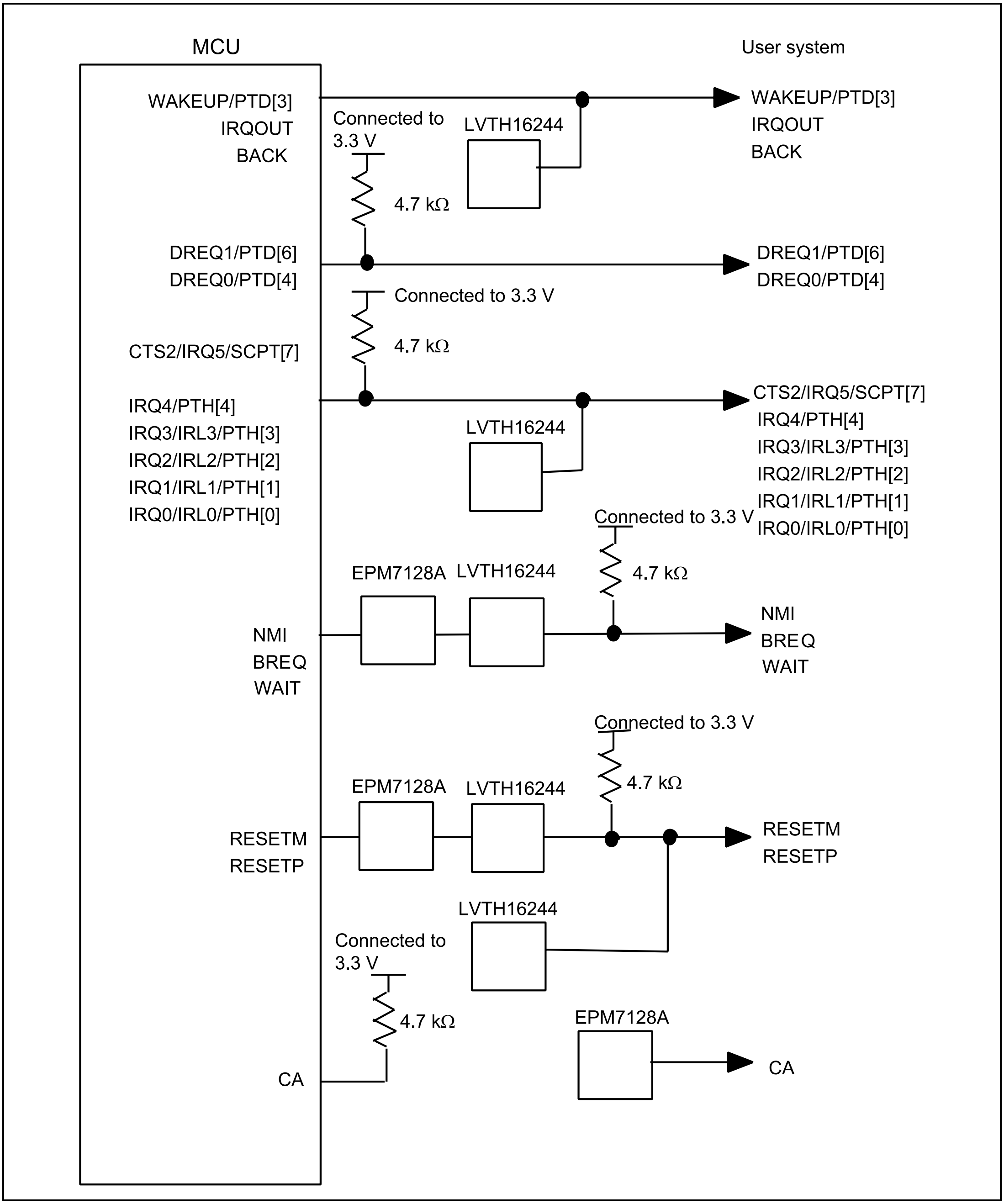


Figure 4.2 User System Interface Circuits (cont)

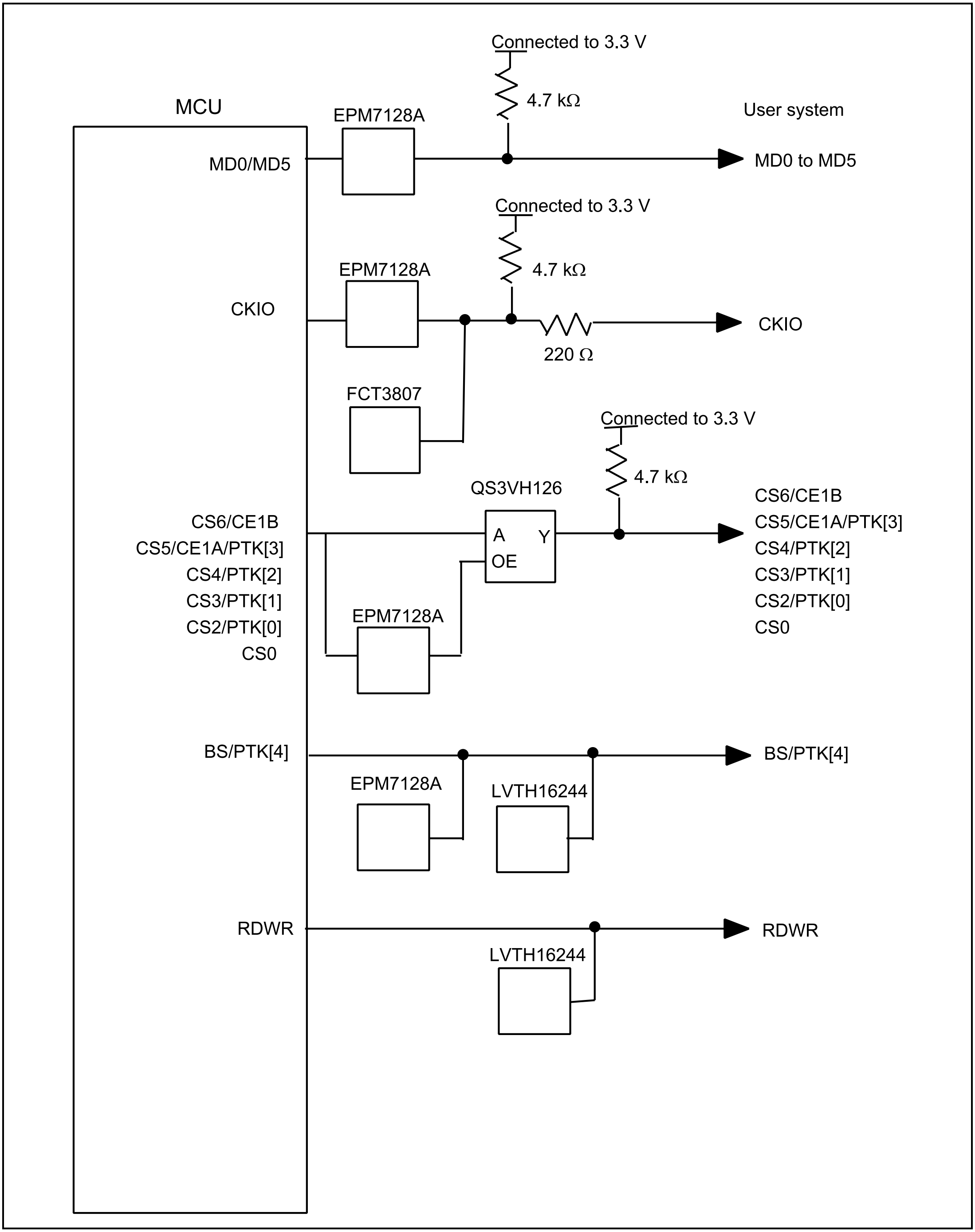


Figure 4.2 User System Interface Circuits (cont)

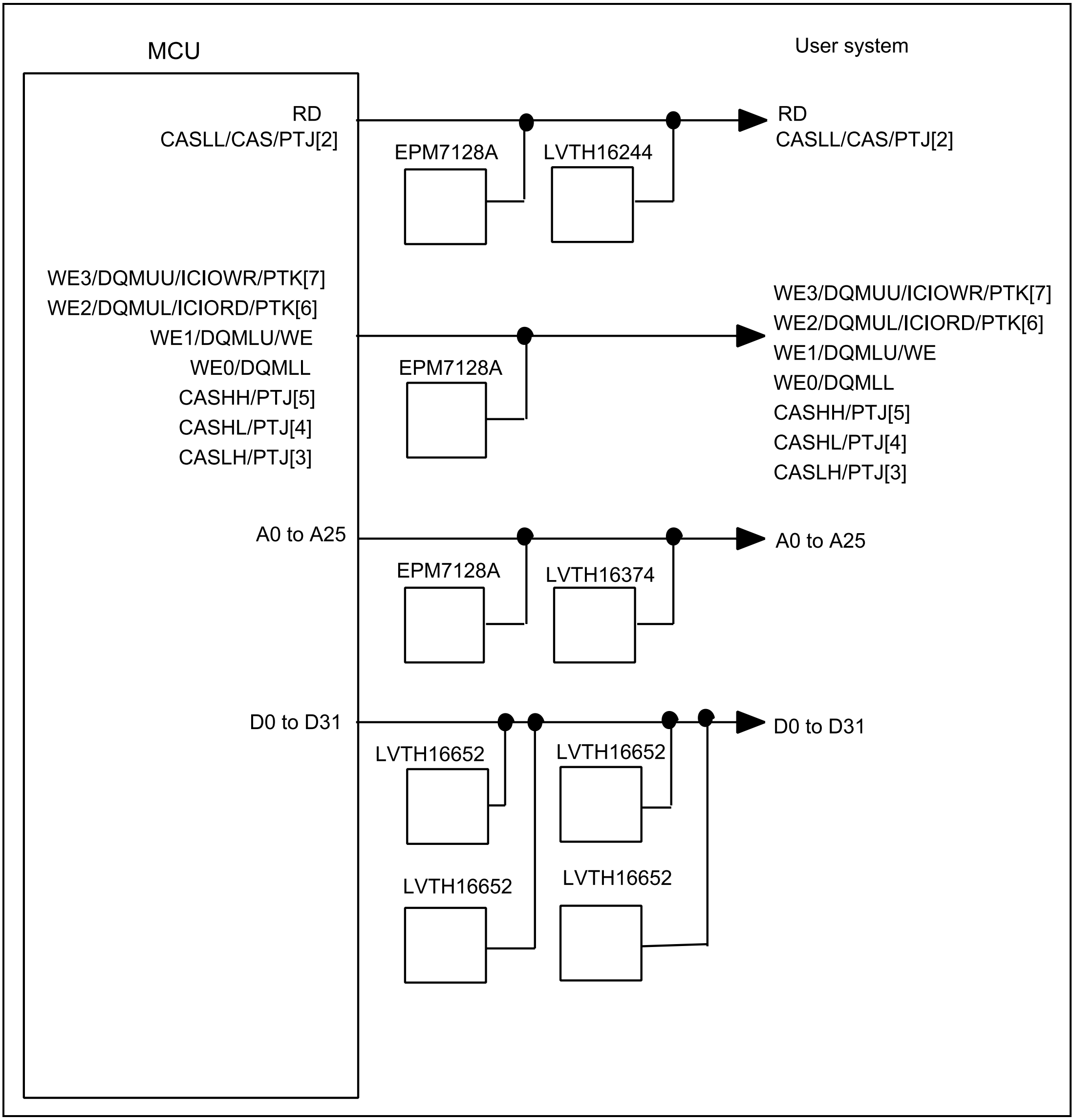


Figure 4.2 User System Interface Circuits (cont)

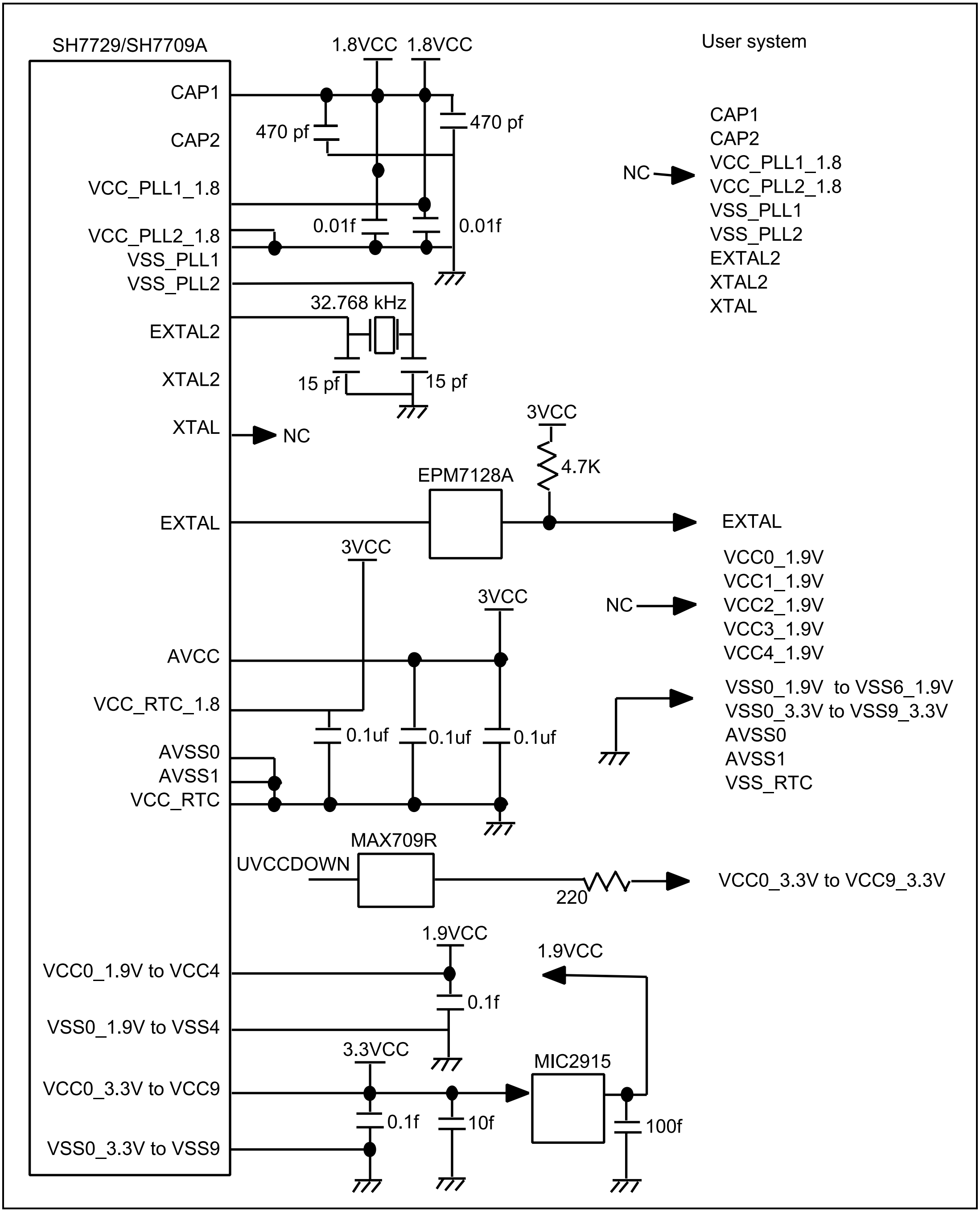


Figure 4.2 User System Interface Circuits (cont)

Section 5 Troubleshooting

The emulator internal system test checks the emulator's internal RAM and registers at power-on and at system program initiation.

5.1 Internal System Test

Internal System Test at Power-On: The emulator checks its internal RAM and registers at power-on. While tests are in progress, the following messages are displayed:

```
E8000 MONITOR (HS8000EST02SR) Vm.n
Copyright (C) Hitachi, Ltd. 1995          (a)
Licensed Material of Hitachi, Ltd.

TESTING                                   (b)
RAM 0123

START E8000
 S : START E8000
 F : FLASH MEMORY TOOL                    (c)
 L : SET LAN PARAMETER
 T : START DIAGNOSTIC TEST
      (S/F/L/T) ?
```

(a) Emulator monitor start message

(b) Internal RAM and registers are being tested.

— A number from 0 to 3 is displayed as each of the four internal RAM blocks has been tested. If an error occurs, the address, write data, and read data are displayed as follows:

   ** RAM ERROR ADDR=xxxxxxxx W-DATA=xxxxxxxx R-DATA=xxxxxxxx

— After RAM testing is completed, the registers are tested. No data will be displayed if an error does not occur. If an error occurs, the following message is displayed:

   *** xxxx REGISTER ERROR W-DATA=xx R-DATA=xx

   xxxx: Name of emulator internal register where an error occurs

(c) The emulator monitor is in command input wait state.

Note: Operation continues if an error occurs in step (b), but the error should be investigated according to section 5.2, Troubleshooting Procedure, without loading the emulator system program.

Internal System Test at Emulator System Program Initiation: The emulator system program performs internal system tests, mainly on the emulator registers, at its initiation.

```
SH7729 E8000   (HS7729EDD81SF)  Vm.n
Copyright (C)  Hitachi, Ltd. 1998                                   (a)
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING                                          (b)
HARDWARE REGISTER READ/WRITE CHECK                                  (c)
FIRMWARE SYSTEM LOADING                                             (d)
EMULATOR FIRMWARE TEST                                              (e)
**  RESET BY E8000 !                                                (f)
CLOCK = 8.25MHz                                                     (g)
MPU NAME=SH7729  MODE=xx  (MD5-0 = xx)                              (h)
MONITOR START ADDRESS=03FC0000 MONITOR BUS WIDTH=32BIT              (h)
AREA2 MEMORY=NORMAL AREA3 MEMORY=NORMAL SDRAM TYPE=1MB-16BIT        (h)
DRAM BUS WIDTH=32BIT                                                (h)
CS2/PTK[0] PIN=CS2 CS3/PTK[1] PIN=CS3 CS4/PTK[2] PIN=CS4            (h)
CS5/CE1A/PTK[3] PIN=CS5 STATUS[1:0]/PTJ[7:6] PIN=STATUS[1:0]        (h)
WE[2:3]/PTK[6:7] PIN=WE[2:3] BS/PTK[4] PIN=BS                       (h)
CAS[LL:HH]/PTJ[2:5] PIN=CAS[LL:HH]                                  (h)
CAS[LL:2H]/PTJ[2:3],PTE[3,6] PIN=CAS[LL:2H]                         (h)
JTAG CLOCK=xx.xMHz                                                  (h)
FALED AT xxxx                                                       (i)
REMAINING EMULATION MEMORY S=8MB                                    (j)
:_                                                                  (k)
```

(a) Emulator system program start message. Vm.n indicates the version number.

(b) Configuration file is being loaded. If an invalid configuration file is assigned, the following message is displayed:

*** 54:INVALID CONFIGURATION FILE

If no configuration file is contained in the memory, the following message is displayed:

*** 55:CONFIGURATION FILE NOT FOUND

Reinstall the configuration file.

(c) The emulator control registers are being checked. If an error occurs, one of the following messages is displayed:

```
***  INVALID DCONT BOARD                                                          (I)
***  DEVICE CONTROL BOARD DISCONNECTION                                           (ii)
***  EVACHIP BOARD DISCONNECTION                                                  (iii)
***  xxxxxxxx   REGISTER ERROR W-DATA = xxxx R-DATA = xxxx                        (iv)
***  SHARED RAM ERROR ADDR = xxxxxx  W-DATA = xxxxxxxx   R-DATA = xxxxxxxx        (v)
***  BxTBM ERROR  ADDR= xxxxxx  W-DATA= xxxxxxxx   R-DATA = xxxxxxxx              (vi)
*** FIRM RAM ERROR  ADDR= xxxxxx  W-DATA= xxxxxxxx   R-DATA = xxxxxxxx          (vii)
```

(i) Another device control board is connected. Please check the MCU type and install the suitable emulator system program, or change the device control board.

(ii) The device control board is disconnected.

(iii) The evaluation chip board is disconnected.

(iv) An error occurred in the register.

xxx: Name of emulator internal register where an error occurs

B0TRAR, ECT, B0CNR, B0MDCNR, B0MASCR, B0CECR, B1CNR, B1MDCNR, B1MASCR, B1CECR, MAPR0, MAPR1, MAPR2, MAPR3

(v) An error occurred in the shared RAM.

(vi) An error occurred in the trace buffer memory.

(vii) An error occurred in the firm RAM area.

(d), (e) A program operating in the device control board is being loaded and the device control board is being tested. If an error occurs, the following message is displayed:

```
*** INVALID FIRMWARE SYSTEM                       (i)
*** EMULATOR FIRMWARE NOT READY                   (ii)
*** FIRMWARE SYSTEM FILE NOT FOUND                (iii)
```

(i) Another MCU firmware has been installed. Reinstall the correct emulator system program.

(ii) A program operating in the device control board is not operating correctly. Please check that the evaluation chip board is connected correctly.

(iii) A program operating in the device control board does not exist. An incorrect system program has been registered in the flash memory. Reinstall the system program and restart the emulator.

Note: If the (CTRL) + C keys or (BREAK) key is pressed during testing for the device control board, the test is aborted.

(f) The RESETP signal is input to the MCU.

Note: (f) is not executed if an error has occurred in step (c), (d), or (e).

(g) The specified clock type is displayed.

(h) The MCU operating mode on the emulator and the status of user system mode selection pins.

(i) MCU pin status is checked.

Note: (i) is not executed if an error has occurred in step (c), (d), or (e).

(j) The remaining emulation memory size that can be assigned.

(k) Emulator system program initiation is completed, and the emulator waits for emulation command input.

Emulator System Failure: If an invalid exception occurs during emulator monitor or emulator system program execution, the system shuts down. No key input from the key board will be received but the following message is displayed:

```
<exception>    PC=xxxxxxxx
***  E8000  SYSTEM  DOWN  ***
```

If an error occurs, re-execute using another system disk. If an error still occurs, inform a Hitachi sales agency of the error.

5.2 Troubleshooting Procedure

This section provides a troubleshooting Problem Analysis Diagram (PAD, see figure 5.1) to reduce the time taken by troubleshooting.

As you work through the diagram:

- Follow the instructions that request operator assistance or intervention.
- Note that “system defect” means that the emulator station is malfunctioning. Execute the diagnostic program as described in the Diagnostic Program Manual (HS7729TM81HE), and inform a Hitachi sales agency of the test results in detail because a system defect may be caused by a number of reasons.

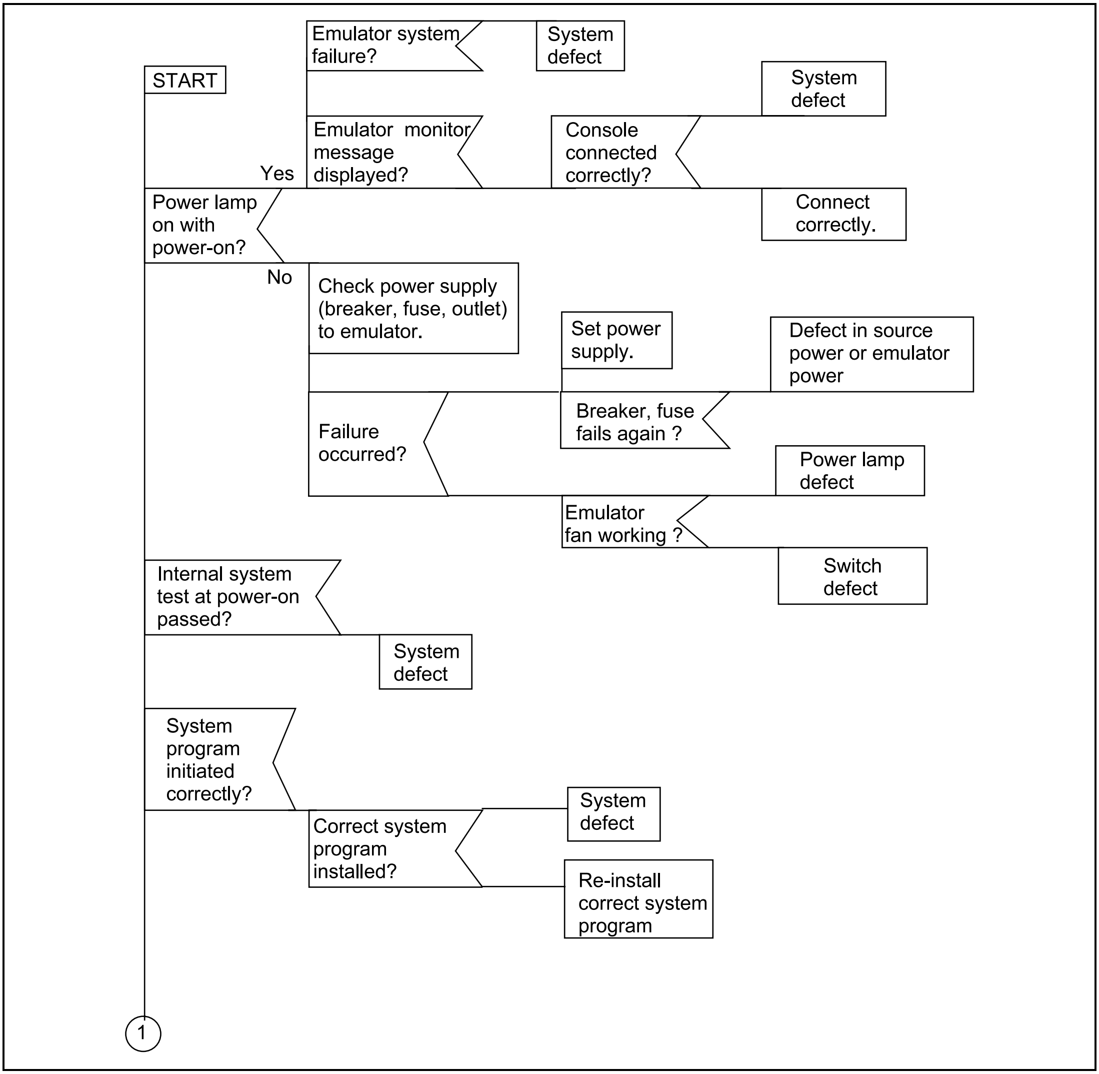

START
Yes
Emulator system failure?
System defect
Emulator monitor message displayed?
Console connected correctly?
System defect
Connect correctly.
Power lamp on with power-on?
No
Check power supply (breaker, fuse, outlet) to emulator.
Set power supply.
Defect in source power or emulator power
Failure occurred?
Breaker, fuse fails again ?
Power lamp defect
Emulator fan working ?
Switch defect
Internal system test at power-on passed?
System defect
System program initiated correctly?
Correct system program installed?
System defect
Re-install correct system program
1

Figure 5.1 Troubleshooting PAD

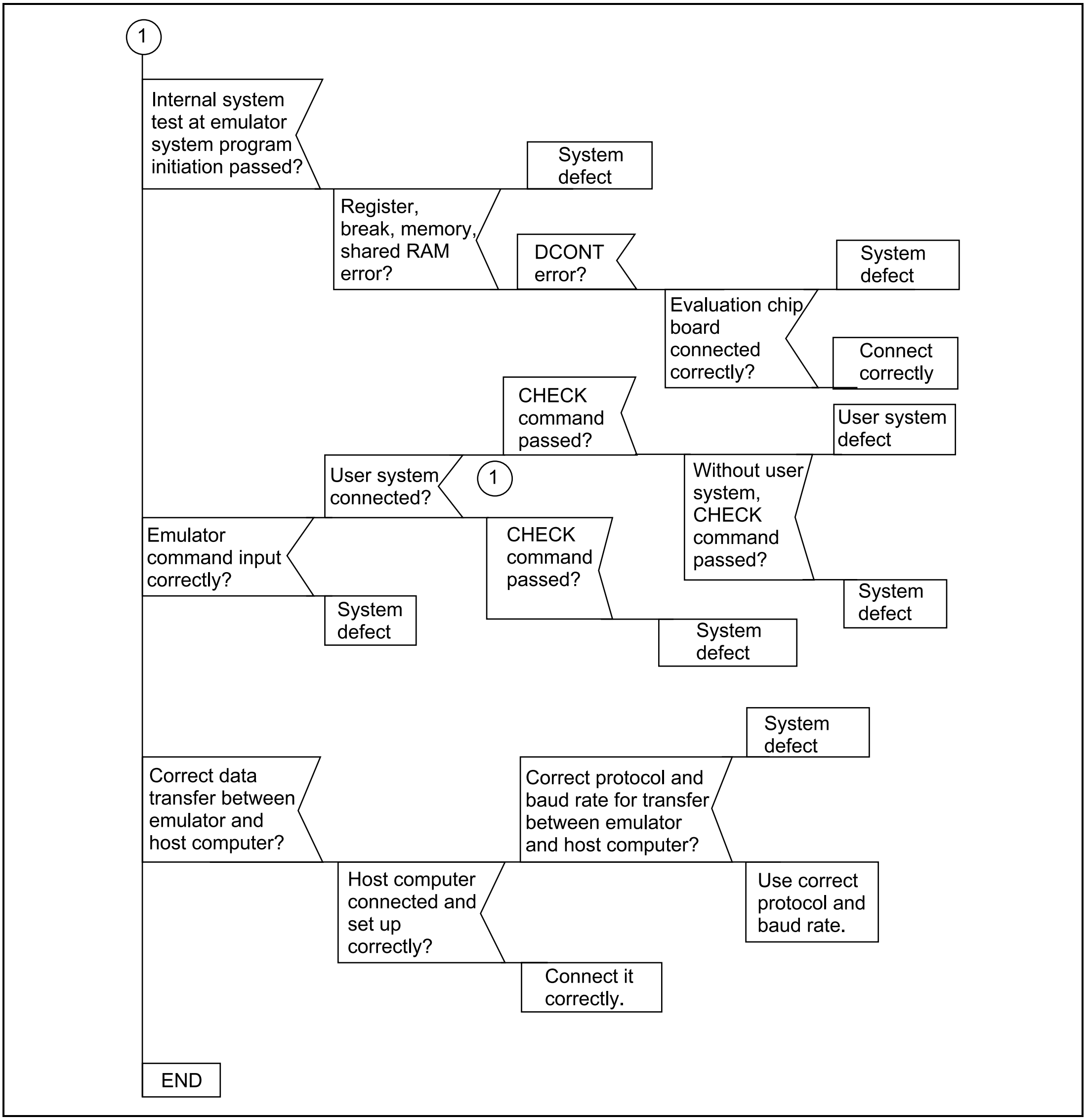

1
Internal system test at emulator system program initiation passed?
System defect
Register, break, memory, shared RAM error?
DCONT error?
System defect
Evaluation chip board connected correctly?
Connect correctly
CHECK command passed?
User system defect
User system connected?
1
Without user system, CHECK command passed?
Emulator command input correctly?
CHECK command passed?
System defect
System defect
System defect
System defect
Correct data transfer between emulator and host computer?
Correct protocol and baud rate for transfer between emulator and host computer?
Host computer connected and set up correctly?
Use correct protocol and baud rate.
Connect it correctly.
END

Figure 5.1 Troubleshooting PAD (cont)

Section 6 Command Input and Display

6.1 Command Syntax

6.1.1 Command Input Format

The emulator command format is as follows:

<command>Δ<parameters>;<option> (RET)

Δ: Space
(RET): (RET) key

Note that each command can be specified in abbreviated form to reduce keyboard operations.

6.1.2 Help Function

All emulator commands can be displayed by entering the HELP command. Any command input format can be displayed by specifying the command name as a parameter of the HELP command.

- To display all emulator commands

```
: HELP (RET)
  <All commands are displayed in their full names and abbreviations>
```

- To display a command input format

```
: HELPΔ<command name> (RET)
  <A command input format is displayed>
```

In this example, an abbreviation of the command name can be entered as <command name>.

6.1.3 Word Definition

Constants or file names can be entered as command parameters or options. Spaces (•) or commas (,) can be inserted between words. Words are described below:

Constants: Numeric constants, character constants, and expressions can be used as constants.

- Numeric constants

  The following shows numeric constant formats. A radix can be entered at the head of a numeric constant.

  S'nnnnnnnn

S:	Radix of a constant
B:	Binary
Q:	Octal
D:	Decimal
H:	Hexadecimal
X:	Fixed-point value
Default:	Value specified with the RADIX command

  nnnnnnnn: Value based on the radix (4-byte value maximum)

  Example: To indicate 100 in decimal:
  D'100

  If the radix is omitted, the radix specified with the RADIX command is automatically used.

  Example: If the radix is omitted while hexadecimal is specified with the RADIX command, entering 10 means H'10.

- Character constants

  Enclosed with single or double quotation marks. If a single or double quotation mark is used as data, add two sequential quotation marks.

  Example 1: ' A' = H'41

  Example 2: ' ' ' = H'27 (single quotation mark ')

  Multiple characters can be included inside the quotation marks within the specified data size as shown below.

  Example: ' AB' = H'4142 (2-byte data)

- Expression

  An expression can be described using numeric constants, character constants, and operators. As an operator, + (addition) or – (subtraction) can be specified.

  Examples: D'10 + H'20
  20 – 4
  –1

File Name: Some commands require a file name as a parameter. The general file name format is as follows:

<drive name>:<file name>.<extension>

6.2 Special Key Input

The emulator supports special key functions to facilitate keyboard operations. In the following description, CTRL + X means pressing the CTRL and X keys simultaneously.

6.2.1 Command Execution and Termination

• Command execution	(RET)	Enters all characters on that line, regardless of the cursor position, and executes the command.
• Command termination	CTRL + C, (BREAK)	Terminates command execution. All characters typed so far are lost and the emulator enters command input wait state.

6.2.2 Display Control

• Display stop	CTRL + S	Suspends display. Resumes display by entering CTRL and Q keys.
• Display restart	CTRL + Q	Resumes display.

6.2.3 Command Re-entry

• Display last entered line	CTRL + L	Redisplays the last line entered. Pressing these keys will repeatedly redisplay up to 16 lines and then return to the last line again.
• Display last entered command	<command name>.	When a period is entered after a command, the previously input parameters of that command are displayed. If two periods are entered after a command, parameters of two commands prior to the entered command are displayed. This key input is useful for executing commands with the same parameters again. (Example) :***D 1000 1010 (RET)*** : Execution of another command :***D. (RET)*** :***D 1000 1010*** : Displays the parameters specified in the previous DUMP command execution and enters command input wait state.

6.2.4 Display Control

• Move cursor backwards	CTRL + H	Moves the cursor one position backwards.
• Move cursor to word starting position	CTRL + T	Moves the cursor to the first position of the word (the character following the space).
• Delete one character	CTRL + D	Deletes a character at the cursor position.
• Cancel line	CTRL + X	Deletes the contents of the entire line.
• Advance cursor	CTRL + W	Moves the cursor one position forwards.
• Insert space	CTRL + U	Inserts a space at the cursor.
• Tab over	CTRL + I	Moves the cursor to the (10’s multiple + 1)th column.

Section 7 Emulation Commands

7.1 Overview

The emulator provides a wide range of functions such as break, trace, and performance analysis. Table 7.1 lists the emulation commands that enable these functions.

Table 7.1 Emulation Commands

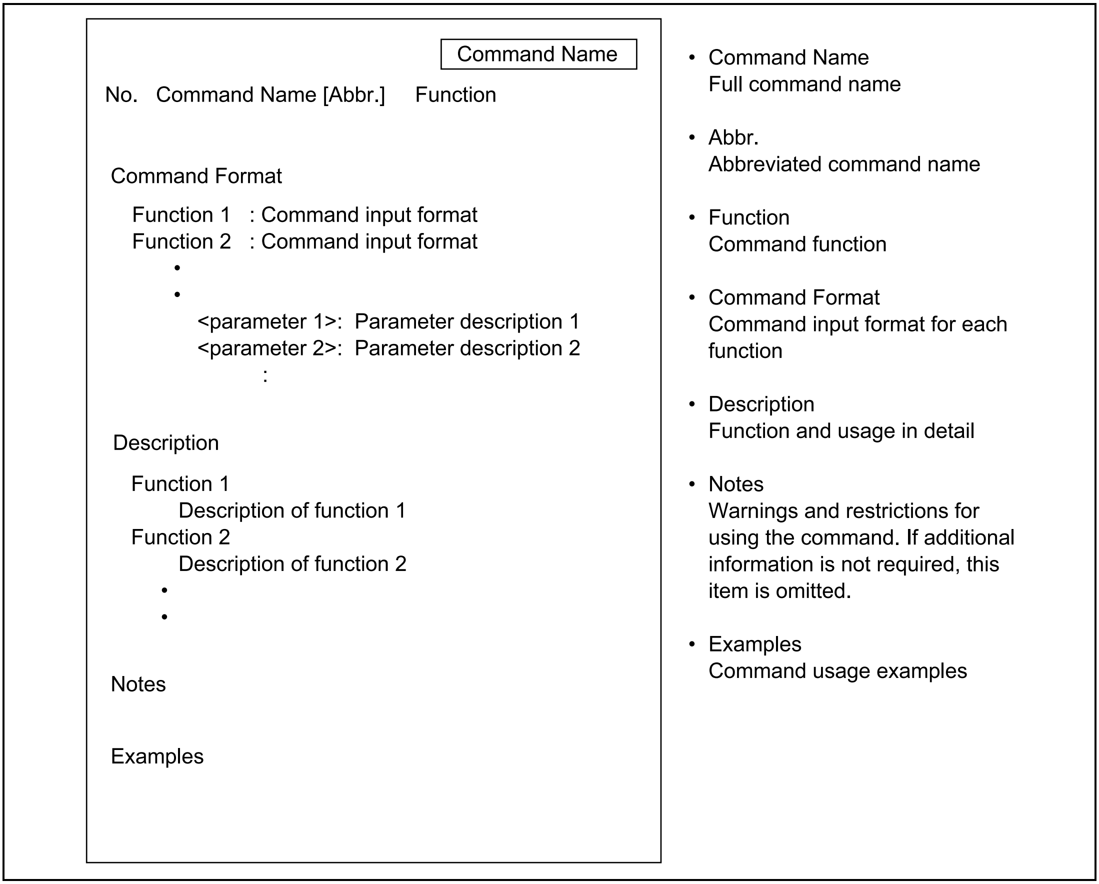
Command	Function	Usable/Unusable in Parallel Mode
.<register>	Modifies and displays register contents	Unusable
ABORT	Terminates emulation in parallel mode	Usable
ALIAS	Sets, displays, and cancels aliases	Usable
ASSEMBLE	Assembles program one line each	Unusable
BACKGROUND_ INTERRUPT	Sets and displays user interrupts in command input wait state	Unusable
BREAK	Sets, displays, and cancels software breakpoints	Only the display function is available
BREAK_CONDITION_ A,B,C	Sets, displays, and cancels hardware break conditions	Only the display function is available
BREAK_CONDITION_ UBC	Sets, displays and cancels hardware break conditions	Only the display function is available
BREAK_SEQUENCE	Sets, displays and cancels software sequential break conditions	Only the display function is available
CHECK	Tests MCU pin status	Unusable
CLOCK	Sets and displays clock	Only display function is available
CONFIGURATION	Saves and restores configuration information, and displays a list	Unusable
CONVERT	Converts data	Usable
DATA_CHANGE	Replaces memory data	Unusable
DATA_SEARCH	Searches for memory data	Unusable
DISASSEMBLE	Disassembles and displays memory contents	Usable
DUMP	Displays memory contents	Usable
END	Cancels parallel mode	Usable
EXECUTION_MODE	Sets and displays execution mode	Unusable
FILL	Writes data to memory	Unusable

Table 7.1 Emulation Commands (cont)

Command	Function	Usable/Unusable in Parallel Mode
GO	Executes realtime emulation	Unusable
HELP	Displays all commands and command format	Usable
HISTORY	Displays all input commands	Usable
ID	Displays the version number of the emulator system program	Usable
MAP	Specifies and displays memory attribute	Unusable
MEMORY	Displays and modifies memory contents	Usable
MODE	Specifies and displays the SH7055 operating mode	Unusable
MOVE	Transfers memory contents	Unusable
PERFORMANCE_ ANALYSIS	Specifies, cancels, initializes, and displays performance analysis data	Usable
QUIT	Terminates emulator system program	Unusable
RADIX	Specifies and displays radix for numeric input	Usable
REGISTER	Displays register contents	Unusable
RESET	Resets the MCU	Unusable
RESULT	Displays execution results	Unusable
STATUS	Displays emulator execution status	Usable
STEP	Performs single-step execution	Unusable
STEP_INFORMATION	Specifies and displays information during single-step execution	Unusable
STEP_OVER	Performs single-step execution except for subroutines	Unusable
TRACE	Displays trace buffer contents	Usable
TRACE_CONDITION_ A,B,C	Specifies, displays, and cancels trace acquisition conditions	Usable
TRACE_DISPLAY_ MODE	Specifies and displays trace information display mode	Usable
TRACE_MODE	Specifies and displays trace information acquisition mode	Unusable
TRACE_SEARCH	Searches for and displays trace information	Usable
VP_MAP	Sets, displays, and cancels the address conversion table for the emulator	Only the display function is available

7.2 Emulation Commands

This section provides details of emulation commands in the format shown in figure 7.1.



Command Name

No. Command Name [Abbr.] Function

Command Format

Function 1 : Command input format
Function 2 : Command input format
•
•
<parameter 1>: Parameter description 1
<parameter 2>: Parameter description 2
:

Description

Function 1
Description of function 1
Function 2
Description of function 2
•
•

Notes

Examples

- Command Name
  Full command name
- Abbr.
  Abbreviated command name
- Function
  Command function
- Command Format
  Command input format for each function
- Description
  Function and usage in detail
- Notes
  Warnings and restrictions for using the command. If additional information is not required, this item is omitted.
- Examples
  Command usage examples

Figure 7.1 Emulation Command Description Format

Symbols used in the command format have the following meanings:

[ ]: Parameters enclosed by [ ] can be omitted.
(a/b): One of the parameters enclosed by ( ) and separated by /, that is, either a or b must be specified.
< >: Contents shown in < > are to be specified or displayed.
... : The entry specified just before this symbol can be repeated.
Δ: Indicates a space. Used only for command format description.
(RET): Pressing the (RET) key.

Although italic and bold characters are used throughout this manual to indicate input, it is not used in the command format parts of these descriptions.

7.2.1 .<register> [.<register>] Modifies and displays register contents

Command Format

- Modification (direct mode) .<register>[Δ<data>] (RET)
- Modification (interactive mode) .<register> (RET)

<register>: System register, control register, general register, or FPU register to be modified or displayed.

System registers: PC, SPC, PR, MACH, MACL

Control registers: SR, SSR, GBR, VBR, RS, RE, MOD (ME, MS)

General registers: R0, R1, R2, R3, R4, R5, R6, R7, R8,
R9, R10, R11, R12, R13, R14, R15 (SP),
R0_BANK0, R1_BANK0, R2_BANK0, R3_BANK0,
R4_BANK0, R5_BANK0, R6_BANK0, R7_BANK0,
R0_BANK1, R1_BANK1, R2_BANK1, R3_BANK1,
R4_BANK1, R5_BANK1, R6_BANK1, R7_BANK1

DSP registers: DSR, A0G, A0, M0, X0, Y0, A1G, A1, M1, X1, Y1

<data>: The value to be set in the specified register

Note: The RS, RE, MOD (ME, MS), and DSP registers can be displayed or modified only when the SH7729 is selected by using the MODE command.

Description

- Modification

— Direct mode

Sets the specified value in the specified register. SP can be specified instead of R15. Also, MOD can be specified separately as MS and ME; each a 16-bit unit.

```
: .<register> <data> (RET)
```

— Interactive mode

If no data is specified on the command line with <register>, register modification is performed in interactive mode. In this case, the emulator displays the current register value and requests its modification. Registers are processed in the following order (and processing can begin at any register):

R0 to R14, R15 (SP), R0_BANK0 to R7_BANK0, R0_BANK1 to R7_BANK1, PC, SPC, SR, SSR, PR, GBR, VBR, MACH, MACL, RS, RE, MOD, A0G, A0, M0, X0, Y0, A1G, A1, M1, X1, Y1, DSR

Display format for modifying registers in interactive mode is as follows:

```
:.<register> (RET)
 <register>   =xxxxxxxx ?    yyyyyyyy (RET)
 <register>   =xxxxxxxx ?    yyyyyyyy (RET)
     .             .              .
     .             .              .
```

yyyyyyyy <data>: Inputs the value to be set in the specified register

.: Terminates the command

^: Displays the previous register

Only (RET): Does not modify the register; displays the following one

To display all register contents, use the REGISTER command.

Note

Registers are set as follows at emulator initiation:

R0 to R14	: H'00000000	R15 (SP)	: H'00000000
R0_BANK0 to R7_BANK0	: H'00000000	PC	: H'A0000000
R0_BANK1 to R7_BANK1	: H'00000000	SPC	: H'00000000
SR	: H'700000F0	SSR	: H'00000000
PR	: H'00000000	VBR	: H'00000000
GBR	: H'00000000	MACH	: H'00000000
MACL	: H'00000000		
RS, RE	: H'00000000 (when SH7729 is selected)		
MOD (MS, ME)	: H'00000000 (when SH7729 is selected)		
DSR	: H'00000000 (when SH7729 is selected)		
A0, A1	: H'00000000 (when SH7729 is selected)		
A0G, A1G	: H'00000000 (when SH7729 is selected)		
M0, M1	: H'00000000 (when SH7729 is selected)		
X0, X1	: H'00000000 (when SH7729 is selected)		
Y0, Y1	: H'00000000 (when SH7729 is selected)		

Since the initial values of the registers, except PC, SR, VBR, and DSR, in the MCU are not fixed, the initial values must be set by a program.

.<register>

If the MCU is reset by the emulator RESET or CLOCK command, registers are set as follows.

R0 to R14	: Value before reset	R15 (SP)	: Value before reset
R0_BANK0 to R7_BANK0	: Value before reset	PC	: H'A0000000
R0_BANK1 to R7_BANK1	: Value before reset	SPC	: Value before reset
SR	: H'700000F0	SSR	: Value before reset
PR	: Value before reset	VBR	: H'00000000
GBR	: Value before reset	MACH	: Value before reset
MACL	: Value before reset		
RS, RE	: Value before reset (when SH7729 is selected)		
MOD (MS, ME)	: Value before reset (when SH7729 is selected)		
DSR	: H'00000000 (when SH7729 is selected)		
A0, A1	: Value before reset (when SH7729 is selected)		
A0G, A1G	: Value before reset (when SH7729 is selected)		
M0, M1	: Value before reset (when SH7729 is selected)		
X0, X1	: Value before reset (when SH7729 is selected)		
Y0, Y1	: Value before reset (when SH7729 is selected)		

Since the reset values of the registers, except PC, SR, VBR, and DSR, in the MCU are not fixed, the initial values must be set by a program.

Examples

1. To set H'5C60 in PC, H'A00FFE00 in SP, H'FF in R1, and H'11 in R2, and then display all registers:

```
:.PC 5C60 (RET)
:.SP A00FFE00 (RET)
:.R1 FF (RET)
:.R2 11 (RET)
:R (RET)
 PC=00005C60 SR=700000F0:-P1B000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 MOD=00000000
 R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
:
```

2. To modify the contents of control registers in interactive mode:

```
:.PC (RET)
 PC    =00005C60 ? 1000 (RET)
 SR    =700000F0:-P1B000000000000--------IIII00-- ? 600000F0 (RET)
 PR    =00000000 ? . (RET)
:
```

7.2.2 ABORT [AB] Terminates emulation in parallel mode

Command Format

- Termination: ABORT (RET)

Description

- Termination

— Terminates GO command execution in parallel mode (prompt #), and cancels parallel mode.

— When GO command execution is terminated by the ABORT command in parallel mode, BREAK KEY is displayed as the termination cause.

Example

To terminate GO command emulation in parallel mode:

```
:GO RESET (RET)
 ** PC=00001022             (RET)          (To enter parallel mode)
#ABORT (RET)
 **RUNNING
 PC=00005C60  SR=700000F0:-P1B0000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000  VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 MOD=00000000
 R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000 M0=000000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
 RUN-TIME=D' 000H:00M:01S:058335US:000NS
 +++:BREAK KEY
:
```

7.2.3 ALIAS [ALI] Sets, displays, and cancels aliases

Command Format

- Setting ALIASΔ<alias name>Δ<alias definition> (RET)
- Display ALIAS (RET)
- Cancellation ALIAS[Δ]–Δ <alias name> (RET)
  ALIAS[Δ]– (RET)

<alias name>:	Alias definition name
<alias definition>:	Alias definition contents

Description

- Setting

  Sets aliases for commands. Up to 40 aliases can be set. An alias name is defined with up to 16 characters and an alias definition with up to 230 characters.

  ```
  : ALIASΔ<alias name> Δ <alias definition>  (RET)
  ```

- Display

  Displays defined aliases as follows:

  ```
  : ALIAS (RET)
    <alias name 1>:<alias definition 1>
    <alias name 2>:<alias definition 2>
    <alias name 3>:<alias definition 3>
        :          :
  ```

- Cancellation

  — Cancels the specified alias.

  ```
  : ALIAS – Δ<alias name>  (RET)
  ```

  — When no alias name is specified, cancels all aliases.

  ```
  : ALIAS –  (RET)
  ```

Note

An alias itself cannot be included in the alias definition contents.

Examples

1. To define the alias name for the command to display the contents of register FRC0H as SHOW_FRC0H:

```
:ALIAS SHOW_FRC0H D 0D000042 @1;B (RET)
:
```

2. To display all defined aliases:

```
:ALIAS (RET)
 SHOW_FRC0H: D 0D000042 @1;B
 SHOW_FRC0L: D 0D000043 @1;B
 LT: 11 test.abs
:
```

3. To cancel the alias with alias name LT:

```
:ALIAS– LT (RET)
:
```

7.2.4 ASSEMBLE [A] Assembles program one line at a time

Command Format

- Line assembly: ASSEMBLE Δ <address> [;(P/V[=<ASID value>])] (RET)

<address>: The address where the object program is to be written

P: Physical address

V: Virtual address (The address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and depends on the MMU status at command input when the VP_MAP table is disabled.

Description

- Line assembly
  - After displaying the memory contents at the specified address, the emulator enters subcommand input wait state. Line input in subcommand input wait state is assembled into machine code which is written to memory. Assembly is continued until a period (finishing subcommand) is entered. The input and output formats are as follows:

```
:ASSEMBLE <address> (RET)
     ADDR    CODE    MNEMONIC    OPERAND
   xxxxxxxx yyyy   <disassemble display>
   xxxxxxxx yyyy   ?  <subcommand> (RET)
   xxxxxxxx yyyy   ?  <subcommand> (RET)
       :           :      :
      (a)      (b)        (c)
```

(a) Address. When an odd address is specified, it is rounded down to an even address.

(b) Memory contents.

(c) Subcommand (Input the contents shown in table 7.2).

When an assembly sentence is input, a space of one character or more must be input at the beginning of the sentence.

The subcommands listed in table 7.2 can be used with the ASSEMBLE command:

Table 7.2 Subcommands for Line Assembly

Subcommand	Description
<assembly language statement>	Assembles the input line (statement) into machine code and writes it to the displayed address.
/[<address 1>[∆<address 2>]]	Disassembles instructions from <address 1> to <address 2> and displays them. If <address 2> is omitted, the first 16 instructions from <address 1> are displayed. If only a slash (/) is input, the contents from the ASSEMBLE command start address to the current address – 1 are disassembled.
(RET) only	Increments the address (odd address + 1, even address + 2), and re-enters subcommand input wait state.
^	Decrements the address (odd address – 1, even address – 2), and re-enters subcommand input wait state.
.	Terminates the ASSEMBLE command.

Note: Enter a space before an instruction.

— If an undefined label is referenced, the following message is displayed:

```
*** 33:INVALID ASM OPERAND
```

— Even if an odd address is specified, machine codes are written to memory. In that case, the following warning message is displayed:

```
*** 82:ODD ADDRESS
```

— Line assembly with this command can be performed only in areas CS0 to CS6 or the internal memory areas.

— When P is specified, the line assembly is performed at the physical address. The emulator disables the MCU MMU then accesses memory, and returns the MMU to its original state after the memory access is completed.

— When V is specified, the line assembly is performed at the virtual address. The emulator enables the MCU MMU, accesses the memory, then returns the MMU to its original state after the memory access is completed. When the ASID value is specified, line assembly is performed for the virtual address of the specified ASID. The emulator overwrites the ASID value in the MCU PTEH register to the specified value, then accesses the memory, and returns the PTEH register to its original state after the access is completed. When no ASID value is specified, the line assembly is performed at the virtual address of the ASID value in the PTEH register at command input.

- When P or V is not specified, and the VP_MAP table is enabled, the address is converted using the VP_MAP table. If the VP_MAP table is disabled, the line assembly is performed at the physical or virtual address according to the MCU MMU status at command input. The ASID value at virtual address line assembly is the ASID value of the MCU PTEH register at command input.
- If a TLB error occurs while a virtual address is accessed, the error message is displayed and the command is terminated.
- Do not perform line assembly in the monitor program area.

Examples

1. To perform line assembly from address H'1000 (the physical or virtual address and the ASID value depends on the MMU status at command input):

```
:A 1000 (RET)
   ADDR  CODE      MNEMONIC   OPERAND
00001000 0000     .DATA.W    0000
00001000 0000 ?    MOV       R0, R1  (RET)
00001002 0000 ?    ADD       R1, R2  (RET)
00001004 0000 ?    JMP       @R3     (RET)
00001006 0000 ? . (RET)
:
```

2. To perform line assembly from physical address H'1000:

```
:A 1000 ;P (RET)
```

3. To perform line assembly from virtual address H'1000 (the ASID value depends on the MMU status at command input):

```
:A 1000 ;V (RET)
```

4. To perform line assembly from virtual address H'1000 when the ASID value is H'10:

```
:A 1000 ;V=10 (RET)
```

7.2.5 BACKGROUND_INTERRUPT [BI] — Sets and displays user interrupts in command input wait state

Command Format

- Setting BACKGROUND_INTERRUPT [Δ (E[:<loop program address>]/D)] [;C] (RET)
- Display BACKGROUND_INTERRUPT (RET)

E/D: User interrupt accepting mode in command input wait state

E: Enables user interrupts in command input wait state

D: Disables user interrupts in command input wait state (default at emulator shipment)

<loop program address>: Address of the loop program for accepting user interrupts. When omitted, the last address of internal Y-RAM area – 3

C: Stores the settings as configuration information in emulator flash memory

Description

- Setting

— Enables user interrupts in command input wait state and sets the address of the loop program for accepting user interrupts. If the above settings are reset when user interrupts have already been enabled, even in the middle of the user interrupt processing, the emulator forcibly terminates the processing and then initiates the loop program for accepting user interrupts again.

```
: BACKGROUND_INTERRUPT E  (RET)
```

— Enables user interrupts in command input wait state and sets the address of the loop program for accepting user interrupts. The loop program must be stored in the RAM area. If no address is specified, the address specified before is used. After setting, the loop program execution starts.

```
: BACKGROUND_INTERRUPT  E:A5008FFC  (RET)
```

— Disables user interrupts in command input wait state.

```
: BACKGROUND_INTERRUPT D  (RET)
```

— When the C option is specified, the following message is displayed to confirm with the user whether to overwrite the existing configuration information in the emulator flash memory.

```
CONFIGURATION STORE OK (Y/N) ?      (a) (RET)
```

(a) Y: Stores the specifications as configuration information in emulator flash memory. Hereafter, when the emulator is activated, the saved specifications go into effect.

N: Does not overwrite configuration information. The existing specifications are valid.

— When user interrupts are enabled in command input wait state (E is specified), only commands usable in parallel mode and the BACKGROUND_INTERRUPT command can be executed.

- Display

— Displays user interrupt accepting mode in command input wait state and the executing address of the loop program for accepting user interrupts. If a break has occurred during user interrupt processing and the loop program has been stopped, the register values at termination and the cause of termination are displayed in the following format.

```
:BACKGROUND_INTERRUPT  (RET)
 USER INTERRUPT=x  LOOP PROGRAM ADDRESS=yyyyyyyy
 [<cause of termination>]
```

x: User interrupt accepting mode

E: User interrupts are enabled (the loop program is being executed)

D: User interrupts are disabled (the loop program has been stopped)

S: A break has occurred during user interrupt processing (the loop program has been stopped)

yyyyyyyy: Address of loop program for accepting user interrupts

<cause of termination>: Register values and the cause of termination (listed in table 7.3) at loop program termination (displayed only when S is selected above)

— Displays the register and the cause of the termination when a break occurs during a user interrupt operation. The loop program for user interrupt acceptance is terminated:

```
-PC=00005C60  SR=600000F0:-P1-000000000000--------IIII00--
-SPC=00000000 SSR=00000000 ASIC=00
-GBR=00000000  VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
–RS=00000000  RE=00000000  MOD=00000000
-R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
-R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
-R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
-R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
-R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
-R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
-DSR=000000F1:--------------------------------------GZNVCOBD
-A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
-A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
+++: <cause of termination>
```

Table 7.3 Causes of BACKGROUND_INTERRUPT Command Termination

Message	Termination Cause
ILLEGAL INSTRUCTION	An illegal instruction was executed.
RESET BY E8000	The emulator terminates program execution with the RESETP signal because an error has occurred in the user system.
LOOP PROGRAM ADDRESS IS NOT IN RAM	The executing address of the loop program for accepting user interrupts is not in the RAM area; therefore, the loop program cannot be executed.
STOPPED IN INTERRUPT PROCESS	A break occurred during the user interrupt processing.

Notes

1. In command input wait state, a BRA $ and NOP instructions (instruction code: H'AFFE0009) are set to the address of the loop program for accepting user interrupts and executed. Note the following:
   - Do not specify the address of the loop program for accepting user interrupts in the ROM area. If the specified address is in the ROM area, the loop program cannot be executed. Specify an address within the RAM area and enable user interrupts (select option E) again.
   - Specify the address of the loop program for accepting user interrupts within an area that is not used by the user program. The loop program requires a 4-byte area.
   - When the address of the loop program for accepting user interrupts is specified, the memory contents before this specification are not stored. Therefore, the contents of the loop program address is a BRA $ instruction even after user interrupts are disabled or after the loop program address is changed.
2. When one of the causes of termination listed in table 7.3 occurs during interrupt processing in command input wait state, the interrupt processing stops there. If an emulation command is executed in this state, the following message is displayed after the emulation command execution. In this case, either change the interrupt processing program and enable user interrupts, or disable user interrupts.

   ```
   *** 66: STOPPED THE BACKGROUND INTERRUPT
   ```
3. Do not use this command when using a system, such as an OS, that does not return from the user interrupt processing to the routine where the interrupt has occurred. If used, execution does not return to the loop program for accepting user interrupts even after the user interrupt processing has terminated.
4. Do not generate a reset exception when user interrupts are enabled. If generated, the user program is initiated and execution does not return to the loop program for accepting user interrupts.
5. During user interrupt processing in command input wait state, the software breakpoints (set with the BREAK or BREAK_SEQUENCE command) and hardware break conditions become invalid.
6. During user interrupt processing in command input wait state, no trace information is acquired.

Examples

1. To specify the executing address of the loop program for accepting user interrupts to H'FFFC, and begin to accept user interrupts in command input wait state:

```
:BI E:FFFC (RET)
:
```

2. To display the current user interrupt accepting mode in command input wait state:

```
:BI (RET)
 USER INTERRUPT=S LOOP PROGRAM ADDRESS=0000FFFC
–PC=00005C60  SR=600000F0:-P1-000000000000--------IIII00--
–SPC=00000000 SSR=00000000 ASID=00
–GBR=00000000  VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
–RS=00000000 RE=00000000 MOD=00000000
–R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
–R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
–R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
–R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
–R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
–R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
–DSR=000000F1:------------------------GZNVCOBD
–A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
–A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
+++:ILLEGAL INSTRUCTION
:
```

7.2.6 BREAK [B] Sets, displays, and cancels software breakpoints

Command Format

- Setting BREAKΔ<address>[Δ<number of times>][;(P/V[=ASID value>])] (RET)
- Display BREAK (RET)
- Cancellation BREAK[Δ]–[<software breakpoint to be cancelled>] [;(P/V[=ASID value>])] (RET)

<address>: Software breakpoint address

Note: When an odd address is specified, it is rounded down to an even address.

<number of times>: How many times the specified software breakpoint is to be passed (H'1 to H'FFFF) (Default: H'1)

P: Physical address

V: Virtual address (The address is converted using MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

<software breakpoint to be cancelled>: Address of the software breakpoint to be cancelled

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and depends on the MMU status at command input when the VP_MAP table is disabled.

Description

- Setting
  - Sets a software breakpoint at the specified address by replacing its contents with a break instruction (H'0000). GO command emulation terminates when the break instruction is executed. (The instruction at the software breakpoint itself is not executed.) A maximum of 255 breakpoints can be set in total. A software breakpoint can only be set in a RAM area (including standard emulation memory) because the contents of the specified address is replaced with a break instruction to cause a break. Do not set software breakpoints at any of the addresses below:
    - Address that holds an illegal instruction (H'0000)
    - The same address as the BREAK_SEQUENCE command
    - Areas other than CS0 to CS6 (excluding internal RAM area)
    - Address where the BREAK_CONDITION_UBC2 command settings are satisfied (refer to the following descriptions)

- Address containing a slot delayed branch instruction (refer to the following descriptions)
- The lower 16 bits of the 32-bit DSP instruction

Note: If a software breakpoint is set at a slot delayed branch instruction, a slot illegal instruction exception occurs instead of terminating program execution. Make sure not to set a software breakpoint at a slot delayed branch instruction.

— By specifying the number of times a breakpoint must be reached when setting the breakpoint, program execution terminates when reaching the breakpoint for the specified number of times.

Note: When multiple passes are specified for a breakpoint, the program must be temporarily stopped each time a software breakpoint is passed to update the pass count, and user program emulation continues until the number of times the breakpoint must be passed is satisfied. As a result, realtime emulation is not performed.

Example: To generate a break when the instruction at address 300 is executed five times

: ***BREAK 300 5 (RET)***

- Display

  Display format is as follows:

```
: BREAK   (RET)
<ADDR>   <ASID>    <CNT>    <PASS>
xxxxxxxx  *nnmm     yyyy      zzzz
  (a)       (b)      (c)       (d)
```

(a) Address to which a breakpoint is set

(b) Physical or virtual address specification, and the ASID value.

P: Physical address specification

V:mm: Virtual address specification (mm is the ASID value)

* A breakpoint is valid during program execution by the GO command.

* is not displayed when a break instruction could not be embedded.

(c) Specified pass count (hexadecimal)

Note: The pass counter is cleared by the next GO command.

(d) Value of pass counter (shows how many times the specified address has been passed at GO command termination, in hexadecimal)

- Cancellation

  Cancels software breakpoints. Breakpoints can be cancelled in the following two ways:

  — Cancellation of specified software breakpoints. A breakpoint can be cancelled with one command execution.

  : ***BREAK–<software breakpoint> (RET)***

  If P, V, or ASID value is omitted when canceling a breakpoint, all breakpoints that the address matches are cleared.

  — Cancellation of all software breakpoints.

  : ***BREAK– (RET)***

Notes

1. In parallel mode, if a command that accesses memory is executed and the emulation stops at a pass point or the reset point at the same time, command execution may not take place. In this case,

   ```
   *** 78: EMULATOR BUSY
   ```

   is displayed. Re-enter the command. If the termination interval is short, the emulator may not enter parallel mode or commands cannot be executed in parallel mode.
2. Even when the memory contents are modified such as by the LOAD command, breakpoints are not cancelled.
3. During STEP or STEP_OVER command execution, a software breakpoint is invalidated, and the pass counter is not updated.
4. If a software breakpoint is set only at either the cache area or the through area, the break occurs only at the specified area.
5. Do not set a software breakpoint at the instruction in which BREAK_CONDITION_UBC2 is satisfied. When an instruction set by this command is executed, BREAK_CONDITION_UBC2 is invalidated.
6. Do not set a software breakpoint at a slot instruction of the delay branch instruction. If it is set and the user program is executed, the execution breaks, but the PC value at break is incorrect.
7. When P is specified, the software breakpoint is set at a physical address. At GO command program execution, the emulator disables the MCU MMU, sets the breakpoint, then returns the MMU to its original state.

8. When V is specified, the software breakpoint is set at a virtual address. At GO command program execution, the emulator enables the MCU MMU, sets the breakpoint, then returns the MMU to its original state. When the ASID value is specified, the software breakpoint is set at a virtual address according to the specified ASID value. The emulator overwrites the ASID value with the specified value, sets a breakpoint, then returns the ASID value to its original state. When no ASID value is specified, the software breakpoint is set at a virtual address according to the ASID value at command input.
9. When P or V is not specified and the VP_MAP table is enabled, the address is converted according to the VP_MAP table. If it is disabled, the software breakpoint is set at a physical or virtual address according to the MMU status at command input. The ASID value when a software breakpoint is set at a virtual address depends on the ASID value of the MCU PTEH register at command input.
10. When a software breakpoint is set to the cache area, address contents of the breakpoint are written to the cache area immediately before and after the user program execution.
11. When the break instruction cannot be resumed after a user program execution is terminated, a break instruction is left in the user program, and the breakpoint is removed. When the break instruction cannot be resumed due to a TLB error or a TLB miss, the break instruction is left in the user program, but the breakpoint is not removed.
12. Do not set a breakpoint in the monitor program area.

Examples

1. To set a software breakpoint at address H'1000:

```
:B 1000 (RET)
:
```

2. To generate a break when virtual address H'6004 with ASID value H'10 has been passed three times:

```
:B 6004 3;V=10 (RET)
:
```

3. To display set software breakpoints:

```
:B (RET)
 <ADDR>    <ASID>  <CNT>  <PASS>
 00000100  *P       0001   0000
 00006004  *V:10    0003   0000
:
```

4. To cancel the software breakpoint at a physical address H'1000:

```
:B – 1000;P (RET)
:
```

5. To cancel all software breakpoints:

```
:B – (RET)
:
```

7.2.7 BREAK_CONDITION_A,B,C [BCA, BCB, BCC] Specifies, displays, and cancels a hardware break condition

Command Format

- Setting BREAK_CONDITION_(A/B/C)(1/2/3/4/5/6/7/8)Δ<condition>
  [[Δ<condition>] [Δ<condition>]...] (RET)
- Display BREAK_CONDITION_(A/B/C)[(1/2/3/4/5/6/7/8)] (RET)
- Cancellation BREAK_CONDITION_(A/B/C)[(1/2/3/4/5/6/7/8)] [Δ] – (RET)

(A/B/C):	Break type
(1/2/3/4/5/6/7/8):	Break number When omitted, all conditions will be displayed or cancelled.
<condition>:	Hardware break condition (refer to table 7.4 for details)

Description

- Setting
  - — Specifies hardware break conditions (BREAK_CONDITION_A,B,C). Program execution stops when the specified conditions are satisfied. The specifiable conditions for the three types of hardware breaks (BREAK_CONDITION_A,B,C) are summarized in table 7.4.

Table 7.4 Specifiable Conditions (BREAK_CONDITION_A1-A8)

Item and Input Format	Description	Commands that can be Set
Address condition A=<address 1>[:<address 2>] [;NOT][∆P]	• When only <address 1> is specified, the condition is satisfied when the address bus value matches the specified value. • When both <address 1> and <address 2> are specified, the condition is satisfied when the address bus value is in the range from <address 1> to <address 2>. • Condition is satisfied when the address bus value provided by the NOT option setting is not equal to the specified value. Only BCB can be set. • When ;P is specified, the specified <address 1> and <address 2> becomes the physical address condition. When ;P is omitted and the VP_MAP table is enabled, the address value converted by the address set with the VP_MAP command becomes the condition, and when VP_MAP table is disabled, the physical address becomes the condition. • Bits can be masked for the address.	BCA BCB BCC
Data condition <LD>=<4-byte value> [;NOT]	• The condition is satisfied when the data bus value matches the specified value. The access size cannot be specified. A 4-byte value must always be specified. To break with a 1-byte or 2-byte values, mask the unnecessary bits. • Condition is satisfied when the data bus value provided by the NOT option setting is not equal to the specified value. Only BCB can be set. • Bits can be masked for the data.	BCA BCB
Read/Write condition R: Read W: Write	The condition is satisfied in a read cycle (R is specified) or a write cycle (W is specified).	BCA BCB

Table 7.4 Specifiable Conditions (BREAK_CONDITION_A1-A8) (cont)

<table>
<tr><th>Item and Input Format</th><th>Description</th><th>Commands that can be Set</th></tr>
<tr><td>External probe condition
PRB=<value></td><td>The condition is satisfied when all of the emulator's external probe signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to a probe number, as follows:
3 2 1 0 ← Bit
x x x x ← Specified value
| | | |
4 3 2 1 ← Probe number
x: 0 = Low level
1 = High level
The bits can be masked for the value.</td><td>BCA
BCB</td></tr>
<tr><td>External interrupt condition 1
NMI [:L] or NMI: H</td><td>The condition is satisfied when the NMI signal matches the specified level.
NMI or NMI: L: The condition is satisfied when NMI is low
NMI: H: The condition is satisfied when NMI is high</td><td>BCA
BCB</td></tr>
<tr><td>External interrupt condition 2
IRL=<value></td><td>The condition is satisfied when all of the IRL0-3, IRQ4, and IRQ5 signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to an IRL and IRQ number, as follows:
5 4 3 2 1 0 Bit
x x x x x x Specified value
| | | | | |
5 4 3 2 1 0 IRL, IRQ number
x: 0 = Low level
1 = High level
The bits can be masked for the value.</td><td>BCA
BCB</td></tr>
<tr><td>Satisfaction count specification
COUNT = <value>
<value>: H'1 to H'FFFF</td><td>This is set in combination with any of the address, data, read/write, external probe, and external interrupts 1 and 2 conditions. The complete condition combination is satisfied when the specified condition is satisfied the specified number of times.</td><td>BCB</td></tr>
</table>

Table 7.4 Specifiable Conditions (BREAK_CONDITION_A1-A8) (cont)

Item and Input Format	Description	Commands that can be Set
Delay count specification DELAY = <value> <value>: H'1 to H'7FFF	This is set in combination with any of the address, data, read/write, external probe, external interrupts 1 and 2, and count conditions. When the condition is satisfied, the complete condition combination is satisfied after the bus cycle of the specified value is executed. Only BCB7 can be specified.	BCB

— A bit mask in 1-bit or 4-bit units can be specified for the address condition of the BREAK_CONDITION_A,B,C command. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). When <address 2> is not specified for an address condition, <address 1> can be consecutively masked from the lowest bit. It is not possible to mask any desired bit position. Table 7.5 shows address mask specification examples.

Example: The following condition is satisfied when the lower four bits of the address condition are not specified:

```
: BREAK_CONDITION_A1   A=H'400000*  (RET)
```

Table 7.5 Address Mask Specifications (BREAK_CONDITION_A,B,C)

Radix	Mask Unit	Example	Mask Position
Binary	1 bit	B'01110***	Bits 2 to 0 are masked
Hexadecimal	4 bits	H'000F50**	Bits 7 to 0 are masked

Note: When <address 2> is not specified for an address condition, <address 1> can be consecutively masked from the lowest bit. It is not possible to mask any desired bit position, as shown in the following examples.

Examples:

Allowed: BREAK_CONDITION_A1 A = H'10**

Not allowed: BREAK_CONDITION_A1 A = H'1*00

BREAK_CONDITION_A1 A = H'100* :10**

— A bit mask in 1-bit or 4-bit units can be specified for the data, PRB, or IRL condition of the BREAK_CONDITION_A,B,C command. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). Table 7.6 shows these mask specification examples.

Example: The following condition is satisfied when address H’3000000 is the address condition and bit 0 is zero in the byte data condition:

```
: BREAK_CONDITION_A1   A=H'3000000 D=B'*******0 (RET)
```

Table 7.6 Mask Specifications (BREAK_CONDITION_A,B,C)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	Bits 0 and 5 are masked	Data or PRB
Hexa-decimal	4 bits	H'F**50	Bits 15 to 8 are masked	Data or PRB

— If a hardware break condition is satisfied, emulation may stop after two or more instructions have been executed.

— Do not set the address condition in the monitor program area.

- Display

— Displays specified conditions. The character string that was input for specifying conditions will be displayed as it was input. If the break number is omitted, all specified break conditions for that break type are displayed.

— If no break condition is specified, a blank is displayed.

```
: BREAK_CONDITION_B (RET)
 BCB1 <B1 break setting>
 BCB2 <B2 break setting>
 BCB3 <B3 break setting>
 BCB4 <B4 break setting>
 BCB5 <B5 break setting>
 BCB6 <B6 break setting>
 BCB7 <B7 break setting>
 BCB8 <B8 break setting>
```

- Cancellation

  Cancels specified conditions. When break numbers 1 to 8 are omitted, all break conditions are cancelled.

  — Cancels all conditions for the BREAK_CONDITION_A command.

  ```
  :BREAK_CONDITION_A – (RET)
  ```

  — Cancels BREAK_CONDITION_A1 conditions.

  ```
  :BREAK_CONDITION_A1 – (RET)
  ```

Notes

1. When conditions have already been set with the TRACE_CONDITION_A,B command, the same command number cannot be set. For example, when a condition has been set with the TRACE_CONDITION_A1 command, the condition cannot be set with the BREAK_CONDITION_A1 command. If necessary, cancel conditions set with the above commands before setting the break conditions.
2. When conditions have already been set with the TRACE_CONDITION_C or PERFORMANCE_ANALYSIS command, the same command number cannot be set. For example, when a condition has been set with the TRACE_CONDITION_C1 or PERFORMANCE_ANALYSIS1 command, the condition cannot be set with the BREAK_CONDITION_C1 command. If necessary, cancel conditions set with the above commands before setting the break conditions.
3. The address condition must be set in the external area. If it is set in internal areas such as the internal I/O area, X-RAM area, or Y-RAM area, a break will not occur.

Examples

1. To generate a break when byte data H'10 is accessed at address H'F000000:

   ```
   :BCA1 A=F000000 LD=10****** (RET)
   :
   ```

2. To generate a break when data is written to address H'1000000:

   ```
   :BCA2 A=1000000 W (RET)
   :
   ```

3. To generate a break when reading data in address H'2000000:

```
:BCB1 A=2000000 R  (RET)
:
```

4. To display the specified conditions:

```
:BCA (RET)
 BCA1 A=F000000 LD=10******
 BCA2 A=1000000 W
 BCA3
 BCA4
 BCA5
 BCA6
 BCA7
 BCA8
:
```

5. To cancel the specified conditions:

```
:BCA1 – (RET)
:BCB1 – (RET)
:
```

7.2.8 BREAK_CONDITION_UBC [BCU] — Specifies, displays, and cancels hardware break conditions

Command Format

- Setting BREAK_CONDITION_UBC(1/2/3)Δ<condition>[[Δ <condition>] [Δ<condition>]...] (RET)
- Display BREAK_CONDITION_UBC[(1/2/3)] (RET)
- Cancellation BREAK_CONDITION_UBC[(1/2/3)] [Δ] – (RET)

(1/2/3):	UBC break number When omitted, all conditions will be displayed or cancelled.
<condition>:	Hardware break condition (refer to table 7.7 for details)

Description

- Setting
  - — Set the hardware break (BREAK_CONDITION_UBC break) condition. If the specified condition is satisfied, program execution is stopped. In BREAK_CONDITION_UBC break, each condition shown in Table 7.7 can be set. Also, BREAK_CONDITION_UBC break can be used in up to two levels of sequential break conditions. Sequential break is specified using the GO command.

Table 7.7 Specifiable Conditions

Item and Input Format	Description	Specifiable Command
Address condition A=<address>	• The condition is satisfied when the address bus value matches the specified value. • The bits can be masked for the address.	BCU1 and BCU2

Address condition*
XA=<X-bus address>
YA=<Y-bus address>
	• The condition is satisfied when the address matches the specified value at the X-bus or Y-bus access according to the DSP instruction. The address must be specified in word size. When XD or YD is specified as the data condition, only XA or YA can be specified, respectively. This condition cannot be specified with the access type conditions.
	• The bits can be masked for the address.
		BCU1

PC condition PC=<address>[;P]	• The condition is satisfied when the program fetch address bus matches the specified value. When ;P is specified, the execution breaks before the instruction of the specified value. This condition can be specified with the ASID condition or the COUNT condition. • The bits can be masked for the PC.	BCU1 and BCU2
ASID condition ASID=<ASID value>	The condition is satisfied when the ASID value matches the specified value.	BCU1 and BCU2

Data condition
D=<1-byte value>
WD=<2-byte value>
LD=<4-byte value>
	• The condition is satisfied when the data bus value matches the specified value. When D, WD, or LD is specified, the break condition is satisfied when the address is accessed in bytes, words, or longwords, respectively.
	In program fetch cycles, the data condition is not satisfied irrespective of the data bus value.
	• The bits can be masked for the data.
		BCU1

Table 7.7 Specifiable Conditions (cont)

Item and Input Format	Description	Specifiable Command

Data condition*
XD=<X-bus data value>
YD=<Y-bus data value>
	• The condition is satisfied when the data bus at X bus or Y bus access according to the DSP instruction matches the specified value. Data must be specified in word size. When XA or YA is specified as the address condition, only XD or YD can be specified, respectively.
	• The bits can be masked for the data.
		BCU1

Read/Write condition R: Read W: Write	The condition is satisfied in a read cycle (R is specified) or a write cycle (W is specified).	BCU1 and BCU2
Access type DAT: Execution cycle DMA: DMAC cycle	The condition is satisfied when the bus-cycle type matches the specified type. Default: All bus cycles (including the program fetch cycle). The DMA condition can be specified only with BCU1.	BCU1 and BCU2
Satisfaction count specification COUNT=<value> <value>: H'1 to H'FFFF	This condition can be specified in combination with any of the address, PC, ASID, data, read/write, and access type conditions. The complete condition combination is satisfied when the specified condition has been satisfied for the specified number of times.	BCU1
I/O access condition IO	The condition is satisfied when the internal I/O area is accessed.	BCU3
LDTLB condition LDTLB	The condition is satisfied when the LDTLB instruction is executed.	BCU3

Note: Can be specified only when the SH7729 is selected as the MCU by the MODE command.

— The data conditions of the BREAK_CONDITION_UBC1 break are satisfied when the address bus and data bus values match the specified values. The data bus (the MCU internal bus) is always 32 bits long. Note the following when specifying break conditions.

- Longword access

  Longword data is accessed in one bus cycle. Only longword data (LD) and a multiple of four can be specified as the data and address conditions, respectively.

- Word access

  Word data is accessed in one bus cycle. Only word data (WD) and a multiple of two can be specified as the data and address conditions, respectively.

- Byte access

  Byte data is accessed in one bus cycle. Only byte data (D) can be specified as the data condition. Both even and odd address values can be specified as the address condition.

— A bit mask in 1-bit or 4-bit units can be specified for the address, PC, and data conditions of the BREAK_CONDITION_UBC command. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). Table 7.8 shows mask specification examples.

Example 1: The following condition is satisfied when the lower four bits of the address condition are not specified:

```
: BREAK_CONDITION_UBC1   A=H'400000*  (RET)
```

Example 2: The following condition is satisfied when address 3000000 is the address condition and bit 0 is zero in the byte data condition:

```
: BREAK_CONDITION_UBC1   A=H'3000000   D=B'*******0 (RET)
```

Table 7.8 Mask Specifications (BREAK_CONDITION_UBC1,2)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	Bits 0 and 5 are masked	Address, data, or PC
Hexa-decimal	4 bits	H'F**50	Bits 15 to 8 are masked	Address, data, or PC

- Display

  — Displays specified conditions. The character string that was input for specifying conditions will be displayed as it was input. If break numbers are omitted, break conditions for all break types are displayed.

  — If no break condition is specified, a blank is displayed.

```
: BREAK_CONDITION_UBC (RET)
  BCU1 <UBC1 break setting>
  BCU2 <UBC2 break setting>
  BCU3 <UBC3 break setting>
```

- Cancellation

  Cancels specified conditions. When break numbers 1 and 2 are omitted, all break conditions are cancelled.

  — Cancellation of all break conditions

```
: BREAK_CONDITION_UBC – (RET)
```

  — Cancellation of BREAK_CONDITION_UBC2 break conditions

```
: BREAK_CONDITION_UBC2 – (RET)
```

Notes

1. The BREAK_CONDITION_UBC2 settings are ignored when a stop address is specified with the GO command or during STEP and STEP_OVER command execution.
2. Executing addresses containing software breakpoints (set by the BREAK or BREAK_SEQUENCE command) invalidates the BREAK_CONDITION_UBC2 settings. Make sure not to set software breakpoints at addresses where the BREAK_CONDITION_UBC2 settings are satisfied.
3. When a break with PC value specification with the P option (break before execution)is set at a delayed slot branchinstruction, the execution is not terminated before the delayed slot branch instruction execution; it is terminated before branch destination insturction execution.
4. When an address is specified, all bits of the address are checked. Therefore, a break does not occur when the shadow area having the same memory area as the set address is accessed.
5. The user break controller cannot be used when using the emulator since BREAK_CONDITION_UBC1,2,3 is realized by using the MCU user break controller.
6. If reset occurs during user program execution, before the satisfaction of the BREAK_CONDITION_UBC1,2,3 condition, a break does not occur since the user break controller is initialized.

Examples

1. To generate a break when byte data H'10 is accessed at address H'F000000:

```
:BCU1 A=F000000 D=10 (RET)
:
```

2. To generate a break when data is written to address H'1000000:

```
:BCU2 A=1000000 W DAT (RET)
:
```

3. To display the specified conditions:

```
:BCU (RET)
 BCU1 A=F000000 D=10
 BCU2 A=1000000 W DAT
 BCU3
```

4. To cancel the specified conditions:

```
:BCU1 – (RET)
:BCU2 – (RET)
:
```

7.2.9 BREAK_SEQUENCE [BS] Sets, displays, and cancels software breakpoints with pass sequence specification

Command Format

- Setting BREAK_SEQUENCEΔ<pass point>Δ<pass point>[Δ<pass point>
  [Δ<pass point>[Δ<pass point>[Δ<pass point>
  [Δ<pass point>]]]][;(P/V=<ASID value>])] (RET)
  (Pass point setting)
  BREAK_SEQUENCEΔ<reset point>;R [Δ (P/V=<ASID value>])](RET)
  (Reset point setting)
- Display BREAK_SEQUENCE (RET)
- Cancellation BREAK_SEQUENCE[Δ]– (RET) (Pass point cancellation)
  BREAK_SEQUENCE[Δ]–;R (RET) (Reset point cancellation)

<pass point>:	Addresses (two to seven points)
R:	Reset point specification
<reset point>:	Address (one point)
P:	Physical address
V:	Virtual address
ASID value:	ASID value

Note: When an odd address is specified, it is rounded down to an even address.

Description

- Setting

— Sets pass points to enable the break for which the pass sequence is specified (sequential break). GO command emulation terminates when these pass points have been passed in the specified sequence.

Notes: 1. Do not set a pass point or a reset point at any of the addresses below:

- Address specified with the BREAK command
- Address that holds an illegal instruction (H'0000)
- Areas other than CS0 to CS6 (excluding internal RAM area)
- Address where BREAK_CONDITION_UBC2 settings are satisfied (refer to the following description)
- Address containing a slot delayed branch instruction (refer to the following description)
- The lower 16 bits of the 32-bit DSP instruction address

2. Pass points or a reset point are ignored during STEP and STEP_OVER command execution. Therefore, the pass count is not updated during STEP and STEP_OVER command execution.

— If the pass points have not been passed in the specified sequence, break checking begins again from the first pass point.

— When the specified reset point is passed, break checking begins again at the first pass point, even if the remaining pass points are then passed in the assigned sequence.

— When pass points or a reset point are specified, the emulator temporarily stops emulation and analyzes the pass sequence at each point. Therefore, realtime emulation is not performed.

— When execution starts at the address set with the BREAK_SEQUENCE command, immediately after execution starts, the BREAK_CONDITION_UBC2 command settings are invalidated. Therefore, even though a BREAK_CONDITION_UBC2 command setting is satisfied immediately after execution start, GO command execution does not terminate.

— If a pass point is set at a slot delayed branch instruction, instead of terminating program execution, a slot illegal instruction interrupt occurs. Make sure not to set a pass point at a slot delayed branch instruction.

— Do not set a breakpoint in the monitor program area.

- Display

  Displays specified pass points and reset point as follows:

```
: BREAK_SEQUENCE  (RET)
 PASS POINT NO.1   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.2   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.3   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.4   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.5   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.6   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.7   = xxxxxxxx    yyyy    *zzzz
 PASS POINT NO.8   = xxxxxxxx    yyyy    *zzzz
 RESET POINT       = xxxxxxxx    yyyy    *zzzz
                      (a)         (b)    (c) (d)
```

(a) Address (If nothing is specified, a blank is displayed.)

(b) Number of times passed (The number of times the pass point was passed is displayed in hexadecimal. If it exceeds H'3FFF, counting restarts from H'0. The number of times passed is cleared by the next GO command.)

(c) * indicates that the breakpoint is valid during the GO command user program execution. If a break instruction could not be embedded, * is not displayed.

(d) Physical or virtual specification, and the ASID value.

P: Physical address

Vnn: Virtual address (nn: ASID value)

- Cancellation

  Cancels specified pass points or a reset point.

— Cancellation of pass points

```
: BREAK_SEQUENCE– (RET)
```

— Cancellation of a reset point

```
: BREAK_SEQUENCE-;R (RET)
```

Note

In parallel mode, if a command that accesses memory is executed and the emulation stops at a pass point or the reset point at the same time, command execution may not take place. In this case,

*** 78: EMULATOR BUSY

is displayed. Re-enter the command. If the termination period is short, the emulator may not enter parallel mode or commands cannot be executed in parallel mode.

Examples

1. To set pass points at addresses H'4000, H'4100, H'4200, and H'4300 in that order and a reset point at address H'2000:

```
:BS 4000 4100 4200 4300  (RET)
:BS 2000 ;R  (RET)
:
```

2. To display the specified pass points and reset point:

```
:BS  (RET)
 PASS POINT NO1 = 00004000   0000  *P
 PASS POINT NO2 = 00004100   0000  *P
 PASS POINT NO3 = 00004200   0000  *P
 PASS POINT NO4 = 00004300   0000  *P
 PASS POINT NO5 = 00005000   0000  *P
 PASS POINT NO6 = 00005100   0000  *P
 PASS POINT NO7 = 00005200   0000  *P
 PASS POINT NO8 = 00005300   0000  *P
 RESET POINT    = 00002000   0000  *P
:
```

3. To cancel the reset point:

```
:BS – ;R  (RET)
:
```

4. To cancel the pass points and reset point:

```
:BS –  (RET)
:BS – ;R  (RET)
:
```

7.2.10 CHECK [CH] Tests MCU pins

Command Format

- Test CHECK (RET)

Description

- Test

  Tests the status of the MCU pins shown in table 7.9.

Table 7.9 MCU Pin Test

Pin Name	Error Status
RES	RESETP signal is fixed low.
NMI	NMI signal is fixed low.
WAIT	WAIT signal is fixed low.
BREQ	BREQ signal is fixed low.
IRL0	IRL0 signal is fixed low.
IRL1	IRL1 signal is fixed low.
IRL2	IRL2 signal is fixed low.
IRL3	IRL3 signal is fixed low.
IRQ4	IRQ4 signal is fixed low.
IRQ5	IRQ5 signal is fixed low.

If an error occurs,

FAILED AT <pin name>

is displayed.

Example

When the IRL0 signal is low:

```
:CH  (RET)
 FAILED AT IRL0
:
```

7.2.11 CLOCK [CL] Sets or displays clock

Command Format

- Setting CLOCKΔ<clock> (RET)
- Display CLOCK (RET)

<clock>:	One of the following clock signals:	
	8:	Uses emulator internal clock signal (8.25 MHz)
	33:	Uses emulator internal clock signal (33 MHz)
	U:	Uses user system CLOCK signal
	X:	Uses crystal oscillator on the evaluation chip board

Description

- Setting

— Selects emulator clock signals from the user system or from the emulator internal clock (installed in the emulator). Resets the MCU when a clock is selected, and consequently, internal I/O registers and control registers return to their reset values.

Note: When CKIO is selected as the CLOCK signal to be used by the MCU at emulator initiation, the clock cannot be set.

— Displays the specified clock signal. If the user system clock (U) is specified, but the user system clock signal is not input, an error occurs and the emulator internal clock is set instead. At emulator initiation, the user system clock (U), crystal oscillator on the evaluation chip board (X), and emulator internal clock are selected in that order, and the correct clock signal is set. When the emulator internal clock is selected, the CLOCK signal (8) is selected in clock mode 1, and the CLOCK signal (33) is selected in clock modes 0 and 3.

- Display

  Displays the current clock signal.

```
: CLOCK  (RET)
  CLOCK = <used clock>
```

<used clock>:	8.25MHz:	Emulator internal CLOCK (8.25 MHz)
	33MHz:	Emulator internal CLOCK (33 MHz)
	USER:	User system clock
	X'TAL:	Crystal oscillator clock on the evaluation chip board
	CKIO:	CKIO pin input (Displayed when the emulator is initiated at clock mode 7.)

Note

If U (user system clock) is specified and the following clock signal problem occurs, the E8000 system program may terminate. In this case,

```
*** 6: USER SYSTEM NOT READY
```

is displayed. The E8000 system program must be terminated with the QUIT command and restarted.

- User system clock signal is not being received even when U is specified and the user system clock is being used. (Vcc is supplied with no problem)

Examples

1. To use the user system clock signal:

```
:CL U (RET)
 ** RESET BY E8000 !
 CLOCK = USER
:
```

2. To use the emulator internal clock signal:

```
:CL 33 (RET)
 ** RESET BY E8000 !
 CLOCK = 33MHz
:
```

3. To display the current clock signal:

```
:CL (RET)
 CLOCK = 33MHz
:
```

7.2.12 CONFIGURATION [CNF] Saves and restores configuration information, and displays a list

Command Format

- Saving CONFIGURATIONΔ<configuration number>Δ<comment> ;S (RET)
- Restoration CONFIGURATIONΔ<configuration number> (RET)
- List display CONFIGURATION (RET)

<configuration number>:	1 or 2
<comment>:	Comment on the defined configuration information. A comment can contain one to 32 characters (not counting the semicolon (;)).

Description

- Saving

  Saves configuration information (various emulation information) that are listed in table 7.10 in the emulator flash memory.

```
: CONFIGURATION <configuration number> <comment> ;S  (RET)
```

Table 7.10 Saved Configuration Information

Item	Description
Software breakpoints	Information set by the BREAK and BREAK_SEQUENCE commands
Hardware break conditions	Information set by the BREAK_CONDITION_A,B,C and BREAK_CONDITION_UBC commands
Trace conditions	Information set by the TRACE_CONDITION_A,B,C, TRACE_DISPLAY_MODE, and TRACE_MODE commands
Performance analysis condition	Information set by the PERFORMANCE_ANALYSIS command
Memory map	Information set by the MAP command
Emulation operating mode	Information set by the EXECUTION_MODE command
Aliases	Information set by the ALIAS command
Background interrupt condition	Information set by the BACKGROUND_INTERRUPT command

- Restoration

  Restores the configuration information saved in the emulator flash memory.

  ```
          :CONFIGURATION <configuration number> (RET)
  ```

- List display

  Displays the configuration information saved in the emulator flash memory.

  ```
          :CONFIGURATION  (RET)
           1 <comment>
           2 <comment>
          :
  ```

Examples

1. To save configuration information with comment CNF1:

   ```
   :CNF 1 CNF1 ;S (RET)
   :
   ```

2. To restore configuration information saved under configuration number 1:

   ```
   :CNF 1 (RET)
   :
   ```

3. To display the configuration information list:

   ```
   :CNF (RET)
    1 CNF1
    2 ETC
   :
   ```

7.2.13 CONVERT [CV] Converts data

Command Format

- Conversion CONVERTΔ<data> (RET)
  CONVERTΔ<expression> (RET)

<data>: Data to be converted
<expression>: Addition or subtraction
<data>+<data>–<data> ...
–<data>

Description

- Conversion

— Converts data to hexadecimal, decimal, octal, binary, ASCII characters, and fixed-point values. Input data is handled as 4-byte values. If there is no corresponding ASCII character (including undisplayable character), a period (.) is displayed instead.

```
: CONVERT <data> (RET)
  H'xxx...   D'xxx...  Q'xxx...  B'xxx...  xxxx
    (a)        (b)       (c)       (d)      (e)
  X'x.xxx...
    (f)
```

(a) Hexadecimal display
(b) Decimal display
(c) Octal display
(d) Binary display
(e) ASCII display
(f) Fixed-point value display

— If the H', D', Q', B', and X' radix is not specified for <data> at data input, the radix specified with the RADIX command is assumed.

Note

When an expression is entered with a fixed-point value, the conversion result is not correct, since the emulator first converts the fixed-point value as a four-byte value, then converts the expression.

Examples

1. To convert hexadecimal data (H'7F):

```
:CV H'7F (RET)
 H'7F  D'127  Q'177  B'1111111   ....
 X'0.0000000591
:
```

2. To convert the expression:

```
:CV H'31+D'16 (RET)
 H'41  D'65  Q'101  B'1000001   ...A
 X'0.0000000303
:
```

7.2.14 DATA_CHANGE [DC] Replaces memory data

Command Format

- Replacement DATA_CHANGEΔ<data 1>Δ<data 2>Δ<start address>
  (Δ<end address>/Δ@<number of bytes>)
  [;[<size>][Δ(P/V[=<ASID value>])] [ΔY]] (RET)

<data 1>: Old data
<data 2>: New data
<start address>: Start address of the memory area to be changed
<end address>: End address of the memory area to be changed
<number of bytes>: The number of bytes in the memory area to be changed
<size>: Length of data
B: 1 byte
W: 2 bytes
L: 4 bytes
Default: 1 byte
P: Physical address
V: Virtual address (address is converted using the TLB at command input)
<ASID value>: the ASID value (can be specified only when V is specified)
Y: Specify Y if a confirmation message is not necessary. If Y is specified, data in all assigned areas is replaced without a confirmation message.

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Replacement

— Replaces <data 1> in the specified memory area (set by the <start address> and <end address> or the <number of bytes>) with <data 2> and verifies the results.

— If option Y is specified, data is replaced without confirmation messages. If option Y is not specified, the following message is displayed whenever the data specified by <data 1> is found.

```
xxxxxxxx  CHANGE  (Y/N) ?  y  (RET)
```

xxxxxxxx: Address where <data 1> was found.

y: Y: <data 1> is replaced with <data 2>.

N: Data is not replaced; continues to search for another occurrence of the specified data. To terminate this command before reaching <end address>, press the (CTRL) + C keys.

— If <data 1> is not found at any point in the replacement range, the following message is displayed:

```
*** 45: NOT FOUND
```

— Memory modification with this command can be performed only in areas CS0 to CS6 or the internal memory areas.

— By specifying P or V, the physical or virtual address can be specified. When P or V is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and the physical or virtual address is determined based on the MMU status at command input when disabled.

— Do not change data using this command in the monitor program area.

Examples

1. To replace 2-byte data H'6475 in the address range from H'7000 to H'7FFF with H'5308 (with confirmation message):

```
:DC 6475 5308 7000 7FFF ;W  (RET)
 00007508 CHANGE (Y/N) ? Y  (RET)
 00007530 CHANGE (Y/N) ? N  (RET)
:
```

2. To replace 4-byte data 'DATA' in the address range from H'FB80 to H'FE00 with 'DATE' (without confirmation message):

```
:DC 'DATA' 'DATE' FB80 FE00 ;L Y  (RET)
:
```

7.2.15 DATA_SEARCH [DS] Searches for memory data

Command Format

- Search: DATA_SEARCHΔ<data>[Δ<start address>[(Δ<end address>/
  Δ@<number of bytes>)]][;[<size>]
  [Δ(P/V[=<ASID value>])][ΔN]] (RET)

<data>:	Data to be searched for
<start address>:	Search start address (Default: H'0)
<end address>:	Search end address (Default: Maximum address of H'FFFFFFFF)
<number of bytes>:	The number of bytes to be searched for (Default: Maximum address of H'FFFFFFFF (same as <end address>))
<size>:	Length of data to be searched for
B:	1 byte
W:	2 bytes
L:	4 bytes
Default:	1 byte
P:	Physical address
V:	Virtual address (address is converted using the TLB at command input)
<ASID value>:	the ASID value (can be specified only when V is specified)
N:	Data other than the specified data is searched for

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Search
  - Searches for <data> from the start address to the end address (or for the specified <number of bytes>). All addresses where <data> is found are displayed.
  - If data is not found, the following message is displayed:

    ```
    ***  45:NOT FOUND
    ```
  - If the N option is specified, data other than the specified <data> is searched for.

- Search with this command can be performed only in areas CS0 to CS6 or the internal memory areas.

- By specifying P or V, the physical or virtual address can be specified. When P or V is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and the physical and virtual address is determined based on the MMU status at command input when the VP_MAP table is disabled.

- Do not search for data using this command in the monitor program area.

Examples

1. To search for 1-byte data H'20 in the address range from H'FB80 to H'FF7F:

```
:DS 20 H'FB80 H'FF7F  (RET)
 0000FBFB 0000FCCD
:
```

2. To search for data other than 2-byte data H'0 in H'100 bytes starting from address H'1000:

```
:DS 0 1000 @100;W N (RET)
 *** 45:NOT FOUND
:
```

7.2.16 DISASSEMBLE [DA] Disassembles and displays memory contents

Command Format

- Display: DISASSEMBLEΔ<start address>[(Δ<end address>/
  Δ@<number of instructions>)][;(P/V[=<ASID value>])] (RET)

<start address>:	Start address of disassembly
<end address>:	End address of disassembly
<number of instructions>:	The number of instructions to be disassembled
P:	Physical address
V:	Virtual address (address is converted using the MCU TLB at command input)
<ASID value>:	the ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Display

— Disassembles the specified memory contents and displays addresses, machine codes, mnemonics, and operands in the following format. As many lines as necessary are used for the display.

```
    ADDR          CODE          MNEMONIC    OPERAND
  <address>   <machine code>   <mnemonic>   <operand>
```

— If <end address> or <number of instructions> is omitted, 16 instructions are disassembled and displayed.

— If there is no applicable instruction,

DATA.W   xxxx

is displayed.

If <start address> is an odd address,

DATA.B   xx

is displayed.

— Immediately after executing this command (except when it is forcibly terminated by the (CTRL) + C keys or (BREAK) key, or by an error), pressing the (RET) key will disassemble and display the next 16 lines of data.

— Disassemble can be performed only in areas CS0 to CS6 or the internal memory areas.

— By specifying P or V, the physical or virtual address can be specified. When P or V is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and the physical and virtual address is determined based on the MMU status at command input when disabled.

— Do not disassemble using this command in the monitor program area.

Examples

1. To disassemble and display six instructions starting from address H'1000:

```
:DA 1000 @6  (RET)
 ADDR     CODE      MNEMONIC    OPERAND
 00001000 E000      MOV         #00,R0
 00001002 2100      MOV.B       R0,@R1
 00001004 2201      MOV.W       R0,@R2
 00001006 430B      JSR         @R3
 00001008 0009      NOP
 0000100A 3400      CMP/EQ      R0,R4
:
```

2. To disassemble and display 16 instructions starting from address H'1000, and to disassemble and display furthermore 16 instructions by only entering (RET):

```
:DA 1000  (RET)
 ADDR     CODE      MNEMONIC    OPERAND
 00001000 1F01      MOV.L       R0,@(4,R15)
 00001002 6673      MOV         R7,R6
 00001004 E001      MOV         #1,R0
 00001006 3708      SUB         R0,R7
 00001008 1F52      MOV.L       R5,@(8,R15)
 0000100A 1F43      MOV.L       R4,@(C,R15)
 0000100C E00A      MOV         #0A,R0
 0000100E 6053      MOV         R5,R0
 00001010 1658      MOV.L       R5,@(20,R6)
 00001012 5568      MOV.L       @(20,R6),R5
 00001014 6053      MOV         R5,R0
 00001016 880A      CMP/EQ      #0A,R0
 00001018 8902      BT          00001020
 0000101A E001      MOV         #01,R0
 0000101C 380C      ADD         R0,R8
 0000101E 0009      NOP
:(RET)
 ADDR     CODE      MNEMONIC    OPERAND
 00001020 2100      MOV.B       R0,@R1
 00001022 2201      MOV.W       R0,@R2
 00001024 2302      MOV.L       R0,@R3
     :                 :
     :                 :
```

7.2.17 DUMP [D] Displays memory contents

Command Format

- Display DUMPΔ<start address>[(Δ<end address>/Δ[@]<number of bytes>)] [;[<display unit>][Δ(P/V[=<ASID value>])]] (RET)

<start address>: Start address for memory dump

<end address>: End address for memory dump

<number of bytes>: Size of data for memory dump

If @ is omitted, this value is determined as <end address> or <number of bytes> according to the inequalities given below.

Default is 256 bytes, as size.

End address:	<start address> ≤ specified value
Number of bytes:	<start address> > specified value

<display unit>: Size of display unit

B:	1-byte units
W:	2-byte units
L:	4-byte units
Default:	1-byte units

P: Physical address

V: Virtual address (address is converted using the MCU TLB at command input)

<ASID value>: the ASID value (can be specified only when V is specified)

Notes

1: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

2: The DUMP command can be executed in the parallel mode, but realtime operation cannot be performed.

Description

- Display

— Displays a memory dump of the specified area as follows:

```
<ADDRESS>           <DATA>                   <ASCII CODE>
 xxxxxxxx    xx...............................xx     "xxxx...............xx"
   (a)                (b)                        (c)
```

(a) Address

(b) Memory contents

(c) Memory contents displayed as ASCII codes. If there is no applicable ASCII code, a period (.) is displayed instead.

— By specifying P or V, the physical or virtual address can be specified. When P or V is omitted, the address is converted using the VP_MAP table when the VP_MAP table is enabled, and the physical and virtual address is determined based on the MMU status at command input when disabled.

— After this command is completed (except for a forced break), if no other commands are executed, the next address of the previously displayed memory can be displayed by pressing (RET).

— Do not execute this command in the monitor program area.

Note

When the reserved area dump is displayed, the display contents are undefined.

Examples

1. To display a memory dump from addresses H'0 to H'2F:

```
:D 0 2F (RET)
 <ADDRESS>            <    D   A    T   A    >                 <ASCII CODE>
 00000000  20 48 20 49 20 54 20 41   20 43 20 48 20 49 20 20  " H I T A C H I  "
 00000010  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00000020  20 20 20 20 20 20 20 20   20 20 20 45 38 30 30 30  "           E8000"
:
```

2. To display H'20 bytes of memory dump from address H'FB80 in 4-byte units:

```
:D FB80  20 ;L (RET)
 <ADDRESS>            <    D   A    T   A    >           <ASCII CODE>
 0000FB80  00000000   00000001    00000002   00000003    "................"
 0000FB90  00000000   00000001    00000002   00000003    "................"
:
```

7.2.18 END [E] Cancels parallel mode

Command Format

- Cancellation END (RET)

Description

- Cancellation
  - — Cancels parallel mode during GO command execution.
  - — Entering the END command clears old trace information and starts storing new trace information.

Example

To cancel parallel mode during GO command execution:

```
:G (RET)
 ** PC=00003400        (RET)        (Parallel mode entered)
#M FD80 (RET)
 0000FD80  00     ?  FF (RET)       (Command execution in parallel mode)
 0000FD81  00     ?  . (RET)
#E (RET)                            (Parallel mode cancellation)
 ** PC=00003800
:
```

7.2.19 EXECUTION_MODE [EM] Specifies and displays execution mode

Command Format

- Setting EXECUTION_MODE [•BREQ=<BREQ option>][ΔTIME=<TIME option>]
  [ΔTRGB=<TRGB option>][ΔMON=<MON option>]
  [ΔMB=<MB option>] [ΔRES=<RES option>]
  [ΔEMBW=<EMBW option>] [;C] (RET)
- Setting EXECUTION_MODE[;C] (RET)
  (interactive mode)

<BREQ option>: Specifies whether the BREQ (bus request) signal inputs are enabled.

E: Enables the BREQ signal inputs (default at emulator shipment)

D: Disables the BREQ signal inputs

<TIME option>: Specifies the minimum time to be measured for the GO command execution.

1: 1.6 µs

2: 406 ns

3: 20 ns (default at emulator shipment)

<TRGB option>: When BREAK_CONDITION_B or TRACE_CONDITION_B command is satisfied, specifies whether a pulse is output from the trigger output pin of the emulator without a break.

(1/2/3/4/5/6/7/8): Outputs a trigger pulse when the specified channel of the BREAK_CONDITION_B or TRACE_CONDITION_B command is satisfied.

A: Outputs a trigger pulse when either BREAK_CONDITION_B or TRACE_CONDITION_B command is satisfied.

D: Does not output a trigger pulse (default at emulator shipment)

<MON option>: Specifies time interval for execution status display.

0: No display

1: Approximately 200 ms (default at emulator shipment)

2: Approximately 2 s

<MB option>: Specifies whether the multi-break is enabled

E: Enabled

D: Disabled (default at emulator shipment)

<RES option>: Specifies whether the RESETP (power-on reset) or RESETM (manual reset) signal inputs are enabled.

E: Enabled (default at emulator shipment)

D: Disabled

<EMBW option>: Bus width of the area allocated to emulation memory

1: 32-bit bus

2: 16-bit bus (default at emulator shipment)

C: Stores the settings as configuration information in the emulator flash memory.

Description

- Specification

— Enables or disables the BREQ signal (bus request signal) inputs during user program execution.

- To disable the BREQ signal inputs during emulator operation and user program execution:

```
: EXECUTION_MODE  BREQ=D  (RET)
```

- To enable the BREQ signal inputs during emulator operation and user program execution:

```
: EXECUTION_MODE  BREQ=E  (RET)
```

— Specifies the minimum time to be measured for GO command execution.

- To set the minimum time to 1.6 µs:

```
: EXECUTION_MODE  TIME=1  (RET)
```

- To set the minimum time to 406 ns:

```
: EXECUTION_MODE  TIME=2  (RET)
```

- To set the minimum time to 20 ns:

```
: EXECUTION_MODE  TIME=3  (RET)
```

— Specifies whether to output a pulse from the trigger output pin when conditions set by the BREAK_CONDITION_B or TRACE_CONDITION_B command are satisfied.

- To output a trigger pulse when the condition set by the BREAK_CONDITION_B1 or TRACE_CONDITION_B1 command is satisfied:

  : ***EXECUTION_MODE TRGB=1 (RET)***
- To output a trigger pulse when any condition set by the BREAK_CONDITION_B or TRACE_CONDITION_B command is satisfied:

  : ***EXECUTION_MODE TRGB=A (RET)***
- To not output a trigger pulse:

  : ***EXECUTION_MODE TRGB=D (RET)***

  Notes:

  1. **When the BREAK_CONDITION _B condition by which trigger output is specified is satisfied, trigger pulse is output and break does not occur.**

  2. **When trigger output is specified, trace stop function of the TRACE_CONDITION_A,B,C cannot be used.**

— Specifies time interval for execution status display during GO command execution.

- To not display PC:

  : ***EXECUTION_MODE MON=0 (RET)***
- To display PC every 200 ms:

  : ***EXECUTION_MODE MON=1 (RET)***
- To display PC every 2 s:

  : ***EXECUTION_MODE MON=2 (RET)***

— Sets the multi-break function. The "multi-break function" is a function which breaks several E8000 emulators at the same time using trigger pulse output and external probe pin 1.

- To set the multi-break function to "disabled."

  : ***EXECUTION_MODE MB=D (RET)***
- To set the multi-break function to "enabled."

  : ***EXECUTION_MODE MB=E (RET)***

— Specifies whether to enable input of the RESETP (power-on reset) and RESETM (manual reset) signals.

- To disable input of the RESETP and RESETM signals:

  : ***EXECUTION_MODE RES=D (RET)***

- To enable input of the RESETP and RESETM signals:

  : ***EXECUTION_MODE RES=E (RET)***

— Specifies the bus width of the area allocated to emulation memory.

- To specify a 32-bit bus width for the area allocated to emulation memory:

  : ***EXECUTION_MODE EMBW=1 (RET)***

- To specify a 16-bit bus width for the area allocated to emulation memory:

  : ***EXECUTION_MODE EMBW=2 (RET)***

— When the C option is specified, the following message is displayed to confirm with the user whether to overwrite the existing configuration information in the emulator flash memory.

CONFIGURATION STORE OK (Y/N) ? ***(a) (RET)***

(a) Y: Stores the specifications as configuration information in the emulator flash memory. Hereafter, when the emulator is activated, the saved specifications go into effect.

N: Does not overwrite configuration information. The existing specifications are valid.

- Specification (interactive mode)

  When all options are omitted, the current set values are displayed and the emulator enters the interactive mode. Enter the required value for each item. Enter (RET) for the item not to be modified. To exit the interactive mode, enter a period (.). In this case, modifications before entering a period are valid.

```
: EXECUTION_MODE (RET)
  BREQ=E   TIME=1.6us TRGB=D   MON=1 MB=D RES=E EMBW=32BIT   (Displays current values)
  BREQ (D:DISABLE/E:ENABLE) ?  (RET)
  TIME (1:1.6us/2:406ns/3:20ns) ? (RET)
  TRGB (A:ALL/1:B1/2:B2/3:B3/4:B4/5:B5/6:B6/7:B7/8:B8/D:DISABLE) ? (RET)
  MON (0:DISABLE/1:200ms/2:2s) ? (RET)
  MB (D:DISABLE/E:ENABLE) ? (RET)
  RES (D:DISABLE/E:ENABLE) ? (RET)
EMBW (1:32BIT/2:16BIT) ? (RET)
:
```

Examples

1. To enable the BREQ (bus request) signal inputs and store configuration information:

```
:EM BREQ=E;C (RET)
 CONFIGURATION STORE OK(Y/N)? Y(RET)
:
```

2. To display the specified values of the current emulation mode and modify them in interactive mode (command execution can be terminated by entering a period (.)):

```
:EM (RET)
 BREQ=E TIME=20ns TRGB=D MON=1 MB=D RES=E EMBW=32BIT
 BREQ (D:DISABLE/E:ENABLE) ? (RET)                (Input (RET) for no modification)
 TIME (1:1.6us/2:406ns/3:20ns) ? 1 (RET)  (Input 1 to set minimum measure time to 1.6 µs)
 TRGB (A:ALL/1:B1/2:B2/3:B3/4:B4/5:B5/6:B6/7:B7/8:B8/D:DISABLE) ? (RET)
 MON (0:DISABLE/1:200ms/2:2s) ? . (RET)  (Command is terminated and new settings become valid)
:
```

7.2.20 FILL [F] Writes data to memory

Command Format

- Write FILLΔ<start address>(Δ<end address>/Δ@<number of bytes>)[Δ<data>]
  [;[<size>][Δ (P/V[=<ASID value>])][ΔN]] (RET)

<start address>: Write start address

<end address>: Write end address

<number of bytes>: The number of bytes to be written

<data>: Data to be written. Default is H'00.

<size>: Length of data to be written

B: 1 byte

W: 2 bytes

L: 4 bytes

Default: 1 byte

P: Physical address

V: Virtual address (address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

N: No verification

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Write

  — Writes data to the specified memory area. Default value is H'00.

  — After data is written, it is also verified. This command can therefore be used as a memory test. If an error occurs, the following message is displayed and processing is terminated.

```
FAILED AT xxxxxxxx   WRITE = yy..'y..'     READ = zz..'z..'
```

xxxxxxxx:	Error address
yy..'y..':	Write data (hexadecimal and ASCII characters)
zz..'z..':	Read data (hexadecimal and ASCII characters)

  — If W is specified as <size>, but the start address is odd, the lowest bit is rounded down to the preceding even address. If L is specified as <size>, the lower bits are rounded down to become a multiple of four. Writing never exceeds the specified <end address>.

  — Data can be written only to areas CS0 to CS6 or the internal memory areas.

  — By specifying P or V, a physical or virtual address can be specified, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

  — Do not write data to the monitor program area.

Example

To fill the entire area from addresses H'0 to H'6FFF with 1-byte data H'00:

```
:F 0 6FFF 0  (RET)
:
```

7.2.21 GO [G] Provides realtime emulation

Command Format

- Execution GO[Δ[<start address>][;[<break address>][V=<ASID value>]
  [Δ<mode>]] (RET)

<start address>: Start address of realtime emulation, or the word RESET

<break address>: Breakpoint address (Break occurs before the instruction at the break address is executed.)

V=<ASID value>: ASID value of the break address

<mode>: Emulation mode

R=<n>: Cycle reset mode; n = 1 to 12

N: Temporarily invalidates break conditions

SB: BREAK_CONDITION_UBC sequential break mode 1

TB: Causes a break to occur at the timeout value specified with the TIME option of the PERFORMANCE_ANALYSIS1 command

Description

- Execution

— Executes realtime emulation (user program execution) starting from the specified <start address>. The following data can be specified as <start address>.

: ***GO <address> (RET)*** Executes the program from the specified address.

: ***GO (RET)*** When omitting the address, the program executes from the address where the current PC indicates.

: ***GO RESET (RET)*** After a RESETP signal input to the MCU, PC and SP are set to the values specified with the reset vector and program execution starts.

— According to the <mode> specification at GO command input, the user program is executed in one of the following modes. If no <mode> is specified, normal emulation mode is assumed.

- Cycle reset mode

  A RESETP signal is input to the MCU at the intervals given in table 7.11, and program execution continues. In this mode, all break conditions and trace conditions are invalidated.

Table 7.11 Cycle Reset Times

Value of n	Reset Interval
1	6.5 µs
2	9.8 µs
3	50 µs
4	100 µs
5	500 µs
6	1 ms
7	5 ms
8	10 ms
9	50 ms
10	100 ms
11	500 ms
12	1 s

- Temporary invalidation of break conditions

  If the N option is specified, software breakpoints (set with the BREAK or BREAK_SEQUENCE command) and hardware break conditions (set with the BREAK_CONDITION_A,B,C or BREAK_CONDITION_UBC command) are invalidated temporarily, and user program emulation continues. The breakpoints and break conditions are invalidated only within one GO command emulation. If the N option is not specified in the next GO command emulation, breakpoints and break conditions are validated again.

- Sequential break mode

  Realtime emulation stops only when break conditions set with the BREAK_CONDITION_UBC1,2 command are satisfied in the sequence of the BREAK_CONDITION_UBC2 condition followed by the BREAK_CONDITION_UBC1 condition.

- Timeout break mode

  A break occurs when the timeout or execution count condition specified with the PERFORMANCE_ANALYSIS command is satisfied.

The restrictions for each mode at emulation are listed in table 7.12.

Table 7.12 Restrictions for Realtime Emulation Modes

Modes	Restrictions
Cycle reset mode	• Software breakpoints specified with the BREAK or BREAK_SEQUENCE command are ignored. • Hardware break conditions specified with the BREAK_CONDITION_A,B,C or BREAK_CONDITION_UBC command are ignored. • All conditions specified with the TRACE_CONDITION_A,B,C command are ignored. • Parallel mode cannot be entered.
Temporary invalidation of break conditions	• Software breakpoints specified with the BREAK or BREAK_SEQUENCE command are ignored. • Hardware break conditions specified with the BREAK_CONDITION_A,B,C or BREAK_CONDITION_UBC command are ignored.
Sequential break mode	• Software breakpoints specified with the BREAK or BREAK_SEQUENCE command are ignored. • Conditions must be specified with the BREAK_CONDITION_UBC1, 2 command.
Timeout break mode	Software breakpoints specified with the BREAK or BREAK_SEQUENCE command are ignored.

— If <break address> is specified, realtime emulation stops when the specified address matches the PC value. The instruction at the specified address is not executed. This specification is valid for only the current GO command emulation. BREAK_CONDITION_UBC2 command settings are invalid when a break address is specified.

— When V=<ASID value> is specified, the ASID value of the break address can be specified. Even if the MCU MMU is disabled, the execution breaks if the ASID value of the PTEH register and the specified value match.

— Do not specify the monitor program area for the break address.

— During user program execution, program fetch addresses are displayed according to the time interval specified with the MON option in the EXECUTION_MODE command.

```
:GO (RET)
**PC = xxxxxxxx
```

xxxxxxxx: Address bus

— During GO command emulation, pressing the SPACE key or (RET) key enters parallel mode. However, since user program execution stops during memory access, realtime operation is not performed.

— If emulation is terminated, register contents, execution time, and cause of termination are displayed in the following format:

```
PC=00005C60  SR=700000F0:-P1B0000000000000000--------IIII00--                    (a)
SPC=00000000  SSR=00000000  ASID=00
GBR=00000000  VBR=A0000000  MACH=00000000  MACL=00000000  PR=00000000
RS=00000000  RE=00000000  MOD=00000000
R0–7    00000000  000000FF  00000011  00000000  00000000  00000000  00000000  00000000
R8–15  00000000  00000000  00000000  00000000  00000000  00000000  00000000  A00FFE00
R0_BANK0-R3_BANK0  00000000  00000000  00000000  00000000
R4_BANK0-R7_BANK0  00000000  00000000  00000000  00000000
R0_BANK1-R3_BANK1  00000000  00000000  00000000  00000000
R4_BANK1-R7_BANK1  00000000  00000000  00000000  00000000
DSR=000000F1:-------------------------------------------GZNVCOBD
A0G=00  A0=00000000  M0=00000000  X0=00000000  Y0=00000000
A1G=00  A1=00000000  M1=00000000  X1=00000000  Y1=00000000
RUN-TIME=D'0000H:00M:00S:000000US:000NS                                          (b)
+++: <cause of termination>                                                      (c)
```

(a) The register contents at emulation termination.

(b) User program execution time in decimal. According to the TIME option of the EXECUTION_MODE command, the maximum measurable time is 488, 124, or 6 hours, where the minimum measurement time is 1.6 µs, 406 ns, or 20 ns, respectively. If the period exceeds the maximum measurable time, it is displayed as *.

(c) Cause of termination, as listed in table 7.13.

Table 7.13 Causes of GO Command Termination

Message	Termination Cause
BREAK CONDITION A1, 2, 3, 4, 5, 6, 7, 8	Break condition is satisfied by BREAK_CONDITION_A.
BREAK CONDITION B1, 2, 3, 4, 5, 6, 7, 8	Break condition is satisfied by BREAK_CONDITION_B.
BREAK CONDITION C1, 2, 3, 4, 5, 6, 7, 8	Break condition is satisfied by BREAK_CONDITION_C.
BREAK CONDITION UBC1, 2, 3	Break condition is satisfied by BREAK_CONDITION_UBC.
BREAK CONDITION SB	Sequential break condition is satisfied by BREAK_CONDITION_UBC.
BREAK KEY	The (CTRL) + C keys were pressed or the ABORT command was executed for forcible termination.
BREAKPOINT	Emulation stopped at a software breakpoint specified with the BREAK command.
BREAK SEQUENCE	A condition for passing software breakpoints specified with the BREAK_SEQUENCE command was satisfied.
GUARDED OR WRITE PROTECT	Execution is stopped because access to access-inhibited area or write to write-protected area occurred.
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed.
MULTI BREAK	Execution is stopped by multi-break.
NO EXECUTION	The user program was not executed (this message is displayed only for the RESULT command).
RESET BY E8000	The emulator forcibly terminates program execution with the RESETP signal because an error has occurred in the user system.
STOP ADDRESS	Emulation stopped at the break address specified with the GO command.
SUBROUTINE TIMEOUT	The timeout condition specified with the PERFORMANCE_ANALYSIS1 command was satisfied.
SUBROUTINE COUNT OVERFLOW	The execution count limit specified with the PERFORMANCE_ANALYSIS1 command was exceeded.
TRACE BUFFER OVERFLOW	The trace buffer overflowed.

— During user program execution, the MCU execution status is displayed. Displayed contents are shown in table 7.14. Time interval specified when MON option of the EXECUTION_MODE command was specified, and if there is a difference from the previous status, the status is displayed.

Table 7.14 Execution Status Display

Display	Meaning
** PC=xxxxxxxx	During user program execution, the program fetch address is displayed according to the time interval specified with the MON option in the EXECUTION_MODE command.
** RESET	The MCU has been reset. The RESETP signal is low.
** RUNNING	User program execution has started. This message is displayed once when GO command execution starts or when parallel mode is cancelled. Note that this message will be deleted when **PC=xxxxxxxx (second message in this table) is displayed.
** SLEEP	The MCU is in the sleep mode.
** SOFWARE STANDBY	The MCU is in the software standby mode.
** TOUT A = xxxxxxxx	The address bus value is displayed. The bus cycle stops for 1.28 ms or more. Note: This message is displayed when memory access does not occur via the external bus due to the execution of a program in the cache memory or in the internal memory area.
** VCC DOWN	User system Vcc (power voltage) is 2.65 V or less. The MCU is not operating correctly. (Displayed only when the user clock is selected.)
** WAIT A = xxxxxxxx	The WAIT signal is low. The address bus value is displayed. The address is not displayed during the refresh cycle.
**BREQ	The BREQ signal is low.

— If the TB option is specified, user program execution stops when the timeout value or execution count limit specified with the PERFORMANCE_ANALYSIS1 command is exceeded.

Notes

1. When a hardware break condition (set by the BREAK_CONDITION_A,B,C command) is satisfied during program execution, the program does not terminate until several instructions that have been already fetched are executed. If another hardware break is satisfied before the user program terminates, several termination causes will be displayed. For further details, study trace information.
2. At each software breakpoint set with the BREAK command or at each pass point set with the BREAK_SEQUENCE command, the program halts at that address, the emulator analyzes the pass count and pass point of the program, and then the program continues. When the memory access command processing in parallel mode occurs during this termination, memory cannot be accessed. At this time,

   ```
   *** 78: EMULATOR BUSY
   ```

   is displayed, and the command should be re-input.

   However, when the termination period is too short, the PC is not displayed, the emulator cannot enter parallel mode, or commands may not be executed in parallel mode.
3. When the contents of a breakpoint (set by the BREAK command) have been modified by the user program during emulation, that breakpoint will be cancelled at execution stop.
4. **SLEEP or **SOFTWARE STANDBY is displayed as the execution status display only when the STATUS[1:0]/PTJ[7:6] pin is set to STATUS[1:0] by the MODE command. When PTJ[7:6] has been specified, nothing is displayed even when the MCU enters the sleep or software standby mode.
5. When user program execution is halted by using the break key during cycle reset mode, the emulator may not correctly return to the command-input wait state. In this case, switch the power to the emulator off and then on to restart the system.

Examples

1. To reset the MCU and start emulation from the reset vector PC address:

```
:G RESET (RET)
 ** PC=00001130
```

2. To start emulation from address H'1000 and stop emulation just before address H'2020 is executed:

```
:G 1000;2020 (RET)
:
```

3. To start emulation from the current PC address and modify memory contents in parallel mode:

```
:G (RET)
 ** PC=00010204 (RET)
#M FEF0 (RET)
 0000FEF0 FE  ?  FF (RET)
 0000FEF1 FF  ?  . (RET)
#END (RET)
 ** PC=00011456
```

7.2.22 HELP [HE] Displays all commands and command format

Command Format

• Display	HELP (RET)	(All commands are displayed.)
	HELP Δ <command> (RET)	(The specified command format is displayed.)

Description

- Display
  - — Displays all emulator command names and abbreviations.

```
:HELP (RET)
  .<REGISTER>                               *AB           : ABORT
 *ALI             : ALIAS                     A            : ASSEMBLE
  BI              : BACKGROUND_INTERRUPT   **B            : BREAK
**BCA,1,2,3,4,5,6,7,8    : BREAK_CONDITION_A,1,2,3,4,5,6,7,8
**BCB,1,2,3,4,5,6,7,8    : BREAK_CONDITION_B,1,2,3,4,5,6,7,8
**BCC,1,2,3,4,5,6,7,8    : BREAK_CONDITION_C,1,2,3,4,5,6,7,8
**BCU,1,2,3       : BREAK_CONDITION_UBC,1,2,3
**BS              : BREAK_SEQUENCE            CH           : CHECK
**CL              : CLOCK                     CNF          : CONFIGURATION
 *CV              : CONVERT                   DC           : DATA_CHANGE
  DS              : DATA_SERACH              *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
  PA,1,2,3,4,5,6,7,8     : PERFORMANCE_ANALYSIS,1,2,3,4,5,6,7,8
  Q               : QUIT                    *RX            : RADIX
  R               : REGISTER                 RS            : RESET
  RT              : RESULT                  *ST            : STATUS
  S               : STEP                     SI            : STEP_INFORMATION
  SO              : STEP_OVER               *T             : TRACE
 *TCA,1,2,3,4,5,6,7,8    : TRACE_CONDITION_A,1,2,3,4,5,6,7,8
 *TCB,1,2,3,4,5,6,7,8    : TRACE_CONDITION_B,1,2,3,4,5,6,7,8
 *TCC,1,2,3,4,5,6,7,8    : TRACE_CONDITION_C,1,2,3,4,5,6,7,8
 *TDM             : TRACE_DISPLAY_MODE        TMO          : TRACE_MODE
 *TS              : TRACE_SEARCH              L            : LOAD
  SV              : SAVE                      V            : VERIFY
  IL              : INTFC_LOAD                IS           : INTFC_SAVE
  IV              : INTFC_VERIFY            *#ASC          : ASC
*#BIN             : BIN                     *#BYE          : BYE
*#CD              : CD                      *#CLOSE        : CLOSE
 *FTP             : FTP                      *LAN          : LAN
  LH              : LAN_HOST                 #LL           : LAN_LOAD
 #LSV             : LAN_SAVE                 #LV           : LAN_VERIFY
 *LO              : LOGOUT                  *#LS           : LS
*#OPEN            : OPEN                    *#PWD          : PWD
  RTR             : ROUTER                  *#STA          : STA
  SN              : SUBNET
```

Note: *: Usable in parallel mode
No *: Unusable in parallel mode
****: Available only for display in parallel mode**
#: Available when the FTP server is open.

— Displays specific command format when command name is specified:

`:HELP <command name>  (RET)`   (Displays command format)

Example

To display GO command format:

```
:HELP GO  (RET)
 Executes real-time emulation.
      G [<addr1>][;[<breakaddr>][V=<ASID>][ <mode>] <RET>
  <addr1>      : {RESET,<address>}
   RESET       : execute after MPU reset
  <address>    : starting address
                 if deleted executes from current PC
  <breakaddr>  : address when stopping the program
  V=<ASID>  : break address for virtual page number
  <mode>       : R=<n>   - cycle reset mode ( n = 1 to 12 )
               : N       - temporarily invalidates break conditions
               : SB      - sequential break mode UBC
               : TB      - time out break mode
                 default - normal mode
:
```

7.2.23 HISTORY [HT] Displays input command history

Command Format

- Display HISTORY (RET) (Displays all input commands)
  HISTORY <history number> (RET) (Displays the input command of the specified history number)

<history number>: History number (1 to 16)

Description

- Display

— Displays the 16 commands most recently input including the HISTORY command in the input order.

— If <history number> is entered, the command corresponding to <history number> is displayed as shown below and the emulator enters command input wait state. When the (RET) key is pressed, the displayed command is executed.

Note

Subcommands cannot be displayed by the HISTORY command.

Example

```
:HISTORY (RET)
 1 MAP
 2 MAP  0 FFFFFF;U
 3 F 0 1000 FF
 4 B 300
 5 BCA1 A=104
 6 HISTORY
:HISTORY 5 (RET)
:BCA1 A=104_ ------------Enters command input wait state
```

7.2.24 ID [ID] Displays version number of emulator system program

Command Format

- Display ID (RET)

Description

- Display
  Displays the version and revision numbers of the SH7729 E8000 system program.

Example

To display the version and revision numbers of the SH7729 E8000 system program:

```
:ID (RET)
 SH7729 E8000 (HS7729EDD81SF) Vm.n
 Copyright (C) Hitachi, Ltd. 1998
 Licensed Material of Hitachi, Ltd.
:
```

7.2.25 MAP [MP] Specifies and displays memory attribute

Command Format

- Specification MAPΔ<start address>Δ<end address>;<memory attribute> (RET)
- Display MAP[Δ<start address>Δ<end address>] (RET)

<start address>: Start address of memory area whose attribute is to be specified or displayed

<end address>: End address of memory area whose attribute is to be specified or displayed

<memory attribute>: Memory type

U:	Memory in the user system (cancels emulation memory usage)
S:	Standard emulation memory in emulator
SW:	Standard emulation memory in emulator with write protection

Description

- Specification

— Standard emulation memory allocation

Memory can be allocated to the CS0 to CS6 area in 4-Mbyte units. Write prohibited (SW) is also possible. The start address is rounded down to 0 or a multiple of H'400000, and the end address is rounded up to a multiple of H'400000-1.

```
:MAP  400000  H'7FFFFF;S  (RET)
```

After the memory is allocated, the remaining amount of standard emulation memory is displayed in 4-Mbyte units.

```
REMAINING EMULATION MEMORY   S=xMB
```

x: Unused size of standard emulation memory (4-Mbyte units)

— To use memory in the user system, specify U for the memory attribute.

— To cancel the write protection of standard emulation memory (SW), respecify S as the memory attribute.

- Display

— Displays the memory attribute of the area defined by <start address> and <end address>, in the following format:

```
: MAP <start address>  <end address>  (RET)
xxxxxxxx–xxxxxxxx;y                                               (a)
         ...
xxxxxxxx–xxxxxxxx;y
INTERNAL I/O = xxxxxxxx–xxxxxxxx xxxxxxxx–xxxxxxxx                (b)
REMAINING EMULATION MEMORY S=xMB                                  (c)
```

(a) Address range and memory attribute

Displays the addresses to which standard emulation memory is allocated.

y: Standard emulation memory attribute

S: Standard emulation memory in emulator

SW: Standard emulation memory in emulator with write protection

(b) Range of internal I/O area

(c) Unused standard emulation memory size in hexadecimal

The size of standard emulation memory that can be allocated is displayed.

S=xMB (standard emulation memory)

x: Unused size of the standard emulation memory (4-Mbyte units)

— When no address is specified, the memory attributes of all memory areas are displayed in the format shown above.

Notes

1. If there is not enough standard emulation memory to satisfy the specification, the memory attribute is specified only for the memory area available.
2. Standard emulation memory cannot be allocated to areas other than areas CS0 to CS6.
3. A memory attribute cannot be allocated to a range which includes a reserved area.

Examples

1. To allocate standard emulation memory to the address range from H'0 to H'3FFFFF:

```
:MP 0 3FFFFF;S (RET)
 REMAINING EMULATION MEMORY S=4MB
:
```

2. To allocate standard emulation memory to the address range from H'400000 to H'7FFFFF with write protection:

```
:MP 400000 7FFFFF ;SW (RET)
 REMAINING EMULATION MEMORY S=0MB
:
```

3. To display the memory address ranges and attributes of allocated standard emulation memory, the internal memory address ranges, and the internal I/O address range:

```
:MP (RET)
 00000000-003FFFFF;S
 00400000-007FFFFF;SW
 INTERNAL I/O = 04000000-04FFFFFF E0000000-FFFFFFFF
 REMAINING EMULATION MEMORY S=0MB
:
```

4. To cancel write protection for the standard emulation memory allocated to the address range from H'400000 to H'7FFFFF:

```
:MP 400000 7FFFFF ;S (RET)
 REMAINING EMULATION MEMORY S=0MB
:
```

7.2.26 MEMORY [M] Displays or modifies memory contents

Command Format

- Display, modification MEMORYΔ<address>[Δ<data>][;[<option>]
  [(P/V[=<ASID value>])][ΔN]] (RET)

<address>:	Address of memory area whose contents are to be displayed or modified
<data>:	Data to be written to the specified address
<option>:	Length of display or modification units
	B: 1-byte units
	W: 2-byte units
	L: 4-byte units
	XW: 16-bit fixed-point value units
	XL: 32-bit fixed-point value units
	O: Odd address; 1-byte units
	E: Even address; 1-byte units
	Default: 1-byte units
P:	Physical address
V:	Virtual address (Address is converted using the MCU TLB at command input.)
<ASID value>:	ASID value (can be specified only when V is specified)
N:	No verification

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Display, modification

— If <data> is omitted, the emulator displays memory contents at the specified address and enters input wait state of the modification data. The user can then enter data and modify memory contents; this process can then be repeated for the next address. When the N option is not specified, the modified data is read and verified.

Note: 1. The internal I/O areas are always not verified.

2. When no verification is specified and the contents in the reserved area are changed, a verify error occurs because the value which has read the changed data is undefined.

The memory contents display and the modified data input format are as follows:

```
: MEMORY <address>  (RET)
xxxxxxxx   yyyyyyyy   ? [<data>][;<option>]  (RET)
```

xxxxxxxx: Address of data to be modified

yyyyyyyy: Memory contents displayed in modification units.

<data>: New data. Data length is considered to be the same as that of the data displayed on the screen. If only the (RET) key is pressed, data is not modified, and the next address is displayed.

<option>: The unit of display or modification can be changed, or the address can be incremented or decremented. When <data> is specified, <option> is processed after the data is modified. When <data> is not specified, a semicolon (;) can be omitted to specify options L, W, O, ^, =, or . (period). Table 7.15 lists option functions.

Notes

1. In parellel mode, realtime operation is not performed.
2. Do not display or modify memory contents in monitor program area.

Table 7.15 MEMORY Command Options

Option	Description
B	Modification in 1-byte units
W	Modification in 2-byte units
L	Modification in 4-byte units
XW	Modification in 16-bit fixed-point units
XL	Modification in 32-bit fixed-point units
O	Odd address; modification in 1-byte units
E	Even address; modification in 1-byte units
^	Display of previous address contents
=	Display of current address contents
.	Command termination
Default	Display of next address contents

— When specifying <address> and <data>, memory contents are modified immediately and the emulator waits for the next command input.

```
: MEMORY  H'FFF0  H'F8 (RET)
:
```

— By specifying P or V, the physical or virtual address can be specified, respectively. When P or V is omitted, the physical and virtual address are determined based on the MMU status at command input.

Notes

1. The MEMORY command can be executed in the parallel mode, but realtime operation cannot be performed.
2. Do not display or modify memory contents in the monitor program area.

Examples

1. To modify memory contents from address H'1000:

```
:M 1000 (RET)
 00001000 00        ?   FF (RET)
 00001001 01        ?   10 (RET)
 00001002 22        ?   (RET)
 00001003 00        ?   30;W (RET)
 00001004 0000      ?   1234 (RET)
 00001006 1100      ?   ^ (RET)
 00001004 1234      ?   ;L (RET)
 00001004 12341100  ?   12345678 (RET)
 00001008 00000000  ?   . (RET)
:
```

2. To modify memory contents from address H'8000 in 2-byte units without verification:

```
:M 8000 ;W N (RET)
 00008000  0000  ?  FF (RET)
 00008002  0002  ?  1000 (RET)
 00008004  FFF2  ?  . (RET)
:
```

3. To modify memory contents from address H'F000 in 16-bit fixed-point value units:

```
:M F000 ;XW (RET)
 0000F000  0.87544  ?  0.875 (RET)
 0000F002  0.45637  ?  0.5 (RET)
 0000F004  0.39285  ?  . (RET)
:
```

4. To write data H'10 to address H'FE00 without displaying the memory contents:

```
:M FE00 10 (RET)
:
```

7.2.27 MODE [MD] Specifies or displays MCU operating mode

Command Format

- Specification MODE;C (RET)
- Display MODE (RET)

Description

- Specification

— Interactively specifies the MCU operating mode in the emulator as shown below.

```
: MODE;C  (RET)
  CPU NAME (1:SH7729, 2:SH7709A)  ?   (a) (RET)
  E8000 MODE (0-3F)  ?   (b) (RET)
  MONITOR START ADDR  ?   (c) (RET)
  MONITOR BUS WIDTH (1:16BIT, 2:32BIT)  ?  (d) (RET)
  AREA2 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM)  ?  (e) (RET)
  AREA3 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM, 4:EDO-DRAM)  ?  (f) (RET)
  SDRAM TYPE (1:1MB-16BIT or 4MB-16BIT/2:2MB-8BIT or 8M-8BIT 3:4MB-4BIT/4:256KB-16BIT
  or 2M-32BIT) ?  (g) (RET)
  DRAM TYPE  (1:16BIT, 2:32BIT)  ?   (h) (RET)
  CS2/PTK[0] PIN (1:CS2, 2:PTK[0])  ?   (i) (RET)
  CS3/PTK[1] PIN (1:CS3, 2:PTK[1])  ?   (j) (RET)
  CS4/PTK[2] PIN (1:CS4, 2:PTK[2])  ?   (k) (RET)
  CS5/CE1A/PTK[3] PIN (1:CS5/2:CE1A or PTK[3]  ?   (l) (RET)
  STATUS[1:0]/PTJ[7:6] PIN (1:STATUS[1:0]/2:PTJ[7:6])  ?   (m) (RET)
  WE[2:3]/PTK[6:7] PIN (1:WE[2:3], 2:PTK[6:7]) ?   (n) (RET)
  BS/PTK[4] PIN (1:BS, 2:PTK[4])  ?   (o) (RET)
  CAS[LL:HH]/PTJ[2:5] PIN (1:CAS[LL:HH], 2PTJ[2:5])  ?   (p) (RET)
  CAS[LL:2H]/PTJ[2:3], PTE[3, 6] PIN (1:CAS[LL:2H], 2:PTJ[2:3], PTE[3, 6])  ?  (q) (RET)
  JTAG CLOCK (1:16.5MHz/2:8.25MHz/3:4.12MHz)   ?   (r) (RET)
  CONFIGURATION STORE OK (Y/N) ?  (s) (RET)
```

(a) Selects MCU. Example: To select SH7709A, enter 2.

(b) Sets operating mode.

(c) Specifies the start address of the monitor program. The address is rounded down to 0 or a multiple of H’40000.

(d) Specifies the bus size of the monitor program setting area.

(e) Sets the memory type for area 2.

(f) Sets the memory type for area 3.

(g) Sets the SDRAM type for area 3.

(h) Sets the DRAM type for area 3.

(i) Specifies the usage of the CS2/PTK[0] pin.

(j) Specifies the usage of the CS3/PTK[1] pin.

(k) Specifies the usage of the CS4/PTK[2] pin.

(l) Specifies the usage of the CS5/CE1A/PTK[3] pin.

(m) Specifies the usage of the STATUS[1:0]/PTJ[7:6] pin.

(n) Specifies the usage of the WE[2:3]/PTK[6:7] pin.

(o) Specifies the usage of the BS/PTK[4] pin.

(p) Specifies the usage of the CAS[LL:HH]/PTJ[2:5] pin.

(q) Specifies the usage of the CAS[LL:2H]/PTJ[2:3] pin.

(r) Sets the input clock of the JTAG interface.

(s) Stores the set values as the configuration information.

Y: Stores the configuration information in the emulator.

N: Terminates the command without storing the configuration information.

If Y is input in (s), stores the settings as configuration information in the emulator flash memory. When the emulator is initiated after configuration information storage, it emulates in the stored operating mode. The emulator system program terminates after the MCU operating mode is set, and must then be re-initiated.

- Display

  Displays the MCU operating mode in the emulator, the operating mode selection pin (MD5 to MD0) status on the user system, and the operating mode setting method in the following format:

```
: MODE  (RET)
  MCU NAME=aaaaaa  MODE=bb  (MD5-0 = cc)                                (a)
  MONITOR START ADDRESS=dddddddd MONITOR BUS WIDTH=eeeee                (b)
  AREA2 MEMORY=ffffff AREA3 MEMORY=gggggg                               (c)
  SDRAM TYPE=hhhhhhhhhhhhhhhhhhhh, DRAM TYPE=iiiii                      (d)
  CS2/PTK[0] PIN=jjjjjj CS3/PTK[1] PIN=kkkkkk CS4/PTK[2] PIN=llllll    (e)
  CS5/CE1A/PTK[3] PIN=mmmmmm  STATUS[1:0]/PTJ[7:6] PIN=nnnnnnnn         (f)
  WE[2:3]/PTK[6:7] PIN=oooooo  BS/PTK[4] PIN=pppppp                     (g)
  CAS[LL:HH]/PTJ[2:5] PIN=qqqqqqqq                                      (h)
  CAS[LL:2H]/PTJ[2:3], PTE[3:6] PIN=rrrrrrr                             (i)
  JTAG CLOCK=sssssss                                                    (j)
:
```

(a): aaaaaa = SH7729 or SH7709A; MCU type
bb = H’00 to H’3F; operating mode
cc = H’00 to H’3F (MD5-0 value); user operating mode pin (see table 7.16 for status).
When the user system is not connected, the value is undefined.

(b): dddddddd = H’00000000 to H’FFFC0000; monitor program start address
eeeee = 16BIT or 32BIT; bus width of the monitor program setting area

(c): ffffff = NORMAL, DRAM, or SDRAM; Area 2 memory type
gggggg = NORMAL, DRAM, SDRAM, or EDO-DRAM; Area 3 memory type

(d): hhhhhhhhhhhhhhhhhhhh = 1MB-16BIT, 4MB-16BIT, 2MB-8BIT, 8MB-8BIT, 4MB-4BIT, 256KB-16BIT, or 2M-32BIT; Area 3 SDRAM type
iiiii = 16BIT or 32BIT; Area 3 DRAM type

(e): jjjjjj = CS2, PTK[0]; The usage of the CS2/PTK[0] pin
kkkkkk = CS3, PTK[1]; The usage of the CS3/PTK[1] pin
llllll = CS4, PTK[2]; The usage of the CS4/PTK[2] pin

(f): mmmmmm = CS5, CE1A/PTK[3]; The usage of the CS5/ CE1A/PTK[3] pin
nnnnnnnn = STATUS[1:0], PTJ[7:6]; The usage of the STATUS[1:0]/PTJ[7:6] pin

(g) oooooo = WE[2:3]/PTK[6:7]; The usage of the WE[2:3]/PTK[6:7] pin
pppppp = BS/PTK[4]: The usage of the BS/PTK[4] pin

(h) qqqqqqqq = CAS[LL:HH], PTJ[2:5]; The usage of the CAS[LL:HH]/PTJ[2:5] pin

(i) rrrrrrr = CAS[LL:2H], PTJ[2:3], PTE[3:6]; The usage of the CAS[LL:2H]/PTJ[2:3]/PTE[3:6] pin

(j) sssssss = 16.5 MHz, 8.25 MHz, or 4.125 MHz; The input clock for the JTAG interface

Table 7.16 Operating Mode Selection Pin Status and Display

Big/Little Endian	CS0 Area Bus Width		Clock Mode			Display (nn)
MD5	**MD4**	**MD3**	**MD2**	**MD1**	**MD0**	
Low	Low	Low	Low	Low	Low	00
Low	Low	Low	Low	Low	High	01
Low	Low	Low	Low	High	Low	02
:	:	:	:	:	:	:
High	High	High	High	High	High	3F

Notes

1. The emulator operating mode is specified with the MODE command, regardless of the operating mode selection pin (MD5 to MD0) status on the user system.
2. The emulator supports only MCU clock modes 0, 1, 3, and 7. When another mode is selected, the error message

   ```
   ***22:INVALID DATA
   ```

   is displayed, and the emulator enters the operating mode input wait state.

Examples

1. To specify the MCU as SH7729, the MCU operating mode as 19, the monitor program start address as H’A01000000, the bus width as 16 bits, teh area 2 as SDRAM, the area 3 as SDRAM, the SDRAM type as 2MB-16BIT, CS2 to CS5 as CS, STATUS[1:0]/PTJ[7:6] pin as STATUS[1:0], WE[2:3]/PTK[6:7] pin as WE[2:3], BS/PTK[4] pin as BS, CAS[LL:HH]/PTJ[2:5] pin as CAS[LL:HH], CAS[LL:2H]/PTJ[2:3],PTE[3,6] as CAS[LL:2H], and JTAG clock as 16.5 MHz:

```
:MODE;C (RET)
  CPU NAME (1:SH7729, 2:SH7709A)  ?   1 (RET)
  E8000 MODE (0-3F)  ?   19 (RET)
  MONITOR START ADDR  ?   A0100000 (RET)
  MONITOR BUS WIDTH (1:16BIT, 2:32BIT)  ?  1 (RET)
  AREA2 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM)  ?  3 (RET)
  AREA3 MEMORY (1:NORMAL, 2:DRAM, 3:SDRAM, 4:EDO-DRAM)  ?  3 (RET)
  SDRAM TYPE (1:1MB-16BIT or 4MB-16BIT/2:2MB-8BIT or 8M-8BIT 3:4MB-4BIT/4:256KB-16BIT or
  2M-32BIT) ?  1 (RET)
  DRAM TYPE  (1:16BIT, 2:32BIT)  ?   1 (RET)
  CS2/PTK[0] PIN (1:CS2, 2:PTK[0])  ?   1 (RET)
  CS3/PTK[1] PIN (1:CS3, 2:PTK[1])  ?   1 (RET)
  CS4/PTK[2] PIN (1:CS4, 2:PTK[2])  ?   1 (RET)
  CS5/CE1A/PTK[3] PIN (1:CS5/2:CE1A or PTK[3]  ?   1 (RET)
  STATUS[1:0]/PTJ[7:6] PIN (1:STATUS[1:0]/2:PTJ[7:6])  ?   1 (RET)
  WE[2:3]/PTK[6:7] PIN (1:WE[2:3], 2:PTK[6:7]) ?   1 (RET)
  BS/PTK[4] PIN (1:BS, 2:PTK[4])  ?   1 (RET)
  CAS[LL:HH]/PTJ[2:5] PIN (1:CAS[LL:HH], 2PTJ[2:5])  ?   1 (RET)
  CAS[LL:2H]/PTJ[2:3], PTE[3, 6] PIN (1:CAS[LL:2H], 2:PTJ[2:3], PTE[3, 6])  ?  1 (RET)
  JTAG CLOCK (1:16.5MHz/2:8.25MHz/3:4.12MHz)   ?   1 (RET)
  CONFIGURATION STORE OK (Y/N) ?  Y (RET)
START E8000
   S:START E8000
   F:FLASH MEMORY TOOL
   L:SET LAN PARAMETER
   T:START DIAGNOSTIC TEST
         (S/F/L/T)?
```

2. To display the current setting:

```
:MODE (RET)
 MPU NAME=SH7729 MODE=19 (MD5-0 = 3F)
 MONITOR START ADDRESS=A0100000  MONITOR BUS WIDTH=16BIT
 AREA2 MEMORY=SDRAM AREA3 MEMORY=SDRAM
 SDRAM TYPE=1MB-16BIT,4MB-16BIT    DRAM BUS WIDTH=16BIT
 CS2/PTK[0]PIN=CS2  CS3/PTK[1]PIN=CS3  CS4/PTK[2] PIN=CS4
 CS5/CE1A/PTK[3] PIN=CS5   STATUS[1:0]/PTJ[7:6] PIN=STATUS[1:0]
 WE[2:3]/PTK[3] PIN=WE[2:3]  BS/PTK[4] PIN=BS
 CAS[LL:HH]/PTJ[2:5] PIN=CAS{LL:HH]
 CAS[LL:2H]/PTJ[2:3],PTE[3,6] PIN=CAS[LL:2H]
 JTAG CLOCK=16.5MHz
:
```

7.2.28 MOVE [MV] Transfers memory contents

Command Format

- Move data MOVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  Δ<destination address> (RET)

<start address>:	Start address of source area
<end address>:	End address of source area
<number of bytes>:	The number of bytes to be transferred
<destination address>:	Start address of destination

Description

- Move data

— Transfers the contents of the memory area specified with <start address> and <end address> or <number of bytes> to an address range starting with <destination address>. Transfer is usually performed from the <start address>.

— Verifies the transfer. If a verification error occurs,

```
FAILED AT xxxxxxxx     WRITE = yy 'y'   READ = zz 'z'
```

is displayed.

xxxxxxxx:	Address of error
yy 'y':	Write data (hexadecimal and ASCII characters)
zz 'z':	Read data (hexadecimal and ASCII characters)

Notes

1. If areas other than the internal memory areas or areas CS0 to CS6 are included in the destination, transfer is performed to only the internal memory areas and areas CS0 to CS6.
2. Do not transfer memory contents in the monitor program area.

Example

To transfer data in the address range from H'101C to H'10FC to address H'1000:

```
:MV 101C 10FC 1000  (RET)
:
```

7.2.29 PERFORMANCE_ANALYSIS1-8 [PA,1,2,3,4,5,6,7,8] Specifies, cancels, initializes, and displays performance measurement data

Command Format

- Specification PERFORMANCE_ANALYSIS(1/2/3/4/5/6/7/8) Δ<subroutine name>
  Δ<start address>Δ<end address>[ΔTIME=<timeout value>]
  [ΔCOUNT=<count value>];I1[ΔP] (RET)
  (Subroutine execution time measurement mode 1)

  PERFORMANCE_ANALYSIS(1/2/3/4/5/6/7/8) Δ<subroutine name>
  Δ<start address>Δ<end address>[ΔTIME=<timeout value>]
  [ΔCOUNT=<count value>];I2[ΔP] (RET)
  (Subroutine execution time measurement mode 2)

  PERFORMANCE_ANALYSIS(1/3/5/7) Δ<subroutine name>
  Δ<start address range>Δ<end address range>;I3[ΔP] (RET)
  (Subroutine execution time measurement mode 3)
- Cancellation PERFORMANCE_ANALYSIS[(1/2/3/4/5/6/7/8)][Δ]– (RET)
- Initialization PERFORMANCE_ANALYSISΔ;I (RET)
- Display PERFORMANCE_ANALYSIS[Δ;(A/V)] (RET)

n: Subroutine number

<subroutine name>: Name of the subroutine whose execution performance is to be measured

<start address>: Subroutine entry address

<end address>: Subroutine exit address

<timeout value>: Timeout value of execution time measurement. Can be set for only the PERFORMANCE_ANALYSIS1 command.

Display format: xxx[:yy[:zz[:nnnnnn]]]

xxx: Hour
yy: Minute
zz: Second
nnnnnn: Microsecond

Specifiable range:
xxx: 0 to 999
yy: 0 to 59
zz: 0 to 59
nnnnnn: 0 to 999999

<specified count>: Execution count limit. Can be set for only the PERFORMANCE_ANALYSIS1 command.

Specifiable range: H'1 to H'FFFF

<start address range>: Subroutine entry address range

<start address of subroutine entry range>:<end address of subroutine entry range>

<end address range>: Subroutine exit address range

<start address of subroutine exit range>:<end address of subroutine exit range>

P: Specifies the physical address (When omitted, specifies the virtual address. When the VP_MAP table is enabled, the address is translated according to the VP_MAP table. When the VP_MAP table is disabled, the address is translated based on the MMU status at command input.)

I: Initializes performance measurement information.

A: Displays specified subroutine addresses.

V: Displays subroutine execution time and execution count in numerical form. If V is omitted, display is in graph form.

Description

- Specification

— Measures the execution time and count of the specified subroutine during user program execution initiated with the GO command. The following modes can be specified.

a. Subroutine execution time measurement mode 1

Measures the execution time and count of the subroutine defined by <start address> and <end address>. Measurement starts when an address within the range from the start address to the end address is prefetched, halts when an address outside the specified range is prefetched, and restarts when an address within the specified range is prefetched again. The subroutine execution count is incremented every time the subroutine end address is fetched after the start address is passed. The execution time of subroutines called from the specified subroutine is not included in the measurement results.

b. Subroutine execution time measurement mode 2

Measures the execution time and count of the subroutine defined by <start address> and <end address>. Measurement starts when the start address is prefetched and halts when the end address is prefetched. The subroutine execution count is incremented every time the subroutine end address is fetched after the start address is passed. The execution time of subroutines called from the specified subroutine is included in the measurement results.

c. Subroutine execution time measurement mode 3

Measures the execution time and count of the subroutine defined by <start address range> and <end address range>. Measurement starts when an address in the start address range is prefetched and halts when an address in the end address range is prefetched. The subroutine execution count is incremented every time <end address range> is passed after the start address is passed.

Note: The performance analysis is measured by using the address bus value of the prefetch cycle. When the end address is specified for the near address of the instruction next to the branch or delay slot instruction, the measurement is incorrect. Analyze the MCU operation following the cycle which prefetched the branch instruction in the trace bus cycle unit, and do not set the address in the prefetch cycle, which is not executed with the branch instruction, to the end address.

— Table 7.17 lists the measurement modes that can be specified by each PERFORMANCE_ANALYSIS command. When break conditions or trace conditions have been set, subroutines may not be set to their maximum number.

Table 7.17 Measurement Modes for Each Command

Measurement Mode	PA1	PA2	PA3	PA4	PA5	PA6	PA7	PA8
Subroutine execution time measurement mode 1	O	O	O	O	O	O	O	O
Subroutine execution time measurement mode 2	O	O	O	O	O	O	O	O
Subroutine execution time measurement mode 3	O	X	O	X	O	X	O	X

Note: O: Mode can be specified.
X: Mode cannot be specified.

— Up to eight subroutines can be specified when using only subroutine execution time measurement mode 1 or 2 for measurement. However, only up to four subroutines can be specified in subroutine execution time measurement mode 3.

— This command cannot be executed during program execution by the STEP or STEP_OVER command.

— If <timeout value> is specified in the PERFORMANCE_ANALYSIS1 command and the subroutine execution time exceeds the specified timeout value, a break occurs. To enable this, be sure to specify TB as the mode with the GO command.

— If <specified count> is specified in the PERFORMANCE_ANALYSIS1 command and the subroutine execution count reaches the specified count, a break occurs. To enable this, be sure to specify TB as the mode with the GO command.

Note: An execution count that is exceeded is detected when the program passes through the subroutine end address. Consequently, a subroutine execution count that is equivalent to the specified count plus one and the corresponding subroutine execution time are displayed.

- Cancellation

  — Cancels measuring execution performance for the specified subroutine number.

  — If the subroutine number is omitted, all subroutines assigned for execution performance measurement are canceled.

- Initialization

  Clears the current execution time and count for all subroutines, as well as the total run time. The total run time begins to be measured only after a subroutine to be measured by this command is assigned. If no subroutines are assigned, the total run time is not measured.

- Display

  Displays specified subroutine addresses or performance measurement results, in one of the following three formats. If a subroutine name is specified, the subroutine addresses and measurement results are displayed in numerical form or graph form.

— Execution time ratio displayed in graph form. (No option is specified.)

```
: PERFORMANCE_ANALYSIS  (RET)
  NO  NAME   MODE  RATE    0-----10-----20-----30-----40-----50-----60-----70-----80-----90-----100
   1  SUBA    I1    D'10.0%    *****
   2  SUBB    I2    D'20.0%    **********
   3  SUBC    I3    D'20.0%    **********
   5
   6
   7
   8
  (a)  (b)    (c)    (d)        (e)
-----------------------------------------------------------------------------------------------------------
TOTAL RUN-TIME = D'0000H:10M:00S:000020US:250NS     (f)
```

(a) Subroutine number

(b) Subroutine name (up to 8 characters are displayed)

(c) Execution measurement mode

I1: Time measurement mode 1
I2: Time measurement mode 2
I3: Time measurement mode 3

(d) Execution time ratio as a percentage

(e) Execution time ratio in graph form (in units of 2%/asterisk, rounded up)

(f) Total run time displayed as H (hour), M (minutes), S (second), US (microsecond), and NS (nanosecond).

— Execution time ratio displayed. (Option A is specified.)

```
: PERFORMANCE_ANALYSIS ;A  (RET)
 NO   NAME    MODE   ADDRESS
  1   SUBA     I1    00000100     00001FF0    TIME=xxxH:xxM:xxS:xxxxxxUS   COUNT=nnnnnnnn
 (a)   (b)     (c)      (d)          (e)          (f)                          (g)
  2   SUBB     I2    00005000     00007FF0
  3   SUBC     I3    00010000  :  0001008F                                 (h)
                     00020000  :  00020098                                 (i)
  5
  6
  7
  8
-----------------------------------------------------------------------------------------------
TOTAL RUN-TIME = D'0000H:10M:00S:000020US:250NS                            (j)
```

(a) Subroutine number

(b) Subroutine name (up to 8 characters are displayed)

(c) Time measurement mode

I1: Time measurement mode 1

I2: Time measurement mode 2

I3: Time measurement mode 3

(d) Subroutine start address

(e) Subroutine end address

(f) Timeout value (displayed only when the timeout value is set with the TIME option in mode I1 or I2)

(g) Count value (displayed only when the count value is set with the COUNT option in mode I1 or I2)

(h) Start address range in subroutine execution time measurement mode 3

(i) End address range in subroutine execution time measurement mode 3

(j) Total run time

— Execution time and count displayed as numerical values. (Option V is specified.)

```
: PERFORMANCE_ANALYSIS ;V  (RET)
  NO   NAME    MODE   RATE      RUN-TIME                          E-COUNT
   1   SUBA     I2    D'10.0%   D'0000H:00M:05S:001000US:250NS    D'00005
  (a)    (b)    (c)     (d)                  (e)                      (f)
       MAX  D'0000H:00M:05S:001000US:250NS    MIN  D'0000H:00M:05S:001000US:250NS
                (g)                                      (h)
       AVE  D'0000H:00M:05S:001000US:250NS
                (i)
   2   SUBB     I1    D'20.0%   D'0000H:00M:10S:010305US:500NS    D'00010
       AVE  D'0000H:00M:05S:001000US:250NS
   3   SUBC     I3    D'20.0%   D'0000H:00M:10S:010305US:500NS    D'00010
       AVE  D'0000H:00M:05S:001000US:250NS
   5
   6
   7
   8
-----------------------------------------------------------------------------------------
TOTAL RUN-TIME = D'0001H:00M:50S:000020US:250NS                   (i)
```

(a) Subroutine number

(b) Subroutine name (up to 8 characters are displayed)

(c) Time measurement mode

I1: Time measurement mode 1
I2: Time measurement mode 2
I3: Time measurement mode 3

(d) Execution time ratio as a percentage

(e) Execution time

(f) Subroutine maximum execution time (only for the PERFORMANCE_ANALYSIS 1,2,3,4 command in subroutine execution time measurement mode 2 (I2))

(g) Subroutine minimum execution time (only for the PERFORMANCE_ANALYSIS 1,2,3,4 command in subroutine execution time measurement mode 2 (I2))

(h) Subroutine average execution time (only for the PERFORMANCE_ANALYSIS 1,2,3,4 command)

(i) Total run time displayed as H (hour), M (minutes), S (second), US (microsecond), and NS (nanosecond).

Notes

1. According to the TIME option of the EXECUTION_MODE command, the maximum measurable time is 488, 124, or 6 hours, where the minimum measurement time is 1.6 µs, 406 ns, or 20 ns, respectively.
2. When conditions have already been set with the BREAK_CONDITION_C or TRACE_CONDITION_C command, the same command number cannot be set. For example, when a condition has been set with the BREAK_CONDITION_C1 or TRACE_CONDITION_C1 command, the condition cannot be set with the PERFORMANCE_ANALYSIS1 command. If necessary, cancel conditions set with the above commands before setting the break conditions.
3. Only subroutines in the external area can be measured. Subroutines in internal areas such as the X-RAM or Y-RAM areas cannot be measured.

Examples

1. To measure the execution time of subroutines SUBB (H'5000 to H'7FE0) and initialize the performance measurement data:

```
:PA2 SUBB 5000 7FE0 ;I2 (RET)
:PA ;I (RET)
:
```

2. To display addresses of the set subroutines:

```
:PA ;A (RET)
 NO NAME     MODE ADDRESS
  1 SUBA      I1  00000100 00001FF0          COUNT=D'00000
  2 SUBB      I2  00005000 00007FF0
  3 SUBC      I3  00010000:0001008F
                  00020000:00020098
  5
  6
  7
  8
 -------------------------------------------------------------
 TOTAL RUN-TIME = D'0001H:00M:40S:022917US:000NS
```

3. To display execution time ratio in graph form:

```
:PA (RET)
 NO NAME     MODE RATE   0---10---20---30---40---50---60---70---80---90--100
  1 SUBA      I1  D'10.0% *****
  2 SUBB      I2  D'20.0% **********
  3 SUBC      I3  D'20.0% **********
  5
  6
  7
  8
 ---------------------------------------------------------------------------
 TOTAL RUN-TIME = D'0001H:00M:40S:022917US:000NS
```

4. To display execution time and count in numerical form:

```
:PA ;V (RET)
 NO NAME     MODE RATE     RUN-TIME                        E-COUNT
  1 SUBA      I1  D'10.0%  D'0000H:00M:05S:001000US:250NS  D'00005
    AVE D'0000H:00M:05S:001000US:250NS
  2 SUBB      I2  D'20.0%  D'0000H:00M:10S:010305US:500NS  D'00010
    MAX D'0000H:00M:10S:010305US:250NS MIN D'0000H:00M:10S:010305US:250NS
    AVE D'0000H:00M:10S:010305US:250NS
  3 SUBC      I3  D'20.0%  D'0000H:00M:10S:010305US:500NS  D'00010
    AVE D'0000H:00M:10S:010305US:250NS
  5
  6
  7
  8
 -----------------------------------------------------------------
 TOTAL RUN-TIME = D'0001H:00M:40S:022917US:000NS
```

5. To cancel all registered subroutines:

```
:PA - (RET)
:
```

7.2.30 QUIT [Q] Terminates E8000 system program

Command Format

- Termination QUIT (RET)

Description

- Termination

— Terminates the E8000 emulator system program and puts the emulator monitor in command input wait state:

```
:QUIT  (RET)
START  E8000
  S:START  E8000
  F:FLASH  MEMORY  TOOL
  L:SET  LAN  PARAMETER
  T:START  DIAGNOSTIC  TEST
       (S/F/L/T)  ?  _
```

Example

To terminate the emulator system program:

```
:QUIT (RET)
START E8000
 S:START E8000
 F:FLASH MEMORY TOOL
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
    (S/F/L/T) ? _
```

7.2.31 RADIX [RX] Specifies and displays radix for numeric input

Command Format

- Specification RADIXΔ<radix> (RET)
- Display RADIX (RET)

<radix>: Radix to be used for input of numeric values

H:	Hexadecimal (default at system program initiation)
D:	Decimal
Q:	Octal
B:	Binary
X:	Fixed-point value

Description

- Specification

  Specifies the radix used by the emulator to interpret numbers entered on the command line.

  The RADIX command sets the radix to be used for numbers entered simply as numbers. Hexadecimal is used at emulator initiation. Numbers may be entered in any radix at any time, provided that each value is prefixed with the appropriate character.

Table 7.18 Radix Input Example

Radix	Input Example
Binary	B'1010
Octal	Q'2370
Decimal	D'6904
Hexadecimal	H'AF10
Fixed-point value	X’0.6634049566

- Display

  Displays the currently set radix as follows:

  RADIX = Radix character

  Radix character, displayed as one of the following:

  B:BINARY

  Q:OCTAL

  D:DECIMAL

  H:HEXADECIMAL

  X:FIXED POINT

Examples

1. To set the radix to decimal:

```
:RX D  (RET)
:B 10  (RET)                (10 is input in decimal)
:
```

2. To display the current radix:

```
:RADIX  (RET)
 RADIX=D:DECIMAL
:
```

7.2.32 REGISTER [R] Displays register contents

Command Format

- Display REGISTER (RET)

Description

- Display
  Displays contents of all registers.

Note

The DSP registers are displayed when the SH7729 is selected by the MODE command.

Example

To display all register contents (when the SH7729 is selected as the CPU):

```
:R  (RET)
 PC=00005C60 SR=700000F0:-P1B000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 MOD=00000000
 R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:-------------------------GZNVCOBD
 A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
:
```

7.2.33 RESET [RS] Resets MCU

Command Format

- Reset RESET (RET)

Description

- Reset

  Resets the MCU. The system register, control register, general register, and DSP register contents will be reset to the following values:

R0 to R14	: The value before reset	MACH	: The value before reset
R15 (SP)	: The value before reset	MACL	: The value before reset
R0_BANK0 to R7_BANK0	: The value before reset	PC	: H’A0000000
R0_BANK1 to R7_BANK1	: The value before reset	SPC	: The value before reset
SR	: H’700000F0	SSR	: The value before reset
PR	: The value before reset	VBR	: H’00000000
GBR	: The value before reset		
RS, RE	: The value before reset (when SH7729 is selected)		
MOD (MS, ME)	: The value before reset (when SH7729 is selected)		
DSR	: H’00000000 (when SH7729 is selected)		
A0, A1	: The value before reset (when SH7729 is selected)		
A0G, A1G	: The value before reset (when SH7729 is selected)		
M0, M1	: The value before reset (when SH7729 is selected)		
X0, X1	: The value before reset (when SH7729 is selected)		
Y0, Y1	: The value before reset (when SH7729 is selected)		

The internal I/O register contents will also be reset.

Note

In the actual MCU, the initial value of the registers, except for PC, SR, VBR, and DSR, must be set in the user program because the register contents are not defined.

Example

To reset the MCU:

```
:RS  (RET)
 ** RESET BY E8000 !
:
```

7.2.34 RESULT [RT] Displays execution results

Command Format

- Display RESULT (RET)

Description

- Display

  Displays current register contents, execution time, and the GO, STEP, or STEP_OVER command termination cause. The display format is as follows:

```
:RESULT (RET)
–PC=00005C60  SR=700000F0: -P1B000000000000--------IIII00—
–SPC=00000000  SSR=00000000  ASID=00                                                          (a)
–GBR=00000000  VBR=A0000000  MACH=00000000  MACL=00000000  PR=00000000
–RS=00000000  RE=00000000  MOD=00000000
–R0–7   00000000  000000FF  00000011  00000000  00000000  00000000  00000000  00000000
–R8–15  00000000  00000000  00000000  00000000  00000000  00000000  00000000  A00FFE00
–R0_BANK0-R3_BAML0  00000000  00000000  00000000  00000000
–R4_BANK0-R7_BAML0  00000000  00000000  00000000  00000000
–R0_BANK1-R3_BAML1  00000000  00000000  00000000  00000000
–R4_BANK1-R7_BAML1  00000000  00000000  00000000  00000000
–DSR=000000F1:----------------------------------------GZNVCOBD
–A0G=00  A0=00000000  M0=00000000  X0=00000000  Y0=00000000
–A1G=00  A1=00000000  M1=00000000  X1=00000000  Y1=00000000
 RUN-TIME=D'0000H:00M:00S:008932US:000NS                                                      (b)
 +++: <cause of termination>                                                                  (c)
```

(a) The register contents at emulation termination.

Note: RS, RE, MOD, and DSP registers are displayed when SH7729 is selected by the MODE command.

(b) User program execution time in decimal. According to the TIME option of the EXECUTION_MODE command, the maximum measurable time is 488, 124, or 6 hours, where the minimum measurement time is 1.6 µs, 406 ns, or 20 ns, respectively. If the period exceeds the maximum measurable time, it is displayed as *.

(c) Cause of termination.

Note

Displayed register contents show values at program termination, not the current values.

Example

To display execution results:

```
:RT  (RET)
–PC=00005C60 SR=700000F0:-P1B000000000000--------IIII00--
–SPC=00000000 SSR=00000000 ASID=00
–GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
–RS=00000000 RE=00000000 M0D=00000000
–R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
–R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
–R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
–R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
–R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
–R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
–DSR=000000F1:------------------------GZNVCOBD
–A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
–A1G=00 A1=00000000 M1=00000000 X=00000000 Y0=00000000
 RUN-TIME=D'0000H:00M:00S:008932US:020NS
 +++:BREAK POINT
:
```

7.2.35 STATUS [ST] Displays emulator execution status

Command Format

- Display STATUS (RET)

Description

- Display
  Displays emulator execution status in the following format:

```
RADIX=(a)   BREAK=(b)   CLOCK=(c)
HOST=(d)   STEP_INFO=REG:(e)   /A:(f)   /SP:(g)
```

(a) RADIX=xxx: Default input number type

BIN:	Binary
OCT:	Octal
DEC:	Decimal
HEX:	Hexadecimal
FIX:	Fixed-point value

(b) BREAK=D'xxx: Number of breakpoints (decimal)

(c) CLOCK=xxxx: Clock signal type

8.25 MHz:	Emulator internal clock (8.25 MHz)
33 MHz:	Emulator internal clock (33 MHz)
USER:	User system CLOCK signal
X'TAL:	Crystal oscillator clock
CKIO:	CKIO pin input

(d) HOST=x1x2x3x4x5: Interface conditions with serial port

x1: Baud rate (BPS: Bits per second)

1: 2400 BPS 2: 4800 BPS 3: 9600 BPS 4: 19200 BPS 5: 38400 BPS

x2: Data length for one character

8: 8 bits 7: 7 bits

x3: Parity

N: None E: Even O: Odd

x4: Number of stop bits

1: 1 stop bit 2: 2 stop bits

x5: Busy control method

X: X-ON/X-OFF control R: RTS/CTS control

(e) STEP_INFO=REG:x1 x2 x3: Register information displayed with the STEP command

x1 1: Control register (PC, SPC, SR, SSR, PR, GBR, VBR, MACH, MACL, RS, RE, MOD) information is displayed.

Space: No control register (PC, SPC, SR, SSR, PR, GBR, VBR, MACH, MACL, RS, RE, MOD) information is displayed.

x2 2: General register (R0 to R15, R0_BANK0 to R7_BANK0, and R0_BANK1 to R7_BANK1) information is displayed.

Space: No general register (R0 to R15, R0_BANK0 to R7_BANK0, and R0_BANK1 to R7_BANK1) information is displayed.

x3 3: DSP register (DSR, A0, A0G, A1, A1G, M0, M1, X0, X1, Y0, and Y1) information is displayed.

Space: No DSP register (DSR, A0, A0G, A1, A1G, M0, M1, X0, X1, Y0, and Y1) information is displayed.

(f) /A:xxxxxxxx-xxxxxxxx: Memory address range displayed with the STEP command

(g) /SP:xxxxxxxx: Number of bytes used to display the stack contents

Example

To display the emulator execution status:

```
:ST (RET)
 RADIX=HEX  BREAK=D'001  CLOCK=8.25MHz
 HOST=38N1X   STEP_INFO=REG:12/A:00001000-00001FFF  /SP:
:
```

7.2.36 STEP [S] Performs single-step execution

Command Format

- Single step STEP [Δ<number of execution steps>[Δ<start address>]]
  [;[<stop PC>[ΔV=<ASID value>]][Δ<display option>][ΔI]] (RET)

<number of execution steps>: Number of steps to be executed (H'1 to H'FFFFFFFF).

Default: If <stop PC> or <display option> are specified, H'FFFFFFFF is assumed. If not, H'1 is assumed.

<start address>: Start address of single-step execution. Default is the current PC address.

<stop PC>: PC address when single-step execution is terminated.

Default is <number of execution steps>.

<display option>: Specification of instructions to be displayed

J: Displays instructions and register contents only when branch instructions are executed

R: Displays instructions and register contents only within the opening routine

Default: Displays instructions and register contents for all executed instructions

V=<ASID value>: The ASID value of <stop PC>

I: Interrupt is enabled before STEP command execution, and an instruction of the interrupt handler is executed.

Description

- Single step

— Performs single-step execution from <start address> to <stop PC> or from <start address> for <number of execution steps>. The type of emulation performed (described below) depends on the specified parameters and option.

In addition, register and memory contents, address, instruction mnemonic, and termination cause are displayed in the following format:

```
(a) PC=00005C60  SR=700000F0:-P1B000000000000--------IIII00--
    SPC=00000000  SSR=00000000  ASID=00
    GBR=00000000  VBR=A0000000  MACH=00000000  MACL=00000000  PR=00000000
    RS=00000000  RE=00000000  M0D=00000000
    R0-7   00000000  000000FF  00000011  00000000  00000000  00000000  00000000  00000000
    R8-15 00000000  00000000  00000000  00000000  00000000  00000000  00000000  A00FFE00
    R0_BANK0-R3_BANK0  00000000  00000000  00000000  00000000
    R4_BANK0-R7_BANK0  00000000  00000000  00000000  00000000
    R0_BANK1-R3_BANK1  00000000  00000000  00000000  00000000
    R4_BANK1-R7_BANK1  00000000  00000000  00000000  00000000
    DSR=000000F1:-----------------------------------------GZNVCOBD
    A0G=00  A0=00000000  M0=00000000  X0=00000000  Y0=00000000
    A1G=00  A1=00000000  M1=00000000  X1=00000000  Y1=00000000
(b) <address>:<instruction mnemonic>
(c) MEMORY
(d) STACK
(e) +++: <cause of termination>
```

(a) Register information

Note: RS, RE, MOD, and DSP registers are displayed when SH7729 is selected by the MODE command.

(b) Address and mnemonic of the executed instruction

(c) Memory contents display

(d) Stack contents display

(e) Cause of termination (refer to table 7.19)

Information (a) and (c) is displayed according to specifications made with the STEP_INFORMATION command. The termination cause, (d), is displayed only when the STEP command is completed.

Table 7.19 Causes of STEP Command Termination

Message	Termination Cause
BREAK CONDITION UBC1,3	A break condition specified with the BREAK_CONDITION_UBC1,3 command was satisfied.
BREAK CONDITION An	A break condition specified with the BREAK_CONDITION_An command was satisfied (n = 1 to 8).
BREAK CONDITION Bn	A break condition specified with the BREAK_CONDITION_Bn command was satisfied (n = 1 to 8).
BREAK CONDITION Cn	A break condition specified with the BREAK_CONDITION_Cn command was satisfied (n = 1 to 8).
BREAK CONDITION A1, ... ,8	Multiple break conditions specified with the BREAK_CONDITION_A (A1 to A8) commands were satisfied.
BREAK CONDITION B1, ... ,8	Multiple break conditions specified with the BREAK_CONDITION_B (B1 to B8) commands were satisfied.
BREAK CONDITION C1, ... ,8	Multiple break conditions specified with the BREAK_CONDITION_C (C1 to C8) commands were satisfied.
BREAK KEY	The BREAK key or (CTRL) + C keys were pressed for forcible termination.
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed.
RESET BY E8000	The emulator forcibly terminates program execution with the RESETP signal because an error has occurred in the user system.
STEP NORMAL END	The specified number of steps were executed.
STOP ADDRESS	The instruction at <stop PC> was executed.
MULTI BREAK	Multi-break occurred.

— If <stop PC> and <display option> are omitted, instruction mnemonics and register information are displayed for each step executed.

```
:STEP <number of execution steps>  [<start address>]  (RET)
```

— Instruction mnemonics and register information are also displayed for each step when <stop PC> is specified, and single-step emulation is executed until the instruction at <stop PC> is executed.

```
:STEP [<number of execution steps>  [<start address>]]; <stop PC>  (RET)
```

The ASID value of <stop PC> can be specified by specifying V=<ASID value>. When the ASID value is specified, the execution stops when the MCU MMU is enabled and the program counter of the PTEH register and the ASID value match the specified values. When the ASID value is not specified, the execution stops when the program counter matches <stop PC> regardless of the MCU MMU state.

— If the J option is specified, instruction mnemonics and register information are displayed only for branch instructions, and single-step emulation is executed until the instruction at <stop PC> is executed. If <stop PC> is set at the start address of an interrupt, STEP execution may not terminate.

```
:STEP [<number of execution steps>  [<start address>]];[<stop PC>] J (RET)
```

The following instructions are valid when the J option is specified:

BT, BF, BRA, JMP, BSR, JSR, BT/S, BF/S, BRAF, BSRF, RTS, RTE, and TRAPA

— If the R option is specified, instruction mnemonics and register information are displayed only during execution within the opening routine. At that time, single-step execution continues until the instruction at <stop PC> is executed. The jump addresses of branch instructions, such as JSR or BSR, are not displayed. Although this function is similar to the STEP_OVER command function, the latter is recommended because of its faster execution time.

```
:STEP [<number of execution steps>  [<start address>]];[<stop PC>] R (RET)
```

If a break occurs while executing a subroutine with R option specification, the subroutine start address and its instruction mnemonic are displayed.

— No interrupts are accepted during STEP command execution, unless the I option has been specified.

- — After the STEP command has been executed (so long as it was not forcibly terminated), and if no other command has been entered, single-step execution can be continued by simply pressing the (RET) key.
- — Do not specify the monitor program area for <start address> or <stop PC>.

Notes

1. Single-step execution is achieved by using the hardware break function (BREAK_CONDITION_UBC2 command). Accordingly, conditions specified with the BREAK_CONDITION_UBC2 command are invalid when using the STEP command.
2. Software breakpoints specified with the BREAK or BREAK_SEQUENCE command are ignored during single-step execution.
3. If a delayed branch instruction is executed during single-step emulation, single-step execution stops after the instruction immediately following the delayed branch instruction is executed. Therefore, two instruction mnemonics are displayed.
4. If break conditions specified with the BREAK_CONDITION_A,B,C or BREAK_CONDITION_UBC command are satisfied, STEP execution may terminate without executing a single instruction.

Examples

1. To execute a program one step at a time, starting from the address given by the current PC:

```
:S (RET)
 PC=00001002 SR=700000F0:-P1B000000000000--------IIII00—
 SPC=00000000  SSR=00000000  ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000 M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000 M1=00000000 X1=00000000 Y1=00000000
 00001000                  MOV      R0,R1
 +++:STEP NORMAL END
:
```

2. To perform single-step execution from addresses H'1060 to H'1070 with information displayed only for branch instructions:

```
:S  FFFF 1060 ;1070 J (RET)
 PC=0000106A SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 00001064                  JMP       @R0
 00001066                  NOP
 PC=0000106C SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 0000106A                  BT        00001070
```

```
PC=00001070 SR=600000F0:-P1-000000000000--------IIII00--
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
RS=00000000 RE=00000000 M0D=00000000
R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
DSR=000000F1:------------------------GZNVCOBD
A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
0000106E                  NOP
+++:STOP ADDRESS
:
```

7.2.37 STEP_INFORMATION [SI] Specifies and displays information during single-step execution

Command Format

- Specification STEP_INFORMATION[Δ<register information>][ΔA=<start address> [(Δ<end address>/Δ@<number of bytes>)][;[<option>] [(P/V[=<ASID value>])]]] [ΔSP=<stack display byte count>] (RET)
- Display STEP_INFORMATION (RET)

<register information>: Register to be displayed.

1:	Displays PC, SR, PR, GBR, VBR, MACH, MACL, SPC, SSR, ASID, RS, RE, and MOD (RS, RE, and MOD are displayed when the SH7729 is selected)
2:	Displays R0 to R15, R0_BANK0 to R7_BANK0, and R0_BANK1 to R7_BANK1
3:	Displays DSR, A0G, A0, M0, X0, Y0, A1G, A1, M1, X1, and Y1 (when the SH7729 is selected)
ALL:	All register information is output (default at emulator initiation).
–:	No information displayed.
Default:	ALL

<start address>: Start address of memory contents.

<end address>: End address of memory contents. (Default is 16 bytes of memory beginning at <start address>.)

<number of bytes>: Number of bytes of memory contents. (Default is 16 bytes.)

<stack display byte count>: Number of bytes of stack contents.

<option>: Display units

B:	One byte
W:	Two bytes
L:	Four bytes

P: Physical address

V: Virtual address (The address is translated using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is translated using the VP_MAP table when VP_MAP table is enabled, and translated based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Specification

  Displays register information, executed instruction information, memory contents, and cause of termination during STEP and STEP_OVER command execution. This command also selects the register information and memory contents which are to be displayed.

```
(a) PC=0000106A SR=600000F0:-P1-000000000000--------IIII00--

    SPC=00000000 SSR=00000000 ASID=00

    GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000

    RS=00000000 RE=00000000 M0D=00000000

(a) R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000

    R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00

    R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000

    R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000

    R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000

    R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000

(c) DSR=000000F1:------------------------GZNVCOBD

    A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000

    A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000

(d) 00001002              MOV          #00, R0

(e) MEMORY

    0000FF80 00 04 00 FF F0 00 02 00     10 00 02 00 0F 00 00 00  "................"

(f) STACK

    000FFFE0 00 00 00 00 00 00 00 00     00 00 00 00 00 00 00 00  "................"

(g) +++:STEP NORMAL END
```

(a) System and control register information (PC, SR, PR, GBR, VBR, MACH, MACL, SPC, SSR, ASID, RS, RE, and MOD)

(b) General register information (R0 to R15, R0_BANK0 to R7_BANK0, and R0_BANK1 to R7_BANK1)

(c) DSP register (DSR, A0G, A0, M0, X0, Y0, A1G, A1, M1, X1, Y1)

(d) Address and assembler instruction mnemonic of the executed instruction

(e) Memory contents display

(f) Stack contents display

(g) Cause of termination

Notes: RS, RE, MOD and DSP registers are displayed when SH7729 is selected with the MODE command.
Do not specify a memory contents display in the monitor program area.

- Display

  Displays STEP information according to the specified contents.

Examples

1. To display only the contents of system and control registers (PC, SR, PR, GBR, VBR, MACH, MACL, SPC, SSR< ASID, RS< RE< and MOD) during STEP or STEP_OVER command execution:

```
:SI 1 (RET)
:
```

2. To display no register information during STEP or STEP_OVER command execution:

```
:SI - (RET)
:
```

3. To display memory contents from addresses H'FB80 to H'FB87 during STEP or STEP_OVER command execution:

```
:SI A=FB80 FB87 (RET)
:
```

4. To display contents according to the specified display information:

```
:SI (RET)
 PC=0000106A SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:---------------------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 00001002               MOV         #00, R0
 MEMORY
 0000FF80 00 04 00 FF F0 00 02 00     10 00 02 00 0F 00 00 00
"................"
 +++:STEP NORMAL END
:
```

7.2.38 STEP_OVER [SO] Performs single-step execution except for subroutines

Command Format

- Execution STEP_OVER [<start address>][;I] (RET)

<start address>: Start address of single-step execution. Default is the current PC address.

I: Interrupt is enabled during single-step execution

Description

- Execution

— Beginning at <start address>, performs single-step execution of instructions, except for subroutines called by the BSR, JSR, BSRF, or TRAPA instruction. If an instruction other than those called by the BSR, JSR, BSRF, or TRAPA instruction is executed, acts as if the subroutine called by the BSR, JSR, BSRF, or TRAPA instruction is a single instruction. If an instruction other than BSR, JSR, BSRF, or TRAPA is executed, register contents and the executed instruction are shown after each instruction is executed, like in the STEP command.

— If a BSR, JSR, or BSRF instruction is executed, sets a PC break before the instruction following the slot delayed branch instruction for the BSR, JSR, or BSRF instruction, and executes the user program. (The instruction following the slot delayed branch instruction is not executed.)

— During STEP_OVER command execution, register contents can be displayed in the following format. The register information and memory contents are displayed according to the STEP_INFORMATION command specifications.

```
PC=0000106A SR=600000F0:-P1-000000000000--------IIII00--
SPC=00000000 SSR=00000000 ASID=00
GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
RS=00000000 RE=00000000 M0D=00000000
R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
DSR=000000F1:----------------------------------------GZNVCOBD
A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
```

(a) Register information

Note: RS, RE, MOD, and DSP registers are displayed when the SH7729 is selected with the MODE command.

(b) Address and mnemonics of the executed instruction

(c) Memory contents display

(d) Stack contents display

(e) Cause of termination (refer to table 7.20)

— After the STEP_OVER command has been executed (so long as it was not forcibly terminated), and if no other command has been entered, single-step execution can be continued by simply pressing the (RET) key.

— Software breakpoints (specified with the BREAK or BREAK_SEQUENCE command) and hardware break conditions (specified with the BREAK_CONDITION_A,B,C or BREAK_CONDITION_UBC command) are invalid during STEP_OVER command execution.

— Interrupts are not accepted during STEP_OVER command execution, unless the I option is specified.

— If a break occurs during subroutine execution, the address and instruction mnemonics of the instruction calling the subroutine are displayed.

— Do not specify the monitor program area as the start address.

Table 7.20 Causes of STEP_OVER Command Termination

Message	Termination Cause
BREAK KEY	The (CTRL) + C keys were pressed for forcible termination.
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed.
ONE STEP END	Single-step execution was completed.
RESET BY E8000	The emulator forcibly terminates program execution with the RESETP signal because an error occurs in the user system.
SUBROUTINE END	The called subroutine has finished execution.
MULTI BREAK	Multi-break occurred.

Notes

1. When a delayed branch instruction is executed with the STEP_OVER command, execution stops at the instruction immediately following a delayed branch instruction. Therefore, two instruction mnemonics are displayed.
2. Do not use this command when program execution may not return from a subroutine called by a BSR, JSR, BSRF, or TRAPA instruction.

Example

To execute the program one step at a time, starting from the address given by the current PC, and without displaying instructions within the called subroutine:

```
:SO (RET)
 PC=00001002 SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  0000106A 0000106A 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 00001000                MOV       R0,R1
 +++:ONE STEP END
:(RET)
```

```
 PC=00001004 SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 00001002                MOV       #00, R0
 +++:ONE STEP END
:(RET)
  PC=00001008 SR=600000F0:-P1-000000000000--------IIII00--
 SPC=00000000 SSR=00000000 ASID=00
 GBR=00000000 VBR=A0000000 MACH=00000000 MACL=00000000 PR=00000000
 RS=00000000 RE=00000000 M0D=00000000
 R0-7  0000106A 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 A00FFE00
 R0_BANK0-R3_BANK0 00000000 00000000 00000000 00000000
 R4_BANK0-R7_BANK0 00000000 00000000 00000000 00000000
 R0_BANK1-R3_BANK1 00000000 00000000 00000000 00000000
 R4_BANK1-R7_BANK1 00000000 00000000 00000000 00000000
 DSR=000000F1:------------------------GZNVCOBD
 A0G=00 A0=00000000  M0=00000000 X0=00000000 Y0=00000000
 A1G=00 A1=00000000  M1=00000000 X1=00000000 Y1=00000000
 00001004                BSR       00002000      (Branch destination subroutine is not displayed)
 00001006                NOP
 +++:SUBROUTINE END
:
```

7.2.39 TRACE [T] Displays trace information

Command Format

- Display TRACE[∆[–]<start pointer>[:[–]<end pointer>]]
  [;<display information>] (RET)

<start pointer>: Start pointer of trace display. (Default is the PTR option of the TRACE_DISPLAY_MODE command.)

<end pointer>: End pointer of trace display. (Default is the PTR option of the TRACE_DISPLAY_MODE command.)

–: Trace up until the break condition is satisfied is displayed.

(This option is usually necessary, except for displaying trace information during delays when a delay count condition is specified by the BREAK_CONDITION_B or TRACE_CONDITION_B command.)

<display information>: Information to be displayed

B: Displays bus-trace information

A: Displays AUD trace information (branch instruction trace)

S: Displays both bus-trace and AUD trace information

Default: Displays bus-trace information

Description

- Display

— Displays trace information acquired during user program execution. Trace information is displayed for bus-trace, AUD trace, or both, according to the option specified.

— The display range can be specified with pointers in bus-cycle units (bus-cycle pointer) or instruction units (instruction pointer). The pointer value is specified as a relative value from the point where a delay start condition is satisfied (see the following note). Trace information acquired before the delay start condition is satisfied is displayed with a minus (–).

Note: When a delay count condition is specified with the BREAK_CONDITION_B or TRACE_CONDITION_B command, the combination of conditions also specified is handled as a delay start condition. Delay starts to be counted when the delay start condition is satisfied. When no delay start condition has been specified or termination has been caused by another reason, the pointer value will be relative to the latest trace information.

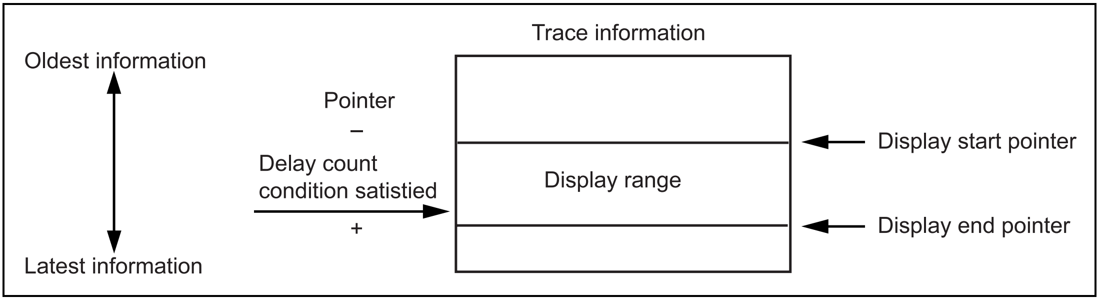


Figure 7.2 Display Range Specified by Pointers in Bus-Trace

The pointer default is as follows:

a. If <start pointer> is omitted, the start pointer specified by the PTR option of the TRACE_DISPLAY_MODE command is used.

b. If <end pointer> is omitted, the end pointer specified by the PTR option of the TRACE_DISPLAY_MODE command is used.

— The display range for AUD trace can be specified with pointers (instruction pointers) in branch-instruction execution units. The pointer value is 0 at the AUD-trace acquisition-stop cycle (the latest information), so a negative value must be specified.

Example: Displays the AUD trace information for the latest 10 instructions from the trace-acquisition stop:

:***T –D’10;A (RET)***

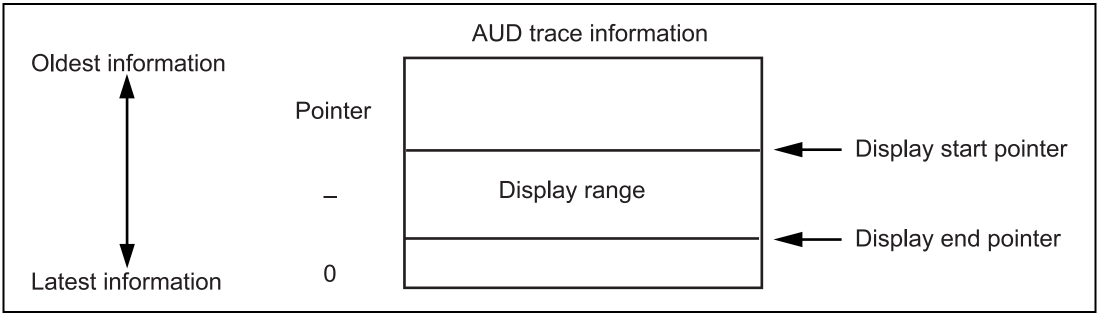


Figure 7.3   Display Range Specified by Pointers in AUD Trace

— To display the trace information of both a bus-trace and an AUD trace, instruction pointers must be specified. To display trace information between 20 instructions to 10 instructions from trace stop:

:***T –D’20:-D’10;S (RET)***

— To display trace information in bus-cycle units, use the following format:

Time Stamp Display:

```
     BP          AB         DB      RW    IRL    NMI   RES   BRQ   VCC   PRB         TIME_STAMP
[-]D'xxxxxx  xxxxxxxx   xxxxxxxx    x   xxxxxx    x     x     x     x   xxxx   xxx  HxxMxxSxxxxxxUxxxN
   (a)         (b)        (c)      (d)   (e)     (f)   (g)   (h)   (i)   (j)   (k)
```

Clock Cycle Display:

```
     BP          AB         DB      RW    IRL    NMI   RES   BRQ   VCC   PRB   CLK
[-]D'xxxxxx  xxxxxxxx   xxxxxxxx    x   xxxxxx    x     x     x     x   xxxx   xx
                                                                               (l)
```

(a) Bus-cycle pointer

Number of bus cycles from an instruction where a delay count condition is satisfied. Although the pointer usually has a negative value (-D'xxxxxx), when a delay count condition is specified as a break or trace condition, the delay will be indicated as a positive value (D'xxxxxx).

(b) Address bus value

(c) Data bus value

(d) Read/write type

R: Read cycle, W: Write cycle

(e) IRL0 to IRL3, IRQ4, and IRQ5 signal level

```
      IRL
x5  x4  x3  x2  x1  x0
```

x5: IRQ5 signal status   x4: IRQ4 signal status
x3: IRL3 signal status   x2: IRL2 signal status
x1: IRL1 signal status   x0: IRL0 signal status

xn 0 : Low level
1 : High level

(f) NMI signal level (0 = low level, 1 = high level)

(g) RESETP signal level (0 = low level, 1 = high level)

(h) BREQ signal level (0 = low level, 1 = high level)

(i) Vcc voltage (0 = Vcc voltage is less than 2.65 V, 1 = Vcc voltage is 2.65 V or more)

(j) External probe signal level (0 = low level, 1 = high level)

(k) Time stamp display

Displayed only when time stamp display is enabled (TIME = E) by the TRACE_DISPLAY_MODE command. The emulator is shipped with time stamp display enabled. The minimum time for the time stamp must be specified (TIME = (1/2/3)) by the TRACE_MODE command. The time cannot be displayed with the number of clock cycles.

(l) The number of clock cycles required from the end of the previous bus cycle to the end of this bus cycle

Up to 255 clock cycles are counted. If the number exceeds 255, it is displayed as **. This information can be displayed when the number of clock cycles is valid (CLK = E) by issuing the TRACE_DISPLAY_MODE command. The emulator is shipped with the display of the number of clock cycles enabled. The TRACE_MODE command must be used to set the number of clock cycles to be acquired (TIME = 0). The number of clock cycles cannot be displayed with the time stamp display.

— To display trace information in bus-cycle units, uses the following format:

IP	TYPE	ADDR	MNEMONIC OPERAND
*-D'xxxxx	xxxxxxxxxxx	xxxxxxxx	xxxxxxxxxxx
(a)	(b)	(c)	(d)

(a): Instruction pointer

This is the relative instruction position originating from the last branch instruction at AUD trace acquisition stop. The pointer value is a negative value (-D'xxxxx) except for the last instruction. Branch source and the branch destination instructions are displayed, but the instruction pointer is displayed only for the branch-source instruction. When the branch-source instruction is not traced, the instruction pointer is displayed for the branch destination.

(b) Address and instruction attribute

BRANCH: Branch source address and branch instruction.

DESTINATION: Branch destination address and branch destination instruction

(c) Instruction address

The address of the branch source or the branch destination. The AUD trace outputs the difference between newly output branch source addresses and the previously output branch source addresses. If the previous branch source address is the same as the upper 16 bits, the lower 16 bits are output. If it matches the upper 24 bits, the lower 8 bits are output. If it matches the upper 28 bits, the lower 4 bits are output. The emulator regenerates the 32-it address from these differences. If the emulator cannot display the 32-bit address, it displays the difference from the previously displayed 32-bit address, and * is displayed for the bit which cannot be displayed.

(d) Instruction mnemonic and operand

The instruction mnemonic and operand for the branch source and branch destination are displayed. However, they are displayed for the cycle where all 32 bits are displayed as the address. They are not displayed in the parallel mode.

— When S is specified for display information, AUD trace information is displayed together with the bus information.

Note: After a cycle for internal area (internal memory) access or a cache hit, AUD trace information may not be correctly displayed with the bus information. In this case, after all of the bus information is displayed, AUD trace information that was not simultaneously obtained will be displayed.

Notes

1. Trace information is not acquired for internal area (internal I/O or internal memory) access cycles.
2. Trace information is not acquired for cycles in which a cache hit occurred.

Examples

1. To specify a display range by bus-cycle pointers, and display bus-cycle information:

```
:T –D’20:-D’10 (RET)
     BP       AB       DB    RW    IRL   NMI RES BRQ VCC PRB       TIME_STAMP
–D'000020 00001082 D10FE000 R 111111  1   1   1   1  1111 000H00M00S000003US660NS
–D'000019 00001084 D10F2102 R 111111  1   1   1   1  1111 000H00M00S000004US240NS
–D'000018 00001086 2102D10F R 111111  1   1   1   1  1111 000H00M00S000004US720NS
–D'000017 00001088 2102412B R 111111  1   1   1   1  1111 000H00M00S000005US300NS
–D'000016 0000108A 412B0009 R 111111  1   1   1   1  1111 000H00M00S000005US780NS
–D'000015 000010C0 412B0000 R 111111  1   1   1   1  1111 000H00M00S000006US360NS
–D'000014 000010C2 00002048 R 111111  1   1   1   1  1111 000H00M00S000006US840NS
–D'000013 0000108C 00000009 R 111111  1   1   1   1  1111 000H00M00S000007US420NS
–D'000012 0000108E 00090009 R 111111  1   1   1   1  1111 000H00M00S000007US920NS
–D'000011 00002048 ****0000 W 111111  1   1   1   1  1111 000H00M00S000008US480NS
–D'000010 0000204A ****0000 W 111111  1   1   1   1  1111 000H00M00S000008US960NS
:
```

2. To display AUD trace information:

```
:T ;A (RET)
    IP      TYPE          ADDR       MNEONIC    OPERAND
*-D'00001 BRANCH        00001000    BRA       00001080
                        00001002    NOP
          DESTINATION   00001080    MOV.L     000010C0,R1
*-D'00000 BRANCH        00001088    JMP       @R1
                        0000108A    NOP
          DESTINATION   00002000    ADD       #01,R2
:
```

3. To display both AUD trace information and bus information from the one branch previous to the trace acquisition stop:

```
:T –1;S (RET)
    IP     TYPE        ADDR         MNEONIC  OPERAND
*-D'00001 BRANCH     00001000     BRA         00001080
                     00001002     NOP
 -D'000025 00001000 1840A03E R   1111  1   1   1   1  1111  00H00M00S000001US040NS
 -D'000024 00001002 A03E0009 R   1111  1   1   1   1  1111  00H00M00S000001US520NS
 -D'000023 00001004 A03E0009 R   1111  1   1   1   1  1111  00H00M00S000002US120NS
 -D'000022 00001006 00090009 R   1111  1   1   1   1  1111  00H00M00S000002US600NS
*-D'00001 DESTINATION 00001080    MOV.L       000010C0,R1
 -D'000021 00001080 0009D10F R   1111  1   1   1   1  1111  00H00M00S000003US180NS
 -D'000020 00001082 D10FE000 R   1111  1   1   1   1  1111  00H00M00S000003US660NS
 -D'000019 00001084 D10F2102 R   1111  1   1   1   1  1111  00H00M00S000004US240NS
 -D'000018 00001086 2102D10F R   1111  1   1   1   1  1111  00H00M00S000004US720NS
*-D'00000 BRANCH     00001088     JMP         @R1
                     0000108A     NOP
 -D'000017 00001088 2102412B R   1111  1   1   1   1  1111  00H00M00S000005US300NS
 -D'000016 0000108A 412B0009 R   1111  1   1   1   1  1111  00H00M00S000005US780NS
 -D'000015 000010C0 412B0000 R   1111  1   1   1   1  1111  00H00M00S000006US360NS
 -D'000014 000010C2 00002048 R   1111  1   1   1   1  1111  00H00M00S000006US840NS
 -D'000013 0000108C 00000009 R   1111  1   1   1   1  1111  00H00M00S000007US420NS
 -D'000012 0000108E 00090009 R   1111  1   1   1   1  1111  00H00M00S000007US920NS
 -D'000011 00002048 ****0000 W   1111  1   1   1   1  1111  00H00M00S000008US480NS
 -D'000010 0000204A ****0000 W   1111  1   1   1   1  1111  00H00M00S000008US960NS
 -D'000009 000010C4 FC000000 R   1111  1   1   1   1  1111  00H00M00S000009US580NS
 -D'000008 000010C6 00002000 R   1111  1   1   1   1  1111  00H00M00S000010US060NS
*-D'00000 DESTINATION 00002000    ADD         #01,R2
 -D'000007 00002000 00007201 R   1111  1   1   1   1  1111  00H00M00S000010US640NS
 -D'000006 00002002 72010009 R   1111  1   1   1   1  1111  00H00M00S000011US120NS
 -D'000005 00002004 72010000 R   1111  1   1   1   1  1111  00H00M00S000011US700NS
 -D'000004 00002006 00000009 R   1111  1   1   1   1  1111  00H00M00S000012US180NS
 -D'000003 00002008 00000009 R   1111  1   1   1   1  1111  00H00M00S000012US780NS
 -D'000002 0000200A 00090009 R   1111  1   1   1   1  1111  00H00M00S000013US280NS
 -D'000001 0000200C 00090009 R   1111  1   1   1   1  1111  00H00M00S000018US220NS
 -D'000000 0000200E 00090009 R   1111  1   1   1   1  1111  00H00M00S000018US780NS
:
```

7.2.40 TRACE_CONDITION_A,B,C [TCA,TCB,TCC] Specifies, displays, and cancels a trace condition

Command Format

```
• Setting        TRACE_CONDITION_(A/B/C)(1/2/3/4/5/6/7/8)Δ<condition>
                                 [[Δ<condition>][Δ<condition>]...];R (RET)
                                                                (Range trace)
                 TRACE_CONDITION_(A/B/C)(1/2/3/4/5/6/7/8)Δ<condition>
                                 [[Δ<condition>][Δ<condition>]...];S (RET)
                                                                (Trace stop)
• Display        TRACE_CONDITION_(A/B/C)[(1/2/3/4/5/6/7/8)] (RET)
                 TRACE_CONDITION_(A/B/C)  (RET)
• Cancellation   TRACE_CONDITION_(A/B/C)[(1/2/3/4/5/6/7/8)] –  (RET)
                 TRACE_CONDITION_(A/B/C) –  (RET)
```

(A/B/C):	Trace condition type
(1/2/3/4/5/6/7/8):	Trace condition number When omitted, all conditions will be displayed or canceled.
<condition>:	Trace conditions to be specified
R:	Range trace mode specification
S:	Trace stop specification

Description

- Setting

— Specifies a trace acquisition condition (trace mode) for user program emulation (GO command execution). The following trace modes can be selected.

Free Trace: Acquires trace information during all bus cycles if no conditions have been set with this command.

Range Trace: Acquires trace information during bus cycles in which the specified condition is satisfied.

Trace Stop: Stops trace information acquisition when the specified condition is satisfied, and enters command input wait state in parallel mode. Though realtime emulation continues, trace information acquisition is not possible in parallel mode. If a trace stop condition is satisfied,

```
** TRACE STOP **
```

is displayed.

— When more than one condition is specified in the range trace mode, the conditions are ORed and the trace information is acquired. When none of the conditions are specified, free trace mode is entered.

— When the specified trace stop condition is satisfied, trace information acquisition stops and the emulator enters parallel mode and waits for command input. To resume trace information acquisition, exit parallel mode with the END command.

— In range trace or trace stop mode, the items shown in tables 7.21 can be specified as <condition> and they can be combined by ANDing them. Several conditions can be specified in any order.

Table 7.21 Specifiable Conditions

Item and Input Format	Description	Commands and modes that can be Set
Address condition A=<address 1>[:<address 2>][;NOT]	When only <address 1> is specified, the condition is satisfied when the address bus value matches the specified value. When both <address 1> and <address 2> are specified, the condition is satisfied when the address bus value is in the range from <address 1> to <address 2>. Condition is satisfied when the address bus value provided by the NOT option setting is not equal to the set value. Only TCB can be set. This condition can be masked.	TCA Range trace Trace stop TCB Range trace Trace stop TCC Range trace Trace stop
Data condition LD=<4-byte value>[;NOT]	The condition is satisfied when the data bus value matches the specified value. Access size cannot be specified; always specify a 4-byte value. To generate a break at a 1-byte or 2-byte value, mask any unnecessary bits. Condition is satisfied when the data bus value provided by the NOT option setting is not equal to the set value. Only TCB can be set. This condition can be masked.	TCA Range trace Trace stop TCB Range trace Trace stop
Read/Write condition R: Read W: Write	The condition is satisfied in a read cycle (R is specified) or a write cycle (W is specified).	TCA Range trace Trace stop TCB Range trace Trace stop

Table 7.21 Specifiable Conditions (cont)

<table>
<tr><th>Item and Input Format</th><th>Description</th><th>Commands and modes that can be Set</th></tr>
<tr><td>External probe condition
PRB=<value&gt;</td><td>The condition is satisfied when all of the emulator's external probe signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to a probe number, as follows:
3 2 1 0 ← Bit
x x x x ← Specified value
| | | |
4 3 2 1 ← Probe number
x: 0 = Low level
1 = High level
This condition can be masked.</td><td>TCA
Range trace
Trace stop
TCB
Range trace
Trace stop</td></tr>
<tr><td>External interrupt condition 1
NMI [:L] or NMI: H</td><td>The condition is satisfied when the NMI signal matches the specified level.
NMI or NMI: L: The condition is satisfied when NMI is low
NMI: H: The condition is satisfied when NMI is high</td><td>TCA
Range trace
Trace stop
TCB
Range trace
Trace stop</td></tr>
<tr><td>External interrupt condition 2
IRL=<value&gt;</td><td>The condition is satisfied when the IRQ5, IRQ4, IRL0 to IRL3 signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to an IRL number, as follows:
5 4 3 2 1 0 ← Bit
x x x x x x ← Specified value
| | | | | | | |
5 4 3 2 1 0 ← IRQ4,5 or IRL[3:0] number
x: 0 = Low level
1 = High level
The condition can be masked.</td><td>TCA
Range trace
Trace stop
TCB
Range trace
Trace stop</td></tr>
</table>

Table 7.21 Specifiable Conditions (cont)

Item and Input Format	Description	Commands and modes that can be Set
Count setting COUNT=<value> <value>: H'1 to H'FFFF	This is set in combination with address, data, read/write, and external probe conditions, external interrupt conditions 1 to 2. The complete condition combination is satisfied when the set condition is satisfied the specified number of times.	TCB Trace stop
Delay count setting DELAY=<value> <value> ... H'1 to H'7FFF	This is set in combination with address, data, read/write, and external probe conditions, external interrupt conditions 1 and 2, and count setting. The complete condition combination is satisfied after the bus cycle of the set value is executed. Only TCB7 can be set.	TCB Trace stop

— A bit mask in 1-bit or 4-bit units can be specified for the address condition of the TRACE_CONDITION_A,B,C command. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). Table 7.22 shows address mask specification examples.

Example: The following condition is satisfied when the lower four bits of the address condition are not specified:

```
: TRACE_CONDITION_A1   A=H'400000* ;S  (RET)
```

Table 7.22 Address Mask Specifications (TRACE_CONDITION_A,B,C)

Radix	Mask Unit	Example	Mask Position
Binary	1 bit	B'01101***	Bits 2 to 0 are masked
Hexadecimal	4 bits	H'F50***	Bits 11 to 0 are masked

Note: When <address 2> is not specified for an address condition, <address 1> can be consecutively masked from the lowest bit. It is not possible to mask any desired bit position, as shown in the following examples.

Examples:

Allowed: TRACE_CONDITION_A1 A = H'10 ;R**

Not allowed: TRACE_CONDITION_A1 A = H'1*00 ;R

TRACE_CONDITION_A1 A = H'100* :10 ;R**

— A bit mask in 1-bit or 4-bit units can be specified for the data, IRL, or PRB condition of the TRACE_CONDITION_A,B,C command. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). Table 7.23 shows these mask specification examples.

Example: The following condition is satisfied when address 4000000 is the address condition and bit 0 is zero in the byte data condition:

```
: TRACE_CONDITION_A1  A=H'4000000 D=B'*******0 ;S  (RET)
```

Table 7.23 Mask Specifications (TRACE_CONDITION_A,B,C)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	Bits 0 and 5 are masked	Data, IRL, or PRB
Hexa-decimal	4 bits	H'F**50	Bits 15 to 8 are masked	Data, IRL, or PRB

— In parallel mode, this command is executed as follows:

Parallel mode is entered by the (RET) key, or the trace stop condition is satisfied:

- This command setting is invalid during parallel mode.
- No trace information is acquired.
- As soon as parallel mode is terminated, this command setting is validated, and trace information acquisition starts. In this case, conditions that have been satisfied are all cleared, and the conditions are rechecked from the beginning. Old trace information is also cleared. At this time,

```
*** 81:TRACE CONDITION RESET
```

  is displayed.

Parallel mode is entered by the (SPACE) key:

- This command setting is valid.
- Trace information is acquired.
- During the following command execution, this command setting is invalid and no trace information is acquired:
  (i) A condition is newly set with the TRACE_CONDITION_A,B,C command
  (ii) TRACE command
  (iii) TRACE_SEARCH command

  As soon as the above command is terminated, this command setting is validated, and trace information acquisition starts. In this case, conditions that have been satisfied are all cleared. Old trace information is also cleared. At this time,

```
*** 81:TRACE CONDITION RESET
```

  is displayed.

- Display

  Displays specified conditions as follows. In addition to condition numbers, character strings that were input for specifying conditions will be displayed as they were input. If no trace condition is specified, a blank is displayed.

```
: TRACE_CONDITION_A (RET)
 TCA1 <A1 trace setting condition>
 TCA2 <A2 trace setting condition>
 TCA3 <A3 trace setting condition>
 TCA4 <A4 trace setting condition>
 TCA5 <A5 trace setting condition>
 TCA6 <A6 trace setting condition>
 TCA7 <A7 trace setting condition>
 TCA8 <A8 trace setting condition>
```

- Cancellation

  Cancels conditions specified with the TRACE_CONDITION_A command.

```
: TRACE_CONDITION_A –  (RET)
```

Notes

1. When conditions have already been set with the BREAK_CONDITION_A,B command, the same command number cannot be set. For example, when a condition has been set with the BREAK_CONDITION_A1 command, the condition cannot be set with the TRACE_CONDITION_A1 command. If necessary, cancel conditions set with the above commands before setting the break conditions.
2. When conditions have already been set with the BREAK_CONDITION_C or PERFORMANCE_ANALYSIS command, the same command number cannot be set. For example, when a condition has been set with the BREAK_CONDITION_C1 or PERFORMANCE_ANALYSIS1 command, the condition cannot be set with the TRACE_CONDITION_C1 command. If necessary, cancel conditions set with the above commands before setting the break conditions.
3. Address condition must only be set in the external area. If it is set in an internal area such as the internal I/O, X-RAM, or Y-RAM area, the condition cannot be set.

Examples

1. To specify a trace stop condition:

```
:TCA1  A=4320 ;S (RET)
:
```

2. To specify a range trace condition:

```
:TCA2  A=2000:27FF ;R (RET)
:
```

3. To display specified trace conditions:

```
:TCA (RET)
 TCA1 A=4320 ;S
 TCA2 A=2000:27FF ;R
 TCA3
 TCA4
 TCA5
 TCA6
 TCA7
 TCA8
:
```

4. To cancel the trace condition specified with the TRACE_CONDITION_A3 command:

```
:TCA3- (RET)
:
```

5. To cancel all trace conditions specified with the TRACE_CONDITION_A command:

```
:TCA- (RET)
:
```

7.2.41 TRACE_DISPLAY_MODE [TDM] — Specifies and displays trace information display mode

Command Format

- Setting TRACE_DISPLAY_MODEΔPTR=[–]<start pointer>
  [:[–]<end pointer>]Δ<display item>=(D/E)
  [[Δ<display item>=(D/E)]...][;C] (RET)
- Display TRACE_DISPLAY_MODE (RET)

<start pointer>: Default start pointer for trace information display and search (emulator shipment: –D'4095)

<end pointer>: Default end pointer for trace information display and search (emulator shipment: D'4095)

<display item>: Information to be displayed at trace information display
A (address bus), D (data bus), RW (read/write),
IRL (IRL signals), NMI (NMI signal),
RES (RESETP signal), BREQ (BREQ signal),
VCC (Vcc voltage state), PRB (external probe),
TIME (time stamp), and CLK (clock cycle)

C: Stores the settings as configuration information in the emulator flash memory

Description

- Setting

— Specifies the default values of start and end pointers for trace information display and search which are used when the pointer values are not specified in the TRACE or TRACE_SEARCH command. Trace information in the emulator is available for approximately 128-kilo bus cycles. Use this command to specify the range of the default values when all trace information is not required. The specified pointers will function as bus-cycle pointers in the TRACE_SEARCH command, and according to the option as instruction or bus-cycle pointers in the TRACE command. The pointer value ranges from –131070 to 131070. When exceeding this range, start and end pointers are automatically specified as –131070 and 131070, respectively.

```
: TRACE_DISPLAY_MODE PTR = –D'2048:D'2048 (RET)
```

— Sets trace items to be displayed as bus-cycle information at trace information display with the TRACE or TRACE_SEARCH command.

```
: TRACE_DISPLAY_MODE∆zzzz = (E/D)  (RET)
```

zzzz: Information to be displayed at trace information display
A, D, RW, IRL, NMI, RES, BREQ, VCC, PRB, TIME, and CLK
E: Display is enabled
D: Display is disabled

Note: TIME and CLK cannot be set as E (display enabled) at the same time.

Table 7.24 shows the default of each trace item display at emulator shipment.

Table 7.24 Shipment Defaults of TRACE_DISPLAY_MODE Command

Trace Items	Default at Shipment
A, D, RW, IRL, NMI, RES, BREQ, VCC, PRB, and TIME	E
CLK	D

— When the C option is specified, the following message is displayed to confirm with the user whether to overwrite the existing configuration information in the emulator flash memory.

```
: TRACE_DISPLAY_MODE ;C  (RET)
: CONFIGURATION STORE OK (Y/N) ?     (a) (RET)
```

(a) Y: Stores the specifications as configuration information in the emulator flash memory. Hereafter, when the emulator is activated, the saved specifications go into effect.

N: Does not overwrite configuration information. The existing specifications are valid.

- Display

  Displays the specified mode as shown below.

```
: TRACE_DISPLAY_MODE (RET)
  PTR = –D'yyyyyy : D'yyyyyy
  DISPLAY ITEM = zzzz zzzz ...
```

yyyyyy: Default values of start and end bus-cycle pointers for trace information display and search

zzzz: Information to be displayed at trace information display

A, D, RW, IRL, NMI, RES, BREQ, VCC, PRB, TIME, and CLK

Notes: 1. To display the number of clock cycles as trace information, the CLK display must be set with this command. Set the number of clock cycles to be acquired (TIME = 0) with the TIME option of the TRACE_MODE command.

2. To display a time stamp with the trace information, TIME display must be set with this command. Set the minimum time for the time stamp (TIME = (1/2/3)) with the TIME option of the TRACE_MODE command.

Examples

1. To set the default values of the pointers to addresses –D'10 and D'10 at trace information display:

```
:TDM PTR=-D'10:D'10 (RET)
:
```

2. To display the specified contents:

```
:TDM (RET)
 PTR=-D'000010:D'000010
 DISPLAY ITEM=A D RW IRL NMI RES BREQ VCC PRB
:
```

3. To specify not to display external probe information (PRB) as bus-cycle information at trace information display with the TRACE or TRACE_SEARCH command:

```
:TDM PRB=D (RET)
:
```

7.2.42 TRACE_MODE [TMO] Specifies and displays trace information acquisition mode

Command Format

- Setting TRACE_MODE [ΔOVFB=(D/E)][ΔTIME=(0/1/2/3)][ΔAUD=(D/R/F)[[:(0/1/2/3)]][ΔSDRAM=(R/E)][;C] (RET)
- Display TRACE_MODE (RET)

OVFB: Specifies whether a break occurs when the trace buffer overflows.

D: A break does not occur when the trace buffer overflows (default at emulator shipment)

E: A break occurs when the trace buffer overflows

TIME: Specifies the minimum time stamp unit.

0: Acquires trace information on the number of clock cycles (CLK) instead of time stamp

1: 20 ns (default at emulator shipment)

2: 1.6 µs

3: 52 µs

AUD: Specifies the acquisition mode of the AUD trace information

D: Does not acquire AUD trace information (default at emulator shipment)

R: Acquires AUD trace information in the realtime trace mode

F: Acquires AUD trace information in the non realtime trace mode

Specifies the clock for acquiring AUD trace information (can be specified when R or F is selected); the AUD clock must be lower than the CPU clock, and higher than a quarter of the CPU clock.

0: 20 MHz (AUD trace information acquisition clock)

1: 33 MHz (AUD trace information acquisition clock)

2: 66 MHz (AUD trace information acquisition clock)

3: Half of the CKIO clock (AUD trace information acquisition clock)

Note: If the MCU clock mode is not 7, do not specify 3.

SDRAM: SDRAM access cycle

R: The address bus and data bus in the SDRAM access cycle are not edited

E: The address bus and data bus in the SDRAM access cycle are edited (default at emulator shipment)

C: Stores the settings as configuration information in the emulator flash memory.

Description

- Specification

— Specifies whether or not to generate a break when the trace buffer overflows.

- To generate a break when the trace buffer overflows:

  : ***TRACE_MODE OVFB=E (RET)***

- To not generate a break when the trace buffer overflows:

  : ***TRACE_MODE OVFB=D (RET)***

— Specifies minimum time stamp unit.

- To acquire trace information on the number of clock cycles. The time stamp is not acquired.

  : ***TRACE_MODE TIME=0 (RET)***
- To set the minimum time stamp unit to 20 ns:

  : ***TRACE_MODE TIME=1 (RET)***
- To set the minimum time stamp unit to 1.6 µs:

  : ***TRACE_MODE TIME=2 (RET)***
- To set the minimum time stamp unit to 52 µs:

  : ***TRACE_MODE TIME=3 (RET)***

— Specifies the acquisition mode for the AUD trace information.

:***TRACE_MODE AUD=x[:y] (RET)***

x: AUD trace information acquisition mode

D: Does not acquire trace information

R: Realtime trace mode

F: Non realtime trace mode

y: Clock for AUD trace information acquisition

0: 20 MHz

1: 33 MHz

2: 66 MHz

3: Half of the CKIO clock

— Specifies whether the address bus and data bus in the SDRAM access cycle is edited before display.

- To not edit the address bus and data bus in the SDRAM cycle before display:

```
: TRACE_MODE  SDRAM=R  (RET)
```

- To edit the address bus and data bus in the SDRAM cycle before display:

```
: TRACE_MODE  SDRAM=E  (RET)
```

- Display

Displays the specified trace mode in the following format:

```
: TRACE_MODE (RET)
 OVFB=a    TIME=bbbbb    AUD=ccccccc:dddddd    SDRAM=eeee
```

a: A break occurs or does not occur when the trace buffer overflows

D: A break does not occur

E: A break occurs

bbbbb: Minimum time stamp unit

CLK: The number of clock cycles is displayed. Trace information on the time stamp is not acquired.

20ns: The minimum acquisition time for the time stamp is 20-ns units.

1.6µs: The minimum acquisition time for the time stamp is 1.6-µs units.

52us: The minimum acquisition time for the time stamp is 52-µs units.

ccccccc: AUD trace information acquisition mode

DISABLE: Does not acquire AUD trace information (The minimum acquisition time is not displayed.)

REAL: Acquires AUD trace information in the realtime mode

FULL: Acquires AUD trace information in non realtime mode

dddddd: Clock for acquiring AUD trace information

20MHz: AUD trace information acquisition clock is 20 MHz

33 MHz: AUD trace information acquisition clock is 33 MHz

66 MHz: AUD trace information acquisition clock is 66 MHz

CKIO/2: AUD trace information acquisition clock is half of CKIO

eeee: Edits or does not edit before displaying the address bus or data bus in the SDRAM access cycle

REAL: Does not edit before displaying address bus or data bus in the SDRAM access cycle

EDIT: Edits before displaying the address bus or data bus in SDRAM access cycle

Note: The AUD trace acquisition clock must satisfy the following condition with respect to the MCU internal clock.

Internal clock/4 ≤ AUD trace acquisition clock ≤ internal clock

Examples

1. To set the minimum time stamp unit to 20 ns:

```
:TMO TIME=1 (RET)
:
```

2. To set the minimum time stamp unit to 1.6 μs and store the set configuration information in the emulator flash memory:

```
:TMO TIME=2 ;C (RET)
 CONFIGURATION STORE OK (Y/N) ? Y(RET)
:
```

3. To set the AUD trace information acquisition mode to realtime mode, and acquisition clock to 33 MHz:

```
:TMO AUD=R:1 (RET)
:
```

4. To display the specified contents:

```
:TMO (RET)
 OVFB=D TIME=1.6ns AUD=REAL :33MHz SDRAM=EDIT
:
```

7.2.43 TRACE_SEARCH [TS] Searches for and displays trace information

Command Format

- Search and display TRACE_SEARCH[Δ<condition>[Δ<condition>...]
  [;[–] <start bus-cycle pointer>[:[–]<end bus-cycle pointer>] [L]]] (RET)

<condition>: Condition governing trace information to be searched for or displayed. If this is omitted, the number of bus cycles and the number of instructions in the trace buffer are displayed.

–: Specified when searching for trace information acquired before the trace or break condition has been satisfied. (This option is usually necessary, except for displaying trace information during delays when a delay count condition is specified by the BREAK_CONDITION_B or TRACE_CONDITION_B command.)

<start bus-cycle pointer>: Start pointer of bus cycle to be searched for or displayed.

<end bus-cycle pointer>: End pointer of bus cycle to be searched for or displayed.

If both <start bus-cycle pointer> and <end bus-cycle pointer> are omitted, bus cycles are searched for or displayed according to the pointers specified with the TRACE_DISPLAY_MODE command.

L: Displays the last bus-cycle information to be searched for.

Description

- Search and display

— Searches for information in the trace buffer under the specified conditions, and displays all applicable bus-cycle information. If <start bus-cycle pointer> and <end bus-cycle pointer> are specified, searches for and displays the bus-cycle information between <start bus-cycle pointer> and <end bus-cycle pointer>. Trace information is displayed in the same format as the bus-cycle information display by the TRACE command.

— If no conditions are specified, the number of bus cycles and instructions saved in the trace buffer are displayed.

```
: TRACE_SEARCH (RET)
 BUS-CYCLE NUMBER = D'xxxxxx
```

xxxxxx: Number of bus cycles (decimal)

- If the L option is specified, displays only the last bus-cycle information to be searched for.

- Items listed in table 7.25 can be specified for <condition>, and they can be combined by ANDing them.

Table 7.25 Specifiable Conditions (TRACE_SEARCH)

Item and Input Format	Description
Address condition A=<address 1>[:<address 2>][;P]	When only <address 1> is specified, the condition is satisfied when the address bus value matches the specified value. When both <address 1> and <address 2> are specified, the condition is satisfied when the address bus value is in the range from <address 1> to <address 2>. When ;P is specified, the specified <address 1> and <address 2> are searched for by the physical address. When the VP_MAP table is enabled, the address is translated using the VP_MAP table, and when the table is disabled, the address is translated based on the MMU status at command input. This condition can be masked.
Data condition LD=<4-byte value>	The condition is satisfied when the data bus value matches the specified value. The access size cannot be specified. Always specify a 4-byte value. To trace in 1-byte or 2-byte units, mask any unnecessary bits. This condition can be masked.
Read/Write condition R: Read W: Write	The condition is satisfied in a read cycle (R is specified) or a write cycle (W is specified).
External probe condition PRB=<value>	The condition is satisfied when all of the emulator's external probe signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to a probe number, as follows: 3 2 1 0 ← Bit x x x x ← Specified value | | | | 4 3 2 1 ← Probe number x: 0 = Low level 1 = High level This condition can be masked.

Table 7.25 Specifiable Conditions (TRACE_SEARCH) (cont)

Item and Input Format	Description
External interrupt condition 1 NMI [:L] or NMI: H	The condition is satisfied when the NMI signal matches the specified level. NMI or NMI: L: The condition is satisfied when NMI is low NMI: H: The condition is satisfied when NMI is high
External interrupt condition 2 IRL=<value>	The condition is satisfied when all of the IRL 0-3 and IRQ4 and IRQ5 signals match the specified values. Specify <value> as 1-byte data. Each bit corresponds to an IRQ number, as follows: 5 4 3 2 1 0 ← Bit x x x x x x ← Specified value | | | | | | 5 4 3 2 1 0 ← IRL[3:0], IRQ4,5 number x: 0 = Low level 1 = High level The condition can be masked.
RESETP condition RES	Searches for a bus cycle in which the RESETP signal is low.
Time stamp TS=<elapsed time 1> [Δ<elapsed time 2>]	Searches for the specified elapsed time. When only <elapsed time 1> is specified, searches for the time specified with <elapsed time 1>. When both <elapsed time 1> and <elapsed time 2> are specified, searches for the time range specified with <elapsed time 1> and <elapsed time 2>. <elapsed time 1> = hhh[:mm[:ss[:uuuuuu]]] <elapsed time 2> = hhh[:mm[:ss[:uuuuuu]]] hhh: Hour mm: Minute ss: Second uuuuuu: Microsecond

— A bit mask in 1-bit or 4-bit units can be specified for address, data, IRL, or PRB condition. When a bit is masked, the condition is satisfied irrespective of its bit value. To implement the mask, specify each digit to be masked at input as an asterisk (*). Table 7.26 shows mask specification examples.

Example: To search for a bus cycle where bit 0 is zero in the byte data condition:

```
: TRACE_SEARCH  A=4000000   D=B'*******0 (RET)
```

Table 7.26 Mask Specifications (TRACE_SEARCH)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	Bits 0 and 5 are masked	Address, data, IRL, or PRB
Hexa-decimal	4 bits	H'F**50	Bits 15 to 8 are masked	Address, data, IRL, or PRB

— The display contents are the same as the bus-cycle display of the TRACE command. However, instruction mnemonics are not displayed.

— If no trace information satisfies the specified condition,

```
*** 45: NOT FOUND
```

is displayed.

— If there is no trace information in the trace buffer,

```
*** 39: BUFFER EMPTY
```

is displayed.

Examples

1. To search for bus cycles where data is written to addresses from H'10000 to H'10050:

```
:TS A=10000:10050 W (RET)
     BP       AB       DB    RW IRL   NMI RES BRQ VCC PRB
 -D'000063 00010003 ******44 W 111111  1   1   1   1  1111
 -D'000062 00010022 ****3344 W 111111  1   1   1   1  1111
 -D'000060 00010040 11223344 W 111111  1   1   1   1  1111
:
```

2. To search for the last bus cycle where IRL0 is low:

```
:TS IRL=B'***0 ;L (RET)
     BP       AB       DB   RW   IRL   NMI RES BRQ VCC PRB
 -D'000063 00010003 ******44 W  111111  1   1   1   1  1111
:
```

7.2.44 VP_MAP [VP] Sets, displays, and cancels emulator’s address translation table

Command Format

- Enable/disable: VP_MAPΔ;(E/D) (RET)
- Setting: VP_MAPΔ<virtual start address>Δ<virtual end address>
  Δ<physical start address> (RET)
- Display: VP_MAP (RET)
- Cancellation: VP_MAP[Δ]-[<cancellation virtual start address>] (RET)

E: Enables address translation table

D: Disables address translation table (setting at emulator initiation)

<virtual start address>: Start address of the virtual area

<virtual end address>: End address of the virtual area

<physical start address>: Start address of the physical area

<cancellation virtual start address>: Virtual start address of the address translation table to be cancelled

Description

- Enable/disable

— Enables or disables the address translation table.

```
: VP_MAPΔ;x (RET)
```

x E: Enables the address translation table.
D: Disables the address translation table.

When the address translation table is enabled, the load module address (the virtual address) is translated to the address at which actual memory exists (the physical address) when the user program is loaded. When the address translation table is disabled, the address is not translated.

The address translation is performed for the following commands:

ASSEMBLE, BREAK, BREAK_CONDITION_A,B,C, BREAK_SEQUENCE, DATA_CHANGE, DATA_SEARCH, DISASSEMBLE, DUMP, FILL, MEMORY, STEP_INFORMATION, INTFC_LOAD, INTFC_SAVE, INTFC_VERIFY, LOAD, SAVE, VERIFY, LAN_LOAD, LAN_SAVE, and LAN_VERIFY

Section 8 Data Transfer from Host Computer Connected by RS-232C Interface

8.1 Overview

When the emulator is connected to a host computer by the RS-232C interface, data can be transferred between the host computer and the emulator or between the host computer and memory in the user system connected to the emulator. This enables the following transmission of host computer load module files:

- Loads a load module file in the host computer to user system memory
- Saves data in the user system memory as a load module file in the host computer

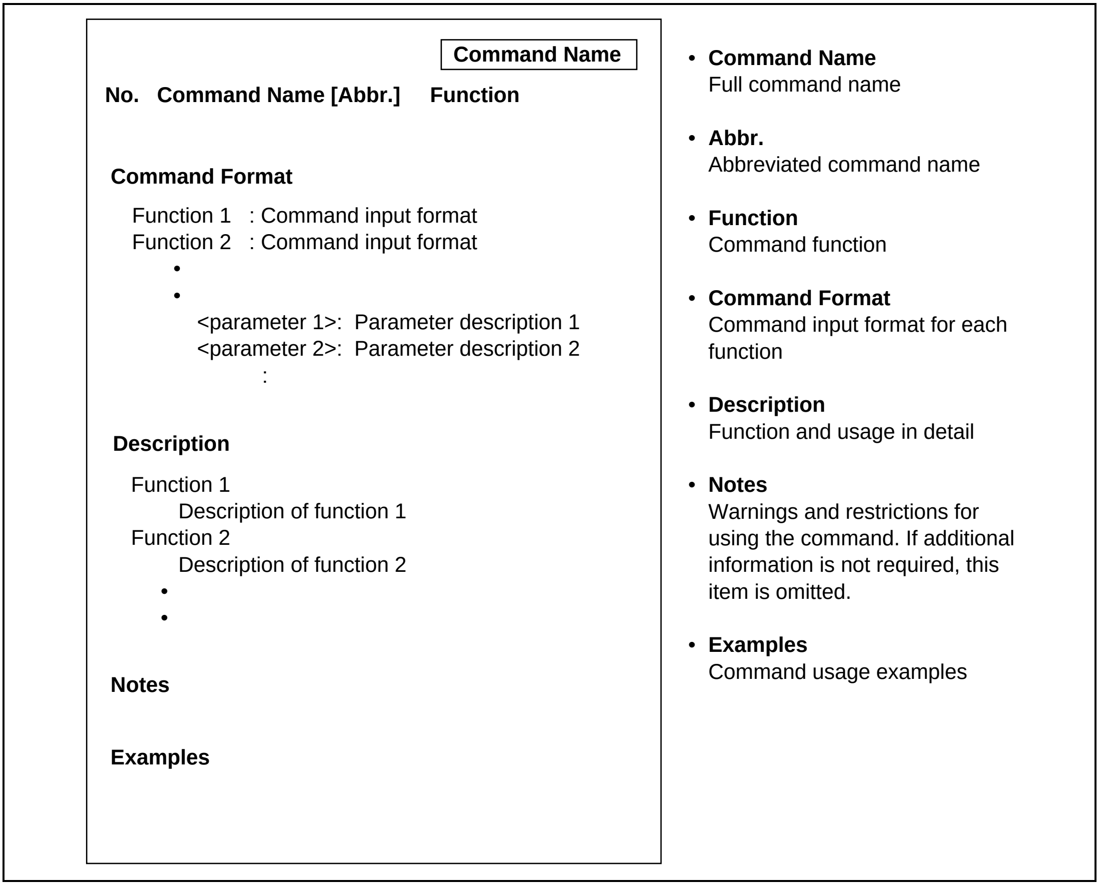
Commands listed in table 8.1 can be used to transfer data between the emulator and host computer.

Table 8.1 Host-Computer Related Commands

Command	Function	Usable/Unusable in Parallel Mode
INTFC_LOAD	Loads program from host computer. — Serial interface	Unusable
INTFC_SAVE	Saves program in host computer. — Serial interface	Unusable
INTFC_VERIFY	Verifies memory contents against host computer file. — Serial interface	Unusable
LOAD	Loads program from host computer. — Bidirectional parallel interface	Unusable
SAVE	Saves program in host computer. — Bidirectional parallel interface	Unusable
VERIFY	Verifies memory contents against host computer file. — Bidirectional parallel interface	Unusable

8.2 Host-Computer Related Commands

This section provides details of host-computer related commands in the format shown in figure 8.1.



Command Name

No. Command Name [Abbr.] Function

Command Format

Function 1 : Command input format
Function 2 : Command input format
•
•
<parameter 1>: Parameter description 1
<parameter 2>: Parameter description 2
:

Description

Function 1
Description of function 1
Function 2
Description of function 2
•
•

Notes

Examples

- **Command Name**
  Full command name
- **Abbr.**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and restrictions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 8.1 Description Format of Host-Computer Related Command

Symbols used in the command format have the following meanings:

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

(a/b): One of the parameters enclosed by ( ) and separated by /, that is, either a or b must be specified.

< >: Contents shown in < > are to be specified or displayed.

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

Δ: Indicates a space. Used only for command format description.

(RET): Pressing the (RET) key.

Although italic and bold characters are used throughout this manual to indicate input, it is not used in the command format parts of these descriptions.

8.2.1 INTFC_LOAD [IL] Loads program from host computer — Serial interface

Command Format

- Load INTFC_LOAD[Δ<offset>][;<load module type>]
  [(P/V)[=<ASID value>]]:<file name> (RET)

<offset>:	Value to be added to the load module address	
<load module type>:	Load module type	
	R:	SYSROF-type load module
	S:	S-type load module
	H:	HEX-type load module
	M:	Memory image file
	E:	ELF-type load module
	Default:	SYSROF-type load module
<file name>:	File name in the host computer	
P:	Physical address	
V:	Virtual address (The address is converted using the MCU TLB at command input)	
<ASID value>:	ASID value (can be specified only when V is specified)	

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Load

— Loads a user program from the host computer into user system memory via the serial interface. Use interface software IPW for the host computer.

:***INTFC_LOAD[;<load module type>]:<file name> (RET)***

When loading is completed, the start and end addresses are displayed as follows:

```
TOP ADDRESS = <start address>
END ADDRESS = <end address>
```

— An offset (value to be added) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

```
:INTFC_LOAD <offset>;S :<file name> (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

— A physical or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. The load module can be loaded only to the internal memory areas or areas CS0 to CS6.
2. Verification is not performed during load. If the program must be verified, use the INTFC_VERIFY command. For details, refer to section 8.2.3, INTFC_VERIFY.

Examples

1. To load SYSROF-type load module F11.ABS:

```
:IL :F11.ABS (RET)
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

2. To load S-type load module ST.MOT:

```
:IL ;S :ST.MOT(RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

8.2.2 INTFC_SAVE [IS] Saves program in host computer — Serial interface

Command Format

- Save INTFC_SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  [;[<load module type>][(P/V)[=<ASID value>]] [ΔLF]]:<file name> (RET)

<start address>: Start memory address

<end address>: End memory address

<number of bytes>: Number of bytes to be saved

<load module type>: Load module type

S:	S-type load module
H:	HEX-type load module
Default:	S-type load module

LF: Adds an LF code (H'0A) to the end of each record

<file name>: File name in the host computer

P: Physical address

V: Virtual address (The address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Save

  — Saves the specified memory contents in the specified load module type in a host computer file via the serial interface. Use interface software IPW for the host computer. An S-type or HEX-type load module can be saved. An SYSROF-type or ELF-type load module cannot be saved.

  ```
  :INTFC_SAVE <start address> <end address>[;<load module type>]
                                              :<file name> (RET)
  ```

  When save is completed, the start and end memory addresses are displayed as follows:

  ```
  TOP ADDRESS = <start address>
  END ADDRESS = <end address>
  ```

  — When the LF option is specified, the emulator adds an LF code (H'0A) to the end of each record in addition to a CR code (H'0D) in the S-type or HEX-type load module.

  — A physical address or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. Data can be saved only in the internal memory areas or areas CS0 to CS6.
2. Verification is not performed after save. If the program must be verified, use the INTFC_VERIFY command. For details, refer to section 8.2.3, INTFC_VERIFY.
3. If the specified file name already exists, an overwrite confirmation message is displayed. If N is entered to halt save, some unnecessary characters may be output to the following line.

Example

To save memory contents in the address range from H'7000 to H'7FFF in host computer file F11.MOT in the S-type load module format:

```
:IS 7000 7FFF :F11.MOT (RET)
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

8.2.3 INTFC_VERIFY [IV] Verifies memory contents against host computer file — Serial interface

Command Format

- Verification INTFC_VERIFY [Δ<offset>][;<load module type>]
  [(P/V)[=<ASID value>]]:<file name> (RET)

<offset>: Value to be added to the address

<load module type>: Load module type

R: SYSROF-type load module

S: S-type load module

H: HEX-type load module

M: Memory image file

E: ELF-type load module

Default: SYSROF-type load module

<file name>: File name in the host computer

P: Physical address

V: Virtual address (The address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Verification

— Verifies data transferred from the host computer against data in memory via the serial interface. Use interface software IPW for the host computer.

:***INTFC_VERIFY[;<load module type>]:<file name> (RET)***

— If a verification error occurs, the address and its contents are displayed as follows:

```
<ADDR>      <FILE>      <MEM>
xxxxxxxx     yy 'y'       zz 'z'
```

xxxxxxxx: Verification error address

yy 'y': Load module data (in hexadecimal and ASCII characters)

zz 'z': Memory data (in hexadecimal and ASCII characters)

— An offset (value to be added or subtracted) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

:***INTFC_VERIFY <offset> ;S :<file name> (RET)***

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

— A physical address or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Note

Data can be verified only in the internal memory areas or areas CS0 to CS6.

Example

To verify SYSROF-type load module F1.ABS against the memory contents:

```
:IV :F1.ABS (RET)
 <ADDR>     <FILE>    <MEM>
 00001012    31'1'     00'.'
:
```

8.2.4 LOAD [L] Loads program from host computer — Bidirectional parallel interface

Command Format

- Load LOAD[Δ<offset>][;<load module type>][(P/V)[=<ASID value>]] :<file name> (RET)

<offset>: Value to be added to the load module address
<load module type>: Load module type
R: SYSROF-type load module
S: S-type load module
H: HEX-type load module
M: Memory image file
E: ELF-type load module
Default: SYSROF-type load module
<file name>: File name in the host computer
P: Physical address
V: Virtual address (The address is converted using the MCU TLB at command input)
<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Load

— Loads a user program from the host computer into user system memory via the bidirectional parallel interface. Use interface software IPW for the host computer to transfer the specified file to the emulator via the bidirectional parallel interface. Enter #B• before the command to request data output to the host computer.

```
:#B LOAD[;<load module type>]:<file name> (RET)
```

When loading is completed, the start and end addresses are displayed as follows:

```
TOP ADDRESS = <start address>
END ADDRESS = <end address>
```

— An offset (value to be added) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

```
:#B LOAD <offset>;S :<file name> (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

— A physical address or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. The load module can be loaded only to the internal memory areas or areas CS0 to CS6.
2. Verification is not performed during load. If the program must be verified, use the VERIFY command. For details, refer to section 8.2.6, VERIFY.

Examples

1. To load SYSROF-type load module F11.ABS:

```
:#B L:F11.ABS (RET)
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

2. To load S-type load module ST.MOT:

```
:#B L;S:ST.MOT (RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

8.2.5 SAVE [SV] Saves program in host computer — Bidirectional parallel interface

Command Format

- Save SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  [;[<load module type>][(P/V)[=<ASID value>]][ΔLF]]:<file name> (RET)

<start address>:	Start memory address
<end address>:	End memory address
<number of bytes>:	Number of bytes to be saved
<load module type>:	Load module type
	S: S-type load module
	H: HEX-type load module
	Default: S-type load module
LF:	Adds an LF code (H'0A) to the end of each record
<file name>:	File name in the host computer
P:	Physical address
V:	Virtual address (The address is converted using the MCU TLB at command input)
<ASID value>:	ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Save

— Saves the specified memory contents in the specified load module type in a host computer file via the bidirectional parallel interface. Use interface software IPW for the host computer. An S-type or HEX-type load module can be saved. An SYSROF-type or ELF-type load module cannot be saved. Enter #NΔ before the command to request data receipt to the host computer.

```
:#N SAVE <start address> <end address>[;<load module type>]:
                                                  <file name> (RET)
```

When save is completed, the start and end memory addresses are displayed as follows:

```
TOP ADDRESS = <start address>
END ADDRESS = <end address>
```

— When the LF option is specified, the emulator adds an LF code (H'0A) to the end of each record in addition to a CR code (H'0D) in the S-type or HEX-type load module.

— A physical address or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. Data can be saved only in the internal memory areas or areas CS0 to CS6.
2. Verification is not performed after save. If the program must be verified, use the VERIFY command. For details, refer to section 8.2.6, VERIFY.

Example

To save memory contents in the address range from H'7000 to H'7FFF in host computer file F11.MOT in the S-type load module format:

```
:#N SV 7000 7FFF :F11.MOT (RET)
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

8.2.6 VERIFY [V] Verifies memory contents against host computer file — Bidirectional parallel interface

Command Format

- Verification VERIFY [Δ<offset>][;<load module type>]
  [(P/V)[=<ASID value>]]:<file name> (RET)

<offset>:	Value to be added to the address	
<load module type>:	Load module type	
	R:	SYSROF-type load module
	S:	S-type load module
	H:	HEX-type load module
	M:	Memory image file
	E:	ELF-type load module
	Default:	SYSROF-type load module
<file name>:	File name in the host computer	
P:	Physical address	
V:	Virtual address (The address is converted using the MCU TLB at command input)	
<ASID value>:	ASID value (can be specified only when V is specified)	

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Verification

— Verifies data transferred from the host computer against data in memory via the bidirectional parallel interface. Use interface software IPW for the host computer. Enter #BΔ before the command to request data output to the host computer.

```
:#B VERIFY[;<load module type>]:<file name> (RET)
```

— If a verification error occurs, the address and its contents are displayed as follows:

```
<ADDR>      <FILE>      <MEM>
xxxxxxxx     yy 'y'       zz 'z'
```

xxxxxxxx: Verification error address

yy 'y': Load module data (in hexadecimal and ASCII characters)

zz 'z': Memory data (in hexadecimal and ASCII characters)

— An offset (value to be added or subtracted) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

```
:#B VERIFY <offset> ;S :<file name> (RET)
```

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

— A physical address or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Note

Data can be verified only in the internal memory areas or areas CS0 to CS6.

Example

To verify SYSROF-type load module F1.ABS against the memory contents:

```
:#B V:F1.ABS (RET)
 <ADDR>     <FILE>    <MEM>
 00001012    31'1'     00'.'
:
```

Section 9 Data Transfer from Host Computer Connected by LAN Interface

9.1 Overview

The optional LAN board supports the FTP client function. This function enables the following data transfer between the emulator and the host computer connected via the LAN interface.

- Loads a load module file in the host computer to user system memory
- Saves data in the user system memory as a load module file in the host computer
- Transfers files between the emulator and host computer

The emulator supports the LAN commands listed in table 9.1 to transfer data between the emulator and the host computer. These commands are explained in section 9.3, LAN Commands.

Table 9.1 LAN Commands

Command	Function	Usable/Unusable in Parallel Mode
ASC	Specifies the file type to be transferred as ASCII	Usable
BIN	Specifies the file type to be transferred as binary	Usable
BYE	Terminates the FTP interface (Re-connects the FTP interface with the FTP command)	Usable
CD	Changes the directory of the FTP server	Usable
CLOSE	Disconnects the host computer from the FTP interface (Re-connects the host computer to the FTP interface with the OPEN command)	Usable
FTP	Connects the host computer and emulator via the FTP interface	Usable
LAN	Displays emulator IP address	Usable
LAN_HOST	Displays all defined host computers	Usable
LAN_LOAD	Loads a load module file from the host computer to memory via the FTP interface	Unusable
LAN_SAVE	Saves the specified memory contents in the host computer connected via the FTP interface	Unusable
LAN_VERIFY	Verifies memory contents against the host computer file connected via the FTP interface	Unusable
LS	Displays the host computer directory connected via the FTP interface	Usable
OPEN	Connects the host computer to the FTP interface	Usable
PWD	Displays the current directory name of the host computer connected via the FTP interface	Usable
ROUTER	Displays routing information	Usable
STA	Displays the type of file to be transferred	Usable
SUBNET	Displays the subnet mask value	Usable
LOGOUT	Disconnects from the TELNET*	Usable

Note: The optional LAN board supports the TELNET server function in addition to the FTP client function. When the emulator is connected to the host computer through TELNET, the emulator can be disconnected from the TELNET with the LOGOUT command. For details on the TELNET interface, refer to section 3.5.1, Power-On Procedure for LAN Interface, in Part I, E8000 Guide. Note that the FTP can be connected via TELNET or the RS-232C interface.

9.2 LAN Data Transfer

9.2.1 Setting the Data Transfer Environment

The optional LAN board enables data transfer between the emulator and host computer via the FTP interface. The transfer environment must be specified before starting data transfer as follows. Note that the optional LAN board supports the FTP client function only.

Procedure:

1. Specify the host computer environment, including the host computer name and IP address, to the network database of the host computer as the host information. For details, refer to the appropriate host computer’s user’s manual.

2. Specify the following emulator environment:

   a. Emulator IP address

      Specify the emulator IP address with the emulator monitor command L. Since the emulator IP address is written to the emulator flash memory, it needs not to be written each time the LAN interface is used. The emulator IP address can be modified as required.

   b. Host computer IP address (host computer connected via FTP interface)

      With the emulator monitor flash memory management tool command LH, specify the name and IP address of the host computer to be connected to the emulator via the FTP interface when initiating the E8000 system program. Since the specified host name and IP address are written to the emulator flash memory, they need not to be written each time the LAN interface is used. The host computer name and IP address can be modified as required.

9.2.2 Data Transfer

Data transfer is performed by connecting the emulator to the host computer via the FTP interface after the environmental settings have been completed. In the FTP interface, the optional LAN board supports only the client function. Therefore, the FTP command must be entered to the emulator and not the host computer to establish the FTP interface. Transfer data using the following procedure.

Procedure:

1. E8000 system program initiation

   Initiate the E8000 system program after confirming that the host computer to be connected has been defined with the emulator monitor flash memory management tool command LH.

2. FTP connection

   Connect the emulator to the designated host computer with the FTP command using the format shown below. Enter the host computer name defined with the emulator monitor flash memory management tool command LH. In addition, enter the user name and password.

```
:FTP <host name> (RET)
 Username <user name> (RET)
 Password <password> (RET)
 login command success
FTP>
```

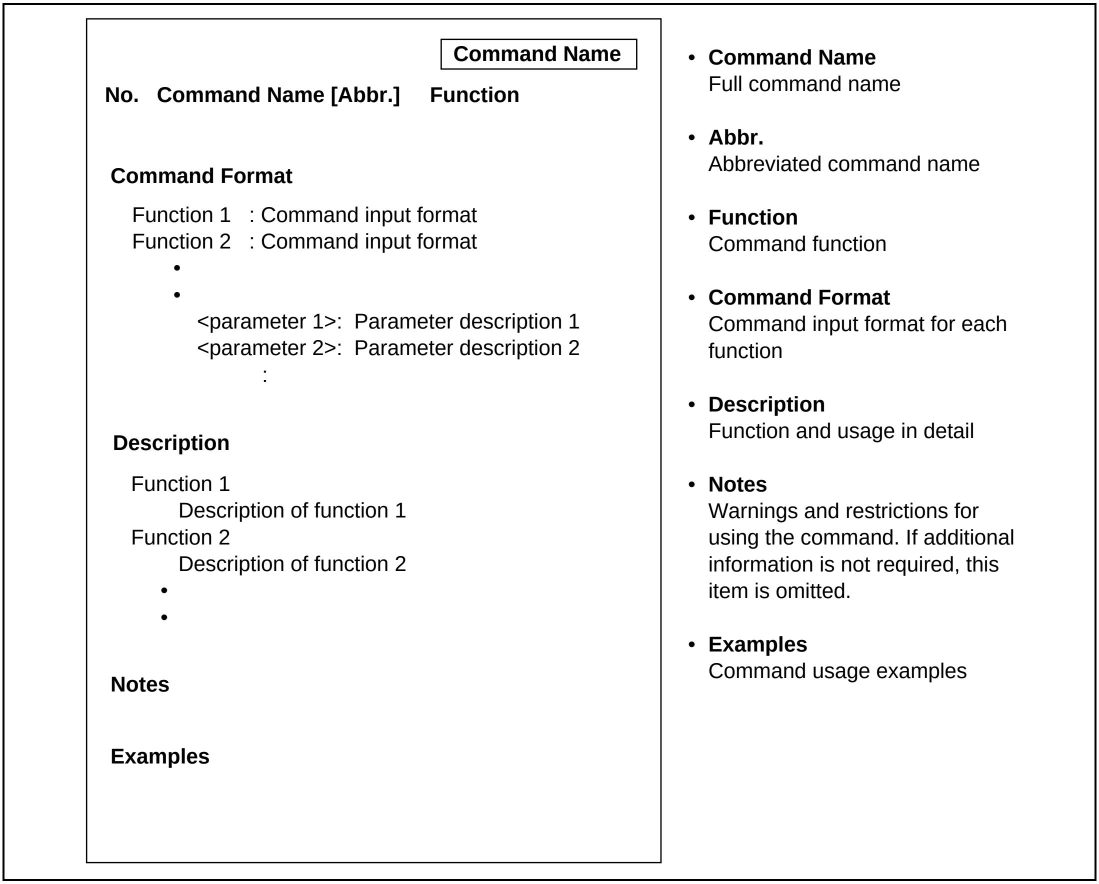
3. Transfer data using the LAN_LOAD, LAN_SAVE, or LAN_VERIFY command after the FTP interface is established. For details, refer to the corresponding command descriptions.

9.2.3 Notes on FTP Interface

Before turning off the emulator power, the FTP interface must be terminated using the BYE command. Otherwise, the host computer interface processing may remain uncompleted. In this case, the FTP interface cannot be re-established correctly even if the emulator is re-initiated.

9.3 LAN Commands

This section provides details of LAN commands in the format shown in figure 9.1.



Command Name

No. Command Name [Abbr.] Function

Command Format

Function 1 : Command input format
Function 2 : Command input format
•
•
<parameter 1>: Parameter description 1
<parameter 2>: Parameter description 2
:

Description

Function 1
Description of function 1
Function 2
Description of function 2
•
•

Notes

Examples

- **Command Name**
  Full command name
- **Abbr.**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and restrictions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 9.1 LAN Command Description Format

Symbols used in the command format have the following meanings:

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

(a/b): One of the parameters enclosed by ( ) and separated by /, that is, either a or b must be specified.

< >: Contents shown in < > are to be specified or displayed.

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

Δ: Indicates a space. Used only for command format description.

(RET): Pressing the (RET) key.

Although italic and bold characters are used throughout this manual to indicate input, it is not used in the command format sections of these descriptions.

9.3.1 ASC [ASC] Specifies the file type as ASCII

Command Format

- Setting ASC (RET)

Description

- Setting

  Specifies the file type as ASCII in the FTP interface. To load an SYSROF-type load module file, binary must be specified with the BIN command.

Example

To set the file type as ASCII in the FTP interface:

```
FTP> ASC (RET)
 asc command success
FTP>
```

9.3.2 BIN [BIN] Specifies the file type as binary

Command Format

- Setting BIN (RET)

Description

- Setting

  Specifies the file type as binary in the FTP interface. This specification is required to transfer files with the LAN_LOAD, LAN_SAVE, or LAN_VERIFY command. To load or verify an SYSROF-type load module file, binary must be specified with this command. Otherwise, a transfer error will occur. At emulator initiation, binary is the default setting.

Example

To set the file type as binary in the FTP interface:

```
FTP> BIN (RET)
 bin command success
FTP>
```

9.3.3 BYE [BYE] Terminates the FTP interface

Command Format

- FTP interface termination BYE (RET)

Description

- FTP interface termination

  Terminates the FTP interface and changes the prompt to a colon (:). To re-establish the FTP interface, enter the FTP command. For details, refer to section 9.3.6, FTP.

Example

To terminate the FTP interface:

```
FTP> BYE (RET)
 bye command success
:
```

9.3.4 CD [CD] Changes the directory name of the FTP server

Command Format

- Directory change CD Δ<directory name> (RET)

  <directory name>: Name of directory to be changed

Description

- Directory change

  Changes the current directory of the FTP server (connected host computer) to the specified directory. The modified directory must be formatted depending on which host computer is connected via the FTP interface.

Example

To change the current directory of the FTP server to subdir:

```
FTP> CD subdir (RET)
 cd command success
FTP>
```

9.3.5 CLOSE [CLOSE] Disconnects the host computer from the FTP interface

Command Format

- FTP interface disconnection CLOSE (RET)

Description

- FTP interface disconnection

  Disconnects the FTP interface from the host computer to which it is currently connected. Before changing host computers, disconnect the FTP interface with this command and re-connect with the OPEN command. For details, refer to section 9.3.13, OPEN.

Example

To disconnect the FTP interface and change the host computer to be connected:

```
FTP> CLOSE (RET)
 bye command success
FTP> OPEN HOST1 (RET)
 Username ABC (RET)
 Password ****** (RET)
 login command success
FTP>
```

9.3.6 FTP [FTP] Connects host computer and emulator via the FTP interface

Command Format

- FTP interface connection FTP <host name> (RET)

<host name>: Name of the host computer to be connected with the FTP server (The host computer name must be already defined with the flash memory management tool.)

Description

- FTP interface connection

— Connects the host computer and emulator via the FTP interface to enable data transfer with the LAN_LOAD, LAN_SAVE, or LAN_VERIFY command. The host name specified in this command must be defined with the flash memory management tool.

— If <host name> has been defined, enter the user name and password in the following format. After FTP command execution, the prompt changes from a colon (:) to FTP>. Emulation commands can be executed even after FTP connection.

```
: FTP <host name>  (RET)
  Username    (a)  (RET)
  Password    (b)  (RET)
  login command success
FTP>          (c)
```

(a) Enter user name.
(b) Enter password.
(c) An FTP> prompt is displayed after FTP connection.

Note

A password must be specified before a host computer can be connected via the FTP interface.

Example

To connect the emulator to host computer HOST1 via the FTP interface:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>
```

9.3.7 LAN [LAN] Displays emulator IP address

Command Format

- Display LAN (RET)

Description

- Display
  - Displays the emulator’s IP (internet) address stored in the emulator, in the following format:

```
:LAN  (RET)
 E8000 INTERNET ADDRESS  xxx.xxx.xxx.xxx
                                (a)
```

(a) Emulator IP address

  - Specify the emulator IP address with the emulator monitor command L.

Example

To display the emulator IP address:

```
:LAN (RET)
 E8000 INTERNET ADDRESS 128.1.1.10
:
```

9.3.8 LAN_HOST [LH] Displays the names and IP addresses of all defined host computers

Command Format

- Display LAN_HOST (RET)

Description

- Display

  Displays the LAN host computer names and internet addresses defined in the emulator flash memory in the following format:

```
: LAN_HOST  (RET)
  NO    <HOST NAME>   <IP ADDRESS>     NO    <HOST NAME>   <IP ADDRESS>
  01      xxxxxx      xxx.xxx.xxx.xxx  02      xxxxxx      xxx.xxx.xxx.xxx
  03      xxxxxx      xxx.xxx.xxx.xxx  04      xxxxxx      xxx.xxx.xxx.xxx
  05      xxxxxx      xxx.xxx.xxx.xxx  06      xxxxxx      xxx.xxx.xxx.xxx
  07      xxxxxx      xxx.xxx.xxx.xxx  08      xxxxxx      xxx.xxx.xxx.xxx
  09      xxxxxx      xxx.xxx.xxx.xxx
```

Example

To display all of the defined host computer names and IP addresses:

```
:LH (RET)
 NO <HOST NAME>  <IP ADDRESS>   NO <HOST NAME>  <IP ADDRESS>
 01  HOST1        128.1.1.1     02  HOST2        128.1.1.4
 03  HOSTX        128.1.1.8     04
 05                             06
 07                             08
 09
:
```

9.3.9 LAN_LOAD [LL] Loads a load module file from the host computer to memory via the FTP interface

Command Format

- Load LAN_LOAD [Δ<offset>][;<load module type>]
  [(P/V)[=<ASID value>]]:<file name> (RET)

<offset>: Value to be added to the load module address

<load module type>: Load module type

R: SYSROF-type load module

S: S-type load module

H: HEX-type load module

M: Memory image file

E: ELF-type load module

Default: SYSROF-type load module

<file name>: File name in the host computer

P: Physical address

V: Virtual address (The address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Load

— Loads a load module file from the host computer to memory via the FTP interface. Before executing this command, the emulator must be connected to the host computer with the FTP command. For details, refer to section 9.3.6, FTP.

— The current load address is displayed as follows:

```
LOADING ADDRESS = xxxxxxxx
```

xxxxxxxx: Current load address (continuously updated)

When loading is completed, the start and end addresses are displayed as follows:

TOP ADDRESS = <start address>

END ADDRESS = <end address>

- An offset (value to be added) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

```
:LAN_LOAD  <offset> ;S:<file name>  (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

- A physical or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. A load module file can be loaded only to the internal memory areas or areas CS0 to CS6.
2. Verification is not performed during load. If the program must be verified, use the LAN_VERIFY command. For details, refer to section 9.3.11, LAN_VERIFY.
3. Before loading an SYSROF-type load module, the file type must be changed to binary code with the BIN command. At emulator initiation, binary code is selected as the default. However, if ASCII is selected with the ASC command, change the file type to binary code with the BIN command before loading. For details, refer to section 9.3.2, BIN.

Example

To load an SYSROF-type load module, enter the following command line. F11.ABS indicates the host computer file name. Before entering the LAN_LOAD command, connect the emulator to the host computer with the FTP command:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP> LL :F11.ABS (RET)
 LOADING ADDRESS   00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

9.3.10 LAN_SAVE [LSV] Saves the specified memory contents in the host computer connected via the FTP interface

Command Format

- Save LAN_SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>) Δ[;[<load module type>][(P/V)[=<ASID value>]][ΔLF]]:<file name> (RET)

<start address>:	Start memory address
<end address>:	End memory address
<number of bytes>:	The number of bytes to be saved
<load module type>:	Load module type S: S-type load module H: HEX-type load module M: Memory image file Default: S-type load module
LF:	Adds an LF code (H'0A) to the end of each record
<file name>:	File name in the host computer
P:	Physical address
V:	Virtual address (The address is converted using the MCU TLB at command input)
<ASID value>:	ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Save

— Saves the specified memory contents in the host computer connected via the FTP interface. An S-type, HEX-type, or M-type load module can be saved. An SYSROF-type or ELF-type load module cannot be saved. Before executing this command, connect the emulator to the host computer with the FTP command.

— The current save address is displayed as follows:

SAVING ADDRESS = xxxxxxxx

xxxxxxxx: Current save address (continuously updated)

When save is completed, the start and end memory addresses are displayed as follows:
TOP ADDRESS = <start address>
END ADDRESS = <end address>

— When the LF option is specified, the emulator adds an LF code (H'0A) to the end of each record of an S- or HEX-type load module in addition to a CR code (H'0D).

— A physical or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. Data can be saved only in the internal memory areas or areas CS0 to CS6.
2. Verification is not performed after save. If the program must be verified, use the LAN_VERIFY command, if necessary. For details, refer to section 9.3.11, LAN_VERIFY.

Example

To save the memory contents in the address range from H'7000 to H'7FFF in the host computer as an S-type load module file (file name: F11.S), enter the following command line. Before entering the LAN_SAVE command, connect the emulator to the host computer with the FTP command:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>LSV 7000 7FFF :F11.S (RET)
 SAVING ADDRESS   00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

9.3.11 LAN_VERIFY [LV] Verifies memory contents against the host computer file connected via the FTP interface

Command Format

- Verification LAN_VERIFY [Δ<offset>][;<load module type>]
  [(P/V)[=<ASID value>]]:<file name> (RET)

<offset>: Value to be added to the load module address

<load module type>: Load module type

R:	SYSROF-type load module
S:	S-type load module
H:	HEX-type load module
M:	Memory image file
E:	ELF-type load module
Default:	SYSROF-type load module

<file name>: File name in the host computer

P: Physical address

V: Virtual address (The address is converted using the MCU TLB at command input)

<ASID value>: ASID value (can be specified only when V is specified)

Note: When P, V, or <ASID value> is omitted, the address is converted using the VP_MAP table when VP_MAP table is enabled, and converted based on the MMU status at command input when the VP_MAP table is disabled.

Description

- Verification
  — Verifies the file in the host computer connected via the FTP interface against data in memory in the following format. Before executing this command, connect the emulator to the host computer with the FTP command.

```
FTP> LAN_VERIFY <load module type>:<file name> (RET)
```

— If a verification error occurs, the address and its contents are displayed as follows:

```
<ADDR>          <FILE>          <MEM>
xxxxxxxx         yy 'y'          zz 'z'
```

xxxxxxxx: Verification error address
yy 'y': Load module data (in hexadecimal and ASCII characters)
zz 'z': Memory data (in hexadecimal and ASCII characters)

— An offset (value to be added or subtracted) can be specified for the address of an SYSROF-type, ELF-type, S-type, or HEX-type load module.

FTP> ***LAN_VERIFY <offset> ;S :<file name> (RET)***

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

— A physical or a virtual address can be specified by specifying P or V, respectively. When P or V is omitted, the physical or virtual address is determined based on the MMU status at command input.

Notes

1. Data can be verified only in the internal memory areas or areas CS0 to CS6.
2. Before verifying an SYSROF-type load module, the file type must be changed to binary code with the BIN command. At emulator initiation, binary code is selected as the default. However, if ASCII is selected with the ASC command, change the file type to binary code with the BIN command before verifying. For details, refer to section 9.3.2, BIN.

Example

To verify SYSROF-type load module file F11.ABS in the host computer against the memory contents:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>LV :F11.ABS (RET)
 VERIFYING ADDRESS   00000C00
 TOP ADDRESS = 00000000
 END ADDRESS = 00000FFF
FTP>
```

9.3.12 LS [LS] Displays the host computer directory connected via the FTP interface

Command Format

- Display LS [Δ<directory name>] (RET)

<directory name>: Name of host computer directory
(Default: Current directory of the host computer)

Description

- Display
  Displays the specified directory contents in the host computer connected via the FTP interface. If <directory name> is omitted, the current directory contents are displayed. Note that the directory name must be specified according to the connected host computer format.

Example

To display the contents of the host computer current directory:

```
FTP>LS (RET)
abc.s
xyz
FTP>
```

9.3.13 OPEN [OPEN] Connects the host computer to the FTP interface

Command Format

- FTP interface connection OPEN <host name> (RET)

<host name>: Name of the host computer to be connected via the FTP interface (The host computer name must be already defined with the flash memory management tool.)

Description

- FTP interface connection

  Connects the emulator to the specified host computer via the FTP interface. This command can also be used to change the host computer connected to the emulator. To change the host computer, first disconnect the current host computer using the CLOSE command and then connect the new host computer using this command.

```
FTP>OPEN <host name> (RET)
  Username    (a) (RET)
  Password    (b) (RET)
  login command success
FTP>
```

(a) Enter user name.

(b) Enter password.

Note

A password must be specified before a host computer can be connected via the FTP interface.

Example

To disconnect the emulator from the current host computer and connect it to the new host computer HOST1:

```
FTP>CLOSE (RET)
 bye command success
FTP>OPEN HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>
```

9.3.14 PWD [PWD] — Displays the current directory name of the host computer connected via the FTP interface

Command Format

- Display PWD (RET)

Description

- Display
  Displays the current directory name of the host computer connected via the FTP interface.

Example

To display the current directory name of the host computer connected via the FTP interface:

```
FTP>PWD (RET)
/usr/e8000
FTP>
```

9.3.15 ROUTER [RTR] Displays the remote network routing information

Command Format

- Display ROUTER (RET)

Description

- Display
  Displays the routing information defined with the emulator monitor flash memory management tool command RTR.

Note

Routing information can be defined with the emulator monitor flash memory management tool command RTR.

Example

To display the defined routing information:

```
:RTR (RET)
 No. IP-ADDRESS    NET-ID      No. IP-ADDRESS    NET-ID
 01  128.1.1.80    168.1.1.0   02  128.1.1.50    160.1.1.0
:
```

9.3.16 STA [STA] Displays the file type to be transferred

Command Format

- Display STA (RET)

Description

- Display

  Displays in the following format, the file type (binary or ASCII) to be transferred by the LAN_LOAD, LAN_SAVE, or LAN_VERIFY command.

```
FTP>STA (RET)
    type mode is BINARY     (Binary)

FTP>STA (RET)
    type mode is ASCII      (ASCII)
```

Example

To display the file type to be transferred:

```
FTP>STA (RET)
 type mode is BINARY
FTP>
```

9.3.17 SUBNET [SN] Displays the subnet mask value

Command Format

- Display SUBNET (RET)

Description

- Display

  Displays the subnet mask value defined with the emulator monitor flash memory management tool command SN.

```
: SUBNET  (RET)
  SUBNET MASK  xxx.xxx.xxx.xxx (H’yy.H’yy.H’yy.H’yy)
                      (a)                  (b)
```

(a) Subnet mask value (in decimal)

(b) Subnet mask value (in hexadecimal)

Note

The subnet mask value can be defined with the emulator monitor flash memory management tool command SN.

Example

To display the defined subnet mask value:

```
:SN (RET)
 SUBNET MASK 255.255.255.128 (H’FF.H’FF.H’FF.H’80)
:
```

9.3.18 LOGOUT [LO] Disconnects from the TELNET

Command Format

- TELNET disconnection LOGOUT (RET)

Description

- TELNET disconnection
  Disconnects the emulator from the TELNET. This command is valid only when the emulator is connected to the host computer via the TELNET interface.

Example

To disconnect the emulator from the TELNET interface:

```
:LO (RET)
```

Section 10 Error Messages

10.1 Emulator Error Messages of E8000 Emulator

The E8000 system program outputs error messages in the format below. Table 10.1 lists error messages, descriptions of the errors, and error solutions.

```
*** nn:   <error message>

          nn:    Error No.
```

Table 10.1 Error Messages

Error No.	Error Message	Description and Solution
1	INTERNAL ERROR (nn)	An error occurred in the E8000 system program or station. Error code nn gives specific details. Contact a Hitachi sales agency and inform them of the code and state.
2	HOST I/O ERROR (nn)	An I/O error occurred in data transfer between the emulator and host computer. Error code nn gives specific details. Refer to table 10.2.
5	INVALID DEVICE CONTROL BOARD	The connected device control board is not supported by this E8000 system program. Check the E8000 system program and device control board numbers.
6	USER SYSTEM NOT READY	The user clock or crystal oscillator clock was not input and therefore could not be selected. The emulator internal clock was used instead. Check if the clock signal is output correctly.
9	INVALID OPTION	The specified option is incorrect. Check the specified option.
10	FLASH MEMORY IS WRITE PROTECTED	Flash memory is write-protected. Remove write protection.
11	FLASH MEMORY WRITE ERROR	An error occurred during write to flash memory.
13	FILE NOT FOUND	The configuration information specified to be restored with the CONFIGURATION command was not found in the emulator flash memory.
15	INVALID FILE	The specified file has invalid contents and cannot be read from or written to. Check the contents of the specified file.
20	SYNTAX ERROR	The command syntax is incorrect. Correct the syntax.
21	INVALID COMMAND	The specified command is invalid, or this command cannot be executed in parallel mode. Correctly enter the command.
22	INVALID DATA	The specified data is invalid. Correctly enter the data.
23	INVALID ADDRESS	The specified address or address range is invalid. Correctly enter the address.
24	DATA OVERFLOW	The specified data is more than 4 bytes. Correctly specify the data.
27	INVALID CONDITION	Invalid conditions are specified. Correctly enter the conditions.

Table 10.1 Error Messages (cont)

Error No.	Error Message	Description and Solution
28	DOUBLE DEFINITION	The item has already been defined. Check the item to be defined.
29	TOO MANY ALIASES	Too many aliases are specified. Delete any unnecessary alias and re-specify.
31	INSUFFICIENT MEMORY	The size of emulation memory to be allocated with the MAP command was not available. Emulation memory was allocated within the available memory size.
32	INVALID ASM MNEMONIC	An instruction mnemonic in an assembly-language statement is invalid. Correct the instruction mnemonic.
33	INVALID ASM OPERAND	An operand in an assembly-language statement is invalid. Correct the operand.
34	ALREADY ASSIGNED	A condition cannot be specified by the BREAK_CONDITION_A,B,C, TRACE_CONDITION_A,B,C, or PERFORMANCE_ANALYSIS command. Too many conditions are specified. Cancel a condition for another command and re-specify.
35	CANNOT USE THIS MODE	GO The GO command cannot be executed because settings for the execution mode are invalid. Correctly specify the settings necessary for the specified execution mode.
37	TOO MANY POINTS	Too many points are specified. Remove any unnecessary settings and re-enter.
38	SET POINT IS NOT IN RAM	A write-inhibited address is specified by the BREAK or BREAK_SEQUENCE command. Specify a correct address.
39	BUFFER EMPTY	TRACE or TRACE_SEARCH The trace buffer is empty. Check trace conditions and execution state, and re-execute. Then display trace information.
42	CANNOT CHANGE ATTRIBUTE OF X/Y MEMORY AREA	An attempt was made to change the memory attribute of the internal memory area. Allocate emulation memory to areas other than the internal memory area.

Table 10.1 Error Messages (cont)

Error No.	Error Message	Description and Solution
43	CANNOT RECOVER A = xxxxxxxx	The break instruction at the address (xxxxxxxx) where a breakpoint is specified with the BREAK or BREAK_SEQUENCE command could not be recovered after GO command execution is terminated. Accordingly, a break instruction remains at the breakpoint address. A hardware error might have occurred. Correct the error, and reload and re-execute the program.
44	VERIFY ERROR	A verification error occurred when modifying memory contents. Writing to ROM was attempted or there was a hardware error. Check the memory area.
45	NOT FOUND	The specified data or information was not found. Correctly specify data.
46	BREAKPOINT ADDRESS	The memory contents of the specified address cannot be modified in parallel mode because the address is used by the BREAK or BREAK_SEQUENCE command.
47	FTP NOT CONNECTED	The command cannot be executed because the FTP interface is not connected. Connect the FTP interface with the FTP command.
48	FTP ALREADY CONNECTION	The FTP interface has already been connected. Disconnect the FTP interface and re-enter the command.
49	CONDITION ALREADY USED	The condition cannot be specified because another command has already specified it.
51	INTERNAL I/O AREA	The internal I/O area was accessed.
52	INTERNAL AREA	An attempt was made to access an area other than CS0 to CS6. This area cannot be accessed with this command. Check the specified address.
54	INVALID CONFIGURATION FILE	The configuration file in emulator flash memory is invalid. Re-install the configuration file from the system disk.
55	CONFIGURATION FILE NOT FOUND	The configuration file was not found in the emulator flash memory. Re-install the configuration file from the system disk.
56	INVALID CONFIGURATION CHECK ERROR	The configuration file in emulator flash memory contains invalid data. Re-install the configuration file from the system disk.

Table 10.1 Error Messages (cont)

Error No.	Error Message	Description and Solution
57	ILLEGAL INSTRUCTION ADDRESS	The memory contents of the address specified with the BREAK or BREAK_SEQUENCE command is a break instruction (H'0000). A breakpoint cannot be specified at this address.
58	CANNOT SELECT EMULATOR CLOCK	An operating mode not supported with this emulator was specified. Check the operating mode.
59	TOO MANY CHARACTERS	Too many characters were specified. Check the number of characters.
61	CANNOT GET INTO PARALLEL MODE	The execution mode specified with the GO command prevents the emulator from entering parallel mode. Change the execution mode.
62	LAN BOARD DISCONNECTION	This command cannot be executed because the LAN board is not installed. Install the optional LAN board and re-enter the command.
63	TLB MISS HIT	TLB miss hit error occurred. Check the address, ASID value, and MCU TLB contents specified with the command.
64	TLB OR ADDRESS ERROR	TLB error or address error occurred. Check the address, ASID value, and MCU TLB contents specified with the command.
66	BACKGROUND INTERRUPT COMMAND STOPPED	The BACKGROUND_INTERRUPT command execution was terminated.
67	LAN I/O ERROR	A LAN I/O error occurred. Refer to table 10.3.
68	INVALID HOST NAME	The specified host name is not defined in flash memory. Define the host name with the emulator monitor command F (flash memory management tool initiation).
69	OUT OF CS AREA ADDRESS	An attempt was made to allocate emulation memory to an area other than CS0 to CS6. Check the specified address.
70	OUT OF CS AREA ADDRESS	An attempt was made to allocate emulation memory to an area other than CS0 to CS6. The emulation memory was allocated within the available area.
71	MAPPING BOUND MUST BE IN 4MB UNITS	Memory was allocated in 4-Mbyte units with the MAP command. For details, refer to the MAP command.
73	BREAK POINT IS DELETED A = xxxxxxxx	A software breakpoint specified at the displayed address was canceled because the contents of the address were modified with the user program.

Table 10.1 Error Messages (cont)

Error No.	Error Message	Description and Solution
74	CANNOT SET A = xxxxxxxx	A breakpoint cannot be specified at the displayed address by the BREAK or BREAK_SEQUENCE command before GO command execution. A hardware error might have occurred or the contents of the memory address might be a break instruction (H'0000). Correct the error, and reload and re-execute the program.
78	EMULATOR BUSY	The emulator was processing a break processing in parallel mode, so another command could not be executed. Re-enter the command. This error occurs when software breakpoints are set with the BREAK or BREAK_SEQUENCE command.
81	TRACE CONDITION RESET	Satisfied trace conditions are all reset when parallel mode is entered. When parallel mode is terminated, the trace conditions are rechecked from the beginning.
82	ODD ADDRESS	An instruction was written to an odd address with the ASSEMBLE command. Processing was initiated from the odd address.
83	INVALID OPERAND SIZE	An invalid operand size was specified with the ASSEMBLE command. Processing was performed with the correct size.
84	INVALID ABSOLUTE ADDRESS	An invalid operand address was specified with the ASSEMBLE command. Processing was performed with the maximum address allowed.
86	INTERNAL AREA	An area other than CS0 to CS6 was also to be processed. Processing specified with the MEMORY command is performed normally, but other command processing are performed for only the CS0 to CS6 areas.
87	INTERNAL I/O AREA	The internal I/O area was accessed.
88	RESERVED AREA	A reserved area was accessed.
92	PERFORMANCE ANALYSIS TABLE BUSY	The minimum unit for execution performance measurement cannot be changed during execution time measurement by the PERFORMANCE_ANALYSIS command. Delete the PERFORMANCE_ANALYSIS command setting and change the minimum unit.

Table 10.2 Host I/O Error Codes

Error Code	Error Name	Description and Solution
D1	Parity error	The parity bit specified with the DIP switch must match the host computer specifications.
D2	Overrun error	The emulator control method is not recognized by the host computer.
D3	Framing error	The baud rate and stop bit specified with the DIP switch must match the host computer specifications.
D4	Load module format error	The load module format of the transferred file is incorrect. Check the file contents.
DC	Timeout error	Check the connection between the emulator and host computer. Also check the operation status of the host computer.

The E8000 system program outputs LAN I/O error messages in the format below. Table 10.3 lists the error messages with brief descriptions.

```
LAN I/O ERROR (E0xx)
socket library error nn: <error message>
```

xx:	Process in which error occurred (see table 10.4)
nn:	Error code
<error message>:	Refer to table 10.3

If an error message other than that listed in table 10.3 is displayed, refer to the description for the host computer error messages.

Table 10.3 LAN I/O Error Messages

Error No.	Error Message	Description
01	not listen	The socket cannot be created.
02	Insufficient Buffer	The internal buffer is insufficient.
03	Socket not Support	The requested function is not supported.
04	Socket is Already	The socket has already been connected.
05	time out error	A timeout error has occurred.
06	Ip Address Nothing	The IP address destination is undefined.
07	Not socket Connection	The socket has not been connected.
08	connection failire	A connection failure has occurred.
09	Illegal IP Address	An illegal IP address has been specified.

Table 10.3 LAN I/O Error Messages (cont)

Error No.	Error Message	Description
10	be Shutdowning	The connection is being terminated.
11	Not Socket Entry	The socket information has not been defined.
12	Socket is already	The socket information has already been defined.
13	HOSTS Name Nothing	The host computer name does not exist.
14	Socket not Assign Connected	The socket cannot be assigned.
15	illegal port No.	The port number is invalid.
16	initialized error	An error has occurred during LAN board initialization.
17	Not Terminate	The LAN board has not been terminated.
18	terminate error	A LAN board termination error has occurred.
19	Not initialized	The LAN board has not been initialized.
20	Illeagal Board	An error has occurred in the LAN board.
21	System Error	A LAN board system error has occurred.
22	Illegal Request	An invalid request has been issued.
23	Parameter Error	The parameter data is invalid.
24	Response Timeout Happend	A response timeout error has occurred.
25	Check Sum Error	A checksum error has occurred.
26	ICMP Error	An ICMP error has occurred.
27	ethernet address error	An Ethernet address error has occurred.
28	not HOST File	The HOSTS information does not exist.
30	illegal initialized	The HOSTS initialization information is invalid.
31	illegal My data	Main station information is invalid.
32	illegal Other Party data	Remote station information is invalid.
33	remote Nothing	Remote station has not been defined.
34	transmission error	A data transfer error has occurred.
35	closing error	A termination error has occurred.
FF	unknown error	An undefined error has occurred.

Table 10.4 Process Code for LAN I/O Error Messages

Error No.	Process
01	Initialization
02	TELNET data transfer
03	TELNET close
04	TELNET open
10	FTP connection
20	File transmission
30	File reception
40	FTP disconnection
50	Directory modification
60	Directory display
70	Current directory display
80	File transfer binary specification
90	File transfer ASCII specification
A0	Forcible termination

10.2 IBM PC Interface Software Error Messages

The IBM PC interface software outputs error messages on the IBM PC. Table 10.5 lists error messages, descriptions of the errors, and error solutions.

Table 10.5 Interface Software Error Messages

Error Message	Description and Solution
INTFC ERROR - ABORT BY BREAK	File transfer has been forcibly terminated by pressing the (BREAK), (STOP), or (CTRL) + C keys.
INTFC ERROR - ALREADY ASSIGNED	The specified command is already being executed. Re-execute the command after command execution has been completed.
INTFC ERROR - EMULATOR NOT READY	The debugger power has been turned off or a cable connected to the debugger has been disconnected. Check that the debugger power is turned on and that cables are connected correctly, and restart. If the same error occurs again, inform a Hitachi sales agency.
INTFC ERROR - ENVIRONMENT NOT SPECIFIED	The specified environment variable name could not be found. Specify the environment variable name with the SET command.
INTFC ERROR - FILE ALREADY EXISTS OVERWRITE ? (Y/N):	The specified IBM PC file already exists. Enter Y to transfer any way after deleting the file; enter N to cancel transfer.
INTFC ERROR - FILE CLOSE ERROR	An error has occurred while closing an IBM PC file.
INTFC ERROR - FILE DELETE ERROR	An error has occurred while deleting an IBM PC file. Check the specified file name.
INTFC ERROR - FILE NOT FOUND	The IBM PC file specified at load cannot be found or the file name contains an error. Check the specified file name.
INTFC ERROR - FILE OPEN ERROR	The directory to which the specified IBM PC file is to be saved is full or the file name contains an error.
INTFC ERROR - FILE READ ERROR	An error has occurred while reading an IBM PC file.
INTFC ERROR - FILE RENAME ERROR	An error has occurred while changing an IBM PC file name. Check the specified file name.
INTFC ERROR - FILE WRITE ERROR	An error has occurred while writing to an IBM PC file. The available memory is insufficient.
INTFC ERROR - INVALID COMMAND	An invalid command has been sent from the debugger.

Table 10.5 Interface Software Error Messages (cont)

Error Message	Description and Solution
INTFC ERROR - I/O ERROR	An I/O error has occurred during file transfer. Check the cable connection and the operating environment, and re-transfer the file.
INTFC ERROR - NO INTERFACE BOARD	The interface board is not inserted in the IBM PC expansion slot. Check the DIP switch setting on the interface board and that the interface board is inserted in the expansion slot correctly, and re-transfer. If the same error occurs again, inform a Hitachi sales agency.
INTFC ERROR - STOP COMMAND CHAIN ? (Y/N):	Automatic command input from the IBM PC file has been completed. Enter Y to terminate command input; enter N to continue command input.
INTFC ERROR - SYNTAX ERROR	An error exists in the IBM PC file name. Refer to the debugger and IBM PC manuals and specify a correct file name.
INTFC ERROR - TIMEOUT ERROR	A timeout error has occurred during data transfer from the debugger. Check the cable connection and re-transfer.

Part III Appendix

Appendix A Connectors

A.1 Serial Connector

Figure A.1 shows the serial connector pin alignment in the emulator station. Table A.1 lists signal names and their usage.

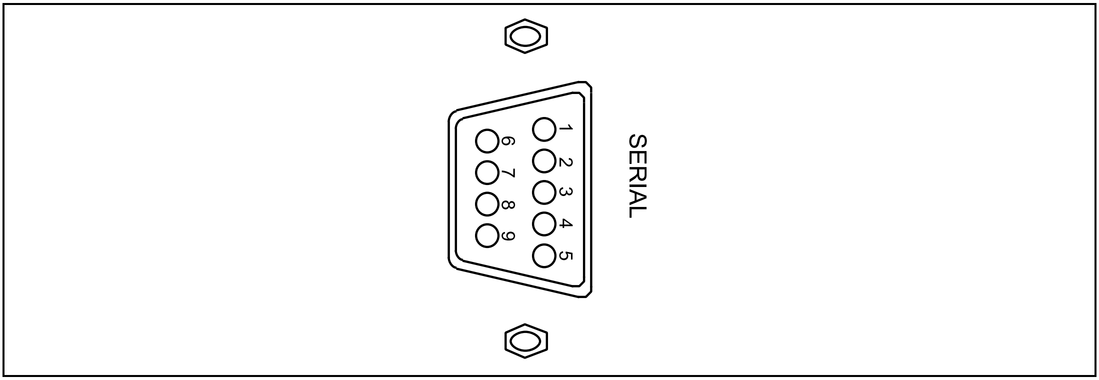


Figure A.1 Serial Connector Pin Alignment at the Emulator Station

Table A.1 Signal Names and Usage of Serial Connector

Pin No.	Signal name	Usage
1	—	Not connected
2	Receive Data (RD)	Data receive line
3	Transmit Data (TD)	Data transmit line
4	Data Terminal Ready (DTR)	High when emulator's power is on.
5	Ground (GND)	Connected to the emulator's frame ground.
6	Data Set Ready (DSR)	Not connected
7	Request To Send (RTS)	High when emulator's power is on.
8	Clear To Send (CTS)	Not connected
9	—	Not connected

A.2 Parallel Connector

Figure A.2 shows the parallel connector pin alignment at the emulator station. Table A.2 lists signal names.

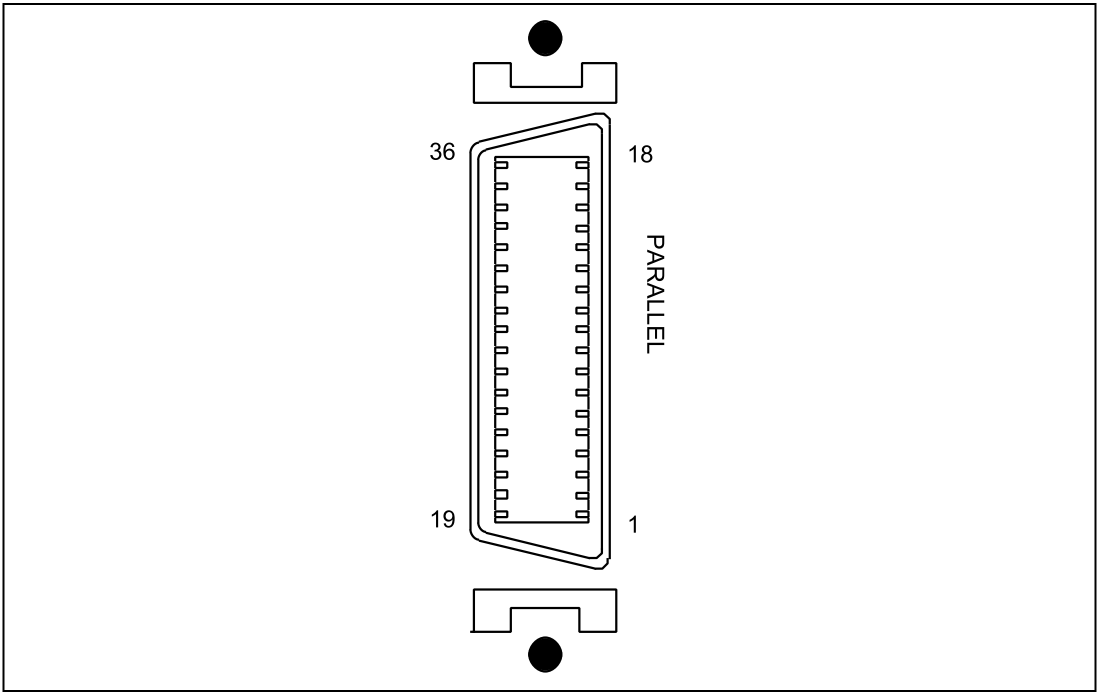


Figure A.2 Parallel Connector Pin Alignment at the Emulator Station

Table A.2 Signal Names of Parallel Connector

Pin No.	Signal Name	Pin No.	Signal Name
1	PeriphAck	19	SignalGround
2	Xflag	20	SignalGround
3	PeriphClk	21	SignalGround
4	nPeriphRequest	22	SignalGround
5	nAckReverse	23	SignalGround
6	Data1 (LSB)	24	SignalGround
7	Data2	25	SignalGround
8	Data3	26	SignalGround
9	Data4	27	SignalGround
10	Data5	28	SignalGround
11	Data6	29	SignalGround
12	Data7	30	SignalGround
13	Data8 (MSB)	31	SignalGround
14	nReverseRequest	32	SignalGround
15	HostClk	33	SignalGround
16	IEEE1284 active	34	SignalGround
17	HostAck	35	SignalGround
18	HostLogicHigh	36	PeripheralLogicHigh

A.3 LAN Connector

Figure A.3 shows the LAN connector pin alignment at the emulator station. Table A.3 lists signal names.

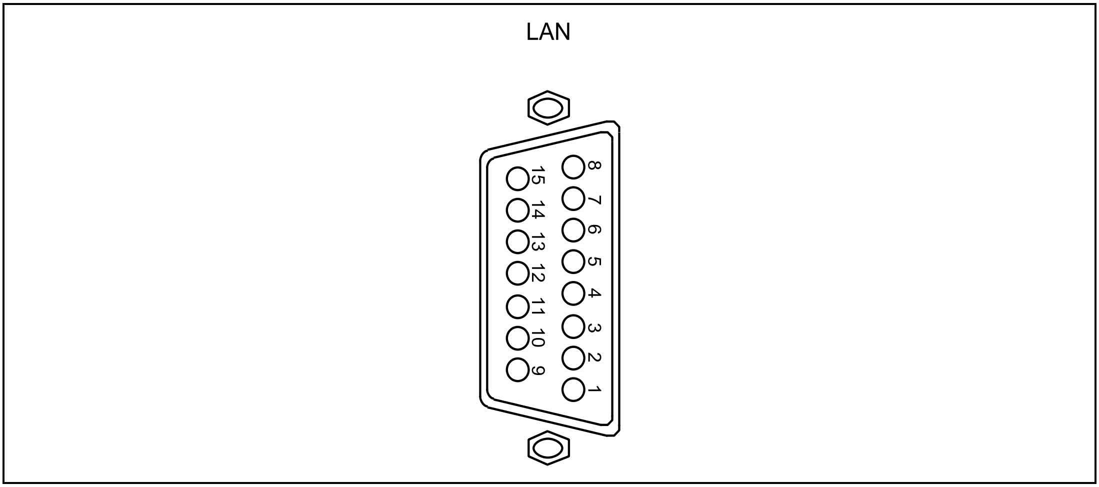


Figure A.3 LAN Connector Pin Alignment at the Emulator Station

Table A.3 Signal Names

Pin No.	Signal Name
1	Not connected
2	COL+
3	TX+
4	—
5	RX+
6	GND
7	—
8	—
9	COL–
10	TX–
11	—
12	RX–
13	+12 V
14	—
15	—

A.4 Serial Interface Cable

Figure A.4 shows the wiring for the serial interface cable.

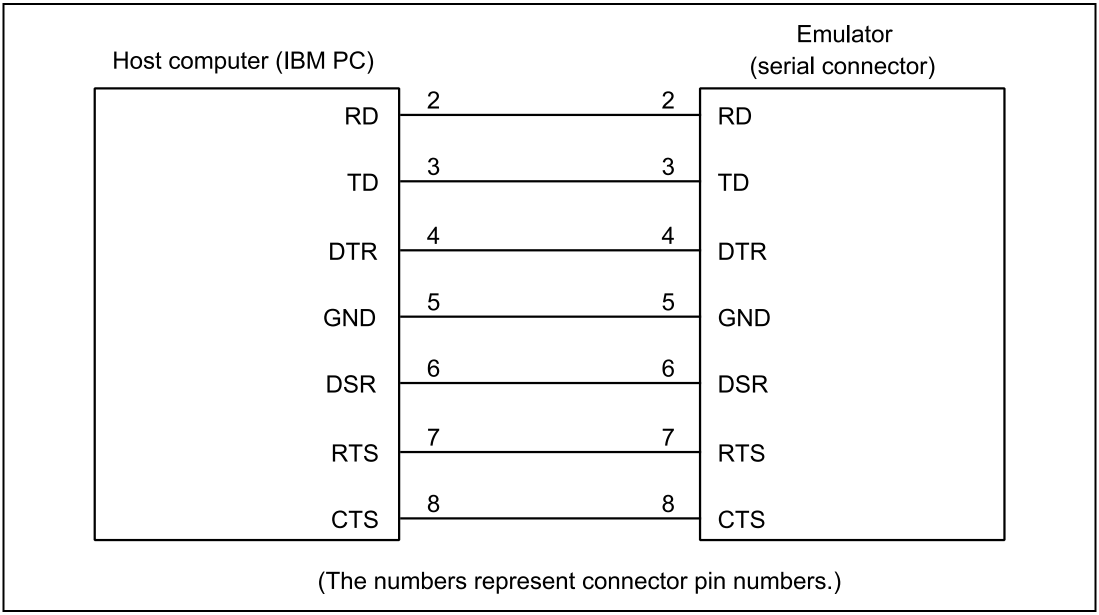


Figure A.4 Serial Interface Cable

Note that the serial interface cable provided may not be suitable for some host computers. In that case, use the wiring shown in figure A.5.

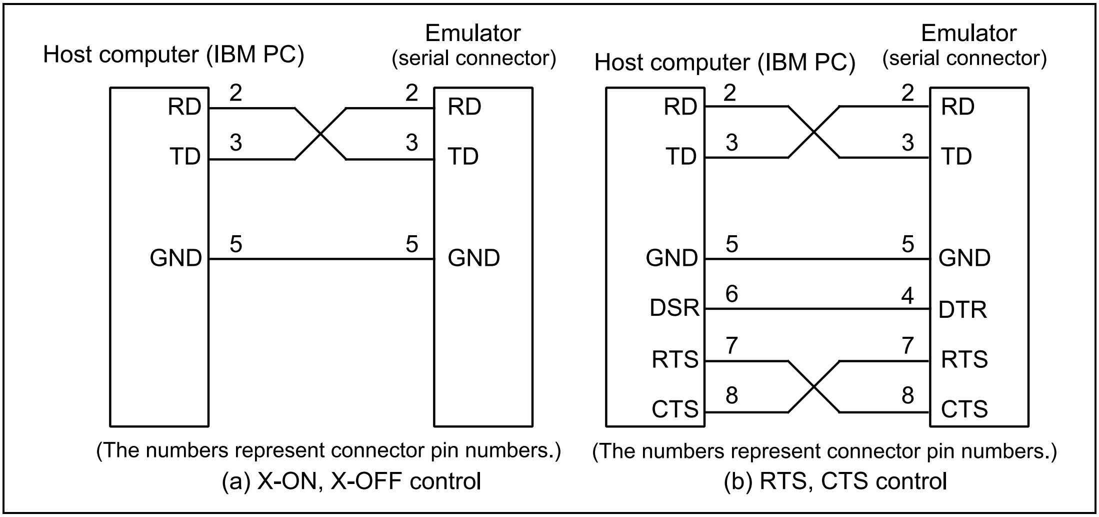


Figure A.5 Serial Interface Cable (Using Other Cables)

Appendix B Emulator External Dimensions and Weight

Figures B.1 and B.2 show the external dimensions and weight of the emulator station and evaluation chip board, respectively.

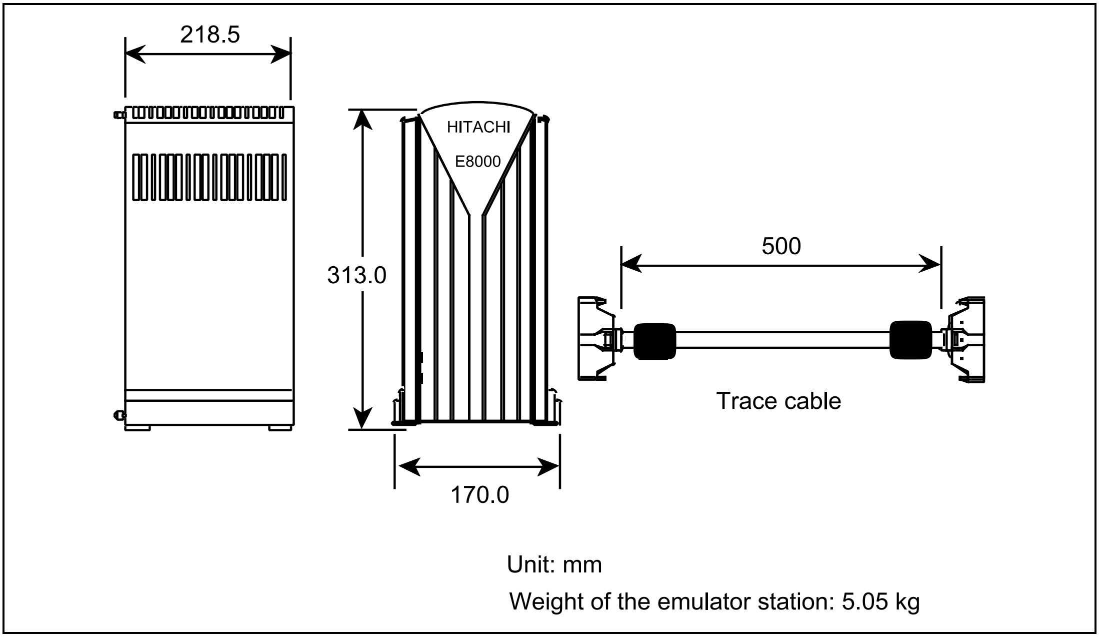


Figure B.1 External Dimensions and Weight of the E8000 Emulator

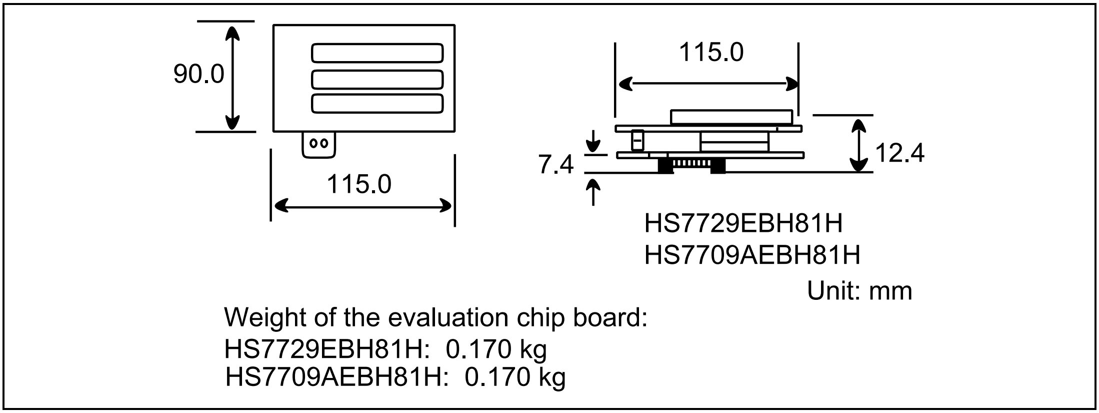


Figure B.2 External Dimensions and Weight of the Evaluation Chip Board

Appendix C Connecting the Emulator to the User System

C.1 Precautions for Evaluation Chip Board Connection

WARNING!

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

Notes:

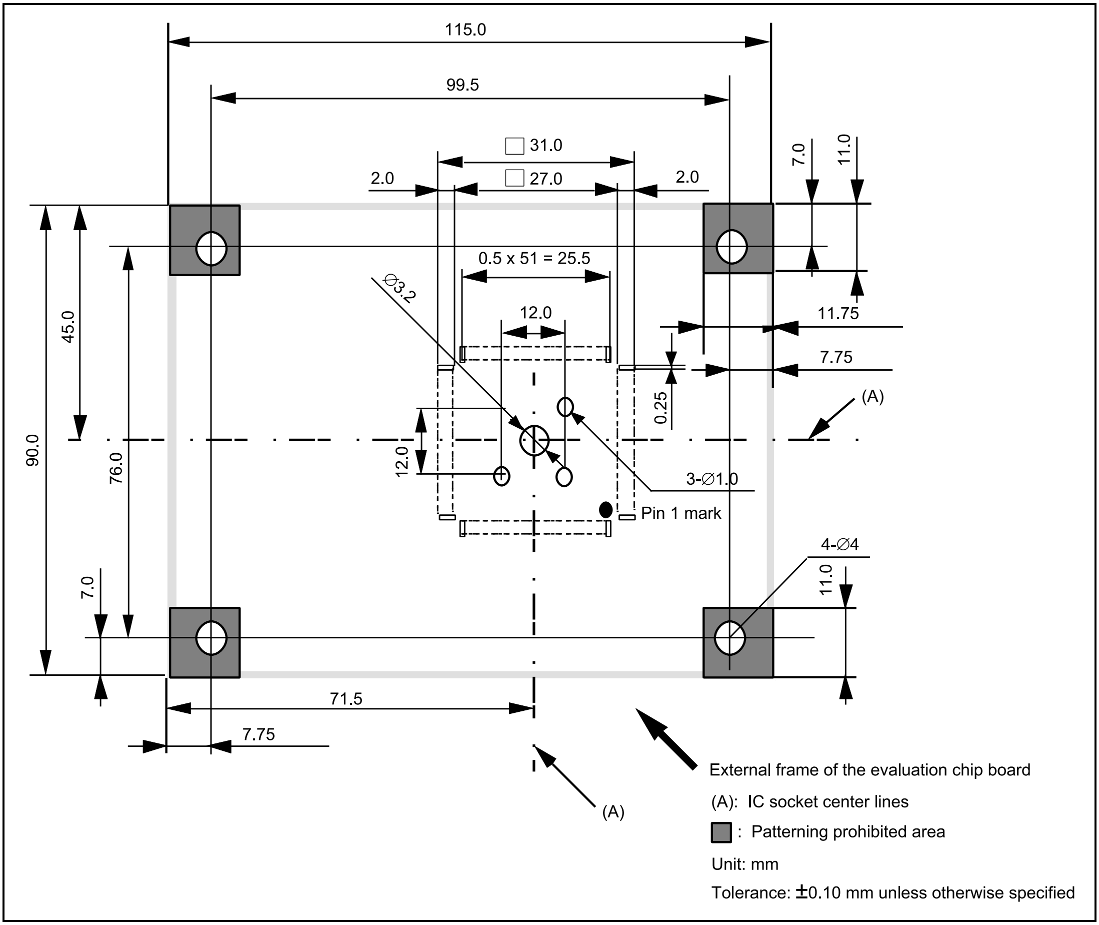
1. For details on the HS7729EBH81H and HS7709AEBH81H, refer to their description notes.
2. Only use the specified QFP socket (NQPACK208SD manufactured by Tokyo Eletech Corporation) to use the evaluation chip board.

The emulator is connected to the user system via the QFP-type evaluation chip board (HS7729EBH81H or HS7709AEBH81H).

Table C.1 Evaluation Chip Boards and User Interfaces

Evaluation Chip Board Type	User Interface
HS7729EBH81H for the SH7729	208-pin QFP (NQPACK208SD manufactured by Tokyo Eletech Corporation)
HS7709AEBH81H for the SH7709A	

When connecting the evaluation chip board to the user system, the foot pattern of the IC socket and the holes for spacers to fix the evaluation chip board are required on the user system. When creating the user system, prepare the IC socket foot pattern and the spacer holes by referring to figure C.1. The pin arrangement of the IC socket is the same as that for the actual SH7729 or SH7709A chip. Refer to the hardware manual of each device for the pin arrangement. Figure C.1 shows the installation allocation of the user system connector and spacers.



External frame of the evaluation chip board

(A): IC socket center lines

■ : Patterning prohibited area

Unit: mm

Tolerance: ±0.10 mm unless otherwise specified

Figure C.1 Installation Location of User System Connector and Spacers

208-pin QFP IC socket NQPACK208SD manufactured by Tokyo Eletech Corporation must be mounted on the user system to connect to the emulator. Figure C.2 shows the connection using the HS7729EBH81H or HS7709AEBH81H.

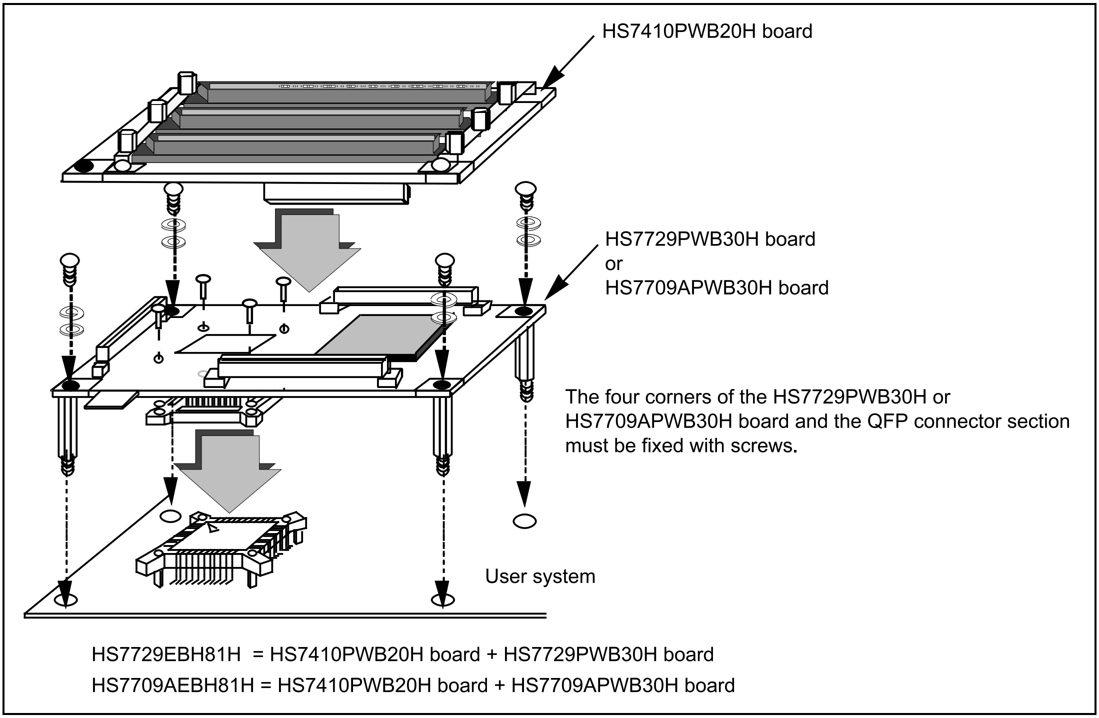

HS7410PWB20H board
HS7729PWB30H board
or
HS7709APWB30H board
The four corners of the HS7729PWB30H or HS7709APWB30H board and the QFP connector section must be fixed with screws.
User system
HS7729EBH81H = HS7410PWB20H board + HS7729PWB30H board
HS7709AEBH81H = HS7410PWB20H board + HS7709APWB30H board

Figure C.2 Connection using the HS7729EBH81H or HS7709AEBH81H

Figure C.3 shows the restrictions on parts location when the HS7729EBH81H or HS7709AEBH81H is used.

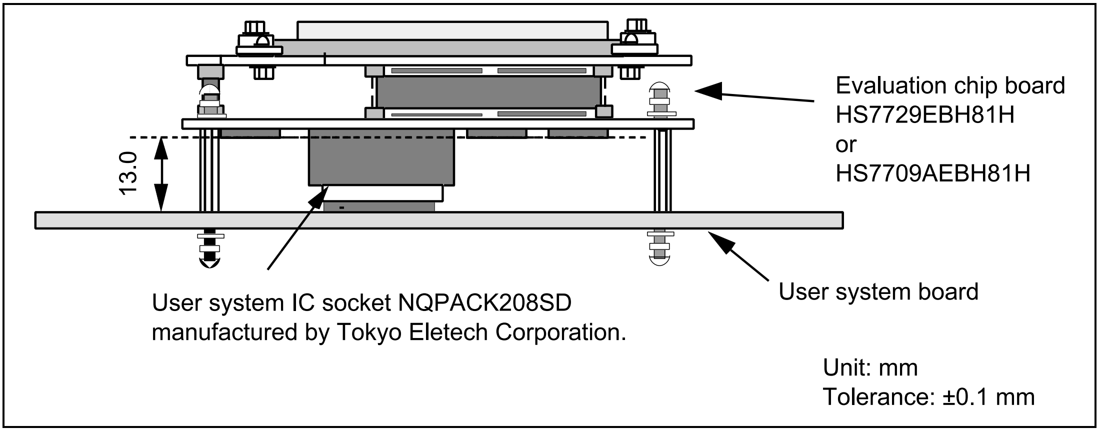


Figure C.3 Restrictions on Parts Location

C.2 Precautions for User System Connection

When connecting the evaluation chip board to the user system, note the following:

1. Secure the E8000 station location.

   Place the E8000 station and evaluation chip board so that the station to evaluation chip board interface cable is not bent or twisted, as shown in figure C.4. A bent or twisted cable will impose stress on the user interface, leading to connection or contact failure. Make sure that the emulator station is placed in a secure position so that it does not move and impose stress on the user interface during use.

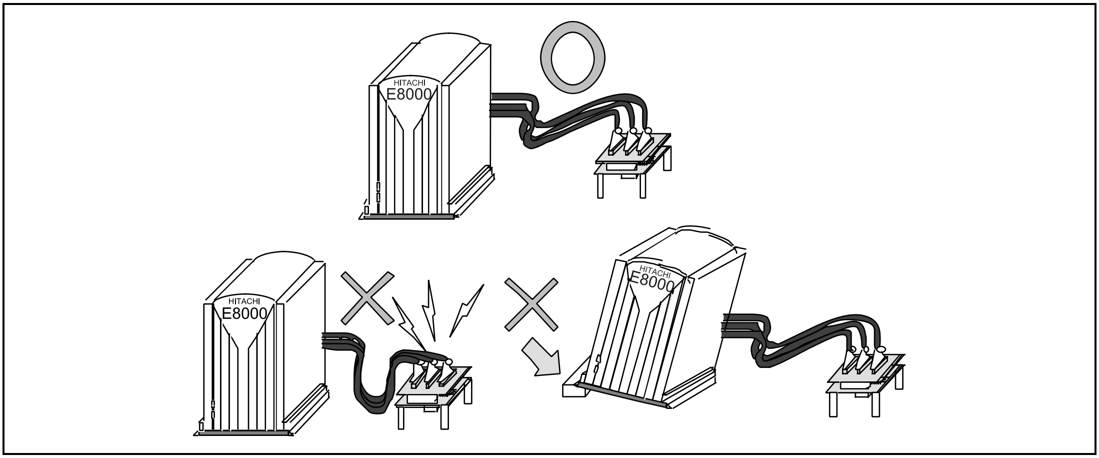


Figure C.4 Examples of Securing the Emulator Station

2. Make sure the power supply is off.

   Before connecting the evaluation chip board to the user system, check that the emulator and the user system are off.

Appendix D Memory Map

The memory map for each area is shown in figure D.1. X-RAM and Y-RAM areas do not exist for the SH7709A.

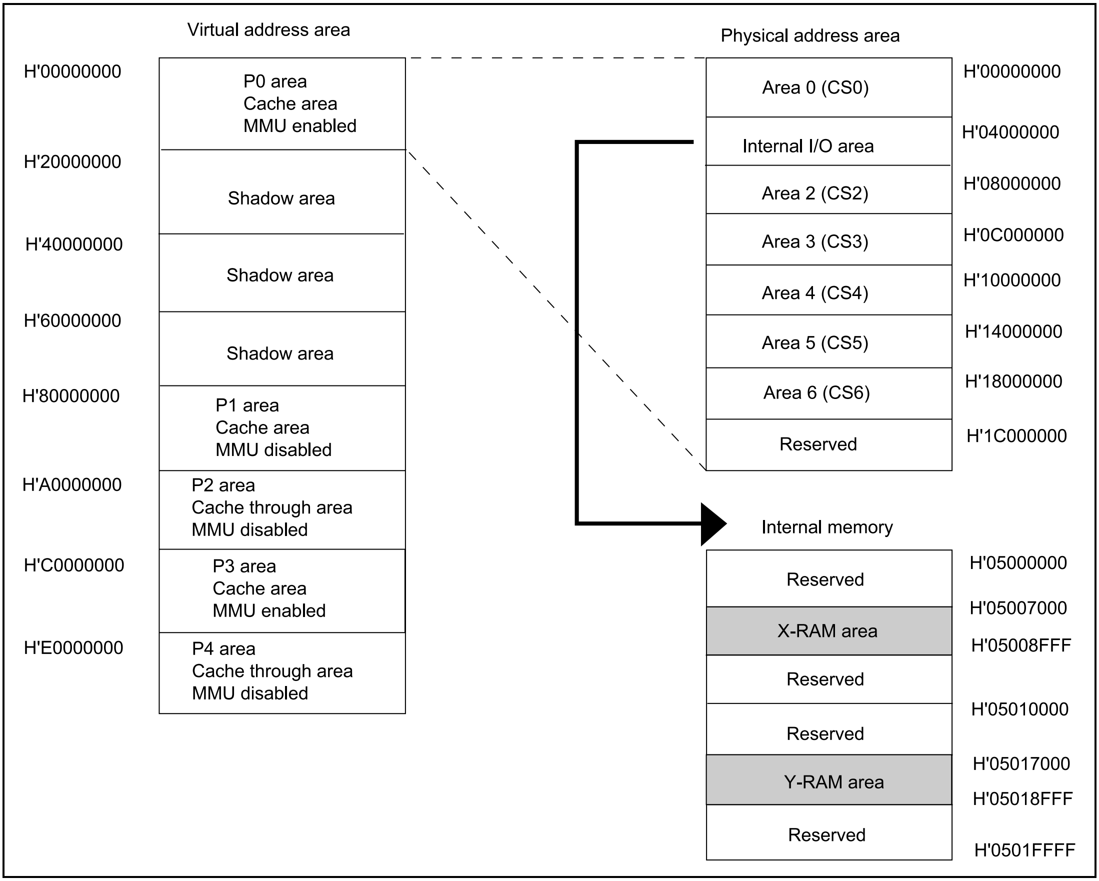


Figure D.1 Memory Map

SH7729/SH7709A E8000 Emulator User's Manual

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